Learning Generalizable Manipulation Policy with Adapter-Based Parameter Fine-Tuning

Kai Lu[1], Kim Tien Ly[2], William Hebberd[2], Kaichen Zhou[1], Ioannis Havoutis[2], Andrew Markham[1]

***Abstract*— This study investigates the use of adapters in reinforcement learning for robotic skill generalization across multiple robots and tasks. Traditional methods are typically reliant on robot-specific retraining and face challenges such as efficiency and adaptability, particularly when scaling to robots with varying kinematics. We propose an alternative approach where a disembodied (virtual) hand manipulator learns a task (i.e., an abstract skill) and then transfers it to various robots with different kinematic constraints without retraining the entire model (i.e., the concrete, physical implementation of the skill). Whilst adapters are commonly used in other domains with strong supervision available, we show how weaker feedback from robotic control can be used to optimize task execution by preserving the abstract skill dynamics whilst adapting to new robotic domains. We demonstrate the effectiveness of our method with experiments conducted in the SAPIEN ManiSkill environment, showing improvements in generalization and task success rates. All code, data, and additional videos are at this GitHub link: https://kl-research.github.io/genrob.**

I. Introduction

Learning generalizable robotic skills is a significant challenge in embodied intelligence, which includes generalization across objects, tasks, and robots. Compared to the widely explored vision-based object generalization [1], generalization across different robots and different task trajectories remains underexplored. This capability has a wide impact, enabling robots to efficiently learn new skills or adapt existing skills to similar domains. However, different robots usually have varying kinematic configurations and morphologies, such as body structure and joint limits, leading to different physical constraints and dynamic properties, posing challenges to skill generalization.

Traditional skill learning methods often consider retraining for new tasks on a specific single robot, such as by using reinforcement learning (RL) [2]–[4] or imitation learning [5]. However, reinforcement learning often encounters problems with sampling efficiency, and imitation learning struggles to find perfectly corresponding robot samples. Recent works designed skill generalization methods across multiple robots and tasks using large robot learning datasets such as Open X-embodiment [6] and foundation models [7], [8], which requires significant computational resources. Another approach is the use of hierarchical or modular network designs [9]–[13], including aligning internal features [9], encoding robotic morphological information [10], and sharing modular policies [11], [12]. However, they often focus on simplified robotic morphologies, such as 2D arms [11] or omnidirectional spherical hands [13], or limited tasks like grasping [10]. In this paper, we aim to explore how a shared global skill policy can effectively be applied on multiple high-degree-of-freedom (high-DoF) mobile robot platforms.

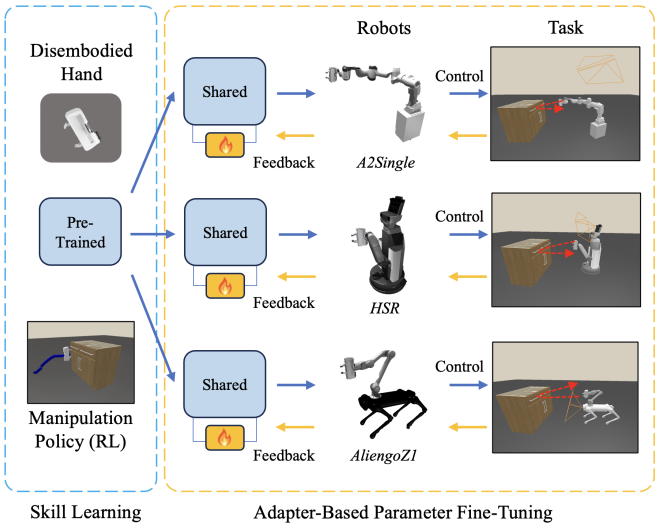


Fig. 1. **Demonstration of the proposed method.** In this work, we study the problem of using adapter-based fine-tuning on pre-trained policy models for generalizable manipulation across different robotic platforms. We teach a disembodied hand to learn tasks like opening a drawer and transfer these skills to a whole-body robot, accounting for the robot's specific constraints.

Our key innovation is to teach an unconstrained, disembodied hand manipulator to learn a skill, such as opening a drawer or cabinet, and then transfer this skill to a constrained whole-body robot. We consider these constraints as the feasibility of the disembodied hand's trajectory on a specific robot. As the disembodied hand policy is not optimized for the specific robot's constraints and kinematic properties, the trajectories generated by this policy are not optimal, or even unfeasible, on new whole-body robots. For instance, a robot with more DoFs and longer arms will have greater adaptability, while smaller robots may find it more difficult to follow the poses generated online by RL. To mitigate this issue, we consider fine-tuning the existing policy network and introducing robotic control feedback to optimize the RL policy.

Recently, parameter-efficient fine-tuning (PEFT), such as the Adapter technique [14]–[17], has proven effective for pre-trained model fine-tuning and is widely used for generalization in natural language processing [18] and visual generation fields [14]. It achieves specific domain adaptation by inserting additional trainable layers into the existing model. This method allows the model to adapt to new domains with a small number of extra parameters while keeping

[1]: K. Lu, K. Zhou and A. Markham are with the Department of Computer Science, University of Oxford. Email: {kai.lu, rui.zhou,andrew.markham}@cs.ox.ac.uk

[2]: K. T. Ly, W. Hebberd and I. Havoutis are with the Oxford Robotics Institute, University of Oxford, Oxford, UK. Email: {ktien, william.hebberd,ioannis}@robots.ox.ac.uk

the original parameters unchanged, which is particularly useful in transfer learning and multi-task learning. Inspired by this, we propose the integration of adapters into the learning of generalizable robotic skills and explore the use of robot feedback in RL to learn these adapters. This approach enables the adaptation to new robots or tasks without the necessity of retraining the entire model. We conjecture that this method can better retain the original model's knowledge of skill dynamics while introducing an understanding of new robots or tasks.

To implement this, we introduce parallel modules into the original network, such as low-rank decomposed (LoRA) [19] adapters, or sequential modules, such as low-dimensional encoder-decoder structure adapters with residual design [20]. Then, in RL training, we design a feedback reward function from the whole-body robotic control, which requires the robot to solve joint configurations using a Newton-Raphson-based Inverse Kinematics (IK) solver [21] at each step. If an unfeasible solution is returned by the IK solver before the robot completes the task, we will assign a negative reward to RL policy. Hence, the adapter learns to optimize the end-effector (EE) trajectory generated by the original RL model. In addition to generalization across robots, we further explore the application of adapter learning techniques between similar tasks. For example, how skills like pulling a drawer can be adapted and transferred to opening a door. We found that this learning method is effective for transferring robotic skill learning.

In summary, in this paper, we explore the following three questions:

- How can adapters be used in robotic reinforcement learning to generalize a global skill across different robotic embodiments?
- What impacts do various adapter architectures and fine-tuning strategies have on skill generalization?
- Does adapter technology have broader application scenarios, such as domain adaptation for similar tasks?

We studied the generalization of the drawer-opening skill across three mobile manipulation robots in the SAPIEN ManiSkill environment [22], including the ManiSkill A2-Single-Arm Robot (A2Single), the Unitree Aliengo robot [23] with Z1 arm [24] (AliengoZ1), and the Toyota Human Support Robot (HSR) [25]. Our research show that adapter learning can effectively generalize among robots with different physical constraints. Particularly, LoRA-type adapters improved the success rate on new robots by 11% on A2Single Arm, 15% on AliengoZ1, and 14% on HSR, compared to vanilla full-finetuning. Our task-variant experiments, including door opening and chair pushing tasks, indicate that adapter learning also improves generalization among tasks.

II. Related Work

A. Learning across robotic embodiments

Given the diverse landscape of robotics, mastering learning across various robotic platforms emerges as a fundamental topic within the field. Based on this concern, a considerable body of work is dedicated to exploring foundational models [7], or delving into the potential of large language models (LLMs) [26]. Additionally, others harness extensive

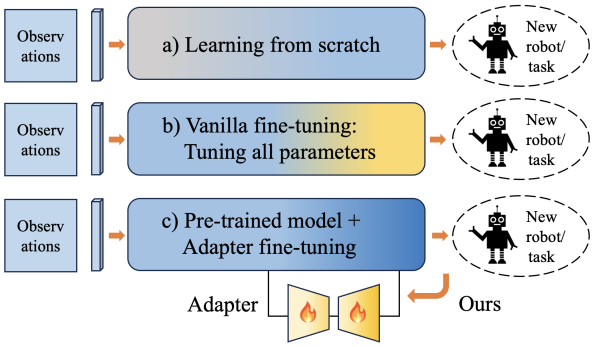


Fig. 2. **Illustration of different approaches.** Commonly seen approaches for new robot/task transfer include learning from scratch and vanilla fine-tuning. In this work, we adopt adapter learning with feedback reward in RL for skill generalization from a pre-trained policy model.

datasets featuring a multitude of robots and demonstrations for comprehensive training approaches, highlighted in the study by the Open X-embodiment [27]. Concurrently, a distinct trajectory in research adopts a structured approach to manipulation policy learning, embracing the hierarchical and modular constructs to bridge the embodiment disparities across varied robotic entities, a notable example being the work by [9]. Nevertheless, these studies often demarcate their focus to either task simplification or specific transfer facets such as 2D kinematic setups [11], end-effector design variants [13], or adaptive visual perception strategies [28]. While there are relevant contributions in adjacent fields such as robotic navigation and locomotion, exemplified by [29] and [30] respectively, these studies tangentially relate to our concentrated objective on robotic manipulation.

B. Generalizable manipulation policy learning

Learning generalizable manipulation skills is a crucial topic in embodied AI research. Numerous works in the visual domain have been proposed to address generalization among objects, including domain-invariant 3D feature distillation [31], 3D affordance learning [32], unified representations of actionable parts [33], and enhancement of both policy and visual modules through interactive perception [34], among others. Additionally, in the domain of policy learning, various works in visual reinforcement learning and imitation learning have been proposed to solve generalization across objects or tasks. For instance, methods that use decoupling or EE action spaces [35], [36] or action primitives [37], [38] have been put forward to increase the efficiency of reinforcement learning and enhance generalizability. On the other hand, modular structures, representational alignment [9], or adversarial generative models [39] have proven effective in generalizing across different objects and tasks. With the advent of large models, recent work has involved integrating Vision-and-Language Models (VLMs) [40] for action and semantic matching, resulting in policies with improved generalizability.

C. Adapter Learning for Policy Model Fine-Tuning

PEFT, such as the adapter technique, is extensively utilized for model domain adaptation. It was first applied in the field of natural language processing; an example being the use of Low-Rank Adaptation (LoRA) [19] for fine-tuning GPT-3 [41], which involves inserting trainable rank decomposition

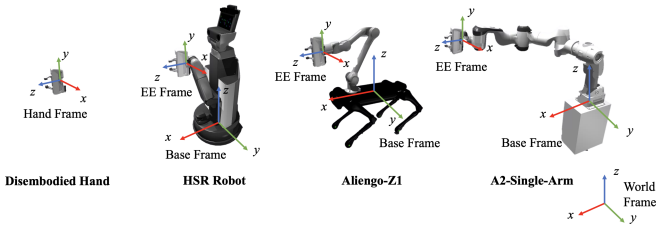


Fig. 3. **Coordinate system of robots and the disembodied hand**. We assess three different mobile manipulators equipped with the same end-effector but featuring varying kinematic configurations.

matrices into each layer. Moreover, adapters are widely adopted in the vision domain [14], [42]–[44]. For instance, ViT-Adapter [42] can achieve performance comparable to that of vision-specific transformers. Adapters are also used in robotic imitation learning, as demonstrated by the work of Liang et al (LoRA-Transformer) [45]. A notable example is TAIL [46], which fine-tunes large, task-specific models. This work is similar to pure vision fields, adapters are used to adjust upstream visual perception modules for manipulation tasks, such as RoboAdapters [47]. Distinct from these applications, our work contemplates the use of adapter technology through RL for generalization across different robotic platforms (as shown in Fig. 2), and we further explore its adaptability across tasks.

III. Methodology

Our methodology focuses on transferring a learned skill from an original robot, e.g., a disembodied hand, to other different robotic platforms, e.g., whole-body mobile manipulators, through the integration of adapters. The adapters serve a dual purpose: fine-tune the model to fit the specific kinematic constraints of a new robot whilst maintaining the integrity of the original learned skill.

A. Problem Statement

Robotic manipulation policy learning can be formulated as a Markov decision process (MDP) [48], which is represented as (S,A,R,T,γ), where S is the set of states, A is the set of actions, $R(s_t,a_t,s_{t+1})$ is the reward function, $T(s_{t+1}|s_t,a_t)$ is the transition function as a probability distribution, and γ is the discount factor for the future rewards. The agent policy $\pi(a|s)$ is the action selecting probability under a given state s. The goal of RL is to maximize the return under the policy $G_\pi = \mathbb{E}_\pi[\sum_t \gamma^t R(s_t,a_t,s_{t+1})]$. In robot learning tasks, we usually need to estimate the task-relevant states from observation O, regarded as $s = f(o)$. This setting is viewed as a partially observable Markov decision process (POMDP) [49] where the policy is $\pi(a|f(o))$.

In this work, we study the problem of generalizing skills across robotic mobile manipulators, where the state space being $s = [s_{obj}, s_{rob}]$ and the action space being $a = [v_{ee}, q_{jaw}]$. We use a shared action space across robots and equip them with the same two-parallel-jaw hand, as shown in Fig. 3.

B. Reinforcement Learning with Disembodied Hand

We first exploit the disembodied hand as the RL agent to learn the abstract skill dynamics, whose DoFs are three virtual prismatic joints and three virtual revolute joints. The action space includes six desired velocities of the virtual

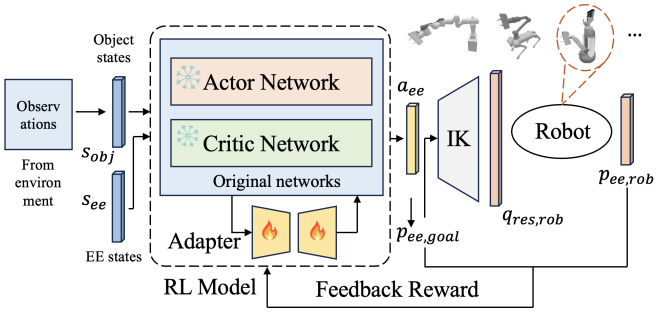


Fig. 4. **Pipeline of our method.** We integrate the adapter module into the RL model and introduce a feedback reward function from the whole-body robotic control. Through this way, the adapter learns to optimize the EE trajectory generated by the original RL model for robot-level generalization.

joints $v_{ee} \in \mathbb{R}^6$ and two desired positions of the finger joints $q_{f,d} \in \mathbb{R}^2$. For the cabinet environment in ManiSkill, we use $s_{obj} = [s_{cab}, s_{link}, s_{hdl}, s_{size}]$, where s_{cab} is the base link pose of the loaded cabinet, s_{link} and s_{hdl} are the current poses and the full poses (i.e., the poses when the drawer is fully opened) of the target drawer link and the handle, and s_{size} is the full length and the opening length of the target drawer. The poses are all represented as world frame coordinates and quaternions. In RL training, $s_{rob} = s_{ee}$ includes the hand joints' positions and velocities. We follow the dense reward function designed in ManiSkill [22] to train the RL model:

$$R_{ms} = \begin{cases} R_{stg} + R_{ee}, \; d > d_{ths}, \\ R_{stg} + R_{ee} + R_{link}, \; d < d_{ths}, \; c < c_{open}, \\ R_{stg} + R_{ee} + R_{link} + R_{stc}, \; d < d_{ths}, \; c > c_{open}, \end{cases} \tag{1}$$

where R_{stg} increases from the first stage to the final and the stage is defined by the distance between EE and the handle of the target drawer $d \in \mathbb{R}$ and the opening extent of the target drawer $c_{op} \in [0,1]$. More detailed definitions and coefficients can be found in ManiSkill. The model is trained using soft actor-critic (SAC) [50] algorithm. We view this process as the model pre-training in this work and focus on how to fine-tune the learned policy. The policy is represented as:

$$a_{ee} = \pi_\phi(s_{obj}, s_{ee}). \tag{2}$$

C. Robotic Control Feedback Reward

Given that the RL agent is a disembodied free-floating hand learning the dynamics of skills without considering the constraints of any specific embodied robot control, trajectories that are unfeasible for the robot may occur during skill transfer. Since different robots have different kinematic properties, it is difficult to design a unified constraint condition on the disembodied hand agent without leading to highly limited or sub-optimal trajectories. Therefore, we choose to introduce adapter techniques to fine-tune the original model, which requires us to design a feedback loop to optimize the pre-trained policy network, as shown in Fig. 4.

We first parallel the environment of the disembodied hand with that of the whole-body robot. The action $a_{ee,t}$ executed in the hand environment yields the next pose x_{t+1}. We aim to synchronize the robot's EE to the pose of the disembodied hand, which is expressed as:

$$x_{t+1} = p_{t+1} = p(q_t + \Delta q), \tag{3}$$

where $p(q)$ denotes the forward kinematics equation, and q represents the robot joint position. This desired joint position can be obtained by computing the Jacobian matrix, where:

$$p_{t+1} = p(q_{t+1}) \approx p(q_t) + J(q_t)\Delta q \tag{4}$$

therefore, the approximated value is:

$$\Delta q \approx J^{+}(q_t)\Delta p, \tag{5}$$

where J^{+} is the pseudoinverse of the Jacobian matrix. We use an IK solver based on the Newton-Raphson method to iterate and find a numerical solution Δq_{res} s.t.:

$$p(q_t + \Delta q_{res}) - x_{t+1} < k_{errtol} \tag{6}$$

For robots with high DoFs (considered to be more than 6 here), it is generally not easy to fall into a situation without any IK solutions. However, a global-searching IK solver cannot guarantee a smooth solution between the two EE positions of consecutive time steps. Therefore, we added a restriction to prevent the robot's current configuration from deviating more than k_{maxdev} along each axis. With such a setting, the IK solver iteratively operates through the Jacobian inverse technique. We use the IK-solve-nearby function from the Klampt library [21] to achieve the above setting, represented as:

$$q_{res}, z_{res} = F_{ik}(q, \Delta p, k_{maxdev}, k_{errtol}, k_{maxiter}), \tag{7}$$

where $z_{res} = 1$ when the IK has a feasible solution otherwise $z_{res} = 0$. q_{res} is the solution joint configuration.

For each RL simulation step, we perform the IK calculation for the robot and then control the robot to the solution joint positions before the next RL prediction. Thus, the EE poses that the robot cannot reach through the above IK setting are considered non-compliant with the robot's kinematic constraints, and we use a reward function:

$$R_{ik} = \begin{cases} +1, \ z_{res} = 1, \\ -1, \ z_{res} = 0. \end{cases} \tag{8}$$

This reward varies for different robots, as their kinematic parameters differ, such as joint limits and the structure and number of joints and links. Therefore, the final reward during the fine-tuning process can be expressed as:

$$R = \omega_{ms} R_{ms} + \omega_{ik} R_{ik}. \tag{9}$$

D. Adapter Modules for Parameter Fine-Tuning

When considering fine-tuning the policy network to a specific robot domain, the vanilla approach involves tuning all of the parameters of the original network. However, this method might overfit to the new domain, reducing the model's capability. Adapter technology offers an alternative by introducing a small number of trainable parameters to adjust the original network. A key benefit of this approach is that transferring skills among different robots only requires attaching the corresponding adapter network. This process typically leaves the original network structure and inference speed unaffected. Furthermore, the parameters of the pre-trained policy network can be shared, streamlining the integration process. Common adapter structures include parallel adapters, such as LoRA, and sequential adapters, such as the residual encoder-decoder, which can improve the model's generalizability.

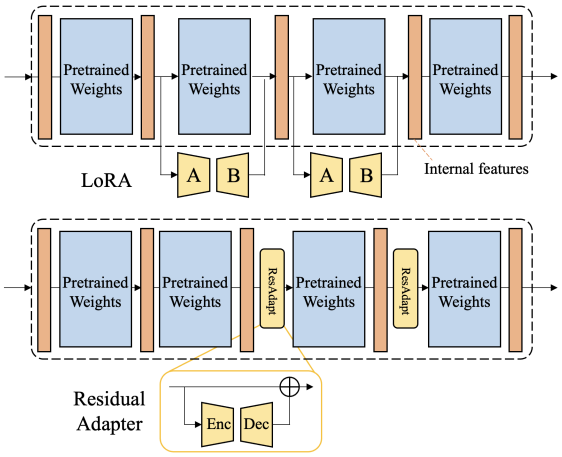


Fig. 5. **Two types of adapters incorporated in this work.** In our implementation, we incorporate the adapters in both the policy (actor) network and the Q-function network in the RL model.

The principle of LoRA is to parallel two low-dimensional matrices in the network's linear layers, represented as A and B, with B initialized to zero. LoRA hypothesizes that the rank of the parameters that need to be adjusted in the original linear layer matrix is likely not high, e.g., 1/10 of the original matrix, so the adjustment can be achieved through learning A and B. Another commonly seen adapter pattern is the encoder-decoder with non-linear activation functions using residual structure (ResAdapter) to connect between original layers. The scale parameter of the residual connection points is initialized to zero. Both structures are widely used to adjust networks based on MLP or Transformer architectures.

In this work, we integrate LoRA or ResAdapter into the linear layers of our MLP, as shown in Fig. 5, and then we use the above reward to finetune the networks. However, we point out that the precise choice of the adapter is not critical to our framework and that our method could relatively easily incorporate other adapters. Since we are using the SAC algorithm, we need to update the parameters of both the actor and critic networks:

$$\begin{aligned} \pi_{\phi_{nn}, \phi_{adapter}} &: \phi_{adapter} \leftarrow \phi'_{adapter}, \\ Q_{\theta_{nn}, \theta_{adapter}} &: \theta_{adapter} \leftarrow \theta'_{adapter}, \end{aligned} \tag{10}$$

where θ represents the critic (Q-function) parameters, and ϕ represents the actor (policy) parameters.

E. Adjusted Adapter and Optimization Techniques for Task-Specific Constraints

In addition to skill generalization across different robots, we further explore more challenging scenarios, namely applying the adapter technique to generalization across tasks. Different tasks typically have distinct trajectories, for instance, as shown in Fig. 6, where pulling a drawer and opening a door are characterized by circular and linear motions respectively, while pushing a chair is usually in the opposite direction to pulling a drawer. Due to significant changes in skill dynamics, we designed a more complex adapter module based on ResAdapter (as shown in Fig. 7) to achieve this adaptation, which can take new inputs and optimize outputs.

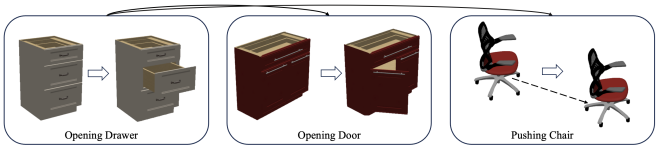


Fig. 6. **Generalization to various tasks.** We demonstrate the adapter technique for generalizing a skill across other manipulation tasks.

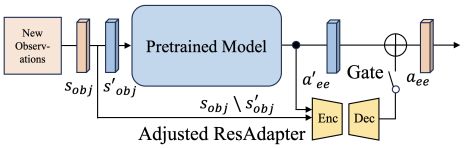


Fig. 7. **Adjusted adapter structure for task-level generalization.** We adjust the structure of ResAdapter to tune the pre-trained model for more challenging scenarios such as new tasks.

To accommodate new tasks, we first align the input states s_{obj} with the corresponding task. For the door opening task, we align the pose of the door handle and door link with the drawer, retaining the original position but leaving the rotation for the Adapter, represented as $\{s_{obj} \setminus s'_{obj}\}$. Additionally, we equate the door's arc length to the drawer's linear length, leaving the door's radius to the Adapter. For the pushing chair task, we map the main body link pose of the chair to the handle and link pose of the drawer but reverse the xy-plane coordinates.

The adjusted residual adapter uses a gate control for the residual connection. The gating signal $d < d_{ths}$ is the distance between the hand and the operational part e.g., the backrest or armrest of the chair. As the new skill dynamics change largely compared to the original skill, in this scenario, we allow the parameters of the original network to be fine-tuned to achieve adaptation to the new states.

IV. Experimental Results

A. Experimental Setups

1) Environments and tasks: We conducted experiments in the SAPIEN ManiSkill simulation environment, choosing the task of opening cabinet drawers as the basic task. The criterion for task success is opening the target joint to $\geq 90\%$ of its extent and that the EE poses are feasible for the whole-body robots. For the drawer-opening task, ManiSkill provides 25 cabinets with different geometries and topologies, of which we randomly select 15 as the training set and 10 as the test set. To evaluate the skill transfer to different robots, the success criteria included the feasibility of the EE poses in the trajectory, which is validated by the Newton-Raphson-based IK solver described in Sec. III-C. We also conducted experiments for task generalization, considering the adaptation of the skill of opening drawers to opening doors and pushing chairs. The success criterion for the door-opening task is to open the specified joint to $\geq \pi/4$ radian. For the pushing chair task, the criterion is that the chair is close to the target position within 0.15 m and remains standing upright. These tasks are also episodic, with a maximum length of 200 steps (each step is 1/20 s).

2) Robotic mobile manipulators: In simulation, we present three mobile manipulation robots, as in Fig. 1:

- **Disembodied Hand**: Modeled as a floating hand with 6-DoF and equipped with two parallel jaws, using the Panda Hand as the collision model.
- **Robot A - A2Single**: Modeled as an 11-DoF robot with a 4-DoF mobile base allowing for x, y, z translations and yaw rotation. It features a Scirus body and a 7-DoF Franka Panda arm.
- **Robot B - HSR**: Modeled as an 8-DoF robot with a 3-DoF mobile base for xy translation and yaw rotation, and it is equipped with a 5-DoF arm.
- **Robot C - AliengoZ1**: Modeled as a 9-DoF robot with a 3-DoF mobile base for xy translation and yaw rotation. Here, we follow the work of Habitat [51], assuming that the base has 3 DoFs of command and ignoring the low-level leg movements. In addition, the robot is fitted with a 6-DoF Z1 robotic arm.

In our real-world experiments, we employ the HSR robot for sim-to-real validation. The outcome and the supplementary videos are available on our project website.

3) Comparison Methods: We compare different methods of generalization to robots: **a) Direct-Transfer**: directly using the model of the trained disembodied hand agent, **b) Full-FineTune**: tuning all the parameters of the original model, **c) ResAdapter**: using the residual adapter with the original model parameters frozen, and **d) LoRA**: using the LoRA adapter with the original model parameters frozen.

B. Effectiveness of adapter learning for multi-robot transfer

Table I shows the average task success rates, episode length in steps, and episode rewards. Comparing Direct-Transfer and LoRA Adapter, we note that introducing robot inverse kinematics control feedback reward can improve the success rate of tasks and accelerate task completion time. This indicates that RL-generated trajectories are optimized such that the introduction of feedback from the specific robot's kinematic constraints improves the success. Meanwhile, by comparing Direct-Transfer and Full-Finetune, we found that if the new scenario differs from the original training scenario, directly fine-tuning a pre-trained model may lead to overfitting. The pre-trained model might already be optimized for the specific characteristics of its original robot, and fine-tuning it on a different robot could cause the model to overly adapt to the new scenario, thus leading to poor generalization on unseen data. Comparing the fine-tuned model performance on different robots, we find that the adapter learning method is more effective for HSR and AliengoZ1 than for A2Single. This might indicate that they have more valid samples, i.e., infeasible poses generate corresponding feedback. Since reward in reinforcement learning is a weak supervisory signal, we consider introducing stronger signals such as auxiliary loss in the future to improve tuning efficiency. Our further experiment on LoRA shown in Fig. 11 illustrates that after applying LoRA on an embodied robot, there is an improvement in the original domain, and it can also be transferred to other robots, demonstrating the effectiveness of the adapter. Meanwhile, the effect of LoRA on the original robot is higher than on other robots, indicating the necessity of applying adapters for each individual robot.

C. Impact of various adapters for skill generalization

Comparing the three methods of fine-tuning, we find that fine-tuning the entire original model based on constraints

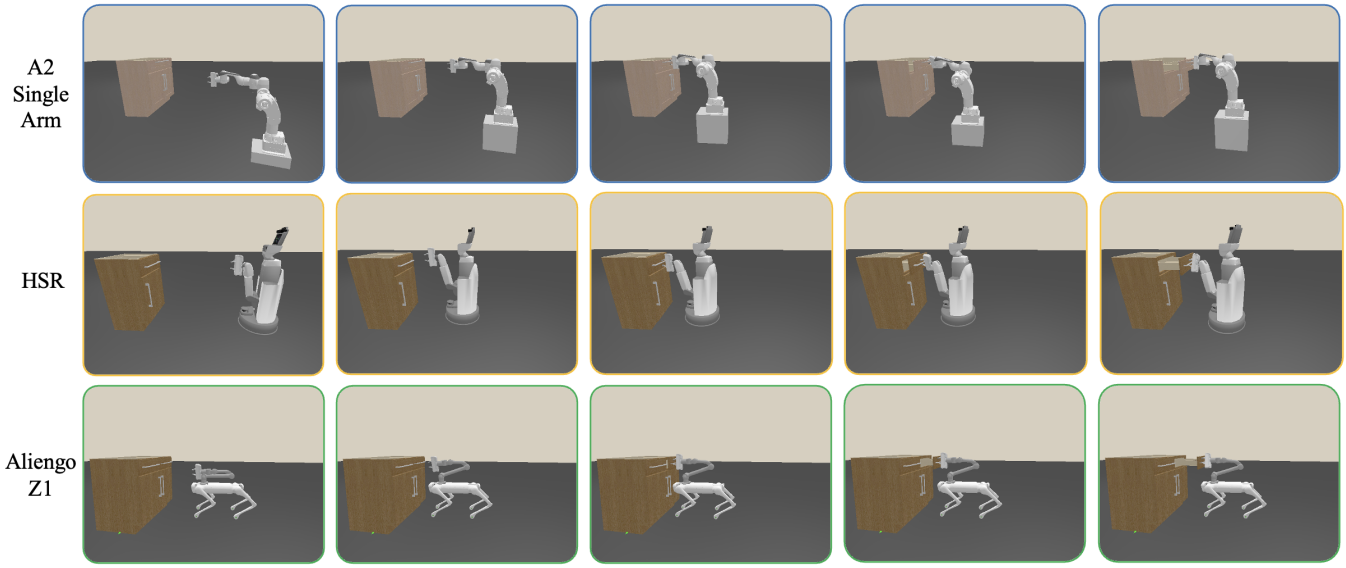


Fig. 8. **Visualization of generalizing a skill across various robotic platforms.**

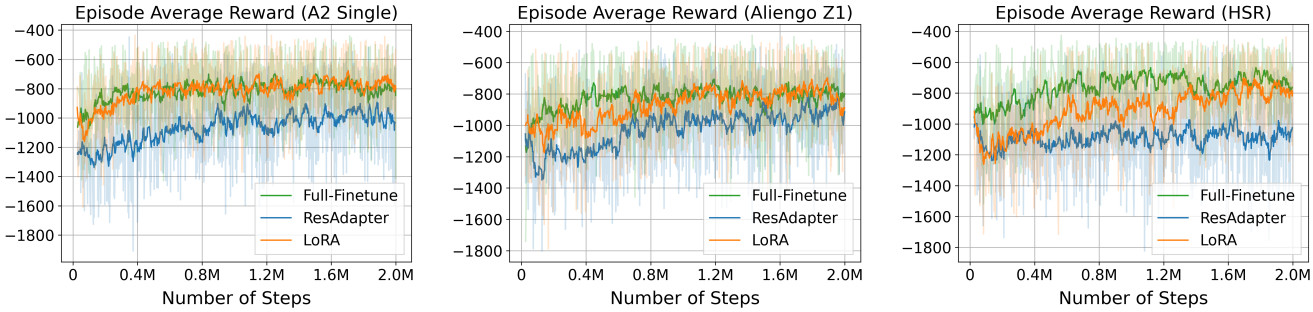


Fig. 9. **Training curve of transferring skill to three different high-DoF robots: A2 Single, Aliengo Z1 and HSR.**

TABLE I
CROSS-ROBOT GENERALIZATION

Robot	Method	Success ↑	Length ↓	Reward ↓
Hand	SAC	68%	115.85	-1084.91
A2Single	Direct-Transfer	32%	155.64	-1500.25
	Full-Finetune	25%	168.19	-1563.99
	LoRA Adapter	**36%**$^{\uparrow 11\%}$	**150.61**	**-1452.61**
AliengoZ1	Direct-Transfer	27%	164.98	-1652.33
	Full-Finetune	22%	169.26	-1702.40
	LoRA Adapter	**37%**$^{\uparrow 15\%}$	**150.85**	**-1503.93**
HSR	Direct-Transfer	26%	163.20	-1508.26
	Full-Finetune	23%	170.51	-1619.22
	LoRA Adapter	**37%**$^{\uparrow 14\%}$	**148.29**	**-1410.24**

presents a high success reward in training as shown in Fig. 9, but relatively lower performance in the test set. This indicates that although it meets the constraints of the new robot, it leads to forgetting the ability to generalize. ResAdapter presents a better performance in the test set, as shown in Fig. 10, by adjusting the intermediate features of the network and deepening it through concatenation. LoRA overall performs the best, indicating that adjusting the parameters of the MLP linear layer in parallel can allow the network to maintain its original good skill generalization ability while meeting the specific kinematic constraints of the robot. We also compare the two adapters' performance on different robots in Table II, and we visualize the task execution by the three robots in Fig. 8, where their policies are fine-tuned by LoRA.

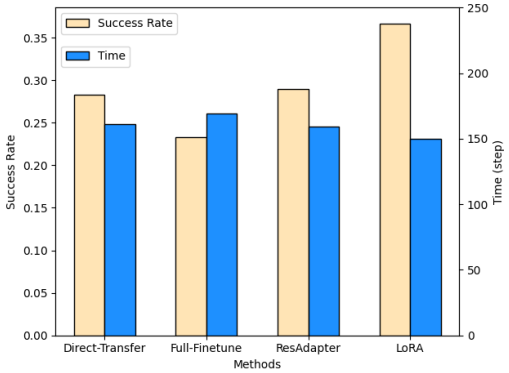
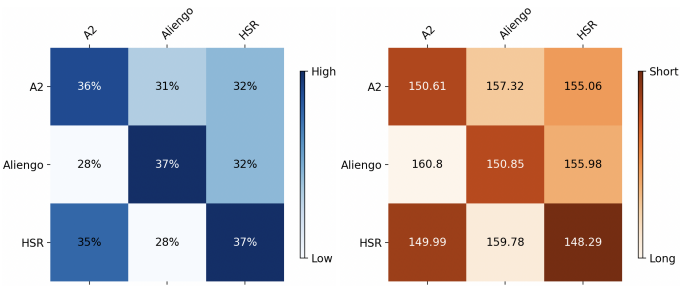


Fig. 10. **Comparison of direct transfer and different tuning methods.** The values are averaged over the three robots.



(a) Task Success Rate (b) Episode length (steps)

Fig. 11. **Cross-robot evaluation.** The adapter is trained on one robot (vertical labels), and tested on the other robot (horizontal labels).

TABLE II

CROSS-ROBOT GENERALIZATION

Robot	Method	Success ↑	Length ↓	Reward ↓
Hand	SAC	68%	115.85	-1084.91
A2Single	ResAdapter	28%	160.18	-1476.89
	LoRA	36%	150.61	-1452.61
AliengoZ1	ResAdapter	27%	162.07	-1558.45
	LoRA	37%	150.85	-1503.93
HSR	ResAdapter	32%	156.35	-1481.21
	LoRA	37%	148.29	-1410.24

TABLE III

CROSS-TASK GENERALIZATION

Robot	Method	Success ↑	Length ↓	Reward ↓
Opening Drawer	SAC	0.68	115.85	-1084.91
Opening Door	Direct-Align	0.	200.00	-2026.08
	Full-Finetune	23%	174.76	-1807.69
	Adapter	**31%**$^{\uparrow 8\%}$	**172.09**	**-1735.80**
Push Chair	Direct-Align	3%	196.61	-3794.65
	Full-Finetune	39%	166.17	-2267.71
	Adapter	**54%**$^{\uparrow 15\%}$	**152.77**	**-1954.63**

D. Adapter-Based Tuning for Cross-Task Generalization

We further explore the generalizability of the adapter technique at the task level, including transfer from opening drawers to opening doors, and from opening drawers to pushing chairs. We consider three methods to generalize the original skill: direct alignment (using the input alignment described in the methodology), fine-tuning the original network, and incorporating adapters for fine-tuning. Fig. 12 shows the training process, where using adapters for input processing and output optimization allows the network to converge faster during training and achieve higher rewards in the door-opening task compared to fine-tuning the original network with new inputs. Table III compares the success rates on the test set, where we find that adapter-based tuning has higher task success rates, indicating that adding adapter modules can enhance the generalizability of the RL policy across tasks. We also notice that transferring the original skill policy to pushing chairs performs better than transferring to doors opening task. This may be because the skill dynamics of the chair are easier to adapt to compared to the originally learned skill dynamics of the network. Fig. 13 shows typical scenarios for the three different tasks, from which we can see that finetuning the pre-trained model can achieve transformations of the skill dynamics.

V. CONCLUSIONS

Our findings indicate that adapter learning is a simple, yet powerful strategy for generalizing robotic skills across different robots and tasks, offering a parameter-efficient alternative to full model retraining. By training first on an entirely virtual disembodied manipulator, the adapter is simply responsible for embodiment-specific adaptation. LoRA-type adapters show improvement in task success rates, supporting their potential for widespread application in robotic learning. Future research may extend these techniques to a broader range of tasks and robots, further enhancing the adaptability and efficiency of robotic systems.

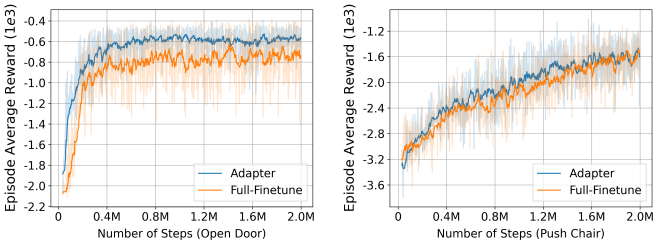


Fig. 12. **Training curves of transferring the opening drawer skill to the opening door task and the pushing chair task.**

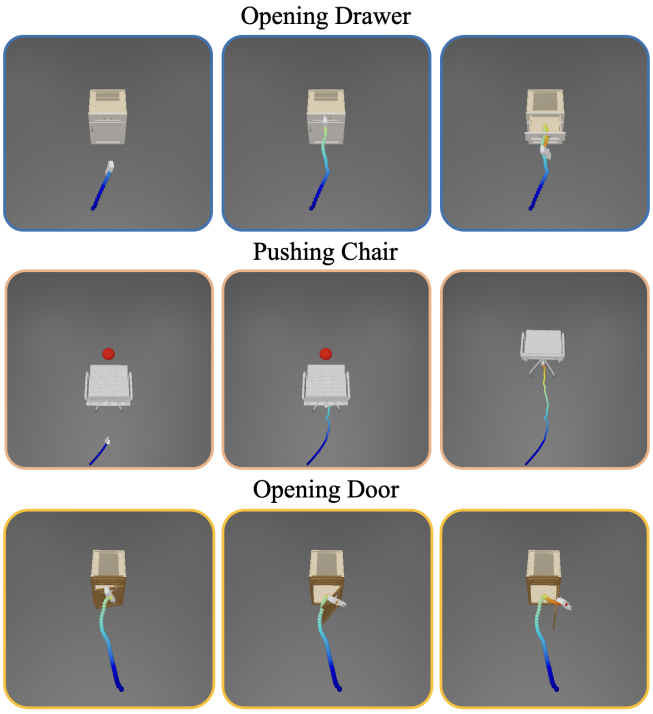


Fig. 13. **Visualization of generalizing a skill across different tasks.**

REFERENCES

[1] Yifeng Zhu, Zhenyu Jiang, Peter Stone, and Yuke Zhu. Learning generalizable manipulation policies with object-centric 3d representations. In *7th Annual Conference on Robot Learning*, 2023.

[2] Vijaykumar Gullapalli, Judy A Franklin, and Hamid Benbrahim. Acquiring robot skills via reinforcement learning. *IEEE Control Systems Magazine*, 14(1), 1994.

[3] Oliver Kroemer, Scott Niekum, and George Konidaris. A review of robot learning for manipulation: Challenges, representations, and algorithms, 2020.

[4] Kaichen Zhou, Shiji Song, Anke Xue, Keyou You, and Hui Wu. Smart train operation algorithms based on expert knowledge and reinforcement learning. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 52(2):716–727, 2020.

[5] Bin Fang, Shidong Jia, Di Guo, Muhua Xu, Shuhuan Wen, and Fuchun Sun. Survey of imitation learning for robotic manipulation. *International Journal of Intelligent Robotics and Applications*, 3, 2019.

[6] Quan Vuong, Sergey Levine, Homer Rich Walke, Karl Pertsch, Anikait Singh, Ria Doshi, Charles Xu, Jianlan Luo, Liam Tan, Dhruv Shah, et al. Open x-embodiment: Robotic learning datasets and rt-x models. In *Towards Generalist Robots: Learning Paradigms for Scalable Skill Acquisition@ CoRL2023*, 2023.

[7] Konstantinos Bousmalis, et al, and Nicolas Heess. RoboCat: A Self-Improving Foundation Agent for Robotic Manipulation, 2023.

[8] Scott Reed, Konrad Zolna, Emilio Parisotto, Sergio Gomez Colmenarejo, Alexander Novikov, Gabriel Barth-Maron, Mai Gimenez, Yury Sulsky, Jackie Kay, Jost Tobias Springenberg, Tom Eccles, Jake Bruce, Ali Razavi, Ashley Edwards, Nicolas Heess, Yutian Chen, Raia

Hadsell, Oriol Vinyals, Mahyar Bordbar, and Nando de Freitas. A Generalist Agent, 2022.

[9] Jonathan Yang, Dorsa Sadigh, and Chelsea Finn. Polybot: Training One Policy Across Robots While Embracing Variability, 2023.

[10] Lin Shao, Fabio Ferreira, Mikael Jorda, Varun Nambiar, Jianlan Luo, Eugen Solowjow, Juan Aparicio Ojea, Oussama Khatib, and Jeannette Bohg. UniGrasp: Learning a Unified Model to Grasp With Multifingered Robotic Hands. *IEEE Robotics and Automation Letters*, 5(2):2286–2293, 2020.

[11] Coline Devin, Abhishek Gupta, Trevor Darrell, Pieter Abbeel, and Sergey Levine. Learning modular neural network policies for multi-task and multi-robot transfer. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, Singapore, 2017.

[12] Wenlong Huang, Igor Mordatch, and Deepak Pathak. One Policy to Control Them All: Shared Modular Policies for Agent-Agnostic Control, 2020.

[13] Gautam Salhotra, I.-Chun Arthur Liu, and Gaurav Sukhatme. Learning Robot Manipulation from Cross-Morphology Demonstration, 2023.

[14] Peng Gao, Jiaming Han, Renrui Zhang, Ziyi Lin, Shijie Geng, Aojun Zhou, Wei Zhang, Pan Lu, Conghui He, Xiangyu Yue, et al. Llama-adapter v2: Parameter-efficient visual instruction model. *arXiv preprint arXiv:2304.15010*, 2023.

[15] Tianrun Chen, Lanyun Zhu, Chaotao Deng, Runlong Cao, Yan Wang, Shangzhan Zhang, Zejian Li, Lingyun Sun, Ying Zang, and Papa Mao. Sam-adapter: Adapting segment anything in underperformed scenes. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3367–3375, 2023.

[16] Chong Mou, Xintao Wang, Liangbin Xie, Yanze Wu, Jian Zhang, Zhongang Qi, Ying Shan, and Xiaohu Qie. T2i-adapter: Learning adapters to dig out more controllable ability for text-to-image diffusion models. *arXiv preprint arXiv:2302.08453*, 2023.

[17] Hu Ye, Jun Zhang, Sibo Liu, Xiao Han, and Wei Yang. Ip-adapter: Text compatible image prompt adapter for text-to-image diffusion models. *arXiv preprint arXiv:2308.06721*, 2023.

[18] Junliang Guo, Zhirui Zhang, Linli Xu, Boxing Chen, and Enhong Chen. Adaptive adapters: An efficient way to incorporate bert into neural machine translation. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 29, 2021.

[19] 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. *arXiv:2106.09685*, 2021.

[20] Jiaxiang Cheng, Pan Xie, Xin Xia, Jiashi Li, Jie Wu, Yuxi Ren, Huixia Li, Xuefeng Xiao, Min Zheng, and Lean Fu. Resadapter: Domain consistent resolution adapter for diffusion models. *arXiv e-prints*, 2024.

[21] Kris Hauser. Klamp't: Kris' Locomotion and Manipulation Planning Toolbox. http://klampt.org, 2022.

[22] Tongzhou Mu, Zhan Ling, Fanbo Xiang, Derek Yang, Xuanlin Li, Stone Tao, Zhiao Huang, Zhiwei Jia, and Hao Su. Maniskill: Generalizable manipulation skill benchmark with large-scale demonstrations. *arXiv:2107.14483*, 2021.

[23] Unitree Aliengo. https://www.unitree.com/aliengo.

[24] Unitree Z1. https://www.unitree.com/z1.

[25] Takashi Yamamoto, Tamaki Nishino, Hideki Kajima, Mitsunori Ohta, and Koichi Ikeda. Human support robot (HSR). Association for Computing Machinery, 2018.

[26] Yifan Zhou, Shubham Sonawani, Mariano Phielipp, Simon Stepputtis, and Heni Ben Amor. Modularity through Attention: Efficient Training and Transfer of Language-Conditioned Policies for Robot Manipulation, 2022.

[27] Abhishek Padalkar, Acorn Pooley, Ajinkya Jain, Alex Bewley, Alex Herzog, Alex Irpan, Alexander Khazatsky, Anant Rai, Anikait Singh, Anthony Brohan, et al. Open x-embodiment: Robotic learning datasets and rt-x models. *arXiv preprint arXiv:2310.08864*, 2023.

[28] Fereshteh Sadeghi, Alexander Toshev, Eric Jang, and Sergey Levine. Sim2real view invariant visual servoing by recurrent control, 2017.

[29] Dhruv Shah, Ajay Sridhar, Arjun Bhorkar, Noriaki Hirose, and Sergey Levine. GNM: A General Navigation Model to Drive Any Robot, 2023.

[30] Vitaly Kurin, Maximilian Igl, Tim Rocktäschel, Wendelin Boehmer, and Shimon Whiteson. My Body is a Cage: The Role of Morphology in Graph-Based Incompatible Control, 2021.

[31] Haoran Geng, Ziming Li, Yiran Geng, Jiayi Chen, Hao Dong, and He Wang. Partmanip: Learning cross-category generalizable part manipulation policy from point cloud observations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023.

[32] Kaichun Mo, Leonidas J Guibas, Mustafa Mukadam, Abhinav Gupta, and Shubham Tulsiani. Where2act: From pixels to actions for articulated 3d objects. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021.

[33] Zhenjia Xu, Zhanpeng He, and Shuran Song. Universal manipulation policy network for articulated objects. *IEEE robotics and automation letters*, 7(2), 2022.

[34] Ruihai Wu, Yan Zhao, Kaichun Mo, Zizheng Guo, Yian Wang, Tianhao Wu, Qingnan Fan, Xuelin Chen, Leonidas Guibas, and Hao Dong. Vat-mart: Learning visual action trajectory proposals for manipulating 3d articulated objects. *arXiv:2106.14440*, 2021.

[35] Kai Lu, Bo Yang, Bing Wang, and Andrew Markham. Decoupling skill learning from robotic control for generalizable object manipulation, 2023.

[36] Roberto Martín-Martín, Michelle A Lee, Rachel Gardner, Silvio Savarese, Jeannette Bohg, and Animesh Garg. Variable impedance control in end-effector space: An action space for reinforcement learning in contact-rich tasks. In *2019 IEEE/RSJ international conference on intelligent robots and systems (IROS)*, pages 1010–1017. IEEE, 2019.

[37] Murtaza Dalal, Deepak Pathak, and Ruslan Salakhutdinov. Accelerating Robotic Reinforcement Learning via Parameterized Action Primitives. *arXiv:2110.15360 [cs]*, October 2021. arXiv: 2110.15360.

[38] Karl Pertsch, Youngwoon Lee, and Joseph J. Lim. Accelerating Reinforcement Learning with Learned Skill Priors. October 2020.

[39] Hao Shen, Weikang Wan, and He Wang. Learning category-level generalizable object manipulation policy via generative adversarial self-imitation learning from demonstrations. *IEEE Robotics and Automation Letters*, 7(4):11166–11173, 2022.

[40] Haoran Geng, Songlin Wei, Congyue Deng, Bokui Shen, He Wang, and Leonidas Guibas. Sage: Bridging semantic and actionable parts for generalizable articulated-object manipulation under language instructions, 2023.

[41] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners, 2020.

[42] Zhe Chen, Yuchen Duan, Wenhai Wang, Junjun He, Tong Lu, Jifeng Dai, and Yu Qiao. Vision transformer adapter for dense predictions. *arXiv preprint arXiv:2205.08534*, 2022.

[43] Yi-Lin Sung, Jaemin Cho, and Mohit Bansal. Vl-adapter: Parameter-efficient transfer learning for vision-and-language tasks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5227–5237, 2022.

[44] Renrui Zhang, Rongyao Fang, Wei Zhang, Peng Gao, Kunchang Li, Jifeng Dai, Yu Qiao, and Hongsheng Li. Tip-adapter: Training-free clip-adapter for better vision-language modeling. *arXiv preprint arXiv:2111.03930*, 2021.

[45] Anthony Liang, Ishika Singh, Karl Pertsch, and Jesse Thomason. Transformer adapters for robot learning. In *CoRL 2022 Workshop on Pre-training Robot Learning*, 2022.

[46] Zuxin Liu, Jesse Zhang, Kavosh Asadi, Yao Liu, Ding Zhao, Shoham Sabach, and Rasool Fakoor. Tail: Task-specific adapters for imitation learning with large pretrained models. In *The Twelfth International Conference on Learning Representations*, 2023.

[47] Mohit Sharma, Claudio Fantacci, Yuxiang Zhou, Skanda Koppula, Nicolas Heess, Jon Scholz, and Yusuf Aytar. Lossless adaptation of pretrained vision models for robotic manipulation. In *The Eleventh International Conference on Learning Representations*, 2023.

[48] Martin L Puterman. Markov decision processes. *Handbooks in operations research and management science*, 2, 1990.

[49] Matthijs TJ Spaan. Partially observable markov decision processes. In *Reinforcement learning: State-of-the-art*. Springer, 2012.

[50] Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In Jennifer Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*. PMLR, 10–15 Jul 2018.

[51] Andrew Szot, Alex Clegg, Eric Undersander, Erik Wijmans, Yili Zhao, John Turner, Noah Maestre, Mustafa Mukadam, Devendra Chaplot, Oleksandr Maksymets, Aaron Gokaslan, Vladimir Vondrus, Sameer Dharur, Franziska Meier, Wojciech Galuba, Angel Chang, Zsolt Kira, Vladlen Koltun, Jitendra Malik, Manolis Savva, and Dhruv Batra. Habitat 2.0: Training Home Assistants to Rearrange their Habitat. *arXiv:2106.14405 [cs]*, June 2021. arXiv: 2106.14405.