Robotics: Science and Systems 2025
Los Angeles, CA, USA, June 21–25, 2025

RLDG: Robotic Generalist Policy Distillation via Reinforcement Learning

Charles Xu[1,†] Qiyang Li[1] Jianlan Luo[1,†] Sergey Levine[1]
[1]University of California, Berkeley. † Project Leads
Emails: {xuc, qcli}@berkeley.edu, {jianlanluo, svlevine}@eecs.berkeley.edu

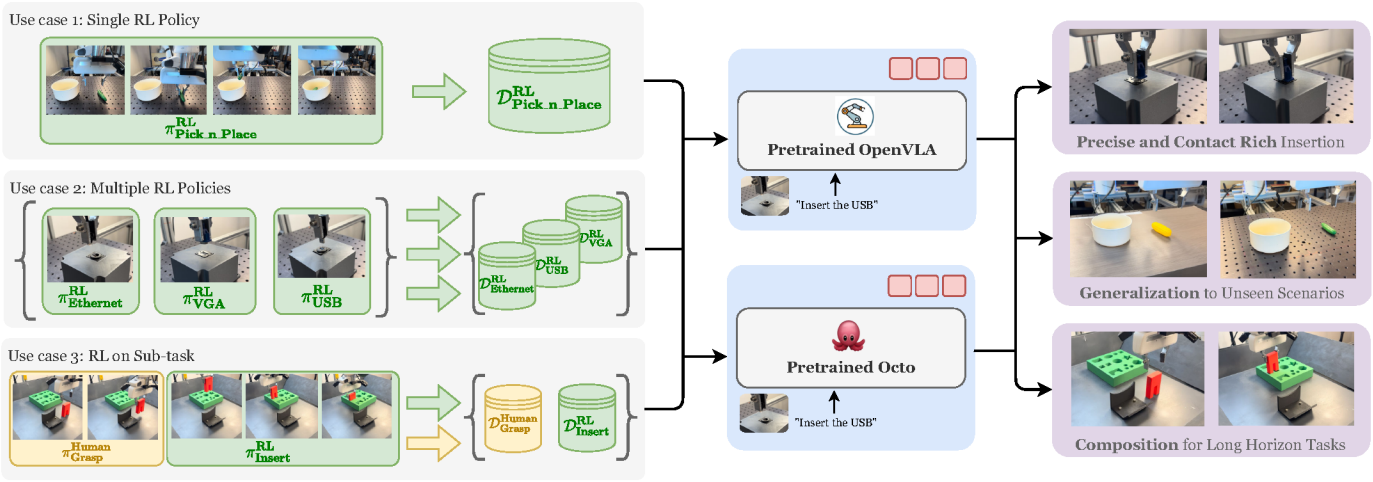

Fig. 1: RLDG improves generalist robot policies like OpenVLA and Octo by training specialist RL policies and using them to generate high-quality fine-tuning datasets. It has the flexibility to distill knowledge from multiple RL policies trained on individual narrowly scoped tasks into a single generalist. It can also be applied to the most critical sub-task of a long-horizon manipulation task, improving the success rate at the "bottleneck" while leveraging human demonstrations on parts of the task where it suffices. The resulting fine-tuned generalist policies are capable of precise manipulation, generalization to unseen scenarios, and composition of skills to solve long-horizon tasks.

***Abstract*—Recent advances in robotic foundation models have enabled the development of generalist policies that can adapt to diverse tasks. While these models show impressive flexibility, their performance heavily depends on the quality of their training data. In this work, we propose Reinforcement Learning Distilled Generalists (RLDG), a method that leverages reinforcement learning to generate high-quality training data for fine-tuning generalist policies. Through extensive real-world experiments on precise manipulation tasks like connector insertion and assembly, we demonstrate that generalist policies trained with RL-generated data consistently outperform those trained with human demonstrations, achieving up to 40% higher success rates while generalizing better to new tasks. We also provide a detailed analysis that reveals this performance gain stems from both optimized action distributions and improved state coverage. Our results suggest that combining task-specific RL with generalist policy distillation offers a promising approach for developing more capable and efficient robotic manipulation systems that maintain the flexibility of foundation models while achieving the performance of specialized controllers. Videos and code can be found on our project website https://generalist-distillation.github.io/.**

I. Introduction

Recent advances in robotic foundation models have demonstrated impressive capabilities in understanding and executing diverse manipulation skills [7, 4, 3, 35, 16, 2, 38, 19, 5]. By leveraging Internet-scale pretraining and grounding with robot actions, these models can achieve zero-shot and few-shot generalization across various domains. Deploying these models typically requires fine-tuning them with task-specific data to adapt to the target task or domain. The quality of this fine-tuning data is therefore critical to the performance of the resulting policies. While human teleoperation is a common and accessible source for such data, human demonstrations often contain inconsistencies in execution quality and style. These variations make it challenging for foundation models to learn robust policies, as they must cope with imperfections and inconsistencies inherent in human demonstrations. This challenge affects all robotic tasks but becomes particularly pronounced in scenarios requiring precise control and dexterity, such as contact-rich manipulation. These tasks demand fine-grained, reactive control to succeed, making the quality and consistency of demonstration data even more crucial for effective policy learning.

To tackle this challenge, we propose Reinforcement Learning Distilled Generalist (RLDG), a simple yet effective method that leverages reinforcement learning to generate high-quality training data for robotic foundation models. While directly fine-tuning foundation models with reinforcement learning is in principle possible, it also presents practical challenges in

terms of optimization stability, computational costs, as well as issues raised in value function pretraining [27].

Instead, our key insight is that RL agents can autonomously generate high-quality trajectories through reward maximization, making them better suited for fine-tuning generalist policies compared to human demonstrations. The approach is straightforward: we first train vision-based manipulation policies using sample-efficient real-world RL frameworks [24, 26] until convergence, then collect data from these policies to fine-tune robotic foundation models. This procedure is simple and flexible, offering several benefits. First, it provides an automated approach to generate large amounts of high-quality training data without requiring the effort of human teleoperation, which is particularly valuable since autonomous RL training is significantly more cost-effective than collecting human demonstrations. Second, by combining the optimization capabilities of RL with the strong generalization of foundation models, RLDG produces policies that achieve superior performance while generalizing to novel scenarios. Finally, RLDG provides a valuable solution for complex multi-stage tasks by using RL data to address the "bottleneck" step that hinders the performance of the overall task.

Through extensive experiments across multiple manipulation tasks with well-defined reward functions, we demonstrate that generalist policies like OpenVLA [16] and Octo [35] achieve superior performance when finetuned with RL data compared to human demonstrations, specifically for tasks where RL can learn effective controllers. For precise manipulation tasks such as tight-fitting connector insertions presented in Fig. 3, RLDG achieves 30% higher success rates on average. This performance gap widens further when evaluating generalization: policies trained with RLDG demonstrate significantly better transfer to novel scenarios, with on average 50% higher success rates. Notably, as we will show in Section IV, achieving comparable performance to RLDG would require 6-10x more human demonstrations. For complex tasks such as precise insertion, RLDG can achieve perfect success rates (100%), while policies trained on human demonstrations plateau at 90% even with significantly more data.

Our key contribution is RLDG, a simple yet effective method that leverages reinforcement learning to generate high-quality training data for fine-tuning pre-trained robotic foundation models, providing an automated alternative to human demonstrations. Through extensive experiments on robotic manipulation tasks, we demonstrate that RLDG achieves 30-50% higher success rates compared to conventional fine-tuning with human demonstrations while requiring 6-10x less data. Additionally, we show that RLDG can be flexibly combined with human demonstrations in multi-stage tasks, enabling better overall performance by using RL data for critical phases while maintaining the benefits of human demonstrations for other phases.

Our results suggest a promising direction for robotic learning: using reinforcement learning as an automated source of high-quality training data for foundation models. This synergy enables more capable robotic systems that can both execute skills precisely and generalize effectively to new scenarios through natural language instructions while reducing reliance on human demonstration data collection.

II. Related Work

Our work finetunes robotic foundation models with data generated by reinforcement learning policies, which could be seen as an instance of policy distillation [31]. By combining these approaches, we develop a general technique for training robust robotic policies that leverage both the performance of RL policies and the flexibility of foundation models. Thus, we survey related work across these three key areas and examine their intersections.

Foundation models for robotics. Recent advances in vision-language foundation models have enabled the development of generalist robotic policies that can understand and execute diverse tasks through natural language instructions. Such models [4, 3, 16, 35, 2, 38, 19, 5, 8] leverage large-scale pretraining on Internet-scale vision-language data followed by finetuning on robot demonstrations. While these approaches show impressive generalization capabilities across a wide range of tasks, our experiments demonstrate that they often struggle with precise manipulation tasks that require careful alignment and contact-rich interactions (see Section IV). This challenge comes from the limitations in the demonstration-based learning approach – human demonstrations, while diverse and adaptable, often lack the precision and repeatability needed for contact-rich manipulation tasks. Thus, robotic systems that can perform these challenging tasks more effectively than humans are highly desirable. By definition, simply imitating human behaviors would not achieve this goal. RLDG addresses this limitation by complementing the semantic understanding of foundation models with the robust behaviors learned through reinforcement learning, enabling precise manipulation while maintaining the flexibility and generalization capabilities of foundation models.

Reinforcement learning for robotic manipulation. Reinforcement learning has been successfully applied to learn complex robotic manipulation skills in the real world through direct interaction with the environment [24, 26, 29, 18, 14, 15, 13, 30, 33]. Prior work has demonstrated RL's effectiveness in learning challenging tasks like precision insertion [22, 39, 21, 20], multi-stage assembly [10], and dexterous in-hand manipulation [14]. A key advantage of RL is its ability to discover optimal action distributions through trial-and-error exploration, leading to more robust and efficient policies compared to pure imitation learning [23, 26, 24]. However, to achieve broader generalization across different tasks and situations, training online RL policies would require careful environment instrumentation such as resetting the environment to different initial conditions for each trial, while this is in principle possible, it would quickly become impractical when the number of tasks and situations grows. RLDG bridges this gap by combining the strengths of both approaches, using RL to learn optimal behaviors for specific challenging tasks,

then distilling these precise capabilities into foundation models while preserving their broad generalization abilities.

Policy distillation and knowledge transfer. The idea of distilling multiple specialized policies into a single more general policy has been explored extensively in the RL and robotics literature [17], including methods that use RL to distill into general-purpose neural networks [31, 28], methods that employ bidirectional constraints between specialists and generalists [36, 9], and methods that focus on continual learning [32, 34].

These approaches have demonstrated that careful distillation can preserve the essential behavioral characteristics of expert policies while potentially adding beneficial properties like improved generalization or reduced computational requirements. While prior work has explored policy distillation in various contexts, our work introduces two key innovations: (1) we show that distilling RL policies into foundation models that leverage large-scale pre-training yields better performance than training from human demonstrations while exhibiting better generalization capabilities than specialized RL policies, and (2) we demonstrate that for precise manipulation tasks, using RL-generated data for fine-tuning foundation models produces superior performance compared to using human demonstrations, even when high-quality demonstrations are available. Together, these findings establish RLDG as a practical approach for enhancing foundation models with specialized RL capabilities while maintaining their broad generalization abilities.

III. Reinforcement Learning Distilled Generalist

Reinforcement Learning Distilled Generalist (RLDG) is a simple yet effective method for enhancing generalist policy performance through the distillation of specialized RL policies. In RLDG, we train RL policies for individual tasks and then use these policies to generate training data that can be used to fine-tune a single generalist robotic manipulation policy, such as OpenVLA [16] or Octo [35]. Although specialized RL policies can achieve high performance on specific tasks, they often lack zero-shot generalization and robustness to disturbances. Conversely, generalist policies excel at generalization but can struggle to achieve high performance when trained on human demonstrations, for example due to suboptimal data or modality mismatches between human demonstrators and robot policies (e.g., different viewpoints, memory, and task knowledge). RLDG bridges this gap through knowledge distillation, resulting in more performant generalists compared to finetuning on human demonstrations, while demonstrating stronger generalization capabilities compared to the original RL policies. This distillation approach through data generation with RL is agnostic to both the choice of RL algorithm and generalist policy architecture, making it flexible to any model choice. Furthermore, it offers flexibility to train and collect data with separate RL policies trained on multiple narrowly scoped tasks (such as one policy for each connector in the `Connector Insertion` task). We can also elect to train RL on the "bottleneck" segments of a long-horizon task that require the most precision and benefit the most from RL-generated data, while leaving the less critical parts for humans to demonstrate. This simplifies the RL training complexity, improves data diversity for better generalist performance, and avoids training RL on unnecessarily long-horizon tasks.

A. Online RL Training

We can formulate each robotic task as a Markov decision process (MDP), where the state s_t consists of RGB images and proprioceptive information, and actions a_t represent desired end-effector movements. The policy objective $\pi(a_t|s_t)$ is to maximize the expected discounted return:

$$J(\pi) = \mathbb{E}_{\substack{s_0\sim\rho_0 \\ a_t\sim\pi(a_t|s_t) \\ s_{t+1}\sim P(s_{t+1}|s_t,a_t)}} \left[\sum_{t=0}^{T} \gamma^t R(s_t, a_t)\right], \quad (1)$$

where ρ_0 defines the initial robot configurations, P represents the system's transition dynamics, and $R : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is a reward function encoding the task objectives.

While RLDG is agnostic to the choice of RL algorithm, we implement RLDG using HIL-SERL [26] motivated by its sample efficiency and high performance for learning precise real-world manipulation skills from pixel input. It incorporates human interventions with RLPD [1] to efficiently learn visuomotor policies that consistently achieve 100% success rate by maximizing (1).

B. Experience Collection

After training RL experts for each of the tasks provided to RLDG, we collect a high-quality fine-tuning dataset by rolling out the converged policies. Since we transfer knowledge from RL into the generalist policy only through this data, we have the flexibility to mix experience from multiple sources. For tasks that involve separate RL policies per manipulation object like `Connector Insertion`, we rolled out each policy and constructed a balanced fine-tuning dataset consisting of equal number of episodes per object. In cases where RL is only trained on a segment of the task like `FMB Assembly`, we combine the RL rollouts with human demonstrations for the remainder of the task.

C. Generalist Policy Finetuning

Robot generalist models are often pre-trained on diverse large-scale datasets before being fine-tuned to improve performance its performance on a particular task while preserving the generalization capabilities form the diverse pre-training. In RLDG, we use the data collected as described above to fine-tune these generalist models. Specifically, suppose we have a pre-trained policy π_0, we fine-tune it with task-specific dataset $D_{(s_t,a_t)}$ with the following supervised learning objective:

$$\mathcal{L}(\theta) = -\mathbb{E}_{(s_t,a_t)\sim\mathcal{D}}[\log \pi_\theta(a_t|s_t)]. \quad (2)$$

We evaluate the efficacy of our method by fine-tuning two pre-trained robot generalist models using different action parametrization.

OpenVLA. OpenVLA [16] is a 7B-parameter vision-language-action model built on Llama 2 [37]. It takes a

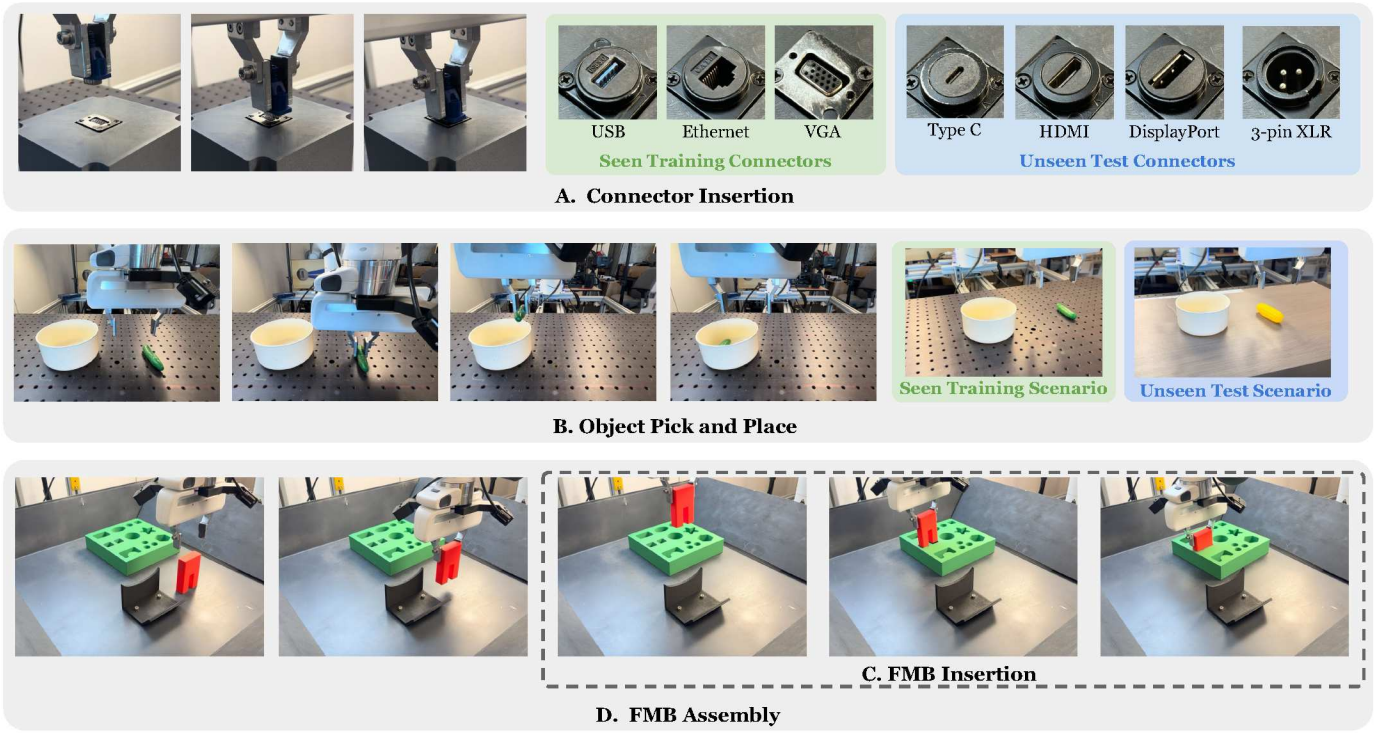

Fig. 3: Illustrations of tasks used to evaluate RLDG. (**A**) Precise `Connector Insertion` includes three training objects and four unseen test objects for evaluating policy generalization. (**B**) `Pick and Place` involves an unseen scenario that tests the policy's visual robustness to different backgrounds and objects. (**C**) `FMB Insertion` involves inserting a pre-grasped object in a moving board while (**D**) `FMB Assembly` starts with the object on the table and involves an additional grasping phase.

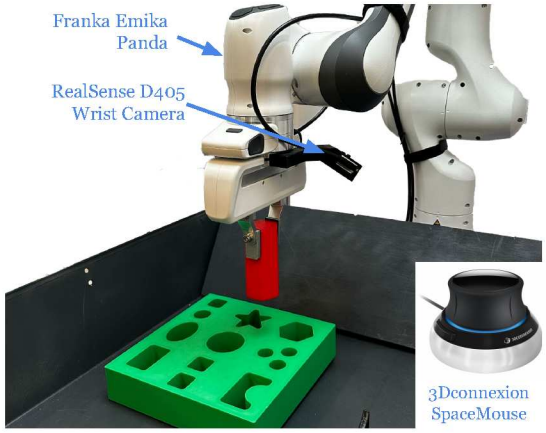

Fig. 4: We use a Franka Emika Panda arm with a parallel jaw gripper teleoperated by a 3Dconnexion SpaceMouse device. There is a single RealSense D405 camera mounted on the robot's wrist for image observations.

single image as observation input along with a language instruction. It predicts 7-dimensional actions which are discretized into 256 bins per dimension autoregressively using the standard cross-entropy loss. To fine-tune the model on our RL-generated dataset, we use the public model weights pre-trained on 970 thousand Open X-Embodiment dataset [7] and apply Low Rank Adaptation (LoRA) [12], a popular parameter-efficient fine-tuning method.

Octo. Octo is another open-source generalist robotic policy, designed to adapt to diverse sensory inputs and action spaces efficiently. Different from OpenVLA, Octo predicts continuous actions with a diffusion head, which excels at modeling multimodal distributions, helpful for imitating human demonstrations [6]. To predict an action, the transformer backbone takes in the tokenized observation and goal, then outputs a readout embedding e, which is used to condition the denoising process trained on the standard DDPM objective [11]. We take the pre-trained Octo-Base model, remove its secondary image tokenizer, and mask out the image goal to match our input modalities, and directly fine-tune the remainder of the network on our RL-generated dataset.

IV. Experiment and Results

Our experiments aim to evaluate RLDG in terms of its ability to improve over both imitation learning methods for training generalist policies (in terms of performance), and its ability to improve over more specialized RL policies (in terms of generalization). We use a test suite of tasks that require precise and delicate manipulation, and thus are particularly challenging for imitation learning methods. We focus on two main questions: (1) Is the RLDG approach for training generalists using data from RL more effective than the conventional approach of training on demonstration data? (2) Is the generalist policy that results from RLDG training more effective at generalizing than the RL policies used to generate the training data?

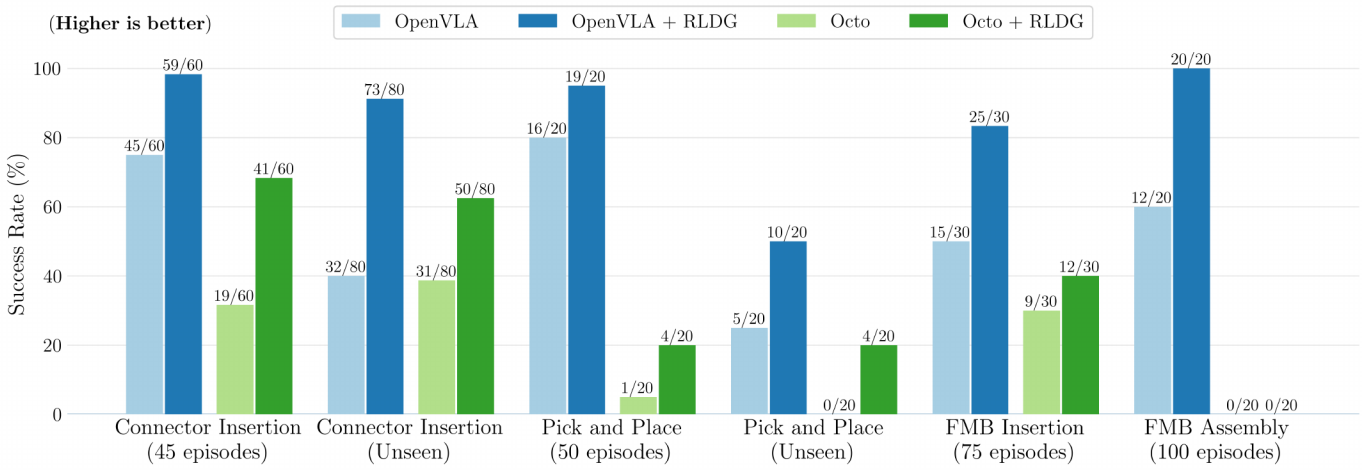

Fig. 5: Success rate comparison of OpenVLA and Octo policies fine-tuned with RLDG versus conventional methods using human demonstrations. Both generalists trained with RLDG consistently outperform their counterparts trained with the same number of successful expert human demonstrations in both training and unseen scenarios.

A. Experimental Setup and Tasks

Our robot setup for all experiments is shown in Fig. 4. The arm tracks end-effector commands with a 1kHz low-level impedance controller. Data collection, RL, and Octo policies command actions at 10Hz, while OpenVLA runs at 4Hz due to inference speed limitations. The action space for all policies is a 6-dimensional end-effector delta pose in the wrist frame and 1 binary gripper action for tasks that involve grasping. We collect expert demonstrations using a SpaceMouse device that streams 6D end-effector twists matching the RL and the generalist policy's action spaces. The RL policy's observation space consists of a single 128×128 wrist RGB image along with end-effector pose, velocity, and wrench measurements. For the generalist policies, we fine-tune only using the wrist camera image as input.

We evaluate RLDG on four real-world manipulation tasks that present distinct challenges. These include high-precision contact-rich tasks that typically challenge generalist models, pick-and-place tasks where we show RLDG can further improve performance, and multi-stage assembly tasks that leverage RLDG's ability to compose skills. Through these tasks, we also evaluate the method's ability to generalize to unseen configurations.

Connector Insertion. This task requires inserting various electronic connectors into their matching ports, which requires sub-millimeter precision and dexterity to deal with the intricate contact dynamics during alignment. We train separate RL policies and use them to collect data on `USB`, `Ethernet`, and `VGA` connectors before distilling them into a single generalist policy. We also use `Type-C`, `HDMI`, `Display Port`, and `3-pin XLR` connectors to evaluate the policy's zero-shot generalization performance.

Pick and Place. We also test our method on a pick-and-place task, where the robot grasps an object from a randomized location and places it in a bowl. To test generalization, we also evaluate on an unseen scenario by replacing the training object and background as shown in Fig. 3. With this experiment, we aim to demonstrate RLDG's effectiveness on tasks that generalist policies are often used for and benchmarked on.

FMB Insertion. We also use the single object insertion task of FMB [25], a common and reproducible benchmark for comparing robotic manipulation methods. This task involves inserting a pre-grasped object into a matching opening with $\pm 1.5mm$ tolerance at randomized positions. We utilize this task primarily for our analysis experiments in Section V.

FMB Single Object Assembly. This task stems from the FMB Single-Object Multi-Stage Assembly, which adds a grasping phase on top of the `FMB Insertion` task above. We use this multi-stage task to demonstrate RLDG's ability to enhance overall task performance by distilling RL data for the precision-critical insertion phase while using human demonstrations for the grasping and transport phases.

More details on the experiment tasks and training procedure can be found in the Appendix.

B. RLDG vs. Conventional Fine-tuning

In this section, we seek to answer Question 1 by comparing generalist policies fine-tuned using RLDG and standard generalist fine-tuning via imitation learning. For each task, we fine-tune OpenVLA and Octo on RL-generated data as described in Sec. III, and on expert human demonstrations. For a fair comparison, we use the same task setup, training configuration, observation and action space, and the number of successful episodes for both methods. The only difference is the source of the data (RL vs. human). We aim to evaluate whether RL policies are a better source of training data for generalist models than conventional human demonstrations in terms of resultant policy performance.

a) Success rate.: We present the success rate of each policy and method in Fig. 5. On each task, both OpenVLA and Octo fine-tuned with RL-generated data consistently achieved higher success rates than their counterparts trained with human demonstrations, in both seen and unseen evaluation scenarios.

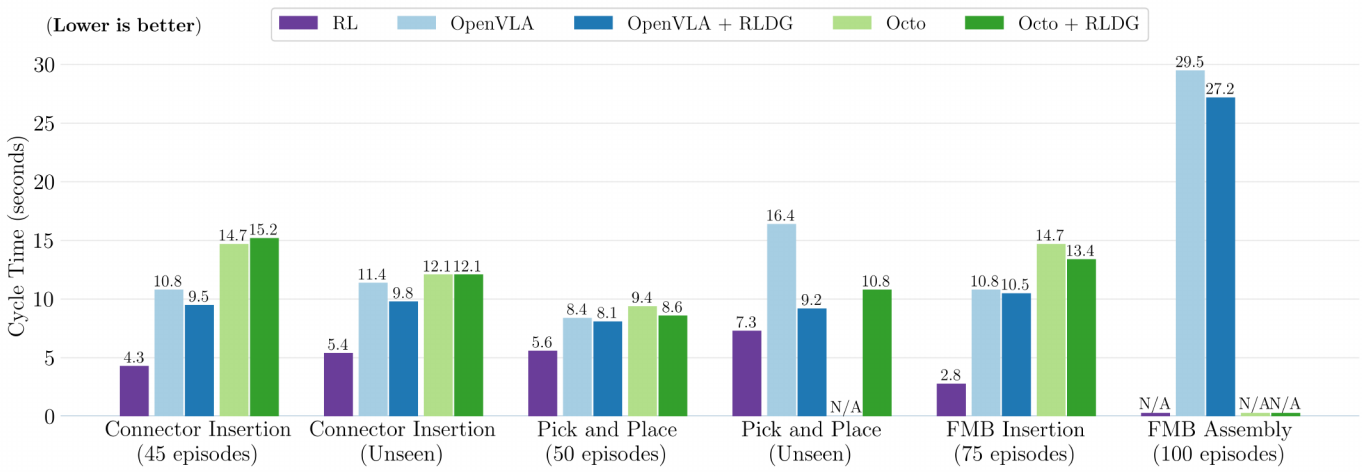

Fig. 6: Cycle time comparison between policies trained with RL data versus human demonstrations. N/A for RL in `FMB Assembly` denotes policy not trained on the whole task, while N/A for fine-tuned policies denotes no successes recorded. The RL-trained policies generally achieve faster execution times across tasks, demonstrating the efficiency benefits of using RL-generated data for policy training.

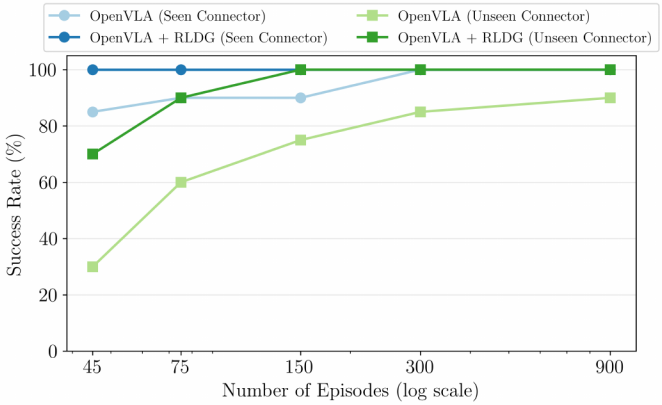

Fig. 7: Success rate of OpenVLA policies fine-tuned on different sizes of RL-generated and human-collected datasets. When evaluated on seen (VGA) and unseen (Type C) `Connector Insertion` tasks, RLDG shows superior sample efficiency, requiring significantly fewer demonstrations to achieve perfect success rate in both scenarios while the performance of conventional method saturates in the unseen case.

On the precise `FMB Insertion` and `Connector Insertion` tasks, where we anticipated the generalist to benefit the most from higher quality training data, OpenVLA with RLDG saw 33% and 23% higher success rates, respectively, compared to the baseline. The benefit of RLDG is equally pronounced for Octo, where it improved the success rate by 10% and 37%, respectively, although the overall success rate is lower than OpenVLA. We found that RLDG also improved the success rate for `Pick and Place` from 16/20 to 19/20 for OpenVLA and 1/20 to 4/20 for Octo. Furthermore, the training task performance boost of RLDG also carried over to unseen evaluation scenarios. OpenVLA with RLDG achieved over 2 times higher success rate than OpenVLA with human data for unseen `Connector Insertion`, while applying RLDG on Octo increased the success rate from 0/20 to 4/20 in the unseen `Pick and Place` task. When we strategically combine human demonstrations with RL-generated data on the `FMB Assembly` task, the resulting OpenVLA policy also significantly outperformed the version trained purely on human demonstrations. It achieved 20/20 successes with RL-generated data compared to 12/20 with human demonstrations, suggesting the flexibility and effectiveness of RLDG for improving the "bottleneck" of a long-horizon task.

b) Scaling analysis.: To further investigate the effectiveness of RLDG, we conduct a scaling experiment studying the success rate of OpenVLA policies on a seen VGA connector and an unseen Type-C connector when fine-tuned on different numbers of RL-generated and human-collected episodes. We present the results in Fig. 7. For the VGA connector, OpenVLA with RLDG achieved a 100% success rate with just 45 RL episodes, compared to 300 required from human demonstrations to achieve the same success rate. Furthermore, the policy trained with 150 RL rollouts on VGA, USB-A, and Ethernet connectors achieved a 100% success rate on the unseen Type-C connector insertion, while OpenVLA trained on human demonstrations plateaued at a 90% success rate even with 900 demonstrations. These results strongly suggest that fine-tuning generalist policies using RLDG is more sample-efficient and leads to higher performance than human demonstrations for both in-distribution and unseen tasks.

c) Cycle time.: As shown in Fig. 6, the generalist policies trained with RL data consistently have faster cycle times compared to those trained with human demonstrations across tasks, although the gap is not as significant. For OpenVLA, RLDG decreased the cycle time between 0.3 to 2.3 seconds per task, while Octo saw little improvement on average. This improvement can be attributed to the inherent speed optimization in RL training through temporal discounting,

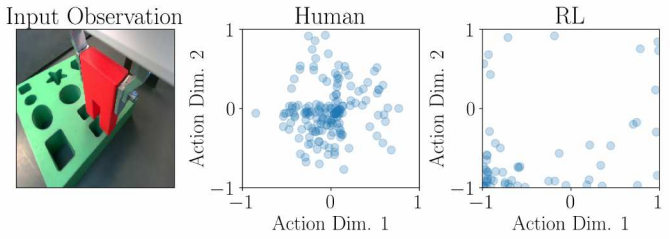

Fig. 8: **Action distribution visualization for RL data and human demo data for the *FMB insertion* task.** We visualize the first two dimensions of the dataset actions after filtering all the transitions in the dataset where the end-effector positions are close to the position shown in the image on the left (x/y coordinates are both within 4mm and z coordinate is within 10mm). The robot arm needs to move in the -x direction and in the -y direction to reach the insertion point. The first two dimensions of the action space corresponds to the control of the x and y position of the end-effector position correspondingly. Human actions are clustered around the center of the action space whereas the RL actions are more optimized, and mostly found near the correct corner (bottom-left) of the action space.

which is then distilled into the generalist policy by collecting trajectories that solve the task faster than the human expert. However, all generalist policies were still much slower at solving the tasks than the original RL policy. For OpenVLA, we primarily attribute this deficiency to the control frequency gap between the RL policy's 10Hz and OpenVLA's 4Hz, changing the system dynamics and lowering the maximum velocity of the arm. We believe the speed of OpenVLA can be significantly improved if inference can be sped up to match the RL policy frequency. For Octo, the fine-tuned policies were unable to fit the fine-tuning dataset perfectly, leading to lower success rate and longer cycle time overall.

C. Generalization of RLDG vs. Original RL Policies

To address Question 2, we compare the generalization performance of generalists trained using RLDG against that of the original RL policies used to generate the data. As shown in Fig. 5, the RL policy success rate quickly degraded from 20/20 for the training scenario to 1/20 for the unseen scenario of the `Pick and Place` task. In contrast, OpenVLA and Octo with RLDG achieved 10/20 and 4/20 success rates respectively on the same task. Additionally, the multi-task capabilities of OpenVLA and Octo allowed fine-tuning on multiple connector data in the `Connector Insertion` task, achieving 73/80 and 50/80, respectively, when evaluated across 4 unseen connectors, whereas the best of the three RL policies trained on single connectors recorded only 49/80 successes.

Our experimental results on challenging dexterous manipulation tasks demonstrated several key advantages of RLDG. Generalist robot policies trained on RL-generated data using RLDG consistently achieve higher performance across all tasks compared to conventional fine-tuning methods using human demonstrations. Compared to directly using the RL policies that generated the data, RLDG also demonstrated much greater generalization capabilities and robustness to unseen test scenarios.

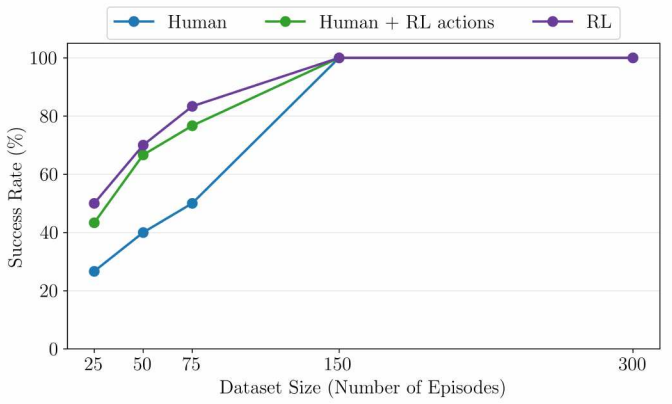

Fig. 9: **Fine-tuning success rate on the *FMB insertion* task with different fine-tuning data sources and varied dataset sizes (from 25 trajectories to 300 trajectories).** *Human*: demo trajectories collected by human teleoperators. *Human + RL actions*: the same human demo trajectories but with all the actions relabeled by a trained RL agent. *RL*: rollouts collected by the RL agent. RL data consistently provide better fine-tuning performance than human data. *Human + RL actions* closes the gap mostly, suggesting that most of the benefits of RL data come from it having better action quality.

V. Analysis: Why is RL Data Better than Human Data?

We have shown that fine-tuning generalist policies with RL data yields superior performance compared to training on human data. However, it is unclear where these benefits are coming from. In this section, we analyze the source of the benefits in two parts. The first part focuses on studying the benefits of RL actions and the state distribution in RL data in isolation. The second part focuses on dissecting the failure modes of the fine-tuned policies on each individual task.

A. Is RL data better because of better action or state distribution?

To answer this question, we use the *FMB insertion* task and create a "mixed" dataset where we take the human data and relabel the actions using action samples from the RL policy. Comparing the fine-tuning performance of the "mixed" dataset with the purely RL data and the purely human demo data would allow us to see the benefits of the actions and the state distribution in isolation. As shown in Figure 9, mixing human states and RL actions yields a better fine-tuning success rate than using fully human data (more than 50% improvements when fine-tuning on 25/50/75 trajectories), while still being worse than using fully RL data. This suggests that while RL action and state distribution *both* contribute to the fine-tuning performance improvements, action quality is the factor that contributes to the performance improvement the most. Figure 8 shows a comparison of human and RL actions. We can see that the RL action distribution assigns more density on the correct direction (bottom-left) that moves the end-effector towards the insertion point whereas human action distribution focuses mostly around the middle with a slight bias towards the correct direction. This suggests that RL actions are more optimal than human actions, resulting in the better sample efficiency for fine-tuning we observe in Figure 9.

B. Qualitative Analysis: Failure Modes

To further understand why RL-generated data leads to better performance, we also analyzed failure modes across tasks. We observed that policies trained with RL-generated data consistently helped overcome alignment issues in precise, contact-rich tasks and reduced premature gripper closure during grasping. Videos of each policy on each task can be found on our project website (https://generalist-distillation.github.io/).

a) Connector and FMB Insertion.: In both tasks, RL data eliminated a "stuck" state where the object contacts the board but fails to align properly. Human demonstration policies often maintained contact pressure without necessary exploratory movements. Furthermore, RL data also improved approach trajectories, preventing early descents that caused connectors to catch on socket lips.

b) Pick and Place.: RL data improved grasp reliability, reducing premature gripper closure seen in human demonstration-trained policies. However, an interesting RL-specific failure mode was observed: objects were sometimes dropped too early, bouncing out of the bowl. This likely resulted from RL's speed optimization, where objects were released immediately after clearing the bowl's edge, but the distilled policy lacked precise timing.

c) FMB Assembly.: While both OpenVLA policies performed similarly in grasping and transport phases as they are trained on the same human data, the performance gap emerged during insertion, with RL data better addressing alignment issues much like in the insertion tasks. Octo's failure was due to consistent grasping errors where the fingers are in front of the object, likely due to the lack of good depth perception.

VI. Discussion and Limitations

In this work, we presented RLDG, a simple method that fine-tunes generalist policies on high-quality data generated by RL policies. We demonstrated that generalist policies fine-tuned using RLDG consistently outperform those trained with human expert demonstrations on a suite of real-world challenging precise manipulation tasks. Our method can be applied in real-world robotic manipulation tasks that require a large amount of training data or where policy performance using human demonstrations saturate. Our work also opens up avenues for making autonomous improvements of generalist policies more scalable and tractable. First, our method assumes access to reward functions for fine-tuning tasks which may present difficulties when the task rewards are hard to specify. Possible future directions include autonomously generating fine-tuning tasks with reward functions (e.g., using pre-trained VLMs) such that there is no need for manual task specification. Furthermore, our RL policies optimize not only for task success but the speed in doing so. Such an objective does not necessarily result in policies that are robust in distillation errors. For example, on the `Pick and Place` task, the policy fine-tuned on RL-generated data always tries to place the object immediately after it has moved close enough to the goal location but sometimes drops the object too early (see Section V-B).

We also note that the human demonstrations in this study were collected by an author with extensive experience using the tele-operation interface. We acknowledge that it may raise concerns about the quality of the demonstrations being suboptimal and the fairness of the comparison to fine-tuning on RL data. However, we stress that the demonstrator made a sincere effort to collect the best possible demonstrations. While we believe this setup reflects a fair comparison, we welcome future work to further validate our findings with demonstrations from a broader set of users.

We hope that our work will serve as a step toward practical RL training of generalist policies, providing both a way to improve generalist policies such as VLAs with a RL and a way to improve the generalization of RL policies via distillation into generalist policies.

Acknowledgments

This research was supported in part by NSF IIS-2150826 and ONR N00014-25-1-2060.

References

[1] Philip J. Ball, Laura Smith, Ilya Kostrikov, and Sergey Levine. Efficient online reinforcement learning with offline data. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett, editors, *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 1577–1594. PMLR, 23–29 Jul 2023. URL https://proceedings.mlr.press/v202/ball23a.html.

[2] Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Lachy Groom, Karol Hausman, Brian Ichter, Szymon Jakubczak, Tim Jones, Liyiming Ke, Sergey Levine, Adrian Li-Bell, Mohith Mothukuri, Suraj Nair, Karl Pertsch, Lucy Xiaoyang Shi, James Tanner, Quan Vuong, Anna Walling, Haohuan Wang, and Ury Zhilinsky. π_0: A vision-language-action flow model for general robot control, 2024. URL https://arxiv.org/abs/2410.24164.

[3] Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Xi Chen, Krzysztof Choromanski, Tianli Ding, Danny Driess, Avinava Dubey, Chelsea Finn, Pete Florence, Chuyuan Fu, Montse Gonzalez Arenas, Keerthana Gopalakrishnan, Kehang Han, Karol Hausman, Alexander Herzog, Jasmine Hsu, Brian Ichter, Alex Irpan, Nikhil Joshi, Ryan Julian, Dmitry Kalashnikov, Yuheng Kuang, Isabel Leal, Lisa Lee, Tsang-Wei Edward Lee, Sergey Levine, Yao Lu, Henryk Michalewski, Igor Mordatch, Karl Pertsch, Kanishka Rao, Krista Reymann, Michael Ryoo, Grecia Salazar, Pannag Sanketi, Pierre Sermanet, Jaspiar Singh, Anikait Singh, Radu Soricut, Huong Tran, Vincent Vanhoucke, Quan Vuong, Ayzaan Wahid, Stefan Welker, Paul Wohlhart, Jialin Wu, Fei Xia, Ted Xiao, Peng Xu, Sichun Xu, Tianhe Yu, and Brianna Zitkovich. Rt-2: Vision-language-action models

transfer web knowledge to robotic control, 2023. URL https://arxiv.org/abs/2307.15818.

[4] Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Joseph Dabis, Chelsea Finn, Keerthana Gopalakrishnan, Karol Hausman, Alex Herzog, Jasmine Hsu, Julian Ibarz, Brian Ichter, Alex Irpan, Tomas Jackson, Sally Jesmonth, Nikhil J Joshi, Ryan Julian, Dmitry Kalashnikov, Yuheng Kuang, Isabel Leal, Kuang-Huei Lee, Sergey Levine, Yao Lu, Utsav Malla, Deeksha Manjunath, Igor Mordatch, Ofir Nachum, Carolina Parada, Jodilyn Peralta, Emily Perez, Karl Pertsch, Jornell Quiambao, Kanishka Rao, Michael Ryoo, Grecia Salazar, Pannag Sanketi, Kevin Sayed, Jaspiar Singh, Sumedh Sontakke, Austin Stone, Clayton Tan, Huong Tran, Vincent Vanhoucke, Steve Vega, Quan Vuong, Fei Xia, Ted Xiao, Peng Xu, Sichun Xu, Tianhe Yu, and Brianna Zitkovich. Rt-1: Robotics transformer for real-world control at scale, 2023. URL https://arxiv.org/abs/2212.06817.

[5] Chi-Lam Cheang, Guangzeng Chen, Ya Jing, Tao Kong, Hang Li, Yifeng Li, Yuxiao Liu, Hongtao Wu, Jiafeng Xu, Yichu Yang, Hanbo Zhang, and Minzhao Zhu. Gr-2: A generative video-language-action model with web-scale knowledge for robot manipulation, 2024. URL https://arxiv.org/abs/2410.06158.

[6] Cheng Chi, Zhenjia Xu, Siyuan Feng, Eric Cousineau, Yilun Du, Benjamin Burchfiel, Russ Tedrake, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion, 2024. URL https://arxiv.org/abs/2303.04137.

[7] Open X-Embodiment Collaboration, Abhishek Padalkar, Acorn Pooley, Ajinkya Jain, Alex Bewley, Alex Herzog, Alex Irpan, Alexander Khazatsky, Anant Rai, Anikait Singh, Anthony Brohan, Antonin Raffin, Ayzaan Wahid, Ben Burgess-Limerick, Beomjoon Kim, Bernhard Schölkopf, Brian Ichter, Cewu Lu, Charles Xu, Chelsea Finn, Chenfeng Xu, Cheng Chi, Chenguang Huang, Christine Chan, Chuer Pan, Chuyuan Fu, Coline Devin, Danny Driess, Deepak Pathak, Dhruv Shah, Dieter Büchler, Dmitry Kalashnikov, Dorsa Sadigh, Edward Johns, Federico Ceola, Fei Xia, Freek Stulp, Gaoyue Zhou, Gaurav S. Sukhatme, Gautam Salhotra, Ge Yan, Giulio Schiavi, Hao Su, Hao-Shu Fang, Haochen Shi, Heni Ben Amor, Henrik I Christensen, Hiroki Furuta, Homer Walke, Hongjie Fang, Igor Mordatch, Ilija Radosavovic, Isabel Leal, Jacky Liang, Jaehyung Kim, Jan Schneider, Jasmine Hsu, Jeannette Bohg, Jeffrey Bingham, Jiajun Wu, Jialin Wu, Jianlan Luo, Jiayuan Gu, Jie Tan, Jihoon Oh, Jitendra Malik, Jonathan Tompson, Jonathan Yang, Joseph J. Lim, João Silvério, Junhyek Han, Kanishka Rao, Karl Pertsch, Karol Hausman, Keegan Go, Keerthana Gopalakrishnan, Ken Goldberg, Kendra Byrne, Kenneth Oslund, Kento Kawaharazuka, Kevin Zhang, Keyvan Majd, Krishan Rana, Krishnan Srinivasan, Lawrence Yunliang Chen, Lerrel Pinto, Liam Tan, Lionel Ott, Lisa Lee, Masayoshi Tomizuka, Maximilian Du, Michael Ahn, Mingtong Zhang, Mingyu Ding, Mohan Kumar Srirama, Mohit Sharma, Moo Jin Kim, Naoaki Kanazawa, Nicklas Hansen, Nicolas Heess, Nikhil J Joshi, Niko Suenderhauf, Norman Di Palo, Nur Muhammad Mahi Shafiullah, Oier Mees, Oliver Kroemer, Pannag R Sanketi, Paul Wohlhart, Peng Xu, Pierre Sermanet, Priya Sundaresan, Quan Vuong, Rafael Rafailov, Ran Tian, Ria Doshi, Roberto Martín-Martín, Russell Mendonca, Rutav Shah, Ryan Hoque, Ryan Julian, Samuel Bustamante, Sean Kirmani, Sergey Levine, Sherry Moore, Shikhar Bahl, Shivin Dass, Shuran Song, Sichun Xu, Siddhant Haldar, Simeon Adebola, Simon Guist, Soroush Nasiriany, Stefan Schaal, Stefan Welker, Stephen Tian, Sudeep Dasari, Suneel Belkhale, Takayuki Osa, Tatsuya Harada, Tatsuya Matsushima, Ted Xiao, Tianhe Yu, Tianli Ding, Todor Davchev, Tony Z. Zhao, Travis Armstrong, Trevor Darrell, Vidhi Jain, Vincent Vanhoucke, Wei Zhan, Wenxuan Zhou, Wolfram Burgard, Xi Chen, Xiaolong Wang, Xinghao Zhu, Xuanlin Li, Yao Lu, Yevgen Chebotar, Yifan Zhou, Yifeng Zhu, Ying Xu, Yixuan Wang, Yonatan Bisk, Yoonyoung Cho, Youngwoon Lee, Yuchen Cui, Yueh hua Wu, Yujin Tang, Yuke Zhu, Yunzhu Li, Yusuke Iwasawa, Yutaka Matsuo, Zhuo Xu, and Zichen Jeff Cui. Open X-Embodiment: Robotic learning datasets and RT-X models. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, Yokohama, Japan, 2024.

[8] Danny Driess, Fei Xia, Mehdi S. M. Sajjadi, Corey Lynch, Aakanksha Chowdhery, Brian Ichter, Ayzaan Wahid, Jonathan Tompson, Quan Vuong, Tianhe Yu, Wenlong Huang, Yevgen Chebotar, Pierre Sermanet, Daniel Duckworth, Sergey Levine, Vincent Vanhoucke, Karol Hausman, Marc Toussaint, Klaus Greff, Andy Zeng, Igor Mordatch, and Pete Florence. Palm-e: An embodied multimodal language model, 2023. URL https://arxiv.org/abs/2303.03378.

[9] Dibya Ghosh, Avi Singh, Aravind Rajeswaran, Vikash Kumar, and Sergey Levine. Divide-and-conquer reinforcement learning, 2018. URL https://arxiv.org/abs/1711.09874.

[10] Abhishek Gupta, Justin Yu, Tony Z. Zhao, Vikash Kumar, Aaron Rovinsky, Kelvin Xu, Thomas Devlin, and Sergey 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, Xi'an, China, May 30 - June 5, 2021*, pages 6664–6671. IEEE, 2021. doi: 10.1109/ICRA48506.2021.9561384. URL https://doi.org/10.1109/ICRA48506.2021.9561384.

[11] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *arXiv preprint arxiv:2006.11239*, 2020.

[12] Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. In *ICLR*. OpenReview.net,

2022. URL http://dblp.uni-trier.de/db/conf/iclr/iclr2022.html#HuSWALWWC22.

[13] Hengyuan Hu, Suvir Mirchandani, and Dorsa Sadigh. Imitation bootstrapped reinforcement learning, 2024. URL https://arxiv.org/abs/2311.02198.

[14] Zheyuan Hu, Aaron Rovinsky, Jianlan Luo, Vikash Kumar, Abhishek Gupta, and Sergey Levine. REBOOT: reuse data for bootstrapping efficient real-world dexterous manipulation. *arXiv preprint arXiv:2309.03322*, 2024.

[15] Tobias Johannink, Shikhar Bahl, Ashvin Nair, Jianlan Luo, Avinash Kumar, Matthias Loskyll, Juan Aparicio Ojea, Eugen Solowjow, and Sergey Levine. Residual reinforcement learning for robot control. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 6023–6029, 2019. doi: 10.1109/ICRA.2019.8794127.

[16] Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan Foster, Grace Lam, Pannag Sanketi, Quan Vuong, Thomas Kollar, Benjamin Burchfiel, Russ Tedrake, Dorsa Sadigh, Sergey Levine, Percy Liang, and Chelsea Finn. Openvla: An open-source vision-language-action model, 2024. URL https://arxiv.org/abs/2406.09246.

[17] Sergey Levine and Pieter Abbeel. Learning neural network policies with guided policy search under unknown dynamics. In Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 27. Curran Associates, Inc., 2014. URL https://proceedings.neurips.cc/paper_files/paper/2014/file/6766aa2750c19aad2fa1b32f36ed4aee-Paper.pdf.

[18] Sergey Levine, Chelsea Finn, Trevor Darrell, and Pieter Abbeel. End-to-end training of deep visuomotor policies. *J. Mach. Learn. Res.*, 17:39:1–39:40, 2016. URL http://jmlr.org/papers/v17/15-522.html.

[19] Songming Liu, Lingxuan Wu, Bangguo Li, Hengkai Tan, Huayu Chen, Zhengyi Wang, Ke Xu, Hang Su, and Jun Zhu. Rdt-1b: a diffusion foundation model for bimanual manipulation, 2024. URL https://arxiv.org/abs/2410.07864.

[20] Jianlan Luo, Eugen Solowjow, Chengtao Wen, Juan Aparicio Ojea, and Alice 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)*, pages 2062–2069. IEEE, 2018.

[21] Jianlan Luo, Eugen Solowjow, Chengtao Wen, Juan Aparicio Ojea, Alice M Agogino, Aviv Tamar, and Pieter Abbeel. Reinforcement learning on variable impedance controller for high-precision robotic assembly. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 3080–3087. IEEE, 2019.

[22] Jianlan Luo, Oleg Sushkov, Rugile Pevceviciute, Wenzhao Lian, Chang Su, Mel Vecerik, Ning Ye, Stefan Schaal, and Jonathan Scholz. Robust multi-modal policies for industrial assembly via reinforcement learning and demonstrations: A large-scale study. In *Proceedings of Robotics: Science and Systems*, Virtual, July 2021. doi: 10.15607/RSS.2021.XVII.088.

[23] Jianlan Luo, Perry Dong, Yuexiang Zhai, Yi Ma, and Sergey Levine. RLIF: Interactive imitation learning as reinforcement learning. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=oLLZhbBSOU.

[24] Jianlan Luo, Zheyuan Hu, Charles Xu, You Liang Tan, Jacob Berg, Archit Sharma, Stefan Schaal, Chelsea Finn, Abhishek Gupta, and Sergey Levine. Serl: A software suite for sample-efficient robotic reinforcement learning. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 16961–16969, 2024. doi: 10.1109/ICRA57147.2024.10610040.

[25] Jianlan Luo, Charles Xu, Fangchen Liu, Liam Tan, Zipeng Lin, Jeffrey Wu, Pieter Abbeel, and Sergey Levine. Fmb: a functional manipulation benchmark for generalizable robotic learning. *arXiv preprint arXiv:2401.08553*, 2024.

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

[27] Mitsuhiko Nakamoto, Yuexiang Zhai, Anikait Singh, Max Sobol Mark, Yi Ma, Chelsea Finn, Aviral Kumar, and Sergey Levine. Cal-ql: Calibrated offline rl pretraining for efficient online fine-tuning, 2024. URL https://arxiv.org/abs/2303.05479.

[28] Emilio Parisotto, Jimmy Lei Ba, and Ruslan Salakhutdinov. Actor-mimic: Deep multitask and transfer learning for robotic control. *arXiv preprint arXiv:1511.06342*, 2015.

[29] Aravind Rajeswaran, Vikash Kumar, Abhishek Gupta, John Schulman, Emanuel Todorov, and Sergey Levine. Learning complex dexterous manipulation with deep reinforcement learning and demonstrations. *CoRR*, abs/1709.10087, 2017. URL http://arxiv.org/abs/1709.10087.

[30] Aravind Rajeswaran, Vikash Kumar, Abhishek Gupta, Giulia Vezzani, John Schulman, Emanuel Todorov, and Sergey Levine. Learning complex dexterous manipulation with deep reinforcement learning and demonstrations. In *Proceedings of Robotics: Science and Systems*, Pittsburgh, Pennsylvania, June 2018. doi: 10.15607/RSS.2018.XIV.049.

[31] Andrei A Rusu, Sergio Gomez Colmenarejo, Caglar Gulcehre, Guillaume Desjardins, James Kirkpatrick, Razvan Pascanu, Volodymyr Mnih, Koray Kavukcuoglu, and Raia Hadsell. Policy distillation. *arXiv preprint arXiv:1511.06295*, 2015.

[32] Andrei A Rusu, Neil C Rabinowitz, Guillaume Desjardins, Hubert Soyer, James Kirkpatrick, Koray Kavukcuoglu, Razvan Pascanu, and Raia Hadsell. Progressive neural networks. In *Advances in Neural Infor-*

mation Processing Systems, 2016.

[33] Gerrit Schoettler, Ashvin Nair, Jianlan Luo, Shikhar Bahl, Juan Aparicio Ojea, Eugen Solowjow, and Sergey 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)*, pages 5548–5555, 2020. doi: 10.1109/IROS45743.2020.9341714.

[34] Jonathan Schwarz, Jelena Luketina, Wojciech M Czarnecki, Agnieszka Grabska-Barwinska, Yee Whye Teh, Razvan Pascanu, and Raia Hadsell. Progress & compress: A scalable framework for continual learning. In *International Conference on Machine Learning*, pages 4528–4537. PMLR, 2018.

[35] Octo Model Team, Dibya Ghosh, Homer Walke, Karl Pertsch, Kevin Black, Oier Mees, Sudeep Dasari, Joey Hejna, Tobias Kreiman, Charles Xu, Jianlan Luo, You Liang Tan, Lawrence Yunliang Chen, Pannag Sanketi, Quan Vuong, Ted Xiao, Dorsa Sadigh, Chelsea Finn, and Sergey Levine. Octo: An open-source generalist robot policy, 2024. URL https://arxiv.org/abs/2405.12213.

[36] Yee Whye Teh, Victor Bapst, Wojciech M Czarnecki, John Quan, James Kirkpatrick, Raia Hadsell, Nicolas Heess, and Razvan Pascanu. Distral: Robust multitask reinforcement learning. *arXiv preprint arXiv:1707.04175*, 2017.

[37] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. Llama 2: Open foundation and fine-tuned chat models, 2023. URL https://arxiv.org/abs/2307.09288.

[38] Junjie Wen, Yichen Zhu, Jinming Li, Minjie Zhu, Kun Wu, Zhiyuan Xu, Ning Liu, Ran Cheng, Chaomin Shen, Yaxin Peng, Feifei Feng, and Jian Tang. Tinyvla: Towards fast, data-efficient vision-language-action models for robotic manipulation, 2024. URL https://arxiv.org/abs/2409.12514.

[39] Tony Z. Zhao, Jianlan Luo, Oleg Sushkov, Rugile Pevceviciute, Nicolas Heess, Jon Scholz, Stefan Schaal, and Sergey Levine. Offline meta-reinforcement learning for industrial insertion. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 6386–6393, 2022. doi: 10.1109/ICRA46639.2022.9812312.

APPENDIX

A. Task Details

a) Connector Insertion.: The robot starts with the male connector pre-grasped in a $10cm \times 10cm$ plane 5 cm above the female port. We use the same D-type fixture for the female portion which is fixed to the table at a consistent pose. The task is completed if the robot aligns and inserts the connector into the socket. We train the policies on `USB`, `Ethernet`, and `VGA` connectors, while using `Type-C`, `HDMI`, `Display Port`, and `3-pin XLR` connectors to evaluate generalization.

b) Pick and Place.: In this task, the robot starts $15cm$ above the table at a fixed pose. The target object is randomly placed within a $18cm \times 18cm$ area on the table and the bowl is at a fixed pose $5cm$ from the edge of the object randomization region. The robot is to pick up the object and put it into the bowl. To test generalization, we evaluate on an out-of-distribution scenario by replacing the green pepper training object with a yellow corn and changing the tabletop to a beige wood grain surface as shown in Fig. 3.

c) FMB Insertion.: The robot starts with the insertion object pre-grasped 15 cm above the assembly board, which is randomly placed within a $35cm \times 35cm$ area with $\pm 15^{\circ}$ rotation. The task is successfully completed if the object is completely inserted into the board. The insertion tolerance in this task is $\pm 1.5mm$. We used the same set of board poses during each evaluation experiment to ensure consistency across rus.

d) FMB Single Object Assembly.: In this task, the assembly board is randomized the same way as in the `FMB Insertion`, but the object is randomly placed in a $3cm \times 7cm$ grasping area approximately $20cm$ from the insertion area. The robot starts $15cm$ above the object at a fixed position. We also use consistent object and board poses during evaluation.

B. Training Procedures

For `FMB Insertion`, `Connector Insertion`, and `Pick and Place`, we first collect positive and negative samples using our SpaceMouse teleoperation device to train a binary success classifier as the reward function for RL. Then, we initialize the RL buffer with 20 demonstrations and train it on the whole task for 1-3 hours until the policy reaches 100% success rate. Next, we roll out the converged RL policy to collect data for generalist policy fine-tuning, filtering out failed trajectories if any exist. We also collect a set of all successful expert human demonstrations to compare against the RL-generated data.

For `FMB Single Object Assembly`, we use the same reward function as in `FMB Insertion` and we only train RL on the insertion stage by starting the episode with the object pre-grasped and the arm $15cm$ above the board within a $5cm \times 5cm$ randomization region. We periodically adjust the grasp pose during training and data collection to build robustness to variations in the grasp. We also collect the same number of human demonstrations for the insertion stage to compare performance. We then separately collect human demonstrations for the grasping phase, starting the robot above the insertion object and ending the episode when the arm has grasped the object and moved within the insertion randomization region above the board. We combine data from the two stages for fine-tuning.

We fine-tuned the OpenVLA weights pre-trained on OXE dataset using LoRA with a rank of 32 applied to each linear layer of the model, which reduces the computational overhead while not sacrificing much performance. We trained using the default fine-tuning configuration with batch size 2 and 3 gradient accumulation steps on a single Nvidia RTX 4090 GPU, which took between 3 to 5 hours to converge. For Octo, we started from the pre-trained Octo-Base model, using the primary image tokenizer to tokenize the wrist camera image and removed the secondary tokenizer. We also mask out the image goal since we do not use it. We applied full fine-tuning with the default hyperparameters and batch size 64 until convergence, which took 3-5 hours on a single Nvidia RTX 4090 GPU.