SRSA: SKILL RETRIEVAL AND ADAPTATION FOR ROBOTIC ASSEMBLY TASKS

Yijie Guo[1], **Bingjie Tang**[2], **Iretiayo Akinola**[1], **Dieter Fox**[1,3], **Abhishek Gupta**[1,3] **& Yashraj Narang**[1]
[1]NVIDIA Corporation, [2]University of Southern California, [3]University of Washington

ABSTRACT

Enabling robots to learn novel tasks in a data-efficient manner is a long-standing challenge. Common strategies involve carefully leveraging prior experiences, especially transition data collected on related tasks. Although much progress has been made for general pick-and-place manipulation, far fewer studies have investigated contact-rich assembly tasks, where precise control is essential. We introduce **SRSA** (**S**kill **R**etrieval and **S**kill **A**daptation), a novel framework designed to address this problem by utilizing a pre-existing skill library containing policies for diverse assembly tasks. The challenge lies in identifying which skill from the library is most relevant for fine-tuning on a new task. Our key hypothesis is that skills showing higher zero-shot success rates on a new task are better suited for rapid and effective fine-tuning on that task. To this end, we propose to predict the transfer success for all skills in the skill library on a novel task, and then use this prediction to guide the skill retrieval process. We establish a framework that jointly captures features of object geometry, physical dynamics, and expert actions to represent the tasks, allowing us to efficiently learn the transfer success predictor. Extensive experiments demonstrate that SRSA significantly outperforms the leading baseline. When retrieving and fine-tuning skills on unseen tasks, SRSA achieves a 19% relative improvement in success rate, exhibits 2.6x lower standard deviation across random seeds, and requires 2.4x fewer transition samples to reach a satisfactory success rate, compared to the baseline. In a continual learning setup, SRSA efficiently learns policies for new tasks and incorporates them into the skill library, enhancing future policy learning. Furthermore, policies trained with SRSA in simulation achieve a 90% mean success rate when deployed in the real world. Please visit our project webpage https://srsa2024.github.io/.

1 INTRODUCTION

Humans excel at solving new tasks with few demonstrations or trial-and-error interactions. In robot learning, a key challenge is to similarly enable robots to learn control policies from sensory input in a data-efficient manner. Achieving data-efficient learning is crucial for deploying robots in diverse real-world environments, such as the household and industry. A compelling approach to efficient policy learning is the development of a foundation model or generalist policy that spans multiple tasks, as the model or policy can offer long-term efficiency gains by providing a strong base for adaptation to novel tasks. Significant advancements have been made in manipulation tasks, particularly in visual pre-training (Parisi et al., 2022; Nair et al., 2022), multi-task policy learning (Shridhar et al., 2022; Goyal et al., 2024), and policy generalization (Jang et al., 2022; Ebert et al., 2021).

Despite this progress, efficiently solving new tasks in contact-rich environments, such as robotic assembly, remains underexplored. Robotic assembly plays a critical role in industries like automotive, aerospace, and electronics, but learning assembly policies is uniquely difficult. These tasks require contact-rich interactions with high levels of precision and accuracy, compounded by the physical complexity of the environments, part variability, and strict reliability standards. Much of the existing research focuses on training specialist (i.e., single-task) policies for individual assembly tasks (Spector & Di Castro, 2021; Spector et al., 2022; Tang et al., 2023). Building on the strengths of these specialist approaches, we propose a novel method for tackling new assembly tasks. Our approach leverages a skill library – a collection of diverse specialist policies and associated information (such as object geometry and task-relevant trajectories) for various assembly tasks. These policies and data, regardless of the training strategies or learning approaches used to develop them, can be harnessed to efficiently solve previously-unseen assembly challenges.

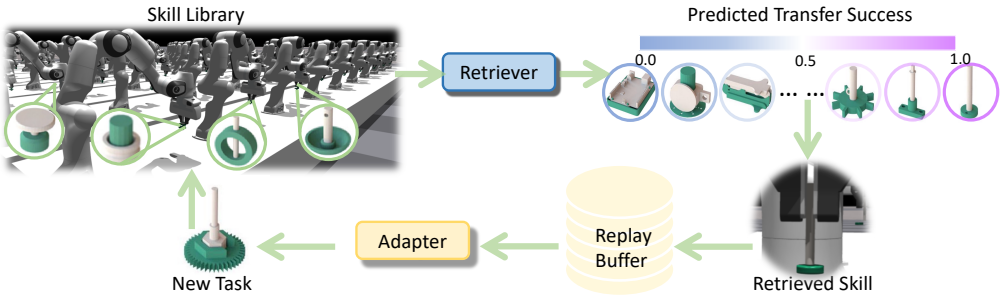


Figure 1: **Overview of SRSA.** We address assembly tasks, where the goal is to use a robot arm to insert diverse *plugs* (i.e., the white parts) into or onto corresponding *sockets* (i.e., the green parts). Specifically, we propose to predict the transfer success of applying prior skills (i.e., policies) to a new task, retrieve the skill with the highest predicted success rate, and fine-tune it on the new task. During fine-tuning, we accelerate and stabilize adaptation by incorporating imitation learning of high-rewarding transitions from the agent's own replay buffer.

To utilize prior task experiences, previous work on general pick-and-place tasks has explored methods such as imitating state-action pairs from expert demonstrations (Du et al., 2023; Lin et al., 2024; Kuang et al., 2024) and encoding sub-task skills as macro-action choices (Lynch et al., 2020; Pertsch et al., 2021; Nasiriany et al., 2022). Unlike these approaches, which focus on reusing data or sub-task skills, our approach centers on adapting *policies* from previous tasks to solve novel tasks. These policies encapsulate essential task-solving knowledge in a generative form, making them a valuable starting point for further refinement. Despite having access to a library of policies, identifying the most relevant ones for fine-tuning on new tasks is still an open question, and the success of fine-tuning hinges on making the right selection. In this paper, we introduce **SRSA** (**S**kill **R**etrieval and **S**kill **A**daptation), a novel framework designed to retrieve policies for similar tasks and adapt them to new tasks, as illustrated in Fig. 1. The key contributions of this paper are as follows:

(1) Skill Retrieval Method: We propose a skill retrieval method that simultaneously and explicitly learns embeddings for three fundamental components of assembly tasks: part geometry, interaction dynamics, and expert action choices. We subsequently introduce a novel objective that leverages these embeddings to predict transfer success between any source policy and target task, implicitly capturing additional critical factors for policy transfer. This approach enables the effective retrieval of relevant skills, resulting in higher zero-shot transfer success when applied to new tasks.

(2) Skill Adaptation Method: We propose a skill adaptation method that fine-tunes retrieved skills on new tasks while incorporating a self-imitation learning method (Oh et al., 2018) to enhance performance and stability during fine-tuning. In a simulation-based, dense-reward setting explored in the leading assembly baseline (Tang et al., 2024), SRSA achieves a relative improvement of 19% in success rate with 2.4x faster training and 2.6x lower standard deviation across random seeds. In simulation-based, sparse-reward settings without demonstrations or curricula (closely aligning with real-world fine-tuning scenarios), SRSA outperforms the baseline with a relative improvement of 135% in success rate. Furthermore, we demonstrate that policies fine-tuned in simulation can be directly transferred to real-world robots, achieving a 90% average success rate without the need for additional training. This capability of effectively fine-tuning policies in simulation on novel tasks, and transferring these policies to the real world in zero-shot, highlights the potential for deploying high-performance solutions in real-world assembly tasks.

(3) Continual Learning with SRSA: Instead of training numerous specialist (i.e., single-task) policies from scratch, we propose gradually expanding a small set of initial skills via retrieval and adaptation to cover a broader range of tasks. This strategy improves sample efficiency by over 80% compared to (Tang et al., 2024) and stays consistently efficient as the skill library and target tasks evolve. Thus, SRSA provides an efficient solution for accumulating a large-scale collection of skills.

2 Related Work

Robotic Assembly Tasks Robotic assembly is a critical manufacturing process in the automotive, aerospace, electronics, and medical device industries, but *adaptive* robotic assembly (e.g., robustness to part types, initial part poses, perceptual noise, control error, and environmental perturbations) is largely unsolved. Research (Beltran-Hernandez et al., 2020; Luo et al., 2021; Narang et al., 2022; Tang et al., 2023; Zhang et al., 2023; Noseworthy et al., 2024) on adaptive assembly has seen significant growth in recent years. Despite advancements in datasets and real-world benchmarks for assembling small, realistic parts (Kimble et al., 2020; 2022; Willis et al., 2022; Tian et al.,

2022), the exploration of policy learning across a wide variety of parts remains relatively limited. Many recent efforts in robotic assembly have concentrated on perception (Fu et al., 2022; Wen et al., 2022) or planning (Tian et al., 2022; 2024), rather than learning policies that are robust to disturbances and noise. Additionally, the policy-learning efforts that have addressed the widest range of assemblies have typically been restricted to <30 parts (Spector & Di Castro, 2021; Spector et al., 2022; Zhao et al., 2022). The largest study, AutoMate (Tang et al., 2024), introduced a diverse dataset featuring 100 assembly tasks with simulation environments and 3D-printable parts, and explores policy learning across these tasks. However, its approach primarily focuses on learning specialist (i.e., single-task) policies *from scratch* without leveraging prior experience or knowledge from related tasks. In contrast, our goal is to solve novel assembly tasks by leveraging skills from previously-solved assembly tasks.

Retrieval-based Policy Learning Many studies have explored techniques for utilizing datasets from other tasks for pretraining, such as visual pretraining (Parisi et al., 2022; Nair et al., 2022; Xiao et al., 2022) and multi-task imitation learning (Jang et al., 2022; Ebert et al., 2021; Shridhar et al., 2022). Recently, in robotic manipulation, some works have investigated how to selectively incorporate offline data from other tasks during policy learning, i.e., retrieving prior data according to expert demonstrations on the target task (Nasiriany et al., 2022; Belkhale et al., 2024; Shao et al., 2021; Zha et al., 2024). For instance, Du et al. (2023) selects pertinent state-action pairs based on visual and action similarity from offline, unlabeled datasets and jointly trains a policy using a small amount of expert demonstrations and the queried data via imitation learning. Lin et al. (2024), on the other hand, emphasizes motion similarity rather than semantic similarity by retrieving state-action pairs based on optical flow representations, followed by few-shot imitation learning with expert demonstrations and the retrieved data. Kuang et al. (2024) takes a different approach by extracting a unified affordance representation from diverse data sources and hierarchically retrieving and transferring 2D affordance information based on language instructions to perform zero-shot robotic manipulation. These works primarily study *data retrieval* for general pick-and-place manipulation tasks. (Zhu et al., 2024) introduces a policy retriever for pick-and-place tasks, which selects policy candidates from a memory bank to align closely with the current input, based on the cosine similarity between instruction and observation features. In contrast to these works, we focus on challenging contact-rich manipulation tasks, especially investigating transfer success predictor for *policy retrieval*

Embedding Learning for Task and Skills *Task* embedding learning has been extensively explored in meta-reinforcement learning and multi-task reinforcement learning problems, where shared knowledge across tasks can significantly enhance learning efficiency for new tasks. Most previous approaches focus on capturing task features related to visual appearance in 2D images or dynamics in transitions (James et al., 2018; Rakelly et al., 2019; Lee et al., 2020). Contrastive learning is often employed to bring similar tasks closer together in the embedding space while pushing dissimilar tasks farther apart (James et al., 2018). *Skill* embedding learning, on the other hand, leverages unstructured prior experiences (i.e., temporally extended actions that encapsulate useful behaviors) and repurposes them to solve downstream tasks. Existing methods typically train a high-level policy where the action space consists of the extracted skills (Pertsch et al., 2021; Nasiriany et al., 2022; Hausman et al., 2018; Sharma et al., 2019; Lynch et al., 2020). Although most previous approaches use skills to solve subtasks and combine sequences of skills for long-horizon tasks, we focus on selecting and adapting a single relevant skill for a new task; our tasks of interest are assembly tasks, which are typically short-horizon but difficult to train due to exploration challenges and precise control requirements. Additionally, we integrate multiple embedding-learning approaches by *jointly* capturing three fundamental components of assembly tasks: part geometry, interaction dynamics, and expert actions. We consolidate these perspectives for more robust task representation.

3 Problem Setup

In this work, we consider the problem setting of solving a new target task leveraging pre-existing skills from a skill library. This library contains policies, each designed to solve a specific previously-encountered task. Our approach is motivated by situations (Rusu et al., 2016; Tirinzoni et al., 2019; Huang et al., 2021) where an agent can draw on knowledge from previously-learned policies to adapt quickly to a new task at hand. Similar to the multi-task reinforcement learning (RL) formulation (Borsa et al., 2016; Sodhani et al., 2021; Calandriello et al., 2014), we consider a task space $\mathcal{T}$ where each task $T \in \mathcal{T}$ is defined as a Markov decision process (MDP) $(\mathcal{S}, \mathcal{A}, p, r, \gamma, \rho)$. In this formulation, $\mathcal{S}$ represents the state space, $\mathcal{A}$ the action space, $p(s_{t+1}|s_t, a_t)$ the transition dynamics, $r(s_t, a_t)$ the reward function, $\gamma \in [0, 1)$ the discount factor, and ρ the initial state distribution.

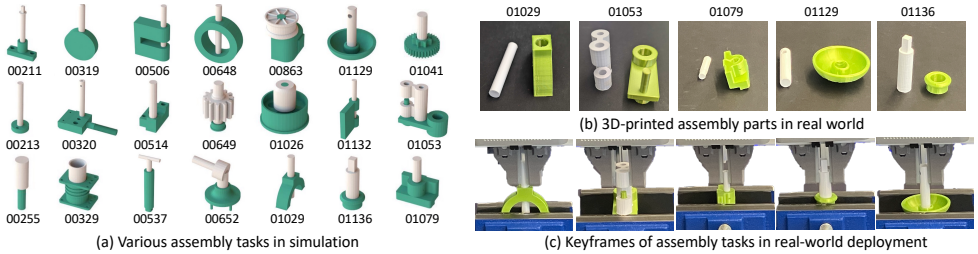


Figure 2: **Illustration of assembly tasks in AutoMate and SRSA**. (a) Samples of assembly tasks in the AutoMate benchmark. (b) 3D-printed parts of corresponding real-world assembly tasks in SRSA. (c) Keyframes from video recordings of our real-world deployments of performant policies.

Our study focuses on two-part assembly tasks, as depicted in Fig. 2. Following the setup of AutoMate (Tang et al., 2024), each environment includes a Franka robot, a *plug* (i.e., a part to be inserted), and a *socket* (i.e., the part that mates with the given plug). In the initial state, we randomize the robot's joint configuration and socket pose, as well as the pose of the plug within the robot's gripper. The goal of each task is to insert a plug into its corresponding socket. (See Appendix A.1)

The state space $\mathcal{S}$ consists of the robot arm's joint angles and velocities, the end-effector pose and its linear/angular velocities, the current plug pose, and the end-effector goal pose. The action space $\mathcal{A}$ consists of incremental pose targets for a task-space impedance controller. As described in (Tang et al., 2024), although assembly trajectories are infeasible to procedurally generate, *disassembly paths* can be easily generated, serving as reverse demonstrations that can be used by an RL agent. Specifically, the RL reward function is composed of terms that penalize the distance to the goal, penalize simulation error, reward task difficulty in a curriculum, and imitate the reversed disassembly paths. The assembly tasks all share the same state space $\mathcal{S}$ and action space $\mathcal{A}$, but vary in part geometries, transition dynamics p, and initial state distribution ρ.

Given a target task $T \in \mathcal{T}$, we assume access to a prior task set $\mathcal{T}_{prior} = \{T_1, T_2, \cdots, T_n\} \subseteq \mathcal{T}$. With policy space $\Pi : \mathcal{S} \to \mathcal{A}$, the *skill library* contains policies $\Pi_{prior} = \{\pi_1, \pi_2, \cdots, \pi_n\} \subseteq \Pi$ that solve each of the prior tasks, respectively. To solve a target task, the goal of RL is to find a policy $\pi(a_t|s_t)$ that produces an action for each state to maximize the expected return. We propose to first retrieve a skill (i.e., policy) for the most relevant prior task (Sec. 4.1), and then rapidly and effectively adapt to the target task by fine-tuning the retrieved skill (Sec. 4.2).

4 Method

4.1 Skill Retrieval

To effectively retrieve skills from Π_{prior} that are useful for a new target task T, we require a means to estimate the potential of applying a source policy $\pi_{src} \in \Pi_{prior}$ to task T. Concretely, we aim to obtain a function $F : \Pi \times \mathcal{T} \to \mathbb{R}$, which takes as input a source policy and a target task and produces a scalar score measuring how well the source policy can be adapted to the target task.

According to the simulation lemma (Agarwal et al., 2019), the difference in expected value when applying the same policy to different tasks partially depends on their difference in transition dynamics and initial state distributions. We execute a source policy π_{src} on both target task T_{trg} and its original source task T_{src}. Let $r_{src,trg}$ denote the zero-shot transfer success of π_{src} on T_{trg} and $r_{src,src}$ its success rate on T_{src}. These success rates reflect the expected value of π_{src} on T_{trg} and T_{src} respectively. Notably, if $r_{src,trg}$ is similar to $r_{src,src}$, it suggests that the transition dynamics and initial state distributions of the two tasks are closely aligned. Since π_{src} is already an expert on T_{src} with a high success rate $r_{src,src}$, a high zero-shot transfer success rate $r_{src,trg}$ indicates strong task similarity. Thus, we use the high transfer success rate as a heuristic indicator of similar dynamics and initial state distributions between source and target tasks. Details are in Appendix A.2.

Subsequently, we hypothesize that fine-tuning a source policy on a target task with similar dynamics will be efficient, as it only requires adaptation to small differences in dynamics. Therefore, we propose using zero-shot transfer success as a metric to gauge the potential of efficiently adapting a source policy on a target task. To identify a source policy with high zero-shot transfer success on a given target task, we propose to learn a function F to predict the zero-shot transfer success for any pair of source policy π_{src} and target task T_{trg}. The prediction $F(\pi_{src}, T_{trg})$ serves as an indicator of whether π_{src} is a strong candidate to initiate fine-tuning for the target task T_{trg}. Below, we describe data collection (Sec. 4.1.1), featurization (Sec. 4.1.2), training (Sec. 4.1.3) and inference(Sec. 4.1.4) for the transfer success predictor F.

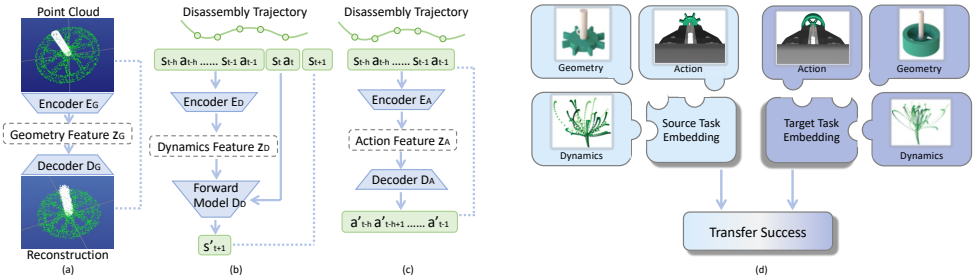


Figure 3: **Illustration of skill retrieval approach.** We decompose skill retrieval into task feature learning(abc) and transfer success prediction(d). (a) Geometry features are learned from point-cloud input using a PointNet autoencoder. (b) Dynamics features are learned from transition segments using a state-prediction objective. (c) Expert-action features are learned from transition segments using an action-reconstruction objective. (d) The zero-shot transfer success rate (of applying a source policy to a target task) is predicted using these task features from source and target tasks.

4.1.1 Dataset Formulation

In order to train the prediction function F, we construct a dataset of tuples $(\pi_{src}, T_{trg}, r_{src,trg})$. We treat any two tasks from the prior task set $\mathcal{T}_{prior}$ as a source-target task pair. For each pair (π_{src}, T_{trg}), we evaluate the source policy π_{src} on the target task T_{trg} to obtain the zero-shot transfer success rate $r_{src,trg}$. In cases where multiple distinct policies exist for the same source task, each solving it in a different manner, policy-specific features would be necessary to capture nuances between different policies. However, in our setting, each policy in the skill library is trained as an expert for a specific source task, with a one-to-one mapping between policies and their corresponding training tasks. Consequently, we use the features of the source task T_{src} as a proxy for representing the source policy π_{src}. This process enables us to collect a training dataset of tuples $(T_{src}, T_{trg}, r_{src,trg})$ from the prior skill library.

4.1.2 Learning Task Features

Given the limited number of (T_{src}, T_{trg}) pairs (specifically, during training, we have $n \times n$ pairs for a total of n tasks in $\mathcal{T}_{prior}$), we need a strong featurization of both the source policy and target task for efficient learning of F. For assembly tasks, each task differs along three fundamental axes: part geometry, interaction dynamics, and expert actions that solve the task. Thus, we propose a framework that jointly captures features of geometry, dynamics, and expert actions to represent the tasks, allowing us to efficiently learn the transfer success predictor F (Fig. 3).

When learning geometry features, we assume access to object meshes for both seen and novel tasks; this assumption is well-grounded in industry, where CAD models are widely available, allowing us to learn embeddings of 3D geometry. However, learning features for dynamics and expert actions poses a unique challenge. For new assembly tasks, we assume that expert demonstrations are *not* available, as these are typically tedious to obtain and often suboptimal for assembly tasks. This deficit prevents us from easily computing dynamics or action embeddings.

We draw insight from (Tian et al., 2022; Tang et al., 2024), which noted that, although procedurally generating assembly demonstrations for new tasks is intractable (narrow-passage problem), *disassembly paths* can be trivially generated by employing a compliant low-level controller to lift an inserted plug from its socket and moving it to a randomized pose. We propose learning features for dynamics and expert actions by using these *disassembly paths* and hypothesize that such features are useful for predicting transfer success for assembly; we later empirically support this hypothesis.

From each task, we randomly sample a certain number of points from the parts' mesh as the point cloud P and and also randomly sample the transition segments τ from disassembly trajectories. Using the point clouds P or transition sequences from disassembly τ, we learn encoders E_G, E_D, and E_A to capture features z_G (representing geometry), z_D (representing forward dynamics), and z_A (representing expert actions). We also train decoders D_G, D_D, and D_A conditioned on these features to predict point cloud for geometry, next state for dynamics, and action sequence for expert action choices. In Appendix A.4, we explain the implementation details for learning these features.

4.1.3 Learning Transfer Success Predictor

We consolidate task features of source T_{src} and target tasks T_{trg} to develop the transfer success predictor F. We feed the sampled point cloud P_{src} and transition segments τ_{src} from T_{src}, and P_{trg}

and τ_{trg} from T_{trg}, into encoders E_G, E_D, and E_A which are pre-trained and frozen. The geometry, dynamics, and expert action features are concatenated together to get task features z_{src} and z_{trg}. We then pass the concatenated task features through an MLP to predict the transfer success $r_{src,trg}$, as illustrated in Fig. 3(d). Formally, we train the function F to minimize the objective function (Eq. 1):

$$\begin{aligned}\mathcal{L} &= \|F(\pi_{src}, T_{trg}) - r_{src,trg}\|^2 = \|MLP(z_{src}, z_{trg}) - r_{src,trg}\|^2 \\ &= \|MLP(E_G(P_{src}), E_D(\tau_{src}), E_A(\tau_{src}), E_G(P_{trg}), E_D(\tau_{trg}), E_A(\tau_{trg})) - r_{src,trg}\|^2 \end{aligned} \quad (1)$$

4.1.4 Inferring Transfer Success for Retrieval

At test time, we use the well-trained function F to predict the transfer success of applying any prior policy to a new task T_{trg} as $F(\pi_{src}, T_{trg})$. As described in Sec. 4.1.2, for each task, we can randomly sample a certain number of points from parts' meshes as point clouds and randomly sample transition segments from disassembly trajectories. For each source and target task pair, we sample the input data for m times and average the output from F to obtain a more robust transfer success prediction. Specifically, we sample point clouds $P_1, P_2, \cdots, P_m$ and transition segments $\tau_1, \tau_2, \cdots, \tau_m$ and then compute the averaged prediction for these samples, i.e. $F(\pi_{src},T_{trg})=\frac{1}{m}\sum_{i=1}^{m} MLP(E_G(P_{src,i}),E_D(\tau_{src,i}),E_A(\tau_{src,i}),E_G(P_{trg,i}),E_D(\tau_{trg,i}),E_A(\tau_{trg,i}))$. In this manner, we infer the predicted transfer success $F(\pi_{src}, T_{trg})$ for any source policies π_{src} in the prior skill library $\Pi_{prior} = \{\pi_1, \pi_2, \cdots, \pi_n\}$.

Although the well-trained F provides transfer success prediction as an effective guidance for retrieval, its predictions may not always be perfectly accurate. To mitigate this, we retrieve the top-k source skills ranked by the predictor F. Among these k candidates, we identify the most relevant skill by evaluating their zero-shot transfer success on the target task, ultimately selecting the skill with the best transfer performance. This technique is grounded in the same intuition as introduced in Sec. 4.1: zero-shot transfer success serves as a reliable metric for skill relevance. In experiments in Sec. 5.2, we set k to 5. Details are in Appendix A.12.

4.2 Skill Adaptation

As mentioned in Sec. 3, our ultimate goal is to solve the new task as an RL problem. The retrieved skill is used to initialize the policy network $\pi_\theta(a_t|s_t)$, and we subsequently use proximal policy optimization (PPO) (Schulman et al., 2017) to fine-tune the policy on the target task. Our initialization provides a strong start for policy learning, as the initial trials with the retrieved skills can achieve a reasonable success rate. Inspired by self-imitation learning (Oh et al., 2018), we fully exploit these positive experiences gained during the initial phase of fine-tuning. We maintain a replay buffer $\mathcal{D} = \{(s_t, a_t, R_t)\}$ to store the transitions encountered throughout training, where $R_t = \sum_{k=t}^{T} \gamma^{k-t} r_k$ is the discounted sum of rewards. We prioritize the state-action pairs (s_t, a_t) based on R_t and imitate those pairs with high rewards. The objective function is defined in Eq. 2:

$$\mathcal{L}^{sil} = \mathbb{E}_{(s,a,R)\in\mathcal{D}}[\mathcal{L}^{sil}_{policy} + \beta\mathcal{L}^{sil}_{value}] \quad (2)$$

where $\mathcal{L}^{sil}_{policy} = -\log \pi_\theta(a|s)(R - V_\psi(s))_+$, $\mathcal{L}^{sil}_{value} = \frac{1}{2}\|(R - V_\psi(s))_+\|^2$, $(\cdot)_+ = \max(\cdot, 0)$, and π_θ and V_ψ are the policy and value function (see details in Appendix A.3).

As training progresses, the agent collects higher rewards on the target task, leading to an expanding replay buffer filled with improved experiences. As analyzed in (Tang, 2020), this self-imitation mechanism accelerates the agent's convergence to encountered high-reward behavior, even though it may introduce some bias into the policy. In our case, the behavior derived from the retrieved skill is advantageous for the target task. We find that self-imitation learning significantly enhances and stabilizes policy fine-tuning, proving especially beneficial in sparse-reward scenarios.

4.3 Continual Learning with Skill-Library Expansion

Continual learning investigates learning various tasks in a sequential fashion. The primary objective is to overcome the forgetting of previously-learned tasks and to leverage earlier knowledge for better performance and/or faster convergence on incoming tasks (Ring, 1994; Xu & Zhu, 2018; Abel et al., 2024). We integrate SRSA in the continual-learning setup and gradually expand the skill library. Specifically, we begin with an initial skill library Π_{prior} corresponding to prior tasks $\mathcal{T}_{prior}$. When faced with a new batch of tasks $T^j = \{T_1, T_2, \cdots, T_k\}$, we apply SRSA to retrieve and fine-tune policies for each new task T_i. The learned policies are then incorporated as $\mathcal{T}_{prior} = \mathcal{T}_{prior} \cup \{T_i\}$; $\Pi_{prior} = \Pi_{prior} \cup \{\pi_i\}$. This approach allows us to efficiently tackle new tasks by leveraging the

skill library, as well as simultaneously prevent the forgetting of all learned tasks by maintaining and expanding the skill library. See Appendix A.3 for the algorithm pseudocode.

5 EXPERIMENTS

We design experiments to answer questions: (1) Compared with baseline retrieval approaches, can SRSA retrieve source policies that achieve a better zero-shot transfer success rate on test tasks? (2) Can policy fine-tuning in SRSA improve learning performance, stability, and efficiency on test tasks? (3) After fine-tuning, can SRSA high-performing policies from simulation be deployed in zero-shot to the real-world? (4) Can SRSA be applied in the continual-learning scenario to improve learning efficiency by gradually expanding a skill library? We investigate these questions on the AutoMate benchmark (Tang et al., 2024), which consists of 100 two-part assembly tasks with diverse parts, enabling us to study challenging contact-rich assembly tasks in simulation and the real world.

5.1 SKILL RETRIEVAL

AutoMate provides meshes and disassembly trajectories for each task. We use these data to learn the task embedding for retrieval. We the following retrieval strategies. **Signature**: retrieve the task with the closest path signature(Barcelos et al., 2024; Chen, 1958; Kidger et al., 2019), which represents disassembly trajectories as a collection of path integrals (Tang et al., 2024). **Behavior**: retrieve the task with the closest VAE embedding of state-action pairs on disassembly trajectories. **Forward**: retrieve the task with the closest latent vector for transition sequence on disassembly trajectories, where the latent vector was trained to to predict forward dynamics. **Geometry**: retrieve the task with the closest PointNet(Qi et al., 2017; Wang et al., 2023) encoding for point clouds of the assembly assets. **SRSA**: retrieve the source task with the highest prediction of transfer success on the target task. Implementation details can be found in Appendix A.4.

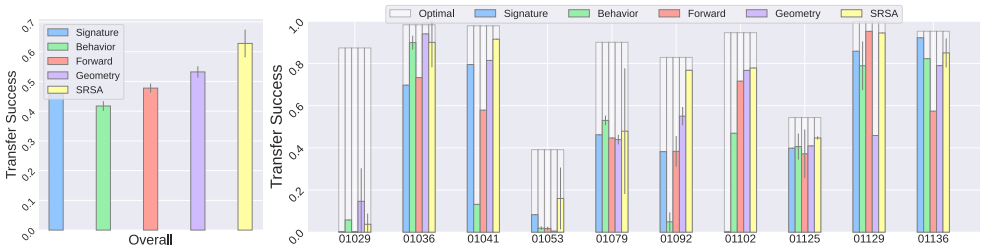


Figure 4: **Zero-shot transfer success of retrieved skills when applied to test tasks**. For each test task, we retrieve a policy from the prior skill library using 5 different approaches (4 baselines and SRSA). If the approach involves training neural networks, we train on 3 random seeds. **Left**: Mean and standard deviation of transfer success rate, averaged over 10 test tasks with 3 seeds each. **Right**: Mean and standard deviation of success rate for each test task, averaged over 3 seeds. Overall, SRSA substantially outperforms baselines. tas

Given the 100 tasks in the AutoMate benchmark, we split the task set into 90 prior tasks (to build the skill library) and 10 test tasks (as the new tasks to solve). For both SRSA and baseline methods, we train the retrieval model over three random seeds and report the average and standard deviation of transfer success over the three seeds. Fig. 4 shows the result on the test task set. SRSA performs best or second-best on all test tasks, except for one very challenging assembly where all methods perform poorly (01029). In Appendix A.5, we show additional comparisons for other splits of prior and test task sets. Overall, SRSA retrieves source policies that obtain around 20% higher success rates on the test tasks, compared with baselines.

5.2 SKILL ADAPTATION

In this section, we investigate policy learning on test tasks given the skill library. We compare AutoMate (i.e., learning specialist policies from scratch (Tang et al., 2024)) and SRSA (i.e., fine-tuning the retrieved specialist policy with self-imitation learning). Details are in Appendix A.4. We consider both the dense-reward setting (identical to AutoMate) with a reward term imitating disassembly demonstrations and a curriculum, and the sparse-reward setting, which only provides a non-zero reward for task success. The sparse-reward setting is designed to emulate the real-world RL fine-tuning setting, where dense-reward information can be much more challenging to acquire.

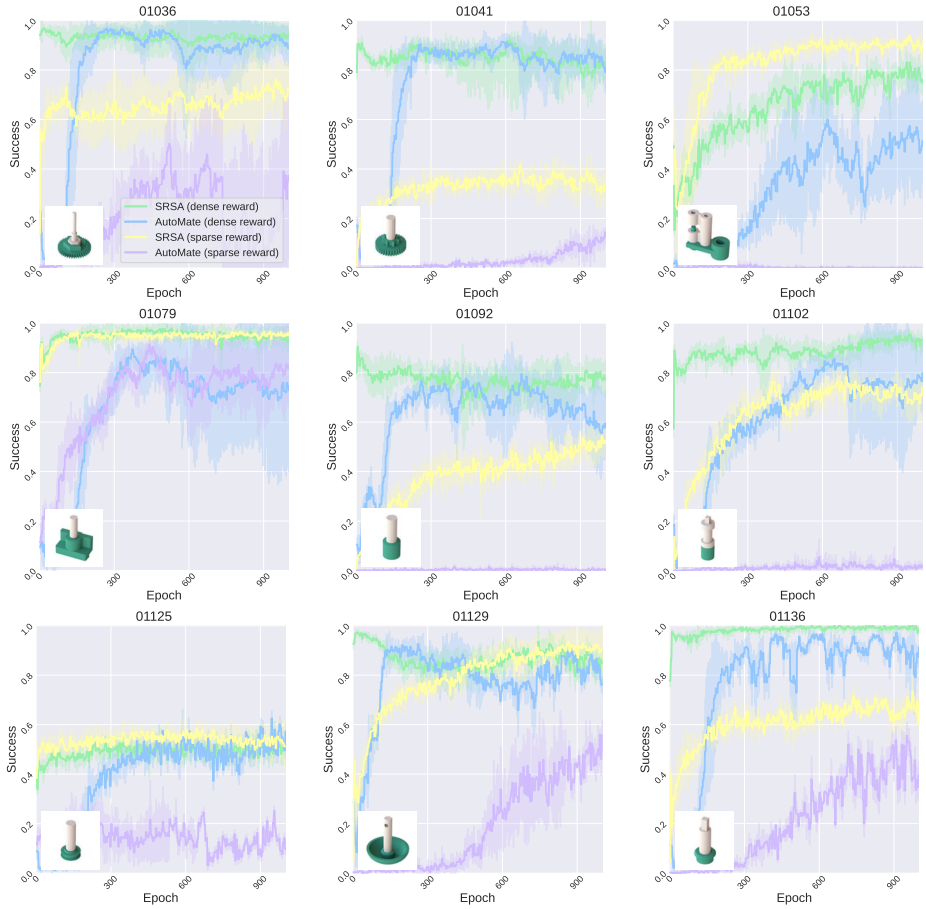


Figure 5: **Learning curves on test tasks.** The x-axis and y-axis represent training epochs (where each epoch consists of 128 environment steps over 256 parallel environments) and success rate, respectively. The solid line shows the mean success rate over 5 runs with different random seeds, and the shaded area denotes the standard deviation.

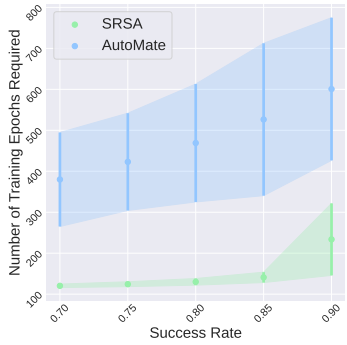


Figure 6: **Sample efficiency on test set**. To achieve a desired success rate (here, 0.70, 0.75, 0.80, 0.85, or 0.90), we identify how many training epochs are required for each run. We illustrate the mean and standard deviation of required epochs across 5 runs with the points and error bars in the figure, averaged over 10 test tasks.

Fig. 5 shows learning curves on the test task set. In the dense-reward setting, SRSA achieves strong performance with a fewer number of training epochs. In the sparse-reward setting, AutoMate struggles to achieve a reasonable success rate, whereas SRSA benefits from the retrieved skill initialization and self-imitation learning to reach higher performance. Additionally, in both settings, the learning curves of AutoMate exhibit instability with fluctuating success rates as training goes on. Tab. 2 and Tab. 3 in Appendix A.5 summarize the mean and standard deviation of the success rate at the last epoch of training, across 5 random seeds, for each test task. In the dense-reward setting, SRSA reaches an average success rate of 82.6% across 10 test tasks, outperforming AutoMate (69.4%), corresponding to a relative improvement of 19% in performance. Moreover, SRSA shows greater stability, as AutoMate exhibits a 2.6x higher standard deviation. In the sparse-reward setting, SRSA delivers a notable 135% relative improvement in average success rate compared to the baseline. Fig. 6 demonstrates the number of training epochs required to reach a desired success rate in the dense-reward setting. Averaged over 10 test tasks and 5 random seeds, SRSA requires far fewer training samples, i.e., at least 2.4 times fewer training epochs, to achieve an arbitrary success threshold.

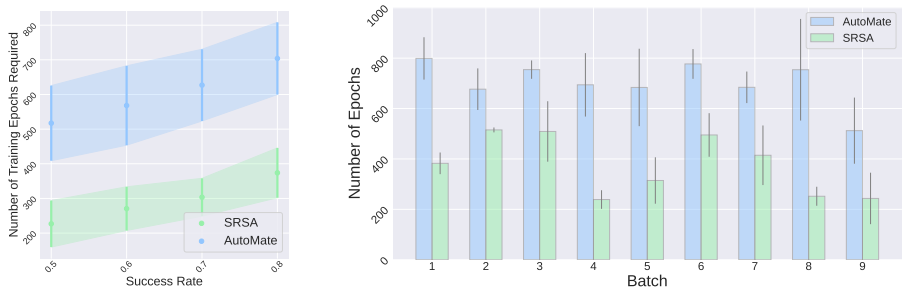
5.3 Real-World Deployment

We now deploy the trained specialist policies in the real world. As in (Tang et al., 2024), we place the robot in lead-through (a.k.a., manual guide mode), grasp a plug, guide it into the socket, and record the pose as a target pose. We then programmatically lift the plug until free from contact, apply perturbations to the position and rotation of the end effector, and deploy a policy to assemble the plug into the socket. Such conditions emulate the perceptual noise and control error that are experienced in full robotic assembly pipelines. In Tab. 7, we take the best checkpoint over 500 training epochs in simulation, and record its performance when deployed in the real world. In this relatively-brief training time, SRSA reaches higher success rates than the baseline on real-world assembly tasks. We show keyframes of the real-world deployments in Fig. 2(c). For videos, please refer to the project website https://srsa2024.github.io/.

Asset ID	01029	01053	01079	01129	01136	Overall
AutoMate	7/10	1/10	7/10	4/10	8/10	54%
SRSA	9/10	8/10	8/10	10/10	10/10	90%

Figure 7: **Real-world evaluation.** We take the best checkpoint of policies across 5 runs within 500 epochs and report the success rate over 10 trials for each task.

5.4 Continual Learning



(a) (b)

Figure 8: **(a) Overall sample efficiency**. We report the number of training epochs required to reach desired success rates (0.5, 0.6, 0.7, 0.8) on all tasks. We calculate the mean and standard deviation of required training epochs over 5 runs, and report the average over 90 tasks. **(b) Sample efficiency in batches**. We sequentially introduce 9 batches of new tasks for policy learning, with each batch containing 10 new tasks. For each batch, we show the mean and standard deviation of training epochs required to reach a success rate of 0.8. SRSA consistently requires fewer training epochs.

We study the continual-learning setting to obtain policies for each of the 100 AutoMate tasks. Rather than training 100 policies from scratch in parallel, we start from a skill library with 10 tasks, and train 10 new policies for 10 new tasks utilizing the skill library. For each new task, we fine-tune the retrieved policy over 5 runs with different random seeds. We pick the best checkpoint with highest success rate over 5 runs as the specialist policy for this new task. We repeat this process for 9 iterations, eventually covering the entire AutoMate benchmark. Essentially, we have a skill library that is gradually expanded with an increasing number of specialist policies.

In Fig. 8, we compare the sample efficiency of SRSA and AutoMate when learning specialist policies for the 90 tasks outside the initial skill library. We consider different desired success rates, and report the number of training epochs required to reach each success rate. Overall, SRSA requires fewer training epochs to reach the desired success rate, demonstrating an 84% relative improvement in sample efficiency (Fig. 8(a)). For each batch of new tasks, SRSA is more efficient than the baseline regardless of the skill library and test tasks (Fig. 8(b)). In Fig. 14, we show the success rates for the highest-reward checkpoints encountered in 5 runs for each task. SRSA achieves an average success rate of 79% compared to AutoMate's 70% across 100 tasks, while also exhibiting better training efficiency. In Appendix A.5, we present learning results for another ordering of batches of tasks, showing that the advantage of SRSA is insensitive to the order of encountering new tasks.

6 Ablation Study

Effect of Skill Retrieval To verify the effect of skill retrieval, we conduct skill adaptation with retrieved skills using only a geometry embedding, i.e., the second best skill-retrieval approach evaluated in Fig. 3. Fig. 9 shows the performance of policy fine-tuning for both SRSA and the geometry-

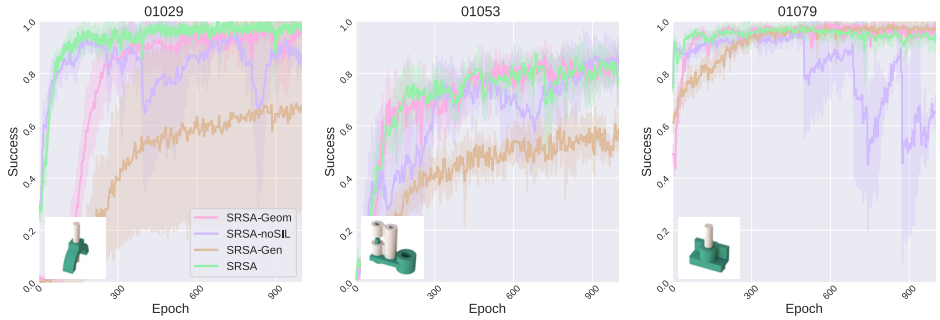


Figure 9: **Comparison across variants of SRSA**. For each method, we train with 5 different random seeds. The learning curves show mean and standard deviation of success rate over these seeds. We show learning curves for more tasks in Appendix A.5.

based skill retrieval (SRSA-Geom). One can observe that retrieving a worse skill hinders learning efficiency, starting from a lower success rate and requiring more training epochs to reach high performance. Our retrieval approach improves adaptation efficiency.

Effect of Self-imitation Learning To demonstrate the benefits of self-imitation learning (SIL) in policy fine-tuning, we compare SRSA to the variant without this component (SRSA-noSIL). In Fig. 9, SRSA outperforms the variant in terms of learning stability. In particular, SRSA-noSIL suffers from more fluctuations during fine-tuning and a larger standard deviation of success rate (shaded area) across runs with different seeds.

Effect of Generalist Policy We analyze whether fine-tuning a generalist policy outperforms fine-tuning a selected specialist policy. For policy initialization, we use the generalist policy for 20 training tasks from (Tang et al., 2024) Although it does not cover numerous tasks, it is the strongest generalist policy reported to date that can solve a diverse set of assembly tasks with an $> 80\%$ success rate. Fig. 9 shows the learning curves of fine-tuning the generalist policy on unseen tasks (SRSA-Gen). We observe that SRSA-Gen provides a weaker initialization compared to SRSA, likely because the generalist policy's knowledge from the training tasks is less specialized than the skills retrieved by SRSA. Furthermore, adaptation is less efficient, possibly due to the larger neural network in the generalist policy, which requires more fine-tuning to adapt to new tasks. As a result, its asymptotic performance is also lower than that of SRSA.

7 Conclusion

Summary: In this work, we propose a pipeline to retrieve and adapt specialist policies to solve new assembly tasks. To learn a retrieval model, we jointly learn features from geometry, dynamics and expert actions to represent tasks, and predict transfer success to implicitly capture other transfer-related factors from tasks. By combining skill retrieval with policy fine-tuning and self-imitation learning, our method efficiently learns high-performance simulation-based policies. We demonstrate that these policies are transferable to real-world robots Additionally, we demonstrate that our approach can continuously expand a skill library through efficient learning of various skills.

Limitations: First, although we train policies for all assembly tasks in a leading benchmark (Tang et al., 2024), we do not address assemblies requiring rotational or helical motion (e.g., nut-and-bolt assembly). Second, we primarily concentrate on learning specialist (i.e., single-task) policies; future work could explore learning generalist (i.e., multi-task) policies, and furthermore, incorporating knowledge from both specialist and generalist policies to solve novel tasks with even greater efficiency. Third, although our real-world success rates outperform the state-of-the-art in sim-to-real transfer for our examined tasks, they still fall short of the 99+% success rates required for industry-level deployment. We believe that RL fine-tuning directly in real-world settings could help bridge the sim-to-real gap and further improve success rates.

Future Extensions: How to utilize existing policies for new tasks (rather than training from scratch) is an open and general question in robotics. This question is relevant not just for insertion tasks, but also for general pick-and-place tasks, dexterous manipulation tasks, advanced assembly tasks, etc. Most robotics tasks are governed by geometry, dynamics, and behavior/action. We believe our ideas of learning task features and predicting zero-shot transfer success for policy transfer can generalize to other domains. For instance, in tool-use tasks, the skill of using scissors may be beneficial for learning to operate pliers due to their similar shape and operation mechanism. We leave it as future work to extend SRSA to these additional robotics applications.

REFERENCES

David Abel, André Barreto, Benjamin Van Roy, Doina Precup, Hado P van Hasselt, and Satinder Singh. A definition of continual reinforcement learning. *Advances in Neural Information Processing Systems*, 36, 2024.

Alekh Agarwal, Nan Jiang, Sham M Kakade, and Wen Sun. Reinforcement learning: Theory and algorithms. *CS Dept., UW Seattle, Seattle, WA, USA, Tech. Rep*, 32:96, 2019.

Lucas Barcelos, Tin Lai, Rafael Oliveira, Paulo Borges, and Fabio Ramos. Path signatures for diversity in probabilistic trajectory optimisation. *The International Journal of Robotics Research*, 43(11):1693–1710, 2024.

Suneel Belkhale, Tianli Ding, Ted Xiao, Pierre Sermanet, Quon Vuong, Jonathan Tompson, Yevgen Chebotar, Debidatta Dwibedi, and Dorsa Sadigh. Rt-h: Action hierarchies using language. *arXiv preprint arXiv:2403.01823*, 2024.

Cristian C. Beltran-Hernandez, Damien Petit, Ixchel G. Ramirez-Alpizar, and Kensuke Harada. Variable compliance control for robotic peg-in-hole assembly: A deep-reinforcement-learning approach. *Applied Sciences*, 2020.

Diana Borsa, Thore Graepel, and John Shawe-Taylor. Learning shared representations in multi-task reinforcement learning. *arXiv preprint arXiv:1603.02041*, 2016.

Daniele Calandriello, Alessandro Lazaric, and Marcello Restelli. Sparse multi-task reinforcement learning. *Advances in neural information processing systems*, 27, 2014.

Kuo-Tsai Chen. Integration of paths–a faithful representation of paths by noncommutative formal power series. *Transactions of the American Mathematical Society*, 89(2):395–407, 1958.

Maximilian Du, Suraj Nair, Dorsa Sadigh, and Chelsea Finn. Behavior retrieval: Few-shot imitation learning by querying unlabeled datasets. *arXiv preprint arXiv:2304.08742*, 2023.

Frederik Ebert, Yanlai Yang, Karl Schmeckpeper, Bernadette Bucher, Georgios Georgakis, Kostas Daniilidis, Chelsea Finn, and Sergey Levine. Bridge data: Boosting generalization of robotic skills with cross-domain datasets. *arXiv preprint arXiv:2109.13396*, 2021.

Bowen Fu, Sek Kun Leong, Xiaocong Lian, and Xiangyang Ji. 6d robotic assembly based on rgb-only object pose estimation. In *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 4736–4742. IEEE, 2022.

Ankit Goyal, Valts Blukis, Jie Xu, Yijie Guo, Yu-Wei Chao, and Dieter Fox. Rvt-2: Learning precise manipulation from few demonstrations. *arXiv preprint arXiv:2406.08545*, 2024.

Yijie Guo, Qiucheng Wu, and Honglak Lee. Learning action translator for meta reinforcement learning on sparse-reward tasks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pp. 6792–6800, 2022.

Karol Hausman, Jost Tobias Springenberg, Ziyu Wang, Nicolas Heess, and Martin Riedmiller. Learning an embedding space for transferable robot skills. In *International Conference on Learning Representations*, 2018.

Biwei Huang, Fan Feng, Chaochao Lu, Sara Magliacane, and Kun Zhang. Adarl: What, where, and how to adapt in transfer reinforcement learning. *arXiv preprint arXiv:2107.02729*, 2021.

Stephen James, Michael Bloesch, and Andrew J Davison. Task-embedded control networks for few-shot imitation learning. In *Conference on robot learning*, pp. 783–795. PMLR, 2018.

Eric Jang, Alex Irpan, Mohi Khansari, Daniel Kappler, Frederik Ebert, Corey Lynch, Sergey Levine, and Chelsea Finn. Bc-z: Zero-shot task generalization with robotic imitation learning. In *Conference on Robot Learning*, pp. 991–1002. PMLR, 2022.

Patrick Kidger, Patric Bonnier, Imanol Perez Arribas, Cristopher Salvi, and Terry Lyons. Deep signature transforms. *Advances in neural information processing systems*, 32, 2019.

Kenneth Kimble, Karl Van Wyk, Joe Falco, Elena Messina, Yu Sun, Mizuho Shibata, Wataru Uemura, and Yasuyoshi Yokokohji. Benchmarking protocols for evaluating small parts robotic assembly systems. *IEEE robotics and automation letters*, 5(2):883–889, 2020.

Kenneth Kimble, Justin Albrecht, Megan Zimmerman, and Joe Falco. Performance measures to benchmark the grasping, manipulation, and assembly of deformable objects typical to manufacturing applications. *Frontiers in Robotics and AI*, 9:999348, 2022.

Yuxuan Kuang, Junjie Ye, Haoran Geng, Jiageng Mao, Congyue Deng, Leonidas Guibas, He Wang, and Yue Wang. Ram: Retrieval-based affordance transfer for generalizable zero-shot robotic manipulation. *arXiv preprint arXiv:2407.04689*, 2024.

Kimin Lee, Younggyo Seo, Seunghyun Lee, Honglak Lee, and Jinwoo Shin. Context-aware dynamics model for generalization in model-based reinforcement learning. In *International Conference on Machine Learning*, pp. 5757–5766. PMLR, 2020.

Li-Heng Lin, Yuchen Cui, Amber Xie, Tianyu Hua, and Dorsa Sadigh. Flowretrieval: Flow-guided data retrieval for few-shot imitation learning. *arXiv preprint arXiv:2408.16944*, 2024.

Jianlan Luo, Oleg Sushkov, Rugile Pevceviciute, Wenzhao Lian, Chang Su, Mel Vecerik, Ning Ye, Stefan Schaal, and Jon Scholz. Robust multi-modal policies for industrial assembly via reinforcement learning and demonstrations: A large-scale study. *arXiv preprint arXiv:2103.11512*, 2021.

Corey Lynch, Mohi Khansari, Ted Xiao, Vikash Kumar, Jonathan Tompson, Sergey Levine, and Pierre Sermanet. Learning latent plans from play. In *Conference on robot learning*, pp. 1113–1132. PMLR, 2020.

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 preprint arXiv:2107.14483*, 2021.

Suraj Nair, Aravind Rajeswaran, Vikash Kumar, Chelsea Finn, and Abhinav Gupta. R3m: A universal visual representation for robot manipulation. *arXiv preprint arXiv:2203.12601*, 2022.

Yashraj Narang, Kier Storey, Iretiayo Akinola, Miles Macklin, Philipp Reist, Lukasz Wawrzyniak, Yunrong Guo, Adam Moravanszky, Gavriel State, Michelle Lu, et al. Factory: Fast contact for robotic assembly. *arXiv preprint arXiv:2205.03532*, 2022.

Soroush Nasiriany, Tian Gao, Ajay Mandlekar, and Yuke Zhu. Learning and retrieval from prior data for skill-based imitation learning. *arXiv preprint arXiv:2210.11435*, 2022.

Michael Noseworthy, Bingjie Tang, Bowen Wen, Ankur Handa, Nicholas Roy, Dieter Fox, Fabio Ramos, Yashraj Narang, and Iretiayo Akinola. Forge: Force-guided exploration for robust contact-rich manipulation under uncertainty. *arXiv preprint arXiv:2408.04587*, 2024.

Junhyuk Oh, Yijie Guo, Satinder Singh, and Honglak Lee. Self-imitation learning. In *International conference on machine learning*, pp. 3878–3887. PMLR, 2018.

Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, et al. Dinov2: Learning robust visual features without supervision. *arXiv preprint arXiv:2304.07193*, 2023.

Simone Parisi, Aravind Rajeswaran, Senthil Purushwalkam, and Abhinav Gupta. The unsurprising effectiveness of pre-trained vision models for control. In *international conference on machine learning*, pp. 17359–17371. PMLR, 2022.

Karl Pertsch, Youngwoon Lee, and Joseph Lim. Accelerating reinforcement learning with learned skill priors. In *Conference on robot learning*, pp. 188–204. PMLR, 2021.

Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 652–660, 2017.

Kate Rakelly, Aurick Zhou, Chelsea Finn, Sergey Levine, and Deirdre Quillen. Efficient off-policy meta-reinforcement learning via probabilistic context variables. In *International conference on machine learning*, pp. 5331–5340. PMLR, 2019.

Mark Bishop Ring. *Continual learning in reinforcement environments*. The University of Texas at Austin, 1994.

Andrei A Rusu, Neil C Rabinowitz, Guillaume Desjardins, Hubert Soyer, James Kirkpatrick, Koray Kavukcuoglu, Razvan Pascanu, and Raia Hadsell. Progressive neural networks. *arXiv preprint arXiv:1606.04671*, 2016.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

Lin Shao, Toki Migimatsu, Qiang Zhang, Karen Yang, and Jeannette Bohg. Concept2robot: Learning manipulation concepts from instructions and human demonstrations. *The International Journal of Robotics Research*, 40(12-14):1419–1434, 2021.

Archit Sharma, Shixiang Gu, Sergey Levine, Vikash Kumar, and Karol Hausman. Dynamics-aware unsupervised discovery of skills. *arXiv preprint arXiv:1907.01657*, 2019.

Mohit Shridhar, Lucas Manuelli, and Dieter Fox. Cliport: What and where pathways for robotic manipulation. In *Conference on robot learning*, pp. 894–906. PMLR, 2022.

Shagun Sodhani, Amy Zhang, and Joelle Pineau. Multi-task reinforcement learning with context-based representations. In *International Conference on Machine Learning*, pp. 9767–9779. PMLR, 2021.

Oren Spector and Dotan Di Castro. Insertionnet-a scalable solution for insertion. *IEEE Robotics and Automation Letters*, 6(3):5509–5516, 2021.

Oren Spector, Vladimir Tchuiev, and Dotan Di Castro. Insertionnet 2.0: Minimal contact multi-step insertion using multimodal multiview sensory input. In *2022 International Conference on Robotics and Automation (ICRA)*, pp. 6330–6336. IEEE, 2022.

Bingjie Tang, Michael A Lin, Iretiayo Akinola, Ankur Handa, Gaurav S Sukhatme, Fabio Ramos, Dieter Fox, and Yashraj Narang. Industreal: Transferring contact-rich assembly tasks from simulation to reality. *arXiv preprint arXiv:2305.17110*, 2023.

Bingjie Tang, Iretiayo Akinola, Jie Xu, Bowen Wen, Ankur Handa, Karl Van Wyk, Dieter Fox, Gaurav S Sukhatme, Fabio Ramos, and Yashraj Narang. Automate: Specialist and generalist assembly policies over diverse geometries. *arXiv preprint arXiv:2407.08028*, 2024.

Yunhao Tang. Self-imitation learning via generalized lower bound q-learning. *Advances in neural information processing systems*, 33:13964–13975, 2020.

Yunsheng Tian, Jie Xu, Yichen Li, Jieliang Luo, Shinjiro Sueda, Hui Li, Karl DD Willis, and Wojciech Matusik. Assemble them all: Physics-based planning for generalizable assembly by disassembly. *ACM Transactions on Graphics (TOG)*, 41(6):1–11, 2022.

Yunsheng Tian, Karl DD Willis, Bassel Al Omari, Jieliang Luo, Pingchuan Ma, Yichen Li, Farhad Javid, Edward Gu, Joshua Jacob, Shinjiro Sueda, et al. Asap: automated sequence planning for complex robotic assembly with physical feasibility. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 4380–4386. IEEE, 2024.

Andrea Tirinzoni, Mattia Salvini, and Marcello Restelli. Transfer of samples in policy search via multiple importance sampling. In *International Conference on Machine Learning*, pp. 6264–6274. PMLR, 2019.

Weikang Wan, Haoran Geng, Yun Liu, Zikang Shan, Yaodong Yang, Li Yi, and He Wang. Unidexgrasp++: Improving dexterous grasping policy learning via geometry-aware curriculum and iterative generalist-specialist learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 3891–3902, 2023.

Ruicheng Wang, Jialiang Zhang, Jiayi Chen, Yinzhen Xu, Puhao Li, Tengyu Liu, and He Wang. Dexgraspnet: A large-scale robotic dexterous grasp dataset for general objects based on simulation. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 11359–11366. IEEE, 2023.

Bowen Wen, Wenzhao Lian, Kostas Bekris, and Stefan Schaal. You only demonstrate once: Category-level manipulation from single visual demonstration. *arXiv preprint arXiv:2201.12716*, 2022.

Karl DD Willis, Pradeep Kumar Jayaraman, Hang Chu, Yunsheng Tian, Yifei Li, Daniele Grandi, Aditya Sanghi, Linh Tran, Joseph G Lambourne, Armando Solar-Lezama, et al. Joinable: Learning bottom-up assembly of parametric cad joints. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 15849–15860, 2022.

Tete Xiao, Ilija Radosavovic, Trevor Darrell, and Jitendra Malik. Masked visual pre-training for motor control. *arXiv preprint arXiv:2203.06173*, 2022.

Ju Xu and Zhanxing Zhu. Reinforced continual learning. *Advances in neural information processing systems*, 31, 2018.

Kuo-Hao Zeng, Zichen Zhang, Kiana Ehsani, Rose Hendrix, Jordi Salvador, Alvaro Herrasti, Ross Girshick, Aniruddha Kembhavi, and Luca Weihs. Poliformer: Scaling on-policy rl with transformers results in masterful navigators. *arXiv preprint arXiv:2406.20083*, 2024.

Lihan Zha, Yuchen Cui, Li-Heng Lin, Minae Kwon, Montserrat Gonzalez Arenas, Andy Zeng, Fei Xia, and Dorsa Sadigh. Distilling and retrieving generalizable knowledge for robot manipulation via language corrections. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 15172–15179. IEEE, 2024.

Xiang Zhang, Changhao Wang, Lingfeng Sun, Zheng Wu, Xinghao Zhu, and Masayoshi Tomizuka. Efficient sim-to-real transfer of contact-rich manipulation skills with online admittance residual learning. In *Conference on Robot Learning*, pp. 1621–1639. PMLR, 2023.

Tony Z Zhao, Jianlan Luo, Oleg Sushkov, Rugile Pevceviciute, Nicolas Heess, Jon Scholz, Stefan Schaal, and Sergey Levine. Offline meta-reinforcement learning for industrial insertion. In *2022 international conference on robotics and automation (ICRA)*, pp. 6386–6393. IEEE, 2022.

Yichen Zhu, Zhicai Ou, Xiaofeng Mou, and Jian Tang. Retrieval-augmented embodied agents. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 17985–17995, 2024.