SkillGen: Automated Demonstration Generation for Efficient Skill Learning and Deployment

Anonymous Author(s)
Affiliation
Address
email

Abstract: Imitation learning from human demonstrations is an effective paradigm for robot manipulation, but acquiring large datasets is costly and resource-intensive, especially for long-horizon tasks. To address this issue, we propose SkillGen, an automated system for generating demonstration datasets from a few human demos. SkillGen segments human demos into manipulation skills, adapts these skills to new contexts, and stitches them together through free-space transit and transfer motion. We also propose a Hybrid Skill Policy (HSP) framework for learning skill initiation, control, and termination components from SkillGen datasets, enabling skills to be sequenced using motion planning at test-time. We demonstrate that SkillGen greatly improves data generation and policy learning performance over a state-of-the-art data generation framework, resulting in the capability to produce data for large scene variations, including clutter, and agents that are on average 24% more successful. We demonstrate the efficacy of SkillGen by generating over 24K demonstrations across 18 task variants in simulation from just 60 human demonstrations, and training proficient, often near-perfect, HSP agents. Finally, we apply SkillGen to 3 real-world manipulation tasks and also demonstrate zero-shot sim-to-real transfer on a long-horizon assembly task. Videos, and more at `https://skillgen.github.io`.

Keywords: Imitation Learning, Manipulation, Planning

1 Introduction

Imitation learning from human demonstrations is an effective approach for training robots to perform different tasks [1, 2]. One popular technique is to have humans teleoperate robot arms to collect datasets for tasks of interest and then subsequently use the data to train robots to perform these tasks autonomously [3,4]. Recent efforts have demonstrated that large, diverse datasets collected by teams of human demonstrators result in impressive and robust robot performance, and even allow the robots to generalize to different objects and tasks [2, 5–8]. However, collecting large datasets in this way is costly and resource-intensive, often requiring multiple human operators, robots, and months of human effort. Acquiring datasets for challenging long-horizon tasks that require sequencing several manipulation behaviors together is even more difficult and costly [9].

The need for large datasets has motivated the development of data generation systems [10–12] that seek to produce task demonstrations with minimal human involvement. For example, some systems combine teleoperation and planning within the same demonstration, partially automating the demonstrating process, which ultimately allows a human to teleoperate several robots in parallel [13]. Alternatively, some systems further reduce human involvement through demonstration adaptation. For example, MimicGen [11], uses a small number of human task demonstrations to automatically generate large datasets by splitting the source human data into object-centric sequences of end-effector targets, and then selectively transforming and sequencing such segments in new settings. However, this and other naive strategies for composing human segments together can produce lower-quality demonstrations with unintended collisions in the environment, and have heterogeneous motions that are difficult for policy learning algorithms to learn from, especially in real-world settings.

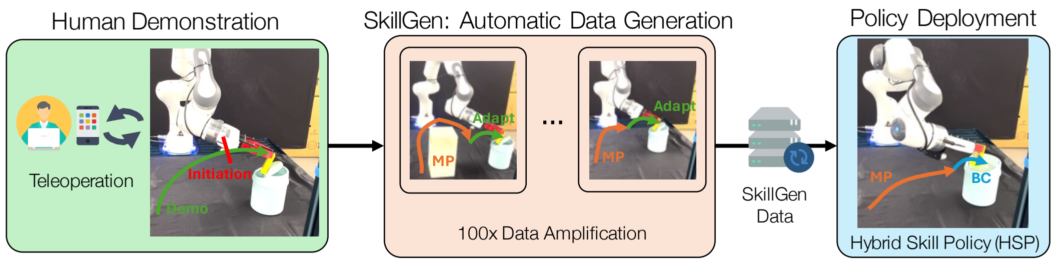


Figure 1: **SkillGen Overview.** SkillGen trains proficient agents with minimal human effort. (*left*) First, a human teleoperator first collects $\sim$ 3 demonstrations of the task and annotates the start and end of the skill segments, where each object interaction happens. (*middle*) Then, SkillGen automatically adapts these local skill demonstrations to new scenes and connects them through motion planning to amplify the number of successful demonstrations. (*right*) These demonstrations are used to train Hybrid Skill Policies (HSP), agents that alternate between closed-loop reactive skills and coarse transit motions carried out by motion planning.

We also seek to minimize the number of required human demonstrations but improve the flexibility and efficacy of adapted demonstrations. To that end, we first observe that control difficulty is often not uniformly spread across a task. Specifically, in order to solve many manipulation tasks, the robot must first move itself in free space in order to reach a state where it can manipulate the world through contact. For example, consider the cleanup task in Fig. 1. The robot must move through free space before picking the butter and also before inserting the butter into the trash can. This kind of free space motion can be easy for planning systems, and greatly reduce the burden on policy learning.

From this observation, we propose SkillGen, a system that leverages the notion of a manipulation *skill* to isolate demonstration adaptation to just contact-rich segments. At data-generation time, SkillGen synthesizes candidate demonstrations by executing several adapted skill segments in sequence, connected through motion planning. At test-time, SkillGen not only learns control policies for these skills but also initiation and termination conditions, enabling them to be sequenced using planning in a similar manner but without any requirements regarding state observability.

We make the following contributions:
• We introduce SkillGen, an automated system for generating demonstration datasets through decomposing tasks into motion segments and skill segments that are adapted from a few human demos.
• We propose a Hybrid Skill Policy (HSP) framework that learns skill initiation, control, and termination components, enabling skills to combined in sequence at at test time using motion planning.
• We show that SkillGen improves data generation and policy learning performance over an existing state-of-the-art data generation framework. Specifically, SkillGen is robust to large scene variation, such as clutter, and produces policies that on average are 24% more successful than MimicGen [11].
• We demonstrate the efficacy of SkillGen by generating 24K+ demonstrations from 60 human demonstrations across 18 task variants in simulation and training proficient, often near-perfect, high-performing HSP agents. Finally, we successfully apply SkillGen to 3 real-world manipulation tasks, and also demonstrate zero-shot sim-to-real transfer on a long-horizon assembly task.

2 Related Work

Data Collection for Robotics. Robot teleoperation [3, 4, 14–23] is a popular method for collecting task demonstrations – here, humans use a teleoperation device to control a robot and guide it through tasks. The robot sensor streams and control actions during operation are logged to a dataset. Several efforts [2, 5–8] have scaled this paradigm up by using a large number of human operators and robot arms over extended periods of time (*e.g.* months). Some works have also allowed for robot-free data collection with specialized hardware [24, 25], but human effort is still required for data collection. In contrast, SkillGen automatically generates data with just a handful of human demonstrations. Other works seek to generate datasets automatically using pre-programmed demonstrators in simulation [10, 26–31], but scaling these approaches to a larger variety of tasks can be difficult.

Imitation Learning and Data Augmentation. Behavioral Cloning (BC) [32] is a typical method for learning policies offline from demonstrations, and has been widely used in robot manipulation [3, 16, 27, 33–45]. Some works leverage offline data augmentation to increase the size of the training dataset for learning policies [1, 46–57]. Instead, SkillGen collects new datasets online.

Imitation Learning with Hybrid Controllers. SayCan [6] composes skills learned from demonstrations using a language model and learns when to begin and end each skill However, each skill starts when the previous one ends – in contrast, our learned skills are local manipulation behaviors and transit is carried out via motion planning. Other works [58–60] learn "keyframe" pose actions from demonstrations and execute them using motion planning, but they lack closed-loop control using learned policies. Some imitation learning methods decompose learning into coarse-grained and fine-grained motions [13, 61–64], but most use naive linear interpolation to carry out coarse-grained motions [61, 62], which is susceptible to collisions. Others [63–65] learn open-loop segments for fine-grained motions, instead of closed-loop skills like our methods. Wang *et al.* [66] learn parametric skills using Gaussian Processes and deploy them in a Task and Motion Planning (TAMP) [67] system. In HITL-TAMP [13], a TAMP planner decides when to employ an agent trained with imitation learning for skill segments; however, it is *TAMP-gated*, meaning that skill start and end conditions are engineering into the TAMP model instead of learned.

MimicGen. MimicGen [11] is a data generation system that takes a small source set of human demonstrations on a task and generates larger sets of demonstrations. It builds on replay-based imitation learning methods [65, 68–74], which address new task instances by adapting and replaying motion from existing human data. MimicGen segments the source demonstrations into a contiguous set of object-centric subtask segments. Then, given a new task instance, MimicGen transforms and replays open-loop subtask segments from the source data one-by-one to generate a new demonstration. However, because MimicGen naively stitches source demonstrations with linear interpolation, it can produce lower quality demonstrations that collide with the environment, and have heterogeneous motions difficult for policy learning. By instead adopting a skill-based framework, SkillGen avoids these pitfalls at data generation time and produces more robust behavior at deployment time.

3 Prerequisites

Imitation Learning. Each robot manipulation task is modeled as a Partially Observable Markov Decision Process (POMDP). We are given a dataset of N demonstrations $\mathcal{D} = \{(s_0^i, o_0^i, a_0^i, s_1^i, o_1^i, a_1^i, ..., s_{H_i}^i)\}_{i=1}^N$ consisting of states $s \in \mathcal{S}$, observations $o \in \mathcal{O}$, and actions $a \in \mathcal{A}$. Each initial state $s_0^i \sim D$ is sampled from the initial state distribution $D \subseteq \mathcal{S}$. We aim to learn a robot control policy $\pi : \mathcal{O} \to \mathcal{A}$ that maps observation space $\mathcal{O}$ to a distribution over action space $\mathcal{A}$. Behavioral Cloning (BC) [32] is a common method to obtain such a policy – it uses optimization to find a policy that maximizes the likelihood of producing the data $\arg\max_\theta \mathbb{E}_{(s,o,a)\sim\mathcal{D}}[\log \pi_\theta(a \mid o)]$. In this work, we train policies via BC and combine them with various mechanisms to exchange control between a learned policy and a motion planner.

Assumptions. Similar to prior work [11], we make the following assumptions. **(A1):** The policy action space $\mathcal{A}$ consists of continuous pose commands for an end effector controller along with a discrete gripper command. This allows us to treat the actions in a human demonstration as a sequence of target poses for a task-space end-effector controller. **(A2):** The task involves a set of manipulable objects $\{O_1, ..., O_k\}$. **(A3):** During data collection, the pose of an object can be observed or estimated prior to the robot making contact with that object.

4 Method

We seek to learn visuomotor policies from demonstrations with minimal human effort by adapting a small number of human demonstrations to a large set of system states to facilitate automated demonstration generation. However, at both demonstration and deployment time, control difficulty is not uniformly spread across an episode. Specifically, in order to solve many manipulation tasks, the robot must first move itself in free space in order to reach a state where it can manipulate the world through contact. Free space motion can easily be carried out via motion planning and greatly reduce the policy learning burden. Thus, we propose decomposing tasks into *motion* and *skill* segments in order to isolate both demonstration generation and learning to just the skill segments, which will improve the quality of demonstrations and learned policies. We accomplish this by learning local manipulation *skills* that we combine in sequence using motion planning (Section 4.1). We show how adopting a skill-based framework allows for more focused demonstration replay (Section 4.4) and ultimately improved policy performance during deployment (Section 4.6).

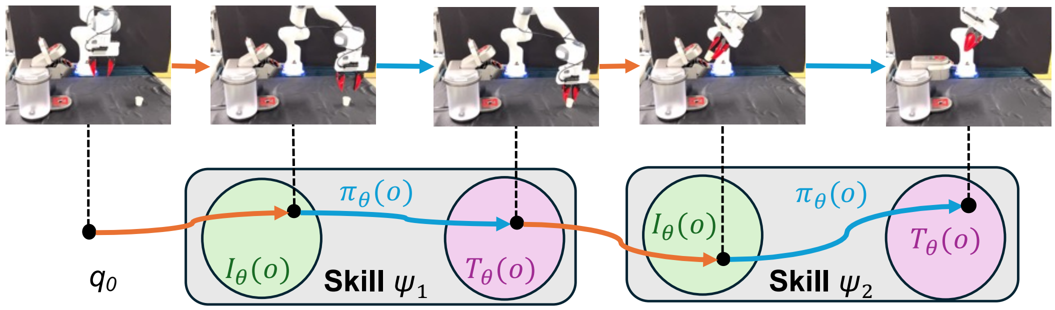


Figure 2: **HSP Deployment.** At test-time SkillGen, executes several learned skills in sequence, using motion planning to connect the termination state of the last skill with an initiation state of the next skill. Each skill consists of the initiation condition I_θ, the closed-loop controller π_θ, and the termination condition T_θ.

4.1 Skills Framework

Building off of the *options* [75] formalism from reinforcement learning, we define a *skill* $\psi = \langle O, \mathcal{I}, \pi, \mathcal{T} \rangle$ as a tuple consisting of an *object* to be manipulated O, *initiation* condition $\mathcal{I}$, *policy* π, and a *termination* condition $\mathcal{T}$. The initiation condition $\mathcal{I}$ defines a set of states where control using policy π can begin. The termination condition $\mathcal{T}$ defines a set of terminal states for policy π. We will use this skill abstraction to model all three phases of SkillGen, namely the initial tele-operation demonstrations (Section 4.3), the automated demonstration adaptation and amplification (Section 4.4), and the system execution at deployment time (Section 4.6).

4.2 Transit and Transfer Motion

Most tasks require performing multiple skills in sequence to complete them, such as the task in Fig. 2, which involves a *pick* skill to grasp the coffee pod and an *insert* skill load the pod in the coffee machine. In order to first reach the pick skill and then move the pod to the pod holder for the insert skill, the robot must perform two kinds of classical free-space motion [76, 77]. The first is *transit* motion, where the robot moves by itself without modifying the world. The second is *transfer* motion, where the robot is grasping an object approximately rigidly and transports the object as it moves. Thus, at both demonstration generation (Section 4.4) and system deployment (Section 4.6) time, SkillGen alternates between transit or transfer motion and manipulation skills.

SkillGen is a bilevel hierarchy where the skill initiation and termination induce the start and end robot configurations (q and q_*) for the motion segments. Namely, the termination condition $\mathcal{T}_i$ from the prior skill ψ_i governs the robot configuration q prior to the motion, and the initiation condition $\mathcal{I}_{i+1}$ of the next skill ψ_{i+1} defines the set of target end-effector poses $T_W^E \in \mathcal{I}_{i+1} \subseteq \mathrm{SE}(3)$, where E is the end-effector frame and W is the world frame. To generate these motions, we first convert task-space pose T_W^E to joint-space configuration q_* using inverse kinematics and then plan and execute a joint-space path from current configuration q to q_* with a motion planner.

4.3 Source Demonstrations

We assume a small *source* dataset of human demonstrations $\mathcal{D}_{\text{src}}$ collected on the task and our aim is to automatically generate a large dataset $\mathcal{D}$ on either the same task or a task variant. We start by annotating each trajectory in the source dataset $\tau \in \mathcal{D}_{\text{src}}$ with the start and end of each skill. This decomposes the demonstration into an alternating sequence of motion and skill trajectories $\tau = (\tau_{1m}, \tau_{1s}, ..., \tau_{Nm}, \tau_{Ns})$, where τ_{im} and τ_{is} denote motion and skill segments respectively. For source demonstrations provided by conventional teleoperation, these annotations can easily be annotated by a human. In our experiments, we choose to use demonstrations from the HITL-TAMP system [13], where the human only demonstrates local skill segments of each task, and the rest is handled by a TAMP system. In this case, annotations can be extracted automatically – each τ_{im} and τ_{is} is a TAMP and human segment respectively. Within each skill segment τ_{is}, each end-effector pose action $T_W^{A_t}$ (Sec. 3, A1) is stored in the frame of skill object O_i as $T_{O_i}^{A_t} \leftarrow (T_W^{O_i})^{-1} T_W^{A_t}$, where $T_W^{O_i}$ is the pose of object O_i observed prior to the skill. The first robot end effector pose in the skill demonstration $T_{O_i}^{E_0} \leftarrow \tau_{is}[0]$ is the *initiation state* and that will be the target end-effector pose for transit and transfer motion planning. The last pose in the demonstration $T_{O_i}^{E_K}$ implicitly defines the *termination state*, which will be learned through binary classification.

4.4 Demonstration Generation

The demonstrations $\mathcal{D}$ are generated through an automated trial-and-error process. Given a new initial state, SkillGen adapts existing skill segments to the new initial state and executes them in sequence with motion segments to generate a new demonstration. First, a reference skill segment τ_{is} is sampled. Next, the corresponding initiation state $T_{O_i}^{E_0}$ is used along with the pose $T_W^{O_i'}$ of object O_i in the new scene to obtain an end-effector pose for where the new skill segment should start, $T_W^{E_0'} \leftarrow T_W^{O_i'} T_{O_i}^{E_0}$. Next, the reference skill segment, expressed as a sequence of end-effector pose actions, $\tau_{is} = (T_{O_i}^{A_0}, ..., T_{O_i}^{A_K})$ is transformed to $\tau_{is}' = (T_W^{A_0'}, ..., T_W^{A_K'})$ where $T_W^{A_t'} \leftarrow T_W^{O_i'} T_{O_i}^{A_t}$. This transformation preserves the new end-effector pose actions with respect to the object frame [11]. The new skill segment τ_{is}' is executed by the end-effector controller. The steps above repeat for each skill, and then SkillGen checks for task success and only keeps the demonstration if it was successful. Seed Appendix O for pseudocode displaying the demonstration generation process.

4.5 Initiation Augmentation

At test time, learned skills trained on the generated data will be responsible for predicting both initiation targets for the motion planner and skill segments by employing a closed-loop agent that decides when to terminate. However, small differences in target pose predictions as well as motion plan tracking errors can cause learned policies to start out-of-distribution, thus reducing their accuracy. To mitigate such issues, SkillGen optionally adds noise to initiation states $T_W^{E_0}$, producing new initiation states $T_W^{E_0'}$, during data generation to broaden the support of the initiation set. To account for changing the initiation state, we consequently plan a *recovery segment* at the start of τ_{is}', consisting of a sequence of pose actions that moves from new $T_W^{E_0'}$ pose to the original pose $T_W^{E_0}$. This ensures that the new initiation state $T_W^{E_0'}$ is connected to the demonstration segment τ_{is}' when training closed-loop skill policies. See Appendix G for full details.

4.6 Policy Learning

Hybrid Skill Policy (HSP): We learn *parameterized skills* $\psi_\theta = \langle O, \mathcal{I}_\theta, \pi_\theta, \mathcal{T}_\theta \rangle$ using the generated datasets (parameterized by θ). The initiation condition $\mathcal{I}_\theta : \mathcal{O} \to \mathrm{SE}(3)$ is trained to predict initiation states $T_W^{E_0}$ from the last observation o on the prior skill. The policy $\pi_\theta : \mathcal{O} \to \mathcal{A}$ is trained on direct observation and action pairs $\langle o, a \rangle$ with BC (see Sec. 5). The termination condition $\mathcal{T}_\theta : \mathcal{O} \to \{0, 1\}$ is a classifier that predicts whether the skill is at a termination state based on the most recent observation o. During task deployment (Fig. 2), for each skill $\psi_\theta \in \Psi$ in a given sequence of skills Ψ, SkillGen predicts the initiation state $T_W^{E_0'} \leftarrow \mathcal{I}_\theta(o)$, plans and executes a path to it using a motion planner, and rolls out the learned policy by predicting actions $a \leftarrow \pi_\theta(o)$ until $\mathcal{T}_\theta(o)$ predicts policy termination. Then, this process repeats with the next skill (pseudocode in Appendix O).

HSP Variants: We consider two approaches for learning initiation conditions $\mathcal{I}_\theta$: *HSP-Reg* and *HSP-Class*. HSP-Reg formulates learning as a regression problem and directly predicts an initiation pose from the last observation. HSP-Class frames learning as classification problem over the initiation states in the source dataset $\mathcal{D}_{\text{src}}$, where the classifier predicts which source demonstration spawned the generated demonstration. Once classified, HSP-Class adapts the predicted initiation state to the current state using the pose adaptation procedure previously described in Section 4.4. However, recall that this requires the current pose $T_W^{O'}$ of object O, and thus HSP-Class assumes that object poses are known or can be estimated at the start of each skill segment. Ultimately, HSP-Class requires an additional observability assumption over HSP-Reg; however, this enables HSP-Class to perform discrete prediction over known pose candidates instead of continuous prediction over $\mathrm{SE}(3)$. Finally, we also consider *HSP-TAMP*, which deploys just the learned policies π_θ within HITL-TAMP [13], without the learned initiation and termination conditions.

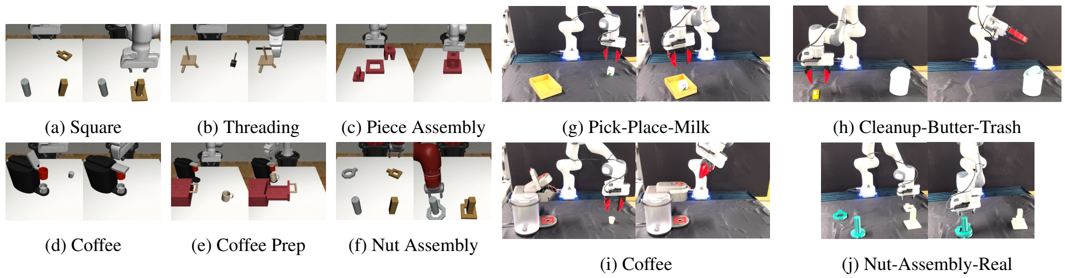


Figure 3: **Tasks.** We deploy SkillGen on 6 simulation tasks (18 task variants, see Appendix J) (a-f) and 4 real-world tasks (g-j). These tasks involve fine-grained insertion (a-d), composing several manipulation behaviors together (e, f), real-world data generation and training (g-i) and zero-shot sim-to-real policy transfer (j).

5 Experiment Setup

Tasks and Task Variants. We applied SkillGen to a broad range of tasks (see Fig. 3, full details in Appendix J) and task variants. Each task has a nominal reset distribution (D_0), and broader, more challenging reset distributions (D_1, D_2) [11]. All simulation tasks are implemented in robosuite [78] using its MuJoCo backend [79]. We experiment on simulated **Fine-Grained Tasks** (Square, Threading, Coffee, Piece Assembly) that require insertion, pulling, and pushing as well as **Long-Horizon Tasks** (Nut Assembly, Coffee Prep) that require chaining multiple behaviors together. Additionally, we experiment on **Real-Robot Tasks** (Pick-Place-Milk, Cleanup-Butter-Trash, Coffee), and **Sim-to-Real Tasks** (Nut-Assembly-Sim, Nut-Assembly-Real) to investigate SkillGen's propensity for zero-shot sim-to-real policy deployment.

Data Generation and Imitation Learning. For most of the experiments, a source dataset of 10 demonstrations was collected for each task on the D_0 variant by a single human operator using the HITL-TAMP teleoperation system [13]. SkillGen was used to generate 1000 successful demonstrations for each task variant (D_0, D_1, D_2) (see Appendix J for details), using each task's source dataset. Motion augmentation (Sec. 4) is only used to generate data to train HSC-Reg agents; HSC-TAMP and HSC-Class agents are trained on datasets generated without motion augmentation. See Appendix H for full policy learning details. The agent control policies used in the hybrid control policies (π_θ) were trained using BC with an RNN architecture [1] with the same hyperparameters from MimicGen. Policy performance is reported as the maximum success rate across all policy evaluations as in Mandlekar *et al.* [1]. All agents are trained with front-view and wrist-view RGB observations along with robot proprioception. Apart from the new task variants, we report the baseline data generation and agent performance statistics present in the MimicGen paper [11].

Motion Planning. In both the simulation and real-world tasks, we use TRAC-IK [80] for inverse kinematics, RRT-Connect [81] for joint-space motion planning, and Operational-Space Control (OSC) for task-space control [82]. In simulation, we check collisions during planning using the ground-truth obstacle collision geometries. In the real world, because collision geometries are not known, we use point-cloud-based collision checking using the segmented point cloud.

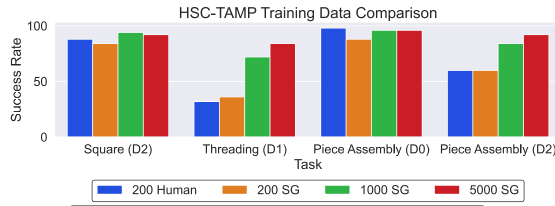
6 Experiments

6.1 SkillGen Features

SkillGen improves data generation rates over MimicGen substantially. MimicGen uses replay-based data generation for the entire trajectory, while SkillGen only uses replay for short skill segments, deferring larger transit motions to a motion planner. This results in substantially higher data generation success rates compared to MimicGen (average 75.4% vs. 40.7%, see Appendix F), especially when the reset distribution is large compared to the source demonstrations. Some compelling examples include Square D_2 (87.7% vs. 31.8%), Threading D_2 (74.3% vs. 21.6%), Three Piece Assembly D_2 (69.3% vs. 31.3%), and Coffee D_2 (70.0% vs. 27.7%).

SkillGen data collection is robust to large object rearrangements and clutter. In Coffee Prep D_2, the drawer containing the coffee pod and the mug are on opposite ends of the table compared to D_0 (source demos), and MimicGen is unable to collect any demonstrations while SkillGen achieves 59.9% data generation success. Additionally, in the Clutter variants of Square and Coffee (Ap-

Task Variant	Src	MG	HSP-T	HSP-C	HSP-R
Square D_0	50.0	90.7	**100.0**	**100.0**	94.0
Square D_1	-	73.3	**100.0**	98.0	62.0
Square D_2	-	49.3	**94.0**	**94.0**	52.0
Threading D_0	64.0	98.0	**100.0**	92.0	94.0
Threading D_1	-	60.7	**72.0**	66.0	60.0
Threading D_2	-	38.0	**62.0**	50.0	**62.0**
Piece Assembly D_0	28.0	82.0	**96.0**	80.0	86.0
Piece Assembly D_1	-	62.7	**88.0**	78.0	78.0
Piece Assembly D_2	-	13.3	**84.0**	74.0	50.0
Coffee D_0	**100.0**	**100.0**	**100.0**	**100.0**	**100.0**
Coffee D_1	-	90.7	**100.0**	**100.0**	**100.0**
Coffee D_2	-	77.3	94.0	**100.0**	98.0
Nut Assembly D_0	22.0	60.0	**100.0**	92.0	94.0
Nut Assembly D_1	-	16.0	72.0	**78.0**	20.0
Nut Assembly D_2	-	12.0	**54.0**	50.0	24.0
Coffee Prep D_0	2.0	**97.3**	92.0	92.0	84.0
Coffee Prep D_1	-	42.0	54.0	**74.0**	64.0
Coffee Prep D_2	-	0.0	80.0	74.0	**84.0**
Average	-	59.1	**85.7**	82.9	72.6



Task	MimicGen [11]	SkillGen
Milk-Bin	-	**95.0**
Butter-Trash	-	**95.0**
Coffee	14.0	**65.0**
Nut-Assembly [Sim]	72.0	**92.0**
Square-Assembly	5.0	**35.0**
Nut-Assembly	0.0	**35.0**

Figure 4: (*left*) **Agent Performance on SkillGen Datasets.** Success rates of agents trained on source demonstrations (with HSP-TAMP), MimicGen [11] data (with BC-RNN [1]), and SkillGen data (with all HSP variants). SkillGen data greatly improves agent performance on D_0 compared to the source data, and SkillGen agents substantially outperform MimicGen agents, especially on more challenging task variants. (*upper right*) **Training Data Comparison.** HSC-TAMP agent performance is comparable on 200 SkillGen demos and 200 human demos, despite SkillGen using just 10 human demos for generation. Generating more SkillGen demonstrations can result in significant performance improvement (also see Appendix E). (*lower right*) **Real-World Manipulation Results.** HSC-Class agents trained on SkillGen data generated in the real world are proficient, and substantially outperform using MimicGen data. They can also be transferred zero-shot from sim-to-real.

pendix D), a large object is placed randomly on the table. SkillGen achieves data generation rates from 49.0% to 72.0% while MimicGen only achieves 4.0% to 16.5%.

SkillGen greatly improves agent performance on the source task. Comparing HSP-TAMP agents trained on the source data vs. on SkillGen data on D_0, we see dramatic improvement (Fig. 4) – some examples include Three Piece Assembly (28% to 96%) and Nut Assembly (22% to 100%).

SkillGen produces more proficient agents through its use of hybrid control. Averaged across all tasks, HSP-TAMP, HSP-Class, and HSP-Reg achieve 85.7%, 82.9%, and 72.6% success rates respectively, compared to 59.1% for agents trained on MimicGen data (Fig. 4). Furthermore, HSP-Class and HSP-Reg make fewer assumptions than HSP-TAMP (see Sec. 4) while retaining the benefits of hybrid control. On Nut Assembly D_1 and D_2, HSP agents trained on SkillGen data outperform agents trained on MimicGen data by up to 62%, and SkillGen is able to train proficient agents (74% to 84%) on Coffee Prep D_2, while MimicGen fails to generate data for this variant (Fig. 4).

SkillGen effectively adapts demonstrations across robots. We use source demonstrations collected on the Panda arm and generate demonstrations for the Sawyer arm. As shown in Appendix N, data generation rates and policy performances are much higher for SkillGen than MimicGen.

6.2 SkillGen Analysis

Can agent performance on SkillGen data match agent performance on an equal amount of human demonstrations? We collected 200 demonstrations with the HITL-TAMP system [13] on each of 4 tasks and compared HSP-TAMP agent performance (the same method from HITL-TAMP) on the 200 human demos vs. 200 SkillGen demos (Fig. 4) generated from just 10 HITL-TAMP demos (which took less than 4 minutes per task to collect, compared to 37-71 minutes). Performance is comparable across all 4 tasks – 10% is the largest deviation, showing that SkillGen generated data is as effective as an equal number of human demos but only requires a small fraction of the effort.

Does agent performance improve by generating more demonstrations? We compared the performance of the different HSP algorithms on 200, 1000, and 5000 SkillGen demonstrations across the same 4 tasks from above – the results are presented in Fig. 4 (HSP-TAMP), and Appendix E (HSP-Class, HSP-Reg). All tasks and methods receive a significant increase from 200 to 1000 demos, and some tasks benefit strongly from 1000 to 5000 demos, notably Square D_2 (52% to 72% on HSP-Reg) and Threading D_1 (60% to 76% on HSP-Reg).

How does performance compare between the different hybrid control learning algorithms? Average task performance between HSP-TAMP and HSP-Class is similar (85.7% vs. 82.9%), and only slightly lower for HSP-Reg (72.2%) despite HSP-Class and HSP-Reg making much fewer assumptions (Fig. 4). HSP-Reg results could improve with more SkillGen data (Appendix E).

6.3 Real World Evaluation

We first demonstrate that SkillGen data generation can be deployed in the real-world and the data enables proficient policies to be learned. Next, we transfer agents trained in simulation with SkillGen zero-shot to the real-world on a long-horizon task, demonstrating that combining SkillGen with more sophisticated sim-to-real approaches is a promising method for robots to acquire real-world manipulation capabilities with minimal human effort. Results are summarized in Fig. 4 (lower right).

Setup. We use a Panda robot arm, a front-view RealSense D415 camera, and a wrist-view RealSense D435 camera. Pose estimates are obtained using FoundationPose [83]. Agents use proprioception and 120x160 camera images (except for sim-to-real agents) and are evaluated over 20 rollouts.

SkillGen Data Generation and Policy Learning in the Real World. We collect 3 source demonstrations with HITL-TAMP teleoperation on each of our tasks (Pick-Place-Milk, Cleanup-Butter-Trash, and Coffee), use SkillGen to generate 100 demonstrations, and train HSP-Class agents on the generated data (Appendix J has full details). These agents obtain near-perfect success rates on the Pick-Place-Milk and Cleanup-Butter-Trash tasks despite large amounts of spatial variation. HSP-Class also obtains 65% on the challenging Coffee task, while the BC-RNN agent trained on MimicGen data from [11] could only obtain 14%. This result is comparable with the 74% reported in HITL-TAMP [13] for an HSP-TAMP agent trained with 100 HITL-TAMP demos. We note the lower human effort (3 human demos vs. 100), that our Coffee task is more challenging (requires agent to learn to grasp the pod, unlike [13]) and our HSP-Class agent makes less assumptions.

Zero-Shot Sim-to-Real Deployment of SkillGen Policies. We designed a simulation task (Nut-Assembly [Sim]) that mirrors our real-world "Nut Assembly" task, where the robot must grasp a square and round nut and fit them onto corresponding square and round pegs. We train agents in simulation by collecting 1 source demo (with HITL-TAMP for SkillGen and with conventional teleoperation for MimicGen), generate 1000 demonstrations with SkillGen and MimicGen, and subsequently train an HSP-Class agent and a MimicGen (BC-RNN) agent (see Fig. 4, lower right). This task is challenging even in simulation, as the trained simulation agents are imperfect (HSP-Class: 92%, MimicGen: 72%). When deployed on the real-world task, the MimicGen agent manages to solve the first insertion task (Square-Assembly) with 5% success rate, but never solves the full task while the HSP-Class agent is able to achieve 35% success rate. This shows the value of SkillGen's hybrid control paradigm in aiding sim-to-real transfer through decomposing tasks into a sequence of local behaviors that are more likely to transfer [84]. More details and discussion in Appendix K.

7 Limitations

SkillGen requires knowledge of a fixed sequence of skills that can complete a task. It assumes that object poses can be observed at the start of each skill segment during data generation. SkillGen was demonstrated on quasi-static tasks involving rigid objects. SkillGen produces the best results when using source human demonstrations collected with the HITL-TAMP system – improving results with conventional teleoperation is left for future work. In the sim-to-real experiment, the agents had limited observability. Namely, agents only observe changes in proprioception, as no pose tracking or visual observations are used during execution. See Appendix C for full discussion.

8 Conclusion

We introduced SkillGen, a data generation system that synthesizes large datasets by adapting select skill segments from a handful of human demonstrations, and a Hybrid Skill Policy (HSP) learning framework to learn from the generated datasets by enabling closed-loop skills to be sequenced using a motion planner. We showed that SkillGen improves over a state-of-the-art data generation system, in both data generation capability and the ability to learn proficient agents from the data. We demonstrated SkillGen on real-world manipulation tasks, including zero-shot sim-to-real transfer.

References

[1] A. Mandlekar, D. Xu, J. Wong, S. Nasiriany, C. Wang, R. Kulkarni, L. Fei-Fei, S. Savarese, Y. Zhu, and R. Martín-Martín, "What matters in learning from offline human demonstrations for robot manipulation," in *Conference on Robot Learning (CoRL)*, 2021.

[2] A. Brohan, N. Brown, J. Carbajal, Y. Chebotar, J. Dabis, C. Finn, K. Gopalakrishnan, K. Hausman, A. Herzog, J. Hsu *et al.*, "Rt-1: Robotics transformer for real-world control at scale," *arXiv preprint arXiv:2212.06817*, 2022.

[3] T. Zhang, Z. McCarthy, O. Jow, D. Lee, K. Goldberg, and P. Abbeel, "Deep imitation learning for complex manipulation tasks from virtual reality teleoperation," *arXiv preprint arXiv:1710.04615*, 2017.

[4] A. Mandlekar, Y. Zhu, A. Garg, J. Booher, M. Spero, A. Tung, J. Gao, J. Emmons, A. Gupta, E. Orbay, S. Savarese, and L. Fei-Fei, "RoboTurk: A Crowdsourcing Platform for Robotic Skill Learning through Imitation," in *Conference on Robot Learning*, 2018.

[5] F. Ebert, Y. Yang, K. Schmeckpeper, B. Bucher, G. Georgakis, K. Daniilidis, C. Finn, and S. Levine, "Bridge data: Boosting generalization of robotic skills with cross-domain datasets," *arXiv preprint arXiv:2109.13396*, 2021.

[6] M. Ahn, A. Brohan, N. Brown, Y. Chebotar, O. Cortes, B. David, C. Finn, K. Gopalakrishnan, K. Hausman, A. Herzog *et al.*, "Do as i can, not as i say: Grounding language in robotic affordances," *arXiv preprint arXiv:2204.01691*, 2022.

[7] E. Jang, A. Irpan, M. Khansari, D. Kappler, F. Ebert, C. Lynch, S. Levine, and C. Finn, "Bc-z: Zero-shot task generalization with robotic imitation learning," in *Conference on Robot Learning*. PMLR, 2022, pp. 991–1002.

[8] C. Lynch, A. Wahid, J. Tompson, T. Ding, J. Betker, R. Baruch, T. Armstrong, and P. Florence, "Interactive language: Talking to robots in real time," *arXiv preprint arXiv:2210.06407*, 2022.

[9] H. Ravichandar, A. S. Polydoros, S. Chernova, and A. Billard, "Recent advances in robot learning from demonstration," *Annual review of control, robotics, and autonomous systems*, vol. 3, pp. 297–330, 2020.

[10] M. Dalal, A. Mandlekar, C. Garrett, A. Handa, R. Salakhutdinov, and D. Fox, "Imitating task and motion planning with visuomotor transformers," *arXiv preprint arXiv:2305.16309*, 2023.

[11] A. Mandlekar, S. Nasiriany, B. Wen, I. Akinola, Y. Narang, L. Fan, Y. Zhu, and D. Fox, "Mimicgen: A data generation system for scalable robot learning using human demonstrations," *arXiv preprint arXiv:2310.17596*, 2023.

[12] R. Hoque, A. Mandlekar, C. R. Garrett, K. Goldberg, and D. Fox, "Interventional data generation for robust and data-efficient robot imitation learning," in *First Workshop on Out-of-Distribution Generalization in Robotics at CoRL 2023*, 2023. [Online]. Available: https://openreview.net/forum?id=ckFRoOaA3n

[13] A. Mandlekar, C. R. Garrett, D. Xu, and D. Fox, "Human-in-the-loop task and motion planning for imitation learning," in *Conference on Robot Learning*. PMLR, 2023, pp. 3030–3060.

[14] A. Mandlekar, J. Booher, M. Spero, A. Tung, A. Gupta, Y. Zhu, A. Garg, S. Savarese, and L. Fei-Fei, "Scaling robot supervision to hundreds of hours with roboturk: Robotic manipulation dataset through human reasoning and dexterity," *arXiv preprint arXiv:1911.04052*, 2019.

[15] A. Mandlekar, D. Xu, R. Martín-Martín, Y. Zhu, L. Fei-Fei, and S. Savarese, "Human-in-the-loop imitation learning using remote teleoperation," *arXiv preprint arXiv:2012.06733*, 2020.

[16] A. Tung, J. Wong, A. Mandlekar, R. Martín-Martín, Y. Zhu, L. Fei-Fei, and S. Savarese, "Learning multi-arm manipulation through collaborative teleoperation," *arXiv preprint arXiv:2012.06738*, 2020.

[17] J. Wong, A. Tung, A. Kurenkov, A. Mandlekar, L. Fei-Fei, S. Savarese, and R. Martín-Martín, "Error-aware imitation learning from teleoperation data for mobile manipulation," in *Conference on Robot Learning*. PMLR, 2022, pp. 1367–1378.

[18] P. Wu, Y. Shentu, Z. Yi, X. Lin, and P. Abbeel, "Gello: A general, low-cost, and intuitive teleoperation framework for robot manipulators," 2023.

[19] T. Z. Zhao, V. Kumar, S. Levine, and C. Finn, "Learning fine-grained bimanual manipulation with low-cost hardware," *arXiv preprint arXiv:2304.13705*, 2023.

[20] J. Aldaco, T. Armstrong, R. Baruch, J. Bingham, S. Chan, K. Draper, D. Dwibedi, C. Finn, P. Florence, S. Goodrich *et al.*, "Aloha 2: An enhanced low-cost hardware for bimanual teleoperation," *arXiv preprint arXiv:2405.02292*, 2024.

[21] Z. Fu, T. Z. Zhao, and C. Finn, "Mobile aloha: Learning bimanual mobile manipulation with low-cost whole-body teleoperation," *arXiv preprint arXiv:2401.02117*, 2024.

[22] A. Iyer, Z. Peng, Y. Dai, I. Guzey, S. Haldar, S. Chintala, and L. Pinto, "Open teach: A versatile teleoperation system for robotic manipulation," *arXiv preprint arXiv:2403.07870*, 2024.

[23] S. Dass, W. Ai, Y. Jiang, S. Singh, J. Hu, R. Zhang, P. Stone, B. Abbatematteo, and R. Martin-Martin, "Telemoma: A modular and versatile teleoperation system for mobile manipulation," *arXiv preprint arXiv:2403.07869*, 2024.

[24] C. Chi, Z. Xu, C. Pan, E. Cousineau, B. Burchfiel, S. Feng, R. Tedrake, and S. Song, "Universal manipulation interface: In-the-wild robot teaching without in-the-wild robots," *arXiv preprint arXiv:2402.10329*, 2024.

[25] H. Fang, H.-S. Fang, Y. Wang, J. Ren, J. Chen, R. Zhang, W. Wang, and C. Lu, "Low-cost exoskeletons for learning whole-arm manipulation in the wild," *arXiv preprint arXiv:2309.14975*, 2023.

[26] S. James, Z. Ma, D. R. Arrojo, and A. J. Davison, "Rlbench: The robot learning benchmark & learning environment," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 3019–3026, 2020.

[27] A. Zeng, P. Florence, J. Tompson, S. Welker, J. Chien, M. Attarian, T. Armstrong, I. Krasin, D. Duong, V. Sindhwani *et al.*, "Transporter networks: Rearranging the visual world for robotic manipulation," *arXiv preprint arXiv:2010.14406*, 2020.

[28] Y. Jiang, A. Gupta, Z. Zhang, G. Wang, Y. Dou, Y. Chen, L. Fei-Fei, A. Anandkumar, Y. Zhu, and L. Fan, "Vima: General robot manipulation with multimodal prompts," *arXiv preprint arXiv:2210.03094*, 2022.

[29] J. Gu, F. Xiang, X. Li, Z. Ling, X. Liu, T. Mu, Y. Tang, S. Tao, X. Wei, Y. Yao *et al.*, "Maniskill2: A unified benchmark for generalizable manipulation skills," *arXiv preprint arXiv:2302.04659*, 2023.

[30] H. Ha, P. Florence, and S. Song, "Scaling up and distilling down: Language-guided robot skill acquisition," *arXiv preprint arXiv:2307.14535*, 2023.

[31] Y. Wang, Z. Xian, F. Chen, T.-H. Wang, Y. Wang, K. Fragkiadaki, Z. Erickson, D. Held, and C. Gan, "Robogen: Towards unleashing infinite data for automated robot learning via generative simulation," *arXiv preprint arXiv:2311.01455*, 2023.

[32] D. A. Pomerleau, "Alvinn: An autonomous land vehicle in a neural network," in *Advances in neural information processing systems*, 1989, pp. 305–313.

[33] A. J. Ijspeert, J. Nakanishi, and S. Schaal, "Movement imitation with nonlinear dynamical systems in humanoid robots," *Proceedings 2002 IEEE International Conference on Robotics and Automation*, vol. 2, pp. 1398–1403 vol.2, 2002.

[34] C. Finn, T. Yu, T. Zhang, P. Abbeel, and S. Levine, "One-shot visual imitation learning via meta-learning," in *Conference on robot learning*. PMLR, 2017, pp. 357–368.

[35] A. Billard, S. Calinon, R. Dillmann, and S. Schaal, "Robot programming by demonstration," in *Springer Handbook of Robotics*, 2008.

[36] S. Calinon, F. D'halluin, E. L. Sauser, D. G. Caldwell, and A. Billard, "Learning and reproduction of gestures by imitation," *IEEE Robotics and Automation Magazine*, vol. 17, pp. 44–54, 2010.

[37] A. Mandlekar, D. Xu, R. Martín-Martín, S. Savarese, and L. Fei-Fei, "Learning to generalize across long-horizon tasks from human demonstrations," *arXiv preprint arXiv:2003.06085*, 2020.

[38] C. Wang, R. Wang, D. Xu, A. Mandlekar, L. Fei-Fei, and S. Savarese, "Generalization through hand-eye coordination: An action space for learning spatially-invariant visuomotor control," *arXiv preprint arXiv:2103.00375*, 2021.

[39] C. Lynch, M. Khansari, T. Xiao, V. Kumar, J. Tompson, S. Levine, and P. Sermanet, "Learning latent plans from play," in *Conference on Robot Learning*, 2019.

[40] K. Pertsch, Y. Lee, Y. Wu, and J. J. Lim, "Demonstration-guided reinforcement learning with learned skills," in *Conference on Robot Learning*, 2021.

[41] A. Ajay, A. Kumar, P. Agrawal, S. Levine, and O. Nachum, "Opal: Offline primitive discovery for accelerating offline reinforcement learning," in *International Conference on Learning Representations*, 2021.

[42] K. Hakhamaneshi, R. Zhao, A. Zhan, P. Abbeel, and M. Laskin, "Hierarchical few-shot imitation with skill transition models," in *International Conference on Learning Representations*, 2021.

[43] Y. Zhu, P. Stone, and Y. Zhu, "Bottom-up skill discovery from unsegmented demonstrations for long-horizon robot manipulation," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 4126–4133, 2022.

[44] S. Nasiriany, T. Gao, A. Mandlekar, and Y. Zhu, "Learning and retrieval from prior data for skill-based imitation learning," in *Conference on Robot Learning (CoRL)*, 2022.

[45] A. Kumar, A. Singh, F. Ebert, Y. Yang, C. Finn, and S. Levine, "Pre-training for robots: Offline rl enables learning new tasks from a handful of trials," *arXiv preprint arXiv:2210.05178*, 2022.

[46] P. Mitrano and D. Berenson, "Data augmentation for manipulation," *arXiv preprint arXiv:2205.02886*, 2022.

[47] M. Laskin, K. Lee, A. Stooke, L. Pinto, P. Abbeel, and A. Srinivas, "Reinforcement learning with augmented data," *arXiv preprint arXiv:2004.14990*, 2020.

[48] I. Kostrikov, D. Yarats, and R. Fergus, "Image augmentation is all you need: Regularizing deep reinforcement learning from pixels," *arXiv preprint arXiv:2004.13649*, 2020.

[49] S. Young, D. Gandhi, S. Tulsiani, A. Gupta, P. Abbeel, and L. Pinto, "Visual imitation made easy," *arXiv e-prints*, pp. arXiv–2008, 2020.

[50] A. Zhan, P. Zhao, L. Pinto, P. Abbeel, and M. Laskin, "A framework for efficient robotic manipulation," *arXiv preprint arXiv:2012.07975*, 2020.

[51] S. Sinha, A. Mandlekar, and A. Garg, "S4rl: Surprisingly simple self-supervision for offline reinforcement learning in robotics," in *Conference on Robot Learning*. PMLR, 2022, pp. 907–917.

[52] S. Pitis, E. Creager, and A. Garg, "Counterfactual data augmentation using locally factored dynamics," *Advances in Neural Information Processing Systems*, vol. 33, pp. 3976–3990, 2020.

[53] S. Pitis, E. Creager, A. Mandlekar, and A. Garg, "Mocoda: Model-based counterfactual data augmentation," *arXiv preprint arXiv:2210.11287*, 2022.

[54] Z. Mandi, H. Bharadhwaj, V. Moens, S. Song, A. Rajeswaran, and V. Kumar, "Cacti: A framework for scalable multi-task multi-scene visual imitation learning," *arXiv preprint arXiv:2212.05711*, 2022.

[55] T. Yu, T. Xiao, A. Stone, J. Tompson, A. Brohan, S. Wang, J. Singh, C. Tan, J. Peralta, B. Ichter *et al.*, "Scaling robot learning with semantically imagined experience," *arXiv preprint arXiv:2302.11550*, 2023.

[56] Z. Chen, S. Kiami, A. Gupta, and V. Kumar, "Genaug: Retargeting behaviors to unseen situations via generative augmentation," *arXiv preprint arXiv:2302.06671*, 2023.

[57] H. Bharadhwaj, J. Vakil, M. Sharma, A. Gupta, S. Tulsiani, and V. Kumar, "Roboagent: Generalization and efficiency in robot manipulation via semantic augmentations and action chunking," *arXiv preprint arXiv:2309.01918*, 2023.

[58] A. Goyal, J. Xu, Y. Guo, V. Blukis, Y.-W. Chao, and D. Fox, "Rvt: Robotic view transformer for 3d object manipulation," *arXiv:2306.14896*, 2023.

[59] M. Shridhar, L. Manuelli, and D. Fox, "Perceiver-actor: A multi-task transformer for robotic manipulation," in *Proceedings of the 6th Conference on Robot Learning (CoRL)*, 2022.

[60] T. Gervet, Z. Xian, N. Gkanatsios, and K. Fragkiadaki, "Act3d: Infinite resolution action detection transformer for robotic manipulation," *arXiv preprint arXiv:2306.17817*, 2023.

[61] S. Belkhale, Y. Cui, and D. Sadigh, "Hydra: Hybrid robot actions for imitation learning," in *Conference on Robot Learning*. PMLR, 2023, pp. 2113–2133.

[62] L. X. Shi, A. Sharma, T. Z. Zhao, and C. Finn, "Waypoint-based imitation learning for robotic manipulation," in *7th Annual Conference on Robot Learning*, 2023. [Online]. Available: https://openreview.net/forum?id=X0cmlTh1Vl

[63] P. Parashar, V. Jain, X. Zhang, J. Vakil, S. Powers, Y. Bisk, and C. Paxton, "Slap: Spatial-language attention policies," in *7th Annual Conference on Robot Learning*, 2023.

[64] Z. Xian, N. Gkanatsios, T. Gervet, T.-W. Ke, and K. Fragkiadaki, "Chaineddiffuser: Unifying trajectory diffusion and keypose prediction for robotic manipulation," in *7th Annual Conference on Robot Learning*, 2023.

[65] S. Cheng, C. Garrett, A. Mandlekar, and D. Xu, "NOD-TAMP: Multi-step manipulation planning with neural object descriptors," in *CoRL 2023 Workshop on Learning Effective Abstractions for Planning (LEAP)*, 2023. [Online]. Available: https://openreview.net/forum?id=DK7TbAS0Wz

[66] Z. Wang, C. R. Garrett, L. P. Kaelbling, and T. Lozano-Pérez, "Learning compositional models of robot skills for task and motion planning," *The International Journal of Robotics Research*, vol. 40, no. 6-7, pp. 866–894, 2021.

[67] C. R. Garrett, R. Chitnis, R. Holladay, B. Kim, T. Silver, L. P. Kaelbling, and T. Lozano-Pérez, "Integrated task and motion planning," *Annual review of control, robotics, and autonomous systems*, vol. 4, pp. 265–293, 2021.

[68] B. Wen, W. Lian, K. Bekris, and S. Schaal, "You only demonstrate once: Category-level manipulation from single visual demonstration," in *Robotics: Science and Systems (RSS)*, 2022.

[69] N. Di Palo and E. Johns, "Learning multi-stage tasks with one demonstration via self-replay," in *Conference on Robot Learning*. PMLR, 2022, pp. 1180–1189.

[70] E. Johns, "Coarse-to-fine imitation learning: Robot manipulation from a single demonstration," in *2021 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2021, pp. 4613–4619.

[71] V. Vosylius and E. Johns, "Where to start? transferring simple skills to complex environments," *arXiv preprint arXiv:2212.06111*, 2022.

[72] A. Chenu, O. Serris, O. Sigaud, and N. Perrin-Gilbert, "Leveraging sequentiality in reinforcement learning from a single demonstration," *arXiv preprint arXiv:2211.04786*, 2022.

[73] E. Valassakis, G. Papagiannis, N. Di Palo, and E. Johns, "Demonstrate once, imitate immediately (dome): Learning visual servoing for one-shot imitation learning," in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2022, pp. 8614–8621.

[74] J. Liang, B. Wen, K. Bekris, and A. Boularias, "Learning sensorimotor primitives of sequential manipulation tasks from visual demonstrations," in *2022 International Conference on Robotics and Automation (ICRA)*. IEEE, 2022, pp. 8591–8597.

[75] M. Stolle and D. Precup, "Learning options in reinforcement learning," in *Abstraction, Reformulation, and Approximation: 5th International Symposium, SARA 2002 Kananaskis, Alberta, Canada August 2–4, 2002 Proceedings 5*. Springer, 2002, pp. 212–223.

[76] R. Alami, T. Simeon, and J.-P. Laumond, "A geometrical approach to planning manipulation tasks. the case of discrete placements and grasps," in *The fifth international symposium on Robotics research*. MIT Press, 1990, pp. 453–463.

[77] Y. Koga, K. Kondo, J. Kuffner, and J.-C. Latombe, "Planning motions with intentions," in *Proceedings of the 21st annual conference on Computer graphics and interactive techniques*, 1994, pp. 395–408.

[78] Y. Zhu, J. Wong, A. Mandlekar, and R. Martín-Martín, "robosuite: A modular simulation framework and benchmark for robot learning," in *arXiv preprint arXiv:2009.12293*, 2020.

[79] E. Todorov, T. Erez, and Y. Tassa, "Mujoco: A physics engine for model-based control," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2012, pp. 5026–5033.

[80] P. Beeson and B. Ames, "{TRAC-IK}: An open-source library for improved solving of generic inverse kinematics," 11 2015.

[81] J. J. Kuffner and S. M. LaValle, "Rrt-connect: An efficient approach to single-query path planning," in *Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No. 00CH37065)*, vol. 2. IEEE, 2000, pp. 995–1001.

[82] O. Khatib, "A unified approach for motion and force control of robot manipulators: The operational space formulation," *IEEE Journal on Robotics and Automation*, vol. 3, no. 1, pp. 43–53, 1987.

[83] B. Wen, W. Yang, J. Kautz, and S. Birchfield, "FoundationPose: Unified 6d pose estimation and tracking of novel objects," in *CVPR*, 2024.

[84] E. Valassakis, N. Di Palo, and E. Johns, "Coarse-to-fine for sim-to-real: Sub-millimetre precision across wide task spaces," in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2021, pp. 5989–5996.