Meta-Residual Policy Learning: Zero-Trial Robot Skill Adaptation via Knowledge Fusion

Peng Hao, Tao Lu, Shaowei Cui, Junhang Wei, Yinghao Cai, and Shuo Wang, *Member, IEEE*

***Abstract*—Adapting the mastered manipulation skill to novel objects is still challenging for robots. Recent works have attempted to endow the robot with the ability to adapt to unseen tasks by leveraging meta-learning. However, these methods are data-hungry in the training phase, which limits their application in the real world. In this paper, we propose Meta-Residual Policy Learning (MRPL) to reduce the cost of policy learning and adaptation. During meta-training, MRPL accelerates the learning process by focusing on the residual task-shared knowledge that is hard to be embedded in the hand-engineered controller. During testing, MRPL achieves fast adaptation on similar unseen tasks through fusing task-specific knowledge in the demonstration with task-shared knowledge in the learned policy. We conduct a series of simulated and real-world peg-in-hole tasks to evaluate the proposed method. The experimental results demonstrate that MRPL outperforms prior methods in robot skill adaptation. Code for this work is available at https://github.com/Bartopt/code4MRPL.**

***Index Terms*—Machine learning for robot control, learning from experience, assembly.**

I. Introduction

ROBOTS struggle to adapt to new tasks quickly in today's industrial environment since robot programs are usually task-specific. Recent studies leverage Reinforcement Learning (RL) [1], Imitation Learning (IL) [2] to improve robotic autonomy. However, these methods often require a large number of expert demonstrations or interactive data, which is not easy to acquire in the real world. Designing robot policies that can quickly adapt to new tasks would significantly broaden the space of manufacturing tasks that can be automated by robots.

Using prior knowledge can speed up the policy learning process of RL and IL [3], [4], and is classified into task-shared knowledge and task-specific knowledge [5]. Previous works leveraged task-specific knowledge to improve the convergence performance of RL algorithms, where the knowledge is in the forms of demonstrations [4], or task-specific reward functions [6]. However, these task-oriented methods cannot achieve skill transfer. Task-shared knowledge endows robots with the ability to adapt to different tasks quickly. Rakelly *et al.* [7] proposed a meta-RL algorithm to transfer the learned policy to new tasks. Unfortunately, meta-RL is time-consuming during meta-training and needs dozens of trials for adapting new tasks since lacking task-specific knowledge. Duan *et al.* [3] and Zhou *et al.* [8] combine meta-learning with IL and achieve one-shot IL in simulation. However, their methods require thousands of demonstrations during meta-training, which are not easy to collect in the real world. Although skill adaptation has achieved many exciting works, there is still a gap between the deployment cost of existing methods and the demands of real-world robot applications.

To tackle these problems, we propose a robot skill adaptation method based on the fusion of multi-source prior knowledge. Our key insight is that the agent's adaptability can benefit from combining task-shared and task-specific knowledge. Numerous studies on human learning have shown the critical role of prior knowledge for encoding and storing new information [9]. Learners who have a large body of information already stored in long-term memory have more ideas to which they can relate their new experiences and so can more easily engage in such processes as meaningful learning [10]. Additionally, the new experience is more effective when it activates and builds on the learner's prior knowledge. However, even if learners have prior knowledge that can relate to new experiences, learners are not always aware of the connections they might make [11].

Inspired by the human learning theories, we propose Meta-Residual Policy Learning (MRPL) to endow robots with the ability to adapt the mastered skill to similar unseen tasks quickly. The overview of MRPL is shown in Fig. 1. MRPL consists of a Task-Conditioned Residual Policy (TCRP) and a Task Extraction Network (TEN). During training, the TCRP is trained to extract and store the residual task-shared knowledge, which refers to the part of task-shared knowledge that is hard to be embedded in the hand-designed controller. Simultaneously, the TEN is learned to infer the task belief from the interaction data, representing the task-specific knowledge. During adaption, we first leverage the TEN to infer the belief of the new task from an

Manuscript received October 25, 2021; accepted January 18, 2022. Date of publication January 31, 2022; date of current version February 15, 2022. This work was supported in part by the National Key R&D Program of China under Grant 2018AAA0103003, in part by the National Natural Science Foundation of China under Grants 61773378, U1913201, and U1713222, and in part by the Strategic Priority Research Program of Chinese Academy of Science under Grant XDB32050100. This letter was recommended for publication by Associate Editor Jim Torresen and Editor Tetsuya Ogata upon evaluation of the reviewers' comments. *(Corresponding author: Shuo Wang.)*

Tao Lu, Shaowei Cui, Junhang Wei, and Yinghao Cai are with the State Key Laboratory of Management and Control for Complex Systems, Institute of Automation, Chinese Academy of Sciences, Beijing 100190, China (e-mail: tao.lu@ia.ac.cn; cuishaowei2017@ia.ac.cn; weijunhang2018@ia.ac.cn; yinghao.cai@ia.ac.cn).

Peng Hao is with the State Key Laboratory of Management and Control for Complex Systems, Institute of Automation, Chinese Academy of Sciences, Beijing 100190, China, and also with the School of Artificial Intelligence, University of Chinese Academy of Sciences, Beijing 100049, China (e-mail: haopeng2017@ia.ac.cn).

Shuo Wang is with the State Key Laboratory of Management and Control for Complex Systems, Institute of Automation, Chinese Academy of Sciences, Beijing 100190, China, and with the School of Artificial Intelligence, University of Chinese Academy of Sciences, Beijing 100049, China, and also with the Center for Excellence in Brain Science and Intelligence Technology, Chinese Academy of Sciences, Shanghai 200031, China (e-mail: shuo.wang@ia.ac.cn).

This letter has supplementary downloadable material available at https://doi.org/10.1109/LRA.2022.3146916, provided by the authors.

Digital Object Identifier 10.1109/LRA.2022.3146916

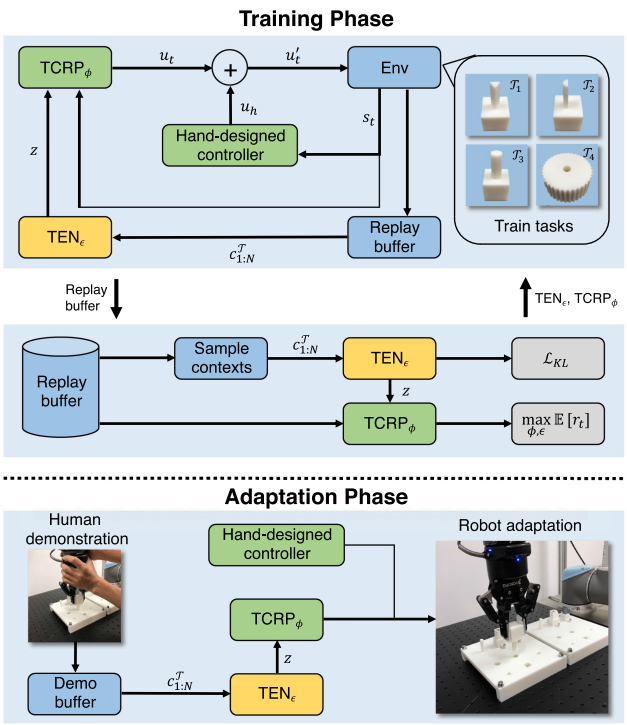


Fig. 1. The overview of the proposed MRPL. In the training phase, the Task-Conditioned Residual Policy (TCRP) is trained to extract the residual task-shared knowledge, and the Task Extraction Network (TEN) learned to infer tasks from experience. In the adaptation phase, TEN infers task-specific knowledge from human demonstrations, and subsequently, TCRP achieves fast adaptation to similar unseen tasks through knowledge fusion.

expert demonstration and then embed it in the trained TCRP to activate the mastered knowledge. By fusing new knowledge with existing knowledge, MRPL achieves robot skill fast adaptation.

We conduct a series of simulated and real-world peg-in-hole assembly tasks to evaluate MRPL. The experimental results show that MRPL has superior training and fast adaptation performance. Furthermore, MRPL achieves zero-trial insertions on new peg-in-hole assembly tasks with a single demonstration. The zero-trial means that the learned policy can adapt solely based on expert trajectories without any on-policy robot trials [12]. The contributions of this paper are summarized as follows:

- We propose MRPL, which fuses task-shared and task-specific knowledge to endow robots with the ability to adapt a learned skill to similar unseen tasks quickly.
- TCRP is designed to accelerate meta-training by only focusing on the residual task-shared knowledge.
- TEN is developed to infer task belief from demonstrations that contain task-specific knowledge. By fusing the inferred task with the TCRP, the robot adapts its skill to similar unseen tasks quickly.
- The simulated and real-world experiments demonstrate that our method has superior training and adaptation performance in peg-in-hole assembly.

The rest of the paper is organized as follows. Section II reviews related works. Section III presents the formulation of the problem. In Section IV, the details of the proposed method are provided, followed by the experiments for validation in Section V. This paper is concluded in Section VI.

II. Related Work

A. Robot Skill Adaptation With Meta-Learning

Kemp *et al.* [13] summarized the challenges in human environments for robot manipulation. One word that kept reappearing was *variation*, describing the challenges for robots when models used and assumptions made during algorithm design or learning differ from reality. Cui *et al.* [14] provided a detailed review of the adaptability of robot manipulation skills. They categorized the variations into known and novel variations based on whether they could be anticipated. They classified robot skill adaptation methods into two classes. One class realizes adaptation via generalization [15] while they can not handle novel variations. The other achieves adaptation on novel variations by gathering data of new tasks [16]. In this work, we focus on the latter to endow robots with the ability to adapt to novel variations rapidly.

Meta-learning aims to learn a model from many different tasks, where the learned model can adapt to a new task from a small amount of new data [16]. Recent works used meta-learning to quickly adapt robot skills to new tasks [7], platforms [17], and environments [18]. Rakelly *et al.* [7] and Li *et al.* [19] have attempted to achieve robot skill adaptation with meta-RL, where the robot can adapt to unseen tasks via its self-exploration. However, RL-based methods generally need dozens of rollouts for adaptation. Leveraging expert experience will significantly accelerate the convergence speed of algorithms [4]. Previous studies introduce human demonstration [4], hand-designed controllers [20] into RL to perform long-horizon, contact-rich robotic tasks. Inspired by this, recent works combine meta-learning with IL to achieve fast robot skill learning, where the simulated robot can learn new tasks from a single human demonstration [3]–[8]–[21]. Unfortunately, meta-IL requires many human demonstrations during meta-training, which are laborious to collect in the real world. Although skill adaptation has achieved many exciting works, existing methods still require massive amounts of training or testing data, limiting their deployment on real robots. In this paper, we propose MRPL to achieve data-efficient robot skill adaptation through knowledge fusion.

B. Peg-in-Hole Assembly

Robotic peg-in-hole assembly is still challenging as there are many inevitable contacts and friction in operation. In this paper, we pay attention to the learning-based control methods of robotic peg-in-hole, consisting of two categories: RL and IL [22]. Schoettler *et al.* [1] designed an image-based reward for the RL algorithm to train a real robotic insertion assembly policy. However, the image-based reward is sparse and is hard to converge. Previous works leveraged demonstration [4]–[23], feedback controllers [20], and fuzzy reward function [6] to assistant the RL learning. Unfortunately, RL-based methods still need several hours to learn a policy, and the learned policy is sensitive to environmental changes. IL acquires policy by mimicking the behavior of experts. Since collecting the real-world demonstration is time-consuming and laborious, learning the policy from a small amount of teaching is crucial. Ho *et al.* [24] proposed Generative Adversarial Imitation Learning (GAIL), which is generally quite sample-efficient on expert data. Recent studies have proposed improved GAIL for learning robotic peg-in-hole assembly from a dozen demonstrations [2]. However,

the generator-based methods have the model collapse problem. Although the peg-in-hole assembly has extensive literature in the robot community, the current methods have high adaptation costs when the peg's type changes, restricting the application scenarios of robotic assembly. In this work, we concentrate on the rapid adaptation of robot skills and validate our approach in robotic assembly tasks with different pegs.

III. Preliminaries

A. Problem Statement and Formulation

We consider the skill adaptation problem where the learned robotic skills can be executed adaptively when the environment changes. In particular, we focuses on external environment variations. For example, the robotic peg-in-hole assembly skills should adjust adaptively when the peg type varies. Our goal is to learn a policy that automatically adapts the robot behavior according to environment variations.

A robot skill can handle multiple specific tasks. For example, the robotic peg-in-hole assembly skill includes USB assembly tasks, RJ45 assembly tasks, etc. Since each task can be formulated as a Markov Decision Process (MDP) [25], a robot skill corresponds to a distribution of MDPs, denoted as $p(\mathcal{T})$. Specifically, each task $\mathcal{T}_i$ is defined by a tuple $(S_i, A_i, R_i, P_i, \gamma)$, where S_i is the state of the robot and the environment, A_i is the action space of the robot, R_i is the reward function, P_i is the transition function, and γ is the discount factor. Given a training set $\mathcal{T}_{train}$ sampled from $p(\mathcal{T})$, the robot skill adaptation algorithm aims to learn a policy π_{skill}. For a new task $\mathcal{T}_n$ sampled from $p(\mathcal{T})$, π_{skill} can adapt to the new task by using few data of the task.

B. Meta-Reinforcement Learning

We now describe preliminaries on meta-RL akin to Finn *et al.* [16], so that we can apply these ideas in the next section to the robot skill adaptation problem. Let $p(\mathcal{T})$ represents the distribution of tasks, where each task $\mathcal{T}$ is defined by an MDP. Specifically, $\mathcal{T} = \{p(s_0), r(s_t, a_t), p(s_{t+1}|s_t, a_t)\}$ includes the initial state distribution $p(s_0)$, reward function $r(s_t, a_t)$, and state transition distribution $p(s_{t+1}|s_t, a_t)$. Given a task set sampled from $p(\mathcal{T})$, the meta-training algorithms aim to learn a policy that can use the experience to quickly adapt to a new task during testing.

IV. Method

This section presents MRPL, which uses meta-RL concepts introduced in Section III to tackle the robot skill adaptation problem. The overview of MRPL is shown in Fig. 1.

A. Task-Conditioned Residual Policy

Leveraging prior knowledge is a significant part of human adaptability. For robots, the policy is the carrier of prior knowledge. Hence, the form of policy is essential to robot skill adaptation as it determines whether the robot can effectively extract and utilize prior knowledge. A well-designed policy should have the following two abilities. First, it can extract and store task-shared knowledge from experience. Secondly, the pre-trained policy can quickly adapt to a new task with task-specific knowledge. In this part, we describe how the proposed TCRP implements these two abilities.

Previous meta-RL works [7] [16] have attempted to extract prior knowledge from exploration data. They generally use the policy in the form of $u = \pi(s)$. However, this kind of policy is learning from scratch and requires a large amount of training data, which is not easy to collect for robotic manipulations. Considering robot control have many mature theories and methods, it is unwise to abandon them altogether. Johannink *et al.* [20] proposed residual RL for robot control. They increase the data efficiency of the real-world RL by decomposing the control problem into a part solved by conventional controllers and the residual part solved by RL. Inspired by their work, we introduce the residual mechanism into meta-RL to solve the data-hungry dilemma of applying meta-RL in robot control. The proposed policy is:

$$u' = \pi_H(s) + \pi_\phi(s) \tag{1}$$

where $\pi_H(s)$ is a hand-engineered controller representing part of the task-shared knowledge, and $\pi_\phi(s)$ is the residual policy designed to extract the residual task-shared knowledge. The workload of extracting task-shared knowledge is effectively reduced by decomposing task-shared knowledge, thus accelerating the training process.

People often leverage demonstrations (e.g., written descriptions, videos, etc.) when learning. These demonstrations include task-specific knowledge, which can boost learning efficiency. Previous works [4] leveraged behavior cloning loss to guide agents imitating experts. However, these fine-grained imitations require online training and are vulnerable to sub-optimal demonstrations. Recent works [7] [26] designed context-based policies where the action is determined by current state and task. They realize policy adaption without additional training steps. Considering the convenience and optimality, we leverage encoding task from the demonstration to accelerate skill adaptation. Based on the residual policy, we proposed our task-conditioned residual policy:

$$u' = \pi_H(s) + \pi_\phi(u|s, z) \tag{2}$$

where z is the inferred task so that TCRP can adapt its behavior according to the task.

B. Task Extraction Network

After designing TCRP, we propose the TEN to infer the salient task information z from demonstrations. We pursue to train the TEN without human participation. To achieve this, we need to figure out what is the salient task information. Rakelly *et al.* [7] explain it as given a task defined by an MDP, the task information should be able to reconstruct the components of the MDP (e.g., reward function, transition function), or maximize the task's return through embedding in the policy. In this paper, we follow their approach that learns the TEN by recovering the state-action value function from the interaction data. Since the demonstration can be regarded as interaction data, the TEN learned without expert data can directly apply to encode demonstrations.

To implement this, we design a network $q_\epsilon(z|c^{\mathcal{T}}_{1:N})$ to estimates the posterior $p(z|c^{\mathcal{T}}_{1:N})$, where ϵ is the network parameter, $c^{\mathcal{T}}_{1:N}$ is the experience of $\mathcal{T}$. Specifically, let $c^{\mathcal{T}}_n = \{s_n, a_n, r_n, s_{n+1}\}$ be one transition of $\mathcal{T}$, $c^{\mathcal{T}}_{1:N}$ consists of N transitions. For simplicity, we use $c^{\mathcal{T}}$ to represent $c^{\mathcal{T}}_{1:N}$. Leveraging replay buffers to encode the MDP is permutation-invariant

with respect to data orders, e.g., the reward and the transition functions can be reconstructed from disordered transitions in the form of $\{s_i, a_i, s_{i+1}, r_i\}$. Therefore, we use a permutation-invariant encoder network architecture to infer task information z from replay buffers [27]. The network is composed of multiple encoders with shared parameters, and z is the product of all encoder outputs, which is as:

$$q_\epsilon(z|c^{\mathcal{T}}) \propto \Pi_{n=1}^{N} q_\epsilon(z|c_n^{\mathcal{T}}) \tag{3}$$

where $q_\epsilon(z|c_n^{\mathcal{T}})$ is the Gaussian posterior. Specifically, $q_\epsilon(z|c_n^{\mathcal{T}}) = \mathcal{N}(f_\epsilon^\mu(c_n^{\mathcal{T}}), f_\epsilon^\sigma(c_n^{\mathcal{T}}))$, where f_ϵ is a neural network with a parameter ϵ used to predict the mean μ and variance σ from $c_n^{\mathcal{T}}$.

C. Network Training

In this part, we describe how to train the designed TCRP and TEN. Sample efficiency is essential to our approach. We not only care about adaptation efficiency but also pursue better training efficiency. We build our training algorithm on top of the soft actor-critic algorithm (SAC) [28]. SAC is an off-policy actor-critic method in the maximum entropy RL framework, and it exhibits good sample efficiency and stability on a range of continuous control benchmarks. To optimize the TEN, we use the amortized variational inference approach same as [7], which is as follows:

$$J_q(\epsilon) = \mathbb{E}_{z\sim q_\epsilon(z|c^{\mathcal{T}})}[J_Q(\theta) + \beta D_{KL}(q_\epsilon(z|c^{\mathcal{T}})||p(z))] \tag{4}$$

where $J_Q(\theta)$ is the critic loss in SAC. Specifically, $J_Q(\theta) = J_Q(\theta_1) + J_Q(\theta_2)$. The KL term is used to constrain z to contain only the essential information for policy adaptation and avoid overfitting on training tasks. The critic parameters can be trained to minimize the soft Bellman residual:

$$J_Q(\theta_i) = \mathbb{E}_{\substack{(s_t,u_t,s_{t+1},r_t)\sim\mathcal{B} \\ z\sim q_\epsilon(z|c^{\mathcal{T}})}}[Q_{\theta_i}(s_t,u_t,z) - \hat{Q}_{\theta_i}(s_t,u_t,z)]^2 \tag{5}$$

where,

$$\hat{Q}_{\theta_i}(s_t,u_t,z) = r_t + \bar{V}_\psi(s_{t+1},z) \tag{6}$$

in equation (5), the second term is the target network and is trained by the squared residual error $J_V(\psi)$, which is:

$$J_V(\psi) = \mathbb{E}_{\substack{s_t\sim\mathcal{B} \\ z\sim q_\epsilon(z|c^{\mathcal{T}})}}[V_\psi(s_t,z) - \hat{V}_\psi(s_t,z)]^2 \tag{7}$$

where,

$$\hat{V}_\psi(s_t,z) = \mathbb{E}_{u_t\sim\pi_\phi}[Q_\theta(s_t,u_t,z) - \log\pi_\phi(u_t|s_t,z)] \tag{8}$$

the actions in equation (7) are sampled according to current policy. By adding additional dependence on z as input, we extend the actor loss of SAC as:

$$J_\pi(\phi) = \mathbb{E}_{\substack{s_t\sim\mathcal{B},u_t\sim\pi_\phi \\ z\sim q_\epsilon(z|c^{\mathcal{T}})}}\left[D_{KL}\left(\pi_\phi(u_t|s_t,z)\|\frac{\exp(Q_\theta(s_t,u_t,z))}{Z_\theta(s_t)}\right)\right] \tag{9}$$

We summarize our training procedure in Algorithm 1. The replay buffers $\mathcal{B}_{rl}$ and $\mathcal{B}_c$ are designed for meta-training. Specifically, $\mathcal{B}_{rl}$ contains all the transitions and is for training the value and the policy network. And $\mathcal{B}_c$ only includes the recent data and is for training the task inference network.

Algorithm 1: MRPL Training.

Input: Training tasks $\{\mathcal{T}_i\}_{i=1\ldots T}$ sampled from $p(\mathcal{T})$, hand-engineered controller π_H, initialized network parameters $\epsilon, \theta, \psi, \phi$, initialized replay buffer $\mathcal{B}_{rl}^i, \mathcal{B}_c^i$ for each task

Output: $q_\epsilon, Q_\theta, V_\psi, \pi_\phi$

for *each iteration* **do**
  for *each* $\mathcal{T}_i$ **do**
    Clear $\mathcal{B}_c^i, c^{\mathcal{T}_i}$
    for $n = 0, \ldots, N$ **do**
      Sample initial state $s_0 \sim E$
      Sample $z \sim q_\epsilon(z|c^{\mathcal{T}_i})$
      for $t = 0, \ldots H-1$ steps **do**
        Get policy action $u_t = \pi_\phi(u_t|s_t, z)$
        Get action to execute $u_t' = u_t + \pi_H(s_t)$
        Get next state $s_{t+1} \sim p(\cdot|s_t, u_t')$
        $\mathcal{B}_{rl}^i \leftarrow \mathcal{B}_{rl}^i \cup \{(s_t, u_t, s_{t+1}, r_t)\}$
        $\mathcal{B}_c^i \leftarrow \mathcal{B}_c^i \cup \{(s_t, u_t, s_{t+1}, r_t)\}$
      end for
      Sample $c^{\mathcal{T}_i} \sim \mathcal{B}_c^i$
    end for
  end for
  for *each training step* **do**
    for *each* $\mathcal{T}_i$ **do**
      Sample $c^{\mathcal{T}_i} \sim \mathcal{B}_c^i$
      Sample trainsition set $\{(s, u, s,' r)\} \sim \mathcal{B}_{rl}^i$
      Sample $z \sim q_\epsilon(z|c^{\mathcal{T}_i})$
      Calculate $J_q^i(\epsilon), J_Q^i(\theta), J_V^i(\psi), J_\pi^i(\phi)$
    end for
    $\epsilon \leftarrow \epsilon - \lambda_q \nabla_\epsilon \sum_i J_q^i(\epsilon)$
    $\theta \leftarrow \theta - \lambda_Q \nabla_\theta \sum_i J_Q^i(\theta)$
    $\psi \leftarrow \psi - \lambda_V \nabla_\psi \sum_i J_V^i(\psi)$
    $\phi \leftarrow \phi - \lambda_\pi \nabla_\phi \sum_i J_\pi^i(\phi)$
  end for
end for

D. Fast Adapatation With Demonstration

After meta-training, we obtain the TCRP that contains task-shared knowledge and the TEN for extracting the task-specific knowledge from the interaction data. In this subsection, we describe how to combine these two components with human demonstration to achieve fast policy adaptation.

We aim to adapt the learned π_ϕ on a new task $\mathcal{T}_n$ quickly by using a single human demonstration in form of $D_{\mathcal{T}} = \{(s_0, u_0, r_0, s_1), \ldots, (s_t, u_t, r_t, s_{t+1})\}$. First, we pre-process the actions in demonstrations to avoid distribution shift. Then, we leverage the TEN to infer the task z from $c^{\mathcal{T}}$ and use z as the input of π_ϕ, which is a significant step as it realizes the fusion of task-shared and task-specific knowledge by embedding the demonstration into task-conditioned policy. Finally, the control policy adapts to the new task with few trials. We summary the whole procedure in Algorithm 2.

V. Experiment

We evaluate our method on simulated and real-world robotic peg-in-hole assembly tasks. The experiments aim to answer the following questions:

Algorithm 2: Fast Adapatation with Demonstration.

Input: Testing task $\mathcal{T}_n$ sampled from $p(\mathcal{T})$, hand-engineered controller π_H, network q_ϵ, π_ϕ, demonstration $D_\mathcal{T}$

for u_t *in* $D_\mathcal{T}$ **do**
  $u_t \leftarrow u_t - \pi_H(s_t)$
end for
Initialize $c^\mathcal{T} = D_\mathcal{T}$
for $n = 0, \ldots, N$
  Sample initial state $s_0 \sim E$
  Sample $z \sim q_\epsilon(z|c^\mathcal{T})$
  for $t = 0, \ldots H-1$ steps
    Get policy action $u_t = \pi_\phi(u_t|s_t, z)$
    Get action to execute $u'_t = u_t + \pi_H(s_t)$
    Get next state $s_{t+1} \sim p(\cdot|s_t, u'_t)$
    $c^\mathcal{T} \leftarrow c^\mathcal{T} \cup \{(s_t, u_t, s_{t+1}, r_t)\}$
  end for
end for

- Can MRPL generate the policy of the new task from only a single demonstration?
- Does MRPL have better training and adaptation performance than baselines?
- What is the contribution of each component in MRPL to the adaptation performance?

A. Ablation and Baselines

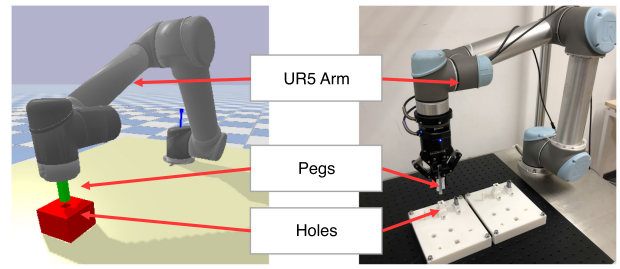
To answer the above questions, we compare the following baseline and ablation methods with MRPL on various tasks.

- *PEARL* [7] A probabilistic context-based meta-RL algorithm, with high adaptation efficiency.
- *MAML* [16] aims to meta-train a model that can quickly adapt to new tasks via a few gradient descent steps by explicitly performing a bi-level optimization.
- *Random Search* [29] The robot randomly samples in a square horizontal plane with a side length of 3 mm and moves down at the sampling position. The robot tries the next sampling point if it failed to insert. At most 10 random insertion attempts are executed in each episode.
- *MRPL-NoResi-NoDemo* is the ablation of our method in which the robot is only controlled by a task-conditioned policy $u_t = \pi_\theta(u|s, z)$. And z is inferred from autonomous exploration.
- *MRPL-NoResi* is the ablation of our method in which the policy is a task-conditioned policy $u_t = \pi_\theta(u|s, z)$.
- *MRPL-NoDemo* is an ablation of MRPL where TCRP is initialized without demonstration during adaptation.
- *Hand-engineered* refers to the approaches that only use feedback controllers. **HE-NG** represents the controller with noisy goals. **HE-DG** represents the controller using the goals extracted from human demonstrations, which are more accurate than noisy goals.

B. Simulation

We first compare our method with other methods on the simulated peg-in-hole assembly tasks, which is easy to generate abundant data for evaluation.

Experimental Setup The environment as shown Fig. 2(a) in is simulated using PyBullet [30]. We focus on controlling the robot moving an already-grasped peg into the target hole, which is the most common setting in actual manufacturing [31]. The manipulated object's pose errors and various peg types are troublesome variables for designing robotic peg-in-hole assembly controllers. We apply MRPL to automated peg-in-hole assembly to tackle these problems. Each task is defined by two parameters.



(a) (b)

Fig. 2. The simulated and real-world experimental setup. (a) Simulated robotic peg-in-hole assembly. (b) Real-world robotic peg-in-hole assembly.

- Hole position errors, which correspond to the variants of the reward function in the MDP. We add random horizontal errors between -2 mm and 2 mm to the hole location to acquire the assumed goal location.
- Peg types, which correspond to variants of transition functions in the MDP. Specifically, the peg's shape is randomly chosen from circles and squares, where the peg's size is sampled between 15 mm and 16 mm. The size of the hole is fixed at 17 mm.

The state is a 3-D vector representing current end-effector positions relative to the assumed goal location. Since this kind of state space unifies the representations of pegs with different shapes, the policy can focus on adapting to dynamic model changes. To account for the peg position errors in the robot gripper, we add a fixed horizontal noise between -1 mm and 1 mm to the state after each reset. The action is also a 3-D vector representing the moving distance of each dimension, and the value range from -2 mm to 2 mm. The reward is the l_2 norm of the state, which represents the estimated distance between the current and the fully inserted position. The horizontal length of each task is 50 steps. We sample 40 tasks from the task distribution for training and 20 unseen tasks for testing. We use controllers with accurate goal positions to collect a demonstration for each task.

Let FC_k represents a fully-connected layer with k outputs. The network for the TCRP has the following architecture: $FC_{300} - FC_{300} - FC_{300} - FC_3$. Each FC layer is followed by Rectified Linear Unit (ReLU), except the final layer FC_3 which has tanh activation unit. The input of TCRP is a 13-D vector including z and s, and its output is a 3-D vector representing a. The network architecture of the TEN is as follows: $FC_{200} - FC_{200} - FC_{200} - FC_{10}$. Each FC layer is followed by ReLU, except the final layer FC_{10} which has no activation unit. The input and output of TEN are two 10-D vectors that represent a replay buffer and inferred z, respectively. The hand-designed controller is a P controller $\pi_H(s) = k_p(s_{ng} - s_{cur})$, where s_{cur}, s_{ng} is the current and the assumed goal positions of the end-effector. We embed the task-shared knowledge of insert after aligning into the designed controller by dynamically adjusting k_p. Specifically, at the first 3 steps of each episode, we set k_p as $[0.2\ 0.2\ 0]^\top$ to align the peg by reducing the horizontal

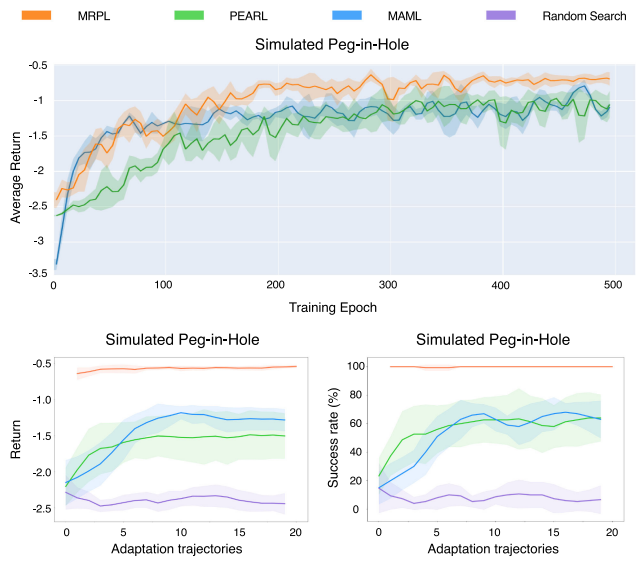


Fig. 3. Comparison of training and adapting performance of different methods on simulated tasks.

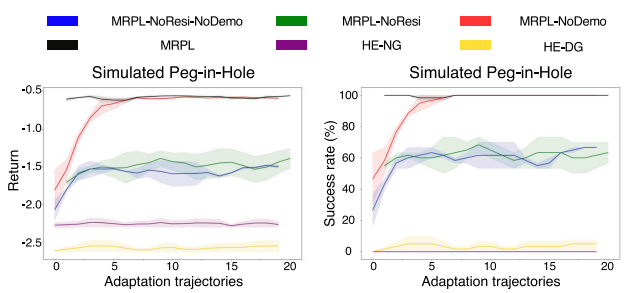


Fig. 4. Comparison of different ablation's adaptation performance on simulation tasks.

error. Then we set k_p to $[0.2\ 0.2\ 0.2]^\top$ to insert the peg by reducing the vertical error. The residual task-shared knowledge (e.g., how to tackle contacts, friction, and state noise) is learned by TCRP.

Results The adaptation and training curves of different methods are shown in Fig. 3. To ensure all methods are compared under identical amounts of data, we translate each adaptation curve along the x-axis according to the number of used demonstrations. The training results show that MRPL has better convergence performance than other baselines, which we attribute to the TCRP reducing the burden of extracting prior knowledge by decomposing it into a part embedded in the hand-engineered controller and the residual part learned from data. MRPL also achieves better adaptation performance under the same amount of data by leveraging the task-specific knowledge contained in the demonstration. Furthermore, MRPL has a high beginning value during adaptation, which means MRPL endows the robot with the ability to realize zero-trial adaptation on similar unseen tasks.

The comparison of all ablations is shown in Fig. 4. The results show that both TCRP and demonstration can improve the adaptation performance. Note that the TCRP representing the task-shared knowledge focuses on improving the convergence value, while the demonstration representing the task-specific knowledge focuses on improving the beginning value.

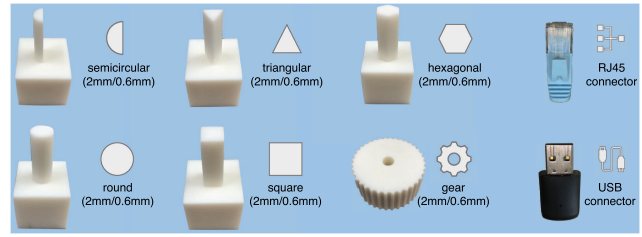


Fig. 5. Pegs used in the real-world experiment and their clearances.

By effectively fusing task-shared and task-specific knowledge, MRPL obtains the superior beginning and convergence value simultaneously, demonstrating excellent adaptability to similar unseen tasks. Furthermore, the hand-engineered methods have poor performance, implying that the TCRP does extract the residual residual task-shared knowledge, which is essential to complete the task.

C. Real-World Assembly

In this section, we conduct a series of real-world peg-in-hole assembly tasks to compare our method with baselines.

Experimental Setup The real-world experiments has the same setting as the simulation, which is shown in Fig. 2(b). Since there is no severe reality gap problem in position-based state, and the TCRP captures the dynamic differences of various pegs during meta-training, the learned TCRP and TEN in the simulation can be directly deployed to the real world for evaluation without additional training. To obtain state and reward in real scenarios, we first manually control the robot to successfully insert to acquire the hole location and add horizontal errors to get the assumed goal location. The assumed goal location corresponds to the noisy goal location estimated by a vision-based measurement system.

We use six different pegs and holes fabricated with a 3-D printer: square, round, triangular, semicircular, hexagonal, and gear. Each peg has two nominal clearances of 2 mm and 0.6 mm. We also use USB and RJ45 connectors for evaluation. There are 14 kinds of pegs in total, as shown in Fig. 5. We perform three independent evaluations for each peg. Similar to the simulated experiment setting, we focus on inserting the grasped peg into the hole. In the implementation, the robot moves to an assembled board to grasp the peg, and a human resets the board after each episode. Since the existence of tolerances, such grasp pipelines will introduce the peg's pose error in the robot gripper, which exists widely in vision-based grasp systems. After grasping, the robot moves to the start position and inserts the peg into the hole. We collected a single demonstration for each kind of peg via kinesthetic teaching, which is shown in Fig. 6.

We control the end-effector Cartesian position with a customized impedance controller to be smooth and tolerant of contacts. The scheme of the controller is summarized in Algorithm 3. In the implementation, we set f_{ts} to 10 N, a_z to 3 mm, ΔT to 3 s, and δ to 0.1 mm. The asynchronous manner means that the algorithm continually runs without waiting for the robot to finish the command. In contrast, the synchronous manner means the algorithm waits for the robot to finish the command before continually running.

TABLE I
COMPARISON OF DIFFERENT METHODS ON VARIOUS PEG-IN-HOLE ASSEMBLY TASKS

	Peg	S-2	S-0.6	R-2	R-0.6	SC-2	SC-0.6	T-2	T-0.6	H-2	H-0.6	G-2	G-0.6	USB	RJ45
Success Rate	MRPL	**100**	**100**	**100**	**100**	**83**	**70**	**97**	**70**	**100**	**80**	**100**	**100**	**73**	**97**
	PEARL [7]	75	65	73	65	76	53	65	57	73	64	100	93	33	90
Insertion Steps	MRPL	**17.3**	**14.7**	**12.9**	**17.1**	**14.8**	**17.4**	**14.8**	**15.7**	**12.5**	**17.8**	**12**	**13.5**	**12.1**	**11.5**
	PEARL [7]	26.3	26.3	22.5	27.1	23.1	26.8	26.6	30.1	21.7	30.3	24.6	25.6	15.3	12
Zero-trial Insertion	MRPL	✓	✓	✓	✓	✓	✗	✓	✓	✓	✓	✓	✓	✓	✓
	PEARL [7]	✗	✗	✓	✗	✓	✗	✗	✗	✓	✗	✓	✓	✗	✓

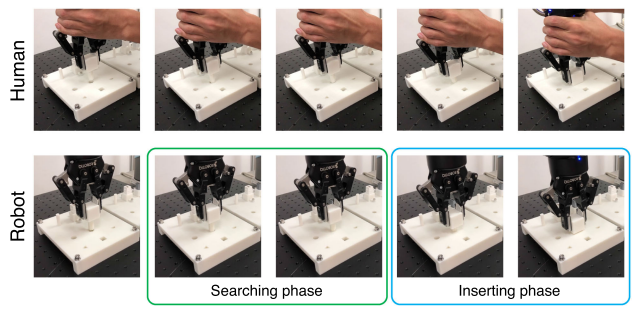


Fig. 6. Snapshots of human demonstration and robot execution on the hexagonal peg-in-hole assembly task.

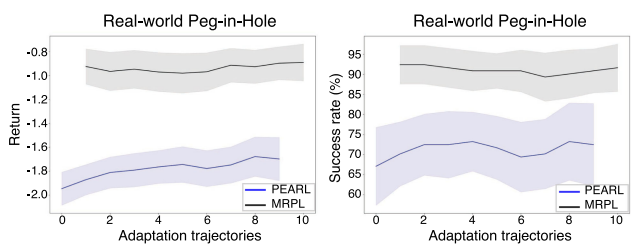


Fig. 7. Comparison of different methods on real-world assembly tasks.

Results The comparison results of different methods are shown in Fig. 7. The results show that MRPL has better convergence and beginning values than baselines. The high beginning value implies MRPL realizes the rapid adaptation in real-world peg-in-hole assembly tasks by fusing the task-specific knowledge in the demonstration and the task-shared knowledge in the TCRP. A more detailed comparison result is shown in TABLE I, where S, R, SC, T, H, G are the abbreviations of the square, round, semicircular, triangular, hexagonal, and gear, respectively, and 2 and 0.6 refer to the assembly tolerance. The results in TABLE I are average performance after the adaptation, where the end of the adaptation refers to the robot inserting the peg into the hole for the first time. The results demonstrate that MRPL has a superior success rate and the average insertion steps in all tasks simultaneously. The insertion steps are essential for practical applications since throughput is a major consideration in industrial settings. SC-0.6, T-0.6, and USB have poor success rates, which we attribute to the small insertable area caused by the assembly clearance and the special shape together.

Furthermore, we evaluate the zero-trial insertion ability of different methods, which means that the robot can complete the insertion without any interaction with the environment at first. The results show that PEARL achieves zero-trial insertions when the tolerance is 2 mm, but it fails when tolerance is reduced to 0.6 mm. Through knowledge fusion, our method achieves zero-trial insertion on almost all tasks, even on challenging USB tasks. Snapshots of the robot assembling a hexagonal peg are displayed in Fig. 6. Videos of all experiments can be viewed in supplementary materials.

Algorithm 3 Impedance Controller Scheme

Input: action u', force threshold f_{ts}, duration time ΔT, control accuracy δ, lifting height a^z

Get the position of the end effector s_{cur}
Get the force of the end effector $f_t = (f_t^x, f_t^y, f_t^z)$
$s_{goal} \leftarrow s_{cur} + u'$
Send s_{goal} to the motion controller asynchronously
while $t < t_0 + \Delta T$ and $(s_{cur} - s_{goal})^2 > \delta$ **do**
  Update s_{cur} and f_t at a frequency of 6 Hz
  if $\|f_t\|_\infty > f_{ts}$ **then**
    Send stop to motion controller
    break
  end if
end while
if $f_t^z > f_{ts}$ **then**
  $s_{goal} \leftarrow s_{cur} + (0, 0, a^z)$
  Send s_{goal} to the motion controller synchronously
end if

VI. CONCLUSION

This paper proposes Meta-Residual Policy Learning (MRPL) to the robot skill adaptation problem. MRPL contains two components. One is the Task-Conditioned Residual Policy, which extracts the residual task-shared knowledge during meta-training. The other is the Task Extraction Network, which infers task-specific knowledge from expert demonstrations during adaptation. After acquiring the two components through robot exploration, MRPL endows the robot with the ability to learn similar unseen tasks quickly via fusing multi-source knowledge. We conduct a series of simulated and real-world peg-in-hole assembly tasks to evaluate the proposed method, where the real-world experiments covered 14 unseen peg types. The experimental results show that through dividing the task-shared knowledge and only focusing on the residual parts, MRPL achieves better sample efficiency than prior methods during meta-training. Furthermore, by extracting task-specific knowledge from human demonstration and fusing it with task-shared knowledge, MRPL can learn unseen robotic peg-in-hole assembly tasks with few or zero trials. The experimental results indicate that the sample efficiency of the robot skill learning algorithm will benefit from the division and fusion of task knowledge. The rapid adaptability of MRPL makes it hope to improve the deployment efficiency of industrial robots.

While MRPL significantly extends the capabilities of prior methods, several limitations remain. One of the limitations is

that we only evaluate MRPL on the robotic peg-in-hole assembly tasks. An interesting avenue of future work would be to increase the diversity of tasks that robots need to adapt. Another limitation is that the current policy is trained in the low-dimensional state space. A potential future direction related to this limitation is to explore robot skill adaptation under multi-modal and high-dimensional observations. Another thing to note is that the hand-engineered controller of MRPL is hard to design in the high-dimensional observation space (e.g., image). A possible way of addressing this challenge is to use a perception module for dimensionality reduction.

References

[1] G. Schoettler *et al.*, "Deep reinforcement learning for industrial insertion tasks with visual inputs and natural rewards," in *Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst.*, Las Vegas, NV, USA, 2020, pp. 5548–5555.

[2] Y. Schroecker, M. Vecerik, and J. Scholz, "Generative predecessor models for sample-efficient imitation learning," in *Proc. Int. Conf. Learn. Representations*, 2019. [Online]. Available: https://openreview.net/pdf?id=SkeVsiAcYm

[3] Y. Duan *et al.*, "One-shot imitation learning," in *Proc. Adv. Neural Inf. Process. Syst.*, Long Beach, CA, USA, 2017, pp. 1087–1098.

[4] A. Nair, B. McGrew, M. Andrychowicz, W. Zaremba, and P. Abbeel, "Overcoming exploration in reinforcement learning with demonstrations," in *Proc. IEEE Int. Conf. Robot. Autom.*, Brisbane, QLD, Australia, 2018, pp. 6292–6299.

[5] E. Johns, "Coarse-to-fine imitation learning: Robot manipulation from a single demonstration," in *Proc. IEEE Int. Conf. Robot. Autom.*, Xi'an, China, 2021, pp. 4613–4619.

[6] J. Xu, Z. Hou, W. Wang, B. Xu, K. Zhang, and K. Chen, "Feedback deep deterministic policy gradient with fuzzy reward for robotic multiple peg-in-hole assembly tasks," *IEEE Trans. Ind. Informat.*, vol. 15, no. 3, pp. 1658–1667, Mar. 2019.

[7] K. Rakelly, A. Zhou, C. Finn, S. Levine, and D. Quillen, "Efficient off-policy meta-reinforcement learning via probabilistic context variables," in *Proc. Int. Conf. Mach. Learn.*, Long Beach, CA, USA, 2019, pp. 5331–5340.

[8] A. Zhou *et al.*, "Watch, try, learn: Meta-learning from demonstrations and rewards," in *Proc. Int. Conf. Learn. Representations*, 2020. [Online]. Available: https://openreview.net/pdf?id=SJg5J6NtDr

[9] J. Hattie, *Visible Learning: A. Synthesis of Over 800 Meta-Analyses Relating to Achievement*. London, U.K.: Routledge, 2008.

[10] J. E. Ormrod and K. M. Davis, *Human Learning*. London, U.K.: Merrill, 2004.

[11] H. A. Spires and J. Donley, "Prior knowledge activation: Inducing engagement with informational texts," *J. Educ. Psychol.*, vol. 90, no. 2, pp. 249–260, 1998.

[12] A. Wu, A. Piergiovanni, and M. Ryoo, "Model-based robot imitation with future image similarity," *Int. J. Comput. Vis.*, vol. 128, no. 5, pp. 1360–1374, 2020.

[13] C. C. Kemp, A. Edsinger, and E. Torres-Jara, "Challenges for robot manipulation in human environments," *IEEE Robot. Autom. Mag.*, vol. 14, no. 1, pp. 20–29, Mar. 2007.

[14] J. Cui and J. Trinkle, "Toward next-generation learned robot manipulation," *Sci. Robot.*, vol. 6, no. 54, 2021, Art. no. eabd9461.

[15] T. M. Mitchell, R. M. Keller, and S. T. Kedar-Cabelli, "Explanation-based generalization: A unifying view," *Mach. Learn.*, vol. 1, no. 1, pp. 47–80, 1986.

[16] C. Finn, P. Abbeel, and S. Levine, "Model-agnostic meta-learning for fast adaptation of deep networks," in *Proc. Int. Conf. Mach. Learn*, Sydney, NSW, Australia, 2017, pp. 1126–1135.

[17] A. Ghadirzadeh, X. Chen, P. Poklukar, C. Finn, M. Björkman, and D. Kragic, "Bayesian meta-learning for few-shot policy adaptation across robotic platforms," in *Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst.*, Prague, Czech Republic, 2021, pp. 1274–1280.

[18] S. M. Richards, N. Azizan, J. E. Slotine, and M. Pavone, "Adaptive-control-oriented meta-learning for nonlinear systems," in *Proc. Robot. Sci. Syst.*, 2021. [Online]. Available: http://www.roboticsproceedings.org/rss17/p056.pdf

[19] K. Li *et al.*, "MURAL: Meta-learning uncertainty-aware rewards for outcome-driven reinforcement learning," in *Proc. Int. Conf. Mach. Learn.*, 2021, pp. 6346–6356.

[20] T. Johannink *et al.*, "Residual reinforcement learning for robot control," in *Proc. IEEE Int. Conf. Robot. Autom.*, Montreal, QC, Canada, 2019, pp. 6023–6029.

[21] B. Wu, F. Xu, Z. He, A. Gupta, and P. K. Allen, "Squirl: Robust and efficient learning from video demonstration of long-horizon robotic manipulation tasks," in *Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst.*, Las Vegas, NV, USA, 2020, pp. 9720–9727.

[22] A. S. Morgan, B. Wen, J. Liang, A. Boularias, A. M. Dollar, and K. E. Bekris, "Vision-driven compliant manipulation for reliable; high-precision assembly tasks," in *Proc. Robot. Sci. Syst.*, 2021. [Online]. Available: http://www.roboticsproceedings.org/rss17/p070.pdf

[23] J. Luo *et al.*, "Robust multi-modal policies for industrial assembly via reinforcement learning and demonstrations: A large-scale study," in *Proc. Robot. Sci. Syst*, 2021. [Online]. Available: http://www.roboticsproceedings.org/rss17/p088.pdf

[24] J. Ho and S. Ermon, "Generative adversarial imitation learning," in *Proc. Adv. Neural Inf. Process. Syst.*, Barcelona, Spain, 2016, pp. 4565–4573.

[25] O. Kroemer, S. Niekum, and G. Konidaris, "A review of robot learning for manipulation: Challenges, representations, and algorithms," *J. Mach. Learn. Res.*, vol. 22, pp. 30:1–30:82, 2021.

[26] A. Bonardi, S. James, and A. J. Davison, "Learning one-shot imitation from humans without humans," *IEEE Robot. Autom. Lett.*, vol. 5, no. 2, pp. 3533–3539, Apr. 2020.

[27] J. Gordon, J. Bronskill, M. Bauer, S. Nowozin, and R. E. Turner, "Meta-learning probabilistic inference for prediction," in *Proc. Int. Conf. Learn. Representations*, 2019. [Online]. Available: https://openreview.net/pdf?id=HkxStoC5F7

[28] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, "Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor," in *Proc. Int. Conf. Mach. Learn.*, Stockholm, Sweden, 2018, pp. 1856–1865.

[29] J. A. Marvel, R. Bostelman, and J. Falco, "Multi-robot assembly strategies and metrics," *ACM Comput. Surv.*, vol. 51, no. 1, pp. 1–32, 2018.

[30] E. Coumans and Y. Bai, "Pybullet, a python module for physics simulation for games, robotics and machine learning," 2016-2019. [Online]. Available: http://pybullet.org

[31] J. Luo *et al.*, "Reinforcement learning on variable impedance controller for high-precision robotic assembly," in *Proc. IEEE Int. Conf. Robot. Autom.*, Montreal, QC, Canada, 2019, pp. 3080–3087.