MimicGen: A Data Generation System for Scalable Robot Learning using Human Demonstrations

Anonymous Author(s)
Affiliation
Address
email

Abstract: Imitation learning from a large set of human demonstrations has proved to be an effective paradigm for building capable robot agents. However, the demonstrations can be extremely costly and time-consuming to collect. We introduce MimicGen, a system for automatically synthesizing large-scale, rich datasets from only a small number of human demonstrations by adapting them to new contexts. We use MimicGen to generate over 50K demonstrations across 18 tasks with diverse scene configurations, object instances, and robot arms from just ∼200 human demonstrations. We show that robot agents can be effectively trained on this generated dataset by imitation learning to achieve strong performance in long-horizon and high-precision tasks, such as multi-part assembly and coffee preparation, across broad initial state distributions. We further demonstrate that the effectiveness and utility of MimicGen data compare favorably to collecting additional human demonstrations, making it a powerful and economical approach towards scaling up robot learning. Videos and additional results at https://sites.google.com/view/corl2023mimicgen.

Keywords: Imitation Learning, Manipulation

1 Introduction

Imitation learning from human demonstrations has become an effective paradigm for training robots to perform a wide variety of manipulation behaviors. One popular approach is to have human operators teleoperate robot arms through different control interfaces [1, 2], resulting in several demonstrations of robots performing various manipulation tasks, and consequently to use the data to train the robots to perform these tasks on their own. Recent attempts have aimed to scale this paradigm by collecting more data with a larger group of human operators over a broader range of tasks [3–6]. These works have shown that imitation learning on large diverse datasets can produce impressive performance, allowing robots to generalize toward new objects and unseen tasks. This suggests that a critical step toward building generally capable robots is collecting large and rich datasets.

However, this success does not come without costly and time-consuming human labor. Consider a case study from robomimic [7], in which the agent is tasked with moving a coke can from one bin into another. This is a simple task involving a single scene, single object, and single robot; however, a relatively-large dataset of 200 demonstrations was required to achieve a modest success rate of 73.3%. Recent efforts at expanding to settings with diverse scenes and objects have required orders of magnitude larger datasets spanning tens of thousands of demonstrations. For example, [3] showed that a dataset of over 20,000 trajectories enables generalization to tasks with modest changes in objects and goals. The nearly 1.5-year data collection effort from RT-1 [5] spans several human operators, months, kitchens, and robot arms to produce policies that can rearrange, cleanup, and retrieve objects with a 97% success rate across a handful of kitchens. Yet it remains unclear how many years of data collection would be needed to deploy such a system to kitchens in the wild.

We raise the question — how much of this data actually contains unique manipulation behaviors? Large portions of these datasets may contain similar manipulation skills applied in different contexts or situations. For example, human operators may demonstrate very similar robot trajectories to grasp a mug, regardless of its location on one countertop or another. Re-purposing these

Submitted to the 7th Conference on Robot Learning (CoRL 2023). Do not distribute.

trajectories in new contexts can be a way to generate diverse data without much human effort. In fact, several recent works build on this intuition and propose imitation learning methods that replay previous human demonstrations [8–11] (**more related work discussion in Appendix D**). While promising, these methods make assumptions about specific tasks and algorithms that limit their applicability. Instead, we seek to develop a general-purpose system that can be integrated seamlessly into existing imitation learning pipelines and improve the performance of a wide spectrum of tasks.

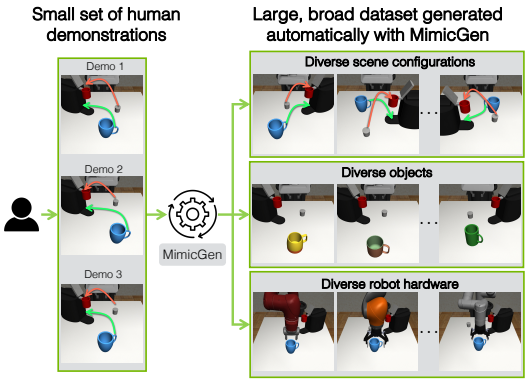


Figure 1: **MimicGen Overview.** We introduce a data generation system that can produce large diverse datasets from a small number of human demonstrations by re-purposing the demonstrations to make them applicable in new settings. We apply MimicGen to generate data across diverse scene configurations, objects, and robot hardware.

In this paper, we introduce a novel data collection system that uses a small set of human demonstrations to automatically generate large datasets across diverse scenes. Our system, **MimicGen**, takes a small number of human demonstrations and divides them into object-centric segments. Then, given a new scene with different object poses, it selects one of the human demonstrations, spatially transforms each of its object-centric segments, stitches them together, and has the robot follow this new trajectory to collect a new demonstration. While simple, we found that this method is extremely effective at generating large datasets across diverse scenes and that the datasets can be used to train capable agents through imitation learning.

We make the following contributions:

• We introduce MimicGen, a system for generating large diverse datasets from a small number of human demonstrations by adapting the human demonstrations to novel settings.

• We demonstrate that MimicGen is able to generate high-quality data to train proficient agents via imitation learning across diverse scene configurations, object instances, and robot arms, all of which are unseen in the original demos (see Fig. 1). MimicGen is broadly applicable to a wide range of long-horizon and high-precision tasks that require different manipulation skills, such as pick-and-place, insertion, and interacting with articulated objects. We generated 50K+ new demonstrations for 18 tasks across 2 simulators and a physical robot arm using only $\sim$200 source human demos.

• Our approach compares favorably to the alternative of collecting more human demonstrations — using MimicGen to generate an equal amount of synthetic data (e.g. 200 demos generated from 10 human vs. 200 human demos) results in comparable agent performance — this raises important questions about when it is actually necessary to request additional data from a human.

2 Problem Setup

Related Work. There are several data collection efforts [2–6, 12–26] and offline data augmentation methods [7, 27–36] for robotics (by contrast, we do online data generation). There are also data generation efforts in other domains [37–43]. We generate data for imitation learning [1, 20, 44–51]. Our method is related to replay-based imitation [8–11, 52–54]. **Full related work in Appendix D.**

Imitation Learning. We consider each robot manipulation task as a Markov Decision Process (MDP), and aim to learn a robot manipulation policy π that maps the state space $\mathcal{S}$ to the action space $\mathcal{A}$. The imitation dataset consists of N demonstrations $\mathcal{D} = \{(s_0^i, a_0^i, s_1^i, a_1^i, ..., s_{H_i}^i)\}_{i=1}^N$ where each $s_0^i \sim D(\cdot)$ is sampled from the initial state distribution D. In this work, we use Behavioral Cloning [44] to train the policy with the objective $\arg\min_\theta \mathbb{E}_{(s,a)\sim\mathcal{D}}[-\log \pi_\theta(a|s)]$.

Problem Statement and Assumptions. Our goal is to use a **source dataset** $\mathcal{D}_{\text{src}}$ that consists of a small set of human demonstrations collected on a task $\mathcal{M}$ and use it to generate a large dataset $\mathcal{D}$ on either the same task or **task variants** (where the initial state distribution D, the objects, or the robot arm can change). To generate a new demo: (1) a start state is sampled from the task we want to generate data for, (2) a demonstration $\tau \in \mathcal{D}_{\text{src}}$ is chosen and adapted to produce a new robot trajectory τ', (3) the robot executes the trajectory τ' on the current scene, and if the task is

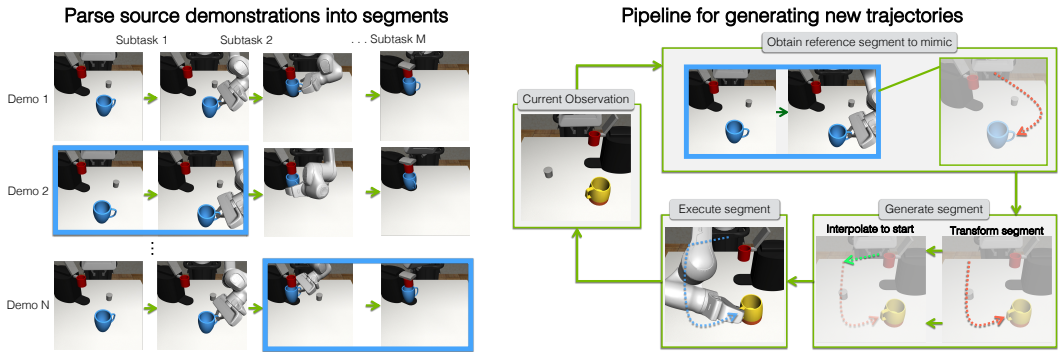


Figure 2: **MimicGen System Pipeline.** (left) MimicGen first parses the demos from the source dataset into segments, where each segment corresponds to an object-centric subtask (Sec. 3.1). (right) Then, to generate new demonstrations for a new scene, MimicGen generates and follows a sequence of end-effector target poses for each subtask by (1) choosing a segment from a source demonstration (chosen segments shown with blue border in figure above), (2) transforming it for the new scene, and (3) executing it (Sec. 3.2).

completed successfully, the sequence of states and actions is added to the generated dataset $\mathcal{D}$ (see Sec. 3 for details of each step). We next outline some assumptions that our system leverages.

Assumption 1: delta end effector pose action space. The action space $\mathcal{A}$ consists of delta-pose commands for an end-effector controller and a gripper open/close command. This is a common action space used in prior work [3–7, 49]. This gives us an equivalence between delta-pose actions and controller target poses, and allows us to treat the actions in a demonstration as a sequence of target poses for the end effector controller (Appendix M).

Assumption 2: tasks consist of a known sequence of object-centric subtasks. Let $\mathcal{O} = \{o_1, ..., o_K\}$ be the set of objects in a task $\mathcal{M}$. As in Di Palo et al. [11], we assume that tasks consist of a sequence of object-centric subtasks $(S_1(o_{S_1}), S_2(o_{S_2}), ..., S_M(o_{S_M}))$, where the manipulation in each subtask $S_i(o_{S_i})$ is relative to a single object's coordinate frame ($o_{S_i} \in \mathcal{O}$). We assume this sequence is known (it is typically easy for a human to specify — see Appendix J).

Assumption 3: object poses can be observed at the start of each subtask during data collection. We assume that we can observe the pose of the relevant object o_{S_i} at the start of each subtask $S_i(o_{S_i})$ during data collection (not, however, during policy deployment).

3 Method

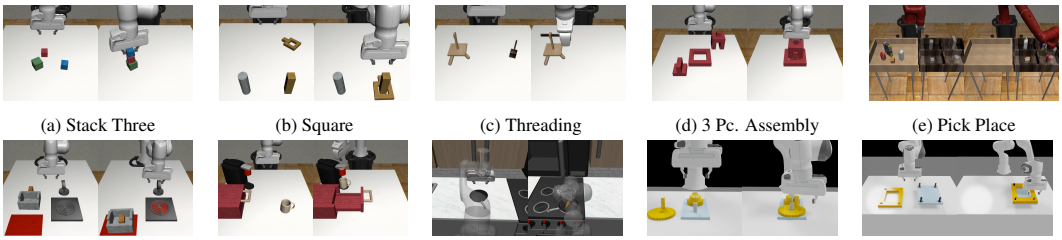
We describe how MimicGen generates new demonstrations using a small source dataset of human demonstrations (see Fig. 2 for an overview). MimicGen first parses the source dataset into segments — one for each object-centric subtask in a task (Sec. 3.1). Then, to generate a demonstration for a new scene, MimicGen generates and executes a trajectory (sequence of end-effector control poses) for each subtask, by choosing a reference segment from the source demonstrations, transforming it according to the pose of the object in the new scene, and then executing the sequence of target poses using the end effector controller (Sec. 3.2).

3.1 Parsing the Source Dataset into Object-Centric Segments

Each task consists of a sequence of object-centric subtasks (Assumption 2, Sec. 2) — we would like to parse every trajectory τ in the source dataset into segments $\{\tau_i\}_{i=1}^{M}$, where each segment τ_i corresponds to a subtask $S_i(o_{S_i})$. In this work, to parse source demonstrations into segments for each subtask, we assume access to metrics that allow the end of each subtask to be detected automatically (see Appendix J for full details). After this step, every trajectory $\tau \in \mathcal{D}_{\text{src}}$ has been split into a contiguous sequence of segments $\tau = (\tau_1, \tau_2, ..., \tau_M)$, one per subtask.

3.2 Transforming Source Data Segments for a New Scene

To generate a task demonstration for a new scene, MimicGen generates and executes a segment for each object-centric subtask in the task. As shown in Fig. 2 (right), this consists of three key steps



(a) Stack Three (b) Square (c) Threading (d) 3 Pc. Assembly (e) Pick Place

(f) Kitchen (g) Coffee Prep (h) Mobile Kitchen (i) Gear Assembly (j) Frame Assembly

Figure 3: **Tasks.** We use MimicGen to generate demonstrations for several tasks — these are a subset. They span a wide variety of behaviors including pick-and-place, insertion, interacting with articulated objects, and mobile manipulation, and include long-horizon tasks requiring chaining several behaviors together.

for each subtask: (1) choosing a reference subtask segment in the source dataset, (2) transforming the subtask segment for the new context, and (3) executing the segment in the scene.

Choosing a reference segment: Recall that MimicGen parses the source dataset into segments that correspond to each subtask $\mathcal{D}_{\text{src}} = \{(\tau_1^j, \tau_2^j, ..., \tau_M^j)\}_{j=1}^N$ where $N = |\mathcal{D}_{\text{src}}|$. At the start of each subtask $S_i(o_{S_i})$, MimicGen chooses a corresponding segment from the set $\{\tau_i^j\}_{j=1}^N$. These segments can be chosen at random or by using the relevant object poses (more details in Appendix M).

Transforming the source subtask segment: We can consider the chosen source subtask segment τ_i for subtask $S_i(o_{S_i})$ as a sequence of target poses for the end effector controller (Assumption 1, Sec. 2). Let T_B^A be the homogeneous 4×4 matrix that represents the pose of frame A with respect to frame B. Then we can write $\tau_i = (T_W^{C_0}, T_W^{C_1}, ..., T_W^{C_K})$ where C_t is the controller target pose frame at timestep t, W is the world frame, and K is the length of the segment. Since this motion is assumed to be relative to the pose of the object o_{S_i} (frame O_0 with pose $T_W^{O_0}$) at the start of the segment, we will transform τ_i according to the new pose of the corresponding object in the current scene (frame O_0' with pose $T_W^{O_0'}$) so that the relative poses between the target pose frame and the object frame are preserved at each timestep ($T_{O_0}^{C_t} = T_{O_0'}^{C_t'}$) resulting in the transformed sequence $\tau_i' = (T_W^{C_0'}, T_W^{C_1'}, ..., T_W^{C_K'})$ where $T_W^{C_t'} = T_W^{O_0}(T_W^{O_0'})^{-1}T_W^{C_t}$ (derivation in Appendix L). As an example, see how the source segment and transformed segment in the right side of Fig. 2 approach the mug in consistent ways. However, the first target pose of the new segment $T_W^{C_0'}$ might be far from the current end-effector pose of the robot in the new scene $T_W^{E_0'}$ (where E is the end-effector frame). Consequently, MimicGen adds an **interpolation segment** at the start of τ_i' to interpolate linearly from the current end-effector pose ($T_W^{E_0'}$) to the start of the transformed segment $T_W^{C_0'}$.

Executing the new segment: Finally, MimicGen executes the new segment τ_i' by taking the target pose at each timestep, transforming it into a delta pose action (Assumption 1, Sec. 2), pairing it with the appropriate gripper open/close action from the source segment, and executing the new action.

The steps above repeat for each subtask until the final segment has been executed. However, this process can be imperfect — small trajectory deviations due to control and arm kinematics issues can result in task failure. Thus, MimicGen checks for task success after executing all segments, and only keeps successful demonstrations. We refer to the ratio between the number of successfully generated trajectories and the total number of attempts as the **data generation rate** (reported in Appendix O).

This pipeline only depends on object frames and robot controller frames — this enables data generation to take place across tasks with different initial state distributions, objects (assuming they have canonical frames defined), and robot arms (assuming they share a convention for the end effector control frame). In our experiments, we designed **task variants** for each robot manipulation task where we vary either the initial state distribution (D), an object in the task (O), or the robot arm (R), and showed that MimicGen enables data collection and imitation learning across these variants.

4 Experiment Setup

We applied MimicGen to a broad range of tasks (see Fig. 3) and task variants, in order to showcase how it can generate useful data for imitation learning across a diverse set of manipulation behaviors, including pick-and-place, contact-rich interactions, and articulation.

Tasks and Task Variants. Each task has a default reset distribution (D_0) (all source datasets were collected on this task variant), a broader reset distribution (D_1), and some have another (D_2), meant to pose even higher difficulty for data generation and policy learning. Consider the Threading task shown in Fig. 5 — in the D_0 variant, the tripod is always initialized in the same location, while in the D_1 variant, both the tripod and needle can move, and in the D_2 variant, the tripod and needle are randomized in novel regions of the workspace. In some experiments, we also applied MimicGen to task variants with a different robot arm (R) or different object instances (O) within a category.

We group the tasks into categories and summarize them below (full tasks and variants in Appendix K). Some tasks are implemented with the robosuite framework [55] (MuJoCo backend [56]) and others are implemented in Factory [57] (Isaac Gym [58] backend). **Basic Tasks** (Stack, Stack Three): a set of box stacking tasks. **Contact-Rich Tasks** (Square, Threading, Coffee, Three Piece Assembly, Hammer Cleanup, Mug Cleanup): a set of tasks that involve contact-rich behaviors such as insertion or drawer articulation. **Long-Horizon Tasks** (Kitchen, Nut Assembly, Pick Place, Coffee Preparation): require chaining multiple behaviors together. **Mobile Manipulation Tasks** (Mobile Kitchen): requires base and arm motion. **Factory Tasks** (Nut-Bolt-Assembly, Gear Assembly, Frame Assembly): a set of high-precision assembly tasks in Factory [57].

Data Generation and Imitation Learning Methodology. For each task, one human operator collected a source dataset of 10 demonstrations on the default variant (D_0) using a teleoperation system [2, 18] (with the exception of Mobile Kitchen, where we used 25 demos due to the large number of object variants, and Square, where we used 10 demos from the robomimic Square PH dataset [7]). MimicGen was used to generate 1000 demonstrations for each task variant, using each task's source dataset (full details in Appendix M). Since data generation is imperfect, each data generation attempt is not guaranteed to result in a task success. Attempts that did not achieve task success were discarded, and data collection kept proceeding for each task variant until 1000 task successes were collected. Each generated dataset was then used to train policies using Behavioral Cloning with an RNN policy [7]. We also adopt the convention from Mandlekar et al. [7] for reporting policy performance — the maximum success rate across all policy evaluations, across 3 different seeds (full training details in Appendix N). All policy learning results are shown on **image-based agents** trained with RGB observations (see Appendix P for low-dim agent results).

5 Experiments

We present experiments that (1) highlight the diverse array of situations that MimicGen can generate data for, (2) show that MimicGen compares favorably to collecting additional human demonstrations, both in terms of effort and downstream policy performance on the data, (3) offer insights into different aspects of the system, and (4) show that MimicGen can work on real-world robot arms.

5.1 Applications of MimicGen

We outline a number of applications that showcase useful properties of MimicGen.

MimicGen data vastly improves agent performance on the source task. A straightforward application of MimicGen is to collect a small dataset on some task of interest and then generate more data for that task. Comparing the performance of agents trained on the small source datasets vs. those trained on D_0 datasets generated by MimicGen, we see that there is substantial improvement across all our tasks (see Fig. 4). Some particularly compelling examples include Square (11.3% to 90.7%), Threading (19.3% to 98.0%), and Three Piece Assembly (1.3% to 82.0%).

MimicGen data can produce performant agents across broad initial state distributions. As shown in Fig. 4), agents trained using datasets generated on broad initial state distributions (D_1, D_2) are performant (42% to 99% on D_1), showing that MimicGen generates valuable datasets on new initial state distributions. In several cases, certain objects in the 10 source demonstrations never moved (the peg in Square, the tripod in Threading, the base in Three Piece Assembly, etc), but data was generated (and policies consequently were trained) on regimes where the objects move in substantial regions of the robot workspace.

MimicGen can generate data for different objects. The source dataset in the Mug Cleanup task contains just one mug, but we generate demonstrations with MimicGen for an unseen mug (O_1) and for a set of 12 mugs (O_2). Policies trained on these datasets have substantial task success rates (90.7% and 75.3% respectively) (full results in Appendix F).

Task	Source	D_0	D_1	D_2
Stack	26.0 ± 1.6	100.0 ± 0.0	99.3 ± 0.9	-
Stack Three	0.7 ± 0.9	92.7 ± 1.9	86.7 ± 3.4	-
Square	11.3 ± 0.9	90.7 ± 1.9	73.3 ± 3.4	49.3 ± 2.5
Threading	19.3 ± 3.4	98.0 ± 1.6	60.7 ± 2.5	38.0 ± 3.3
Coffee	74.0 ± 4.3	100.0 ± 0.0	90.7 ± 2.5	77.3 ± 0.9
Three Pc. Assembly	1.3 ± 0.9	82.0 ± 1.6	62.7 ± 2.5	13.3 ± 3.8
Hammer Cleanup	59.3 ± 5.7	100.0 ± 0.0	62.7 ± 4.7	-
Mug Cleanup	12.7 ± 2.5	80.0 ± 4.9	64.0 ± 3.3	90.7 ± 1.9
Kitchen	54.7 ± 8.4	100.0 ± 0.0	76.0 ± 4.3	-
Nut Assembly	0.0 ± 0.0	53.3 ± 1.9	-	-
Pick Place	0.0 ± 0.0	50.7 ± 6.6	-	-
Coffee Preparation	12.7 ± 3.4	97.3 ± 0.9	42.0 ± 0.0	-
Mobile Kitchen	2.0 ± 0.0	46.7 ± 18.4	-	-
Nut-and-Bolt Assembly	8.7 ± 2.5	92.7 ± 2.5	81.3 ± 8.2	72.7 ± 4.1
Gear Assembly	14.7 ± 5.2	98.7 ± 1.9	74.0 ± 2.8	56.7 ± 1.9
Frame Assembly	10.7 ± 6.8	82.0 ± 4.3	68.7 ± 3.4	36.7 ± 2.5

Figure 4: (left) **Agent Performance on Source and Generated Datasets.** Success rates (3 seeds) of image-based agents trained with BC on the 10 source demos and each 1000 demo MimicGen dataset. There is large improvement across all tasks on the default distribution (D_0) and agents are performant on the broader distributions (D_1, D_2). (top-right) **MimicGen with more source human demonstrations.** We found that using larger source datasets to generate MimicGen data did not result in significant agent improvement. (bottom-right) **Policy Training Dataset Comparison.** Image-based agent performance is comparable on 200 MimicGen demos and 200 human demos, despite MimicGen only using 10 source human demos. MimicGen can produce improved agents by generating larger datasets (200, 1000, 5000 demos), but there are diminishing returns.

MimicGen can generate data for diverse robot hardware. We apply MimicGen to the Square and Threading source datasets (which use the Panda arm) and generate datasets for the Sawyer, IIWA, and UR5e across the D_0 and D_1 reset distribution variants. Interestingly, although the data generation rates differ greatly per arm (range 38%-74% for Square D_0), trained policy performance is remarkably similar across the 4 robot arms (80%-91%, full results in Appendix E). This shows the potential for using human demonstrations across robot hardware using MimicGen, an exciting prospect, as teleoperated demonstrations are typically constrained to a single robot.

Applying MimicGen to mobile manipulation. In the Mobile Kitchen task MimicGen yields a gain from 2.0% to 46.7% (image, Fig. 4) and 2.7% to 76.7% success rate (low-dim, Table P.1 in Appendix), highlighting that our method can be applied to tasks beyond static tabletop manipulation.

MimicGen is simulator-agnostic. We show that MimicGen is not limited to just one simulation framework by applying it to high-precision tasks (requiring **millimeter precision**) in Factory [57], a simulation framework built on top of Isaac Gym [58] to accurately simulate high-precision manipulation. We generate data for and train performant policies on the Nut-and-Bolt Assembly, Gear Assembly, and Frame Assembly tasks. Policies achieve excellent results on the nominal tasks (D_0) (82%-99%), a significant improvement over policies trained on the source datasets (9%-15%), and are also able to achieve substantial performance on wider reset distributions (D_1, D_2) (37%-81%).

MimicGen can use demonstrations from inexperienced human operators and different teleoperation devices. Surprisingly, policies trained on these MimicGen datasets have comparable performance to those in Fig. 4. See Appendix H for the full set of results.

5.2 Comparing MimicGen to using more human data

In this section, we contextualize the performance of agents trained on MimicGen data.

Comparing task performance to prior works. Zhu et al. [59] introduced the Hammer Cleanup and Kitchen tasks and reported agent performance on 100 human demonstrations for their method called BUDS. On Hammer Cleanup, BUDS achieved 68.6% (D_0), while BC-RNN achieves 59.3% on our 10 source demos, 100.0% on our generated 1000 D_0 demos, and 62.7% on the D_1 variant where both the hammer and drawer move substantially. On Kitchen, BUDS achieved 72.0% (D_0), while BC-RNN achieves 54.7% on our 10 source demos, 100.0% on our generated D_0 data, and 76.0% on the D_1 variant, where all objects move in wider regions. This shows that using MimicGen to make effective use of a small number of human demonstrations can improve the complexity of tasks that can be learned with imitation learning. As another example, Mandlekar et al. [2] collected over 1000 human demos across 10 human operators on both the Nut Assembly and Pick Place tasks, but only managed to train proficient policies for easier, single-stage versions of these tasks using a combination of reinforcement learning and demonstrations. By contrast, in this work we are able to

make effective use of just 10 human demonstrations to generate a set of 1000 demonstrations and learn proficient agents from them (76.0% and 58.7% low-dim, 53.3% and 50.7% image).

Agent performance on data generated by MimicGen can be comparable to performance on an equal amount of human demonstrations. We collect 200 human demonstrations on several tasks and compare agent performance on those demonstrations to agent performance on 200 demonstrations generated by MimicGen (see Fig. 4). In most cases, agent performance is similar, despite the 200 MimicGen demos being generated from just 10 human demos — a small number of human demos can be as effective (or even more) than a large number of them when used with MimicGen. MimicGen can also easily generate more demonstrations to improve performance (see Sec. 5.3), unlike the time-consuming nature of collecting more human data. This result also raises important questions on whether soliciting more human demonstrations can be redundant and not worth the labeling cost, and where to collect human demonstrations given a finite labeling bandwidth.

5.3 MimicGen Analysis

We analyze some practical aspects of the system, including (1) whether the number of source demonstrations used impacts agent performance, (2) whether the choice of source demonstrations matters, (3) whether agent performance can keep improving by generating more demonstrations, and (4) whether the data generation success rate and trained agent performance are correlated.

Can dataset quality and agent performance be improved by using more source human demonstrations? We used 10, 50, and 200 source human demonstrations on the Square and Three Piece Assembly tasks, and report the policy success rates in Fig. 4. We see that performance differences are modest (ranging from 2% to 21%). We also tried using just 1 human demo — in some cases performance was much worse (e.g. Square), while in others, there was no significant performance change (e.g. Three Piece Assembly). It is possible that performance could improve with more source human demos if they are curated in an intelligent manner, but this is left for future work.

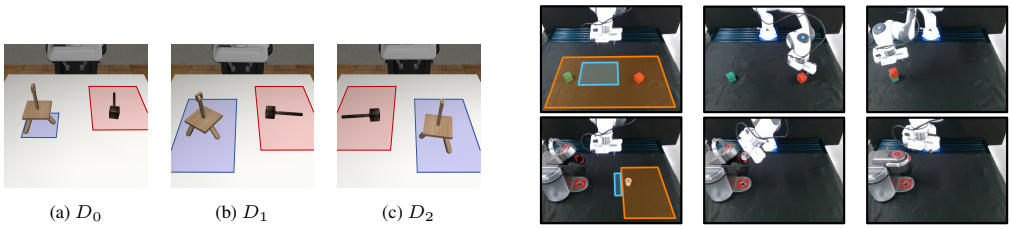
Does the choice of source human demonstrations matter? For each generated dataset, we logged which episode came from which source human demonstration — in certain cases, this distribution can be very non-uniform. As an example, the generated Factory Gear Assembly task (D_1) had over 850 of the 1000 episodes come from just 3 source demonstrations. In the generated Threading task (D_0), one source demo had over 170 episodes while another had less than 10 episodes. In both cases, the number of attempted episodes per source demonstration was roughly uniform (since we picked them at random — details in Appendix M), but some were more likely to generate successful demonstrations than others. Furthermore, we found the source demonstration segment selection technique (Sec. 3.2) to matter for certain tasks (Appendix M). This indicates that both the initial set of source demos provided to MimicGen ($\mathcal{D}_{\text{src}}$), and how segments from these demos are chosen during each generation attempt (τ_i for each subtask, see Sec. 3.1) can matter.

Can agent performance keep improving by generating more demonstrations? In Fig. 4, we train agents on 200, 1000, and 5000 demos generated by MimicGen across several tasks. There is a large jump in performance from 200 to 1000, but not much from 1000 to 5000, showing that there can be diminishing returns on generating more data.

Are the data generation success rate and trained agent performance correlated? It is tempting to think that data generation success rate and trained agent performance are correlated, but we found that this is not necessarily true — there are datasets that had low dataset generation success rates (and consequently took a long time to generate 1000 successes) but had high agent performance after training on the data (Appendix O). A few examples are Object Cleanup (D_0) (29.5% generation rate, 82.0% agent rate), Three Piece Assembly (D_0) (35.6% generation rate, 74.7% agent rate), Coffee (D_2) (27.7% generation rate, 76.7% agent rate), and Factory Gear Assembly (D_1) (8.2% generation rate, 76.0% agent rate). These results showcase the value of using replay-based mechanisms for data collection instead of directly using them to deploy as policy as in prior works [8, 11].

5.4 Real Robot Evaluation

We validate that MimicGen can be applied to real-world robot arms and tasks. We collect 10 source demonstrations for each task in narrow regions of the workspace (D_0) and then generate demonstrations (200 for Stack, 100 for Coffee) for large regions of the workspace (D_1) (see Fig. 5). The



(a) D_0 (b) D_1 (c) D_2

Figure 5: (left) **Reset Distributions.** Each task has a default reset distribution for the objects (D_0), a broader one (D_1), and some had a more challenging one (D_2). The figure shows the sampling regions for the tripod and needle in the Threading task. The tripod is at a fixed location in D_0, and D_2 swaps the relative locations of the tripod and needle. We generate data across diverse scene configurations by taking source demos from D_0 and generating data for all variants. (right) **Real Robot Tasks.** We apply MimicGen to two real robot tasks — Stack (top row) and Coffee (bottom row). In the first column, the blue and orange regions show the source (D_0) and generated (D_1) reset distributions for each task. We use 10 source demos per task, and generate 100 successful demos — MimicGen has a data generation success rate of 82.3% for Stack and 52.1% for Coffee.

generation success rate was 82.3% for Stack (243 attempts) and 52.1% for Coffee (192 attempts), showing that MimicGen works in the real world with a reasonably high success rate. We then trained visuomotor agents using a front-facing RealSense D415 camera and a wrist-mounted RealSense D435 camera (120×160 resolution). Over 50 evaluations, our Stack agent had 36% success rate and Coffee had 14% success rate (pod grasp success rate of 60% and pod insertion success rate of 20%). The lower numbers than from simulation might be due to the larger number of interpolation steps we used in the real world for hardware safety (50 total instead of 5) — these motions are difficult for the agent to imitate since there is little association between the intermediate motion and observations (see Appendix G for experiments that confirm a negative impact on performance).

We also compared these results with the performance of agents trained on the source datasets (10 demos) in the narrow regions (orange regions in Fig. 5) where the training data came from — the Stack source agent had 0% success rate and the Coffee source agent had 0% success rate (with an insertion rate of 0% and pod grasp rate of 94%). The Coffee (D_0) task in particular has barely any variation (the pod can move vertically in a 5cm region) compared to the D_1 task, which is substantially harder (pod placed anywhere in the right half of the workspace). Agents trained with MimicGen data compare favorably to these agents, as they achieve non-zero success rates on broader task reset distributions. Real-world data collection is slow and often bottlenecked by the number of arms and requirements of humans to reset the workspace — instead, another avenue to obtain real-world agents is to generate large diverse sets of demonstrations in simulated settings, train an agent in simulation using this data, and then transfer the agent to the real world. Combining MimicGen with sim-to-real policy deployment methods [60–64] remains an exciting avenue for future work.

6 Limitations

See Appendix C for full set of limitations and discussion. MimicGen assumes knowledge of the object-centric subtasks in a task and requires object pose estimates at the start of each subtask during data generation (Assumption 3, Sec. 2). MimicGen only filters data generation attempts based on task success, so generated datasets can be biased (Appendix Q). MimicGen uses linear interpolation between human segments (Appendix M.2), which does not guarantee collision-free motion, and can potentially hurt agent performance (Appendix G). MimicGen was demonstrated on quasi-static tasks with rigid objects, and novel objects were assumed to come from the same category.

7 Conclusion

We introduced MimicGen, a data generation system that can use small amounts of human demonstrations to generate large datasets across diverse scenes, object instances, and robots, and applied it to generate over 50K demos across 18 tasks from less than 200 human demos, including tasks involving long-horizon and high-precision manipulation. We showed that agents learning from this data can achieve strong performance. We further found that agent performance on MimicGen data can be comparable to performance on an equal number of human demos — this surprising result motivates further investigation into when to solicit additional human demonstrations instead of making more effective use of a small number, and whether human operator time would be better spent collecting data in new regions of the workspace. We hope that MimicGen motivates and enables exploring a more data-centric perspective on imitation learning in future work.

References

[1] 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.

[2] 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.

[3] 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.

[4] 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.

[5] 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.

[6] 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.

[7] 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.

[8] 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.

[9] E. Johns, "Coarse-to-Fine Imitation Learning: Robot Manipulation from a Single Demonstration," *ICRA*, 2021.

[10] 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.

[11] 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.

[12] S. Levine, P. Pastor, A. Krizhevsky, and D. Quillen, "Learning hand-eye coordination for robotic grasping with large-scale data collection," in *ISER*, 2016, pp. 173–184.

[13] L. Pinto and A. Gupta, "Supersizing self-supervision: Learning to grasp from 50k tries and 700 robot hours," in *Robotics and Automation (ICRA), 2016 IEEE Int'l Conference on*. IEEE, 2016.

[14] D. Kalashnikov, A. Irpan, P. Pastor, J. Ibarz, A. Herzog, E. Jang, D. Quillen, E. Holly, M. Kalakrishnan, V. Vanhoucke *et al.*, "Qt-opt: Scalable deep reinforcement learning for vision-based robotic manipulation," *arXiv preprint arXiv:1806.10293*, 2018.

[15] D. Kalashnikov, J. Varley, Y. Chebotar, B. Swanson, R. Jonschkowski, C. Finn, S. Levine, and K. Hausman, "Mt-opt: Continuous multi-task robotic reinforcement learning at scale," *arXiv preprint arXiv:2104.08212*, 2021.

[16] K.-T. Yu, M. Bauza, N. Fazeli, and A. Rodriguez, "More than a million ways to be pushed. a high-fidelity experimental dataset of planar pushing," in *Int'l Conference on Intelligent Robots and Systems*, 2016.

[17] S. Dasari, F. Ebert, S. Tian, S. Nair, B. Bucher, K. Schmeckpeper, S. Singh, S. Levine, and C. Finn, "Robonet: Large-scale multi-robot learning," *arXiv preprint arXiv:1910.11215*, 2019.

[18] 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.

[19] 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.

[20] 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.

[21] 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.

[22] 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.

[23] J. Zhang and K. Cho, "Query-efficient imitation learning for end-to-end autonomous driving," *arXiv preprint arXiv:1605.06450*, 2016.

[24] R. Hoque, A. Balakrishna, C. Putterman, M. Luo, D. S. Brown, D. Seita, B. Thananjeyan, E. Novoseller, and K. Goldberg, "Lazydagger: Reducing context switching in interactive imitation learning," in *2021 IEEE 17th International Conference on Automation Science and Engineering (CASE)*. IEEE, 2021, pp. 502–509.

[25] R. Hoque, A. Balakrishna, E. Novoseller, A. Wilcox, D. S. Brown, and K. Goldberg, "Thriftydagger: Budget-aware novelty and risk gating for interactive imitation learning," *arXiv preprint arXiv:2109.08273*, 2021.

[26] S. Dass, K. Pertsch, H. Zhang, Y. Lee, J. J. Lim, and S. Nikolaidis, "Pato: Policy assisted teleoperation for scalable robot data collection," *arXiv preprint arXiv:2212.04708*, 2022.

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

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

[29] 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.

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

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

[32] 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.

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

[34] 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.

[35] 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.

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

[37] Y. Zhang, H. Ling, J. Gao, K. Yin, J.-F. Lafleche, A. Barriuso, A. Torralba, and S. Fidler, "Datasetgan: Efficient labeled data factory with minimal human effort," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 10 145–10 155.

[38] D. Li, H. Ling, S. W. Kim, K. Kreis, S. Fidler, and A. Torralba, "Bigdatasetgan: Synthesizing imagenet with pixel-wise annotations," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 21 330–21 340.

[39] A. Kar, A. Prakash, M.-Y. Liu, E. Cameracci, J. Yuan, M. Rusiniak, D. Acuna, A. Torralba, and S. Fidler, "Meta-sim: Learning to generate synthetic datasets," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 4551–4560.

[40] J. Devaranjan, A. Kar, and S. Fidler, "Meta-sim2: Unsupervised learning of scene structure for synthetic data generation," in *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XVII 16*. Springer, 2020, pp. 715–733.

[41] S. W. Kim, J. Philion, A. Torralba, and S. Fidler, "Drivegan: Towards a controllable high-quality neural simulation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 5820–5829.

[42] D. Paschalidou, A. Kar, M. Shugrina, K. Kreis, A. Geiger, and S. Fidler, "Atiss: Autoregressive transformers for indoor scene synthesis," *Advances in Neural Information Processing Systems*, vol. 34, pp. 12 013–12 026, 2021.

[43] S. Tan, K. Wong, S. Wang, S. Manivasagam, M. Ren, and R. Urtasun, "Scenegen: Learning to generate realistic traffic scenes," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 892–901.

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

[45] 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.

[46] 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.

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

[48] 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.

[49] 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.

[50] 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.

[51] 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.

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

[53] 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.

[54] 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.

[55] 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.

[56] 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.

[57] Y. Narang, K. Storey, I. Akinola, M. Macklin, P. Reist, L. Wawrzyniak, Y. Guo, A. Moravanszky, G. State, M. Lu *et al.*, "Factory: Fast contact for robotic assembly," *arXiv preprint arXiv:2205.03532*, 2022.

[58] V. Makoviychuk, L. Wawrzyniak, Y. Guo, M. Lu, K. Storey, M. Macklin, D. Hoeller, N. Rudin, A. Allshire, A. Handa *et al.*, "Isaac gym: High performance gpu-based physics simulation for robot learning," *arXiv preprint arXiv:2108.10470*, 2021.

[59] 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.

[60] X. B. Peng, M. Andrychowicz, W. Zaremba, and P. Abbeel, "Sim-to-real transfer of robotic control with dynamics randomization," in *2018 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2018, pp. 3803–3810.

[61] M. Kaspar, J. D. M. Osorio, and J. Bock, "Sim2real transfer for reinforcement learning without dynamics randomization," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2020, pp. 4383–4388.

[62] A. Allshire, M. MittaI, V. Lodaya, V. Makoviychuk, D. Makoviichuk, F. Widmaier, M. Wüthrich, S. Bauer, A. Handa, and A. Garg, "Transferring dexterous manipulation from gpu simulation to a remote real-world trifinger," in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2022, pp. 11 802–11 809.

[63] M. Khansari, D. Ho, Y. Du, A. Fuentes, M. Bennice, N. Sievers, S. Kirmani, Y. Bai, and E. Jang, "Practical imitation learning in the real world via task consistency loss," *arXiv preprint arXiv:2202.01862*, 2022.

[64] A. Handa, A. Allshire, V. Makoviychuk, A. Petrenko, R. Singh, J. Liu, D. Makoviichuk, K. Van Wyk, A. Zhurkevich, B. Sundaralingam *et al.*, "Dextreme: Transfer of agile in-hand manipulation from simulation to reality," *arXiv preprint arXiv:2210.13702*, 2022.

[65] 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.

[66] O. Mees, L. Hermann, E. Rosete-Beas, and W. Burgard, "Calvin: A benchmark for language-conditioned policy learning for long-horizon robot manipulation tasks," *IEEE Robotics and Automation Letters*, vol. 7, no. 3, pp. 7327–7334, 2022.

[67] S. Levine, A. Kumar, G. Tucker, and J. Fu, "Offline reinforcement learning: Tutorial, review, and perspectives on open problems," *arXiv preprint arXiv:2005.01643*, 2020.

[68] 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.

[69] 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.

[70] S. Dasari, J. Wang, J. Hong, S. Bahl, Y. Lin, A. Wang, A. Thankaraj, K. Chahal, B. Calli, S. Gupta *et al.*, "Rb2: Robotic manipulation benchmarking with a twist," *arXiv preprint arXiv:2203.08098*, 2022.

[71] T. Yu, D. Quillen, Z. He, R. Julian, K. Hausman, C. Finn, and S. Levine, "Meta-world: A benchmark and evaluation for multi-task and meta reinforcement learning," in *Conference on robot learning*. PMLR, 2020, pp. 1094–1100.

[72] T. Mu, Z. Ling, F. Xiang, D. Yang, X. Li, S. Tao, Z. Huang, Z. Jia, and H. Su, "Maniskill: Generalizable manipulation skill benchmark with large-scale demonstrations," *arXiv preprint arXiv:2107.14483*, 2021.

[73] 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.

[74] C. Chamzas, C. Quintero-Pena, Z. Kingston, A. Orthey, D. Rakita, M. Gleicher, M. Toussaint, and L. E. Kavraki, "Motionbenchmaker: A tool to generate and benchmark motion planning datasets," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 882–889, 2021.

[75] 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.

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

[77] 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.

[78] 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.

[79] 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.

[80] M. A. Fischler and R. C. Bolles, "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography," *Communications of the ACM*, vol. 24, no. 6, pp. 381–395, 1981.

[81] M. Ester, H.-P. Kriegel, J. Sander, X. Xu *et al.*, "A density-based algorithm for discovering clusters in large spatial databases with noise." in *kdd*, vol. 96, no. 34, 1996, pp. 226–231.

[82] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask r-cnn," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 2961–2969.

[83] M. Danielczuk, M. Matl, S. Gupta, A. Li, A. Lee, J. Mahler, and K. Goldberg, "Segmenting unknown 3d objects from real depth images using mask r-cnn trained on synthetic data," in *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 2019, pp. 7283–7290.

[84] B. Wen, C. Mitash, S. Soorian, A. Kimmel, A. Sintov, and K. E. Bekris, "Robust, occlusion-aware pose estimation for objects grasped by adaptive hands," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 6210–6217.

[85] Z. Zhang, "Iterative point matching for registration of free-form curves and surfaces," *International journal of computer vision*, vol. 13, no. 2, pp. 119–152, 1994.

[86] Y. Liu, Y. Wen, S. Peng, C. Lin, X. Long, T. Komura, and W. Wang, "Gen6d: Generalizable model-free 6-dof object pose estimation from rgb images," in *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXXII*. Springer, 2022, pp. 298–315.

[87] J. Sun, Z. Wang, S. Zhang, X. He, H. Zhao, G. Zhang, and X. Zhou, "Onepose: One-shot object pose estimation without cad models," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 6825–6834.

[88] B. Wen, J. Tremblay, V. Blukis, S. Tyree, T. Muller, A. Evans, D. Fox, J. Kautz, and S. Birchfield, "Bundlesdf: Neural 6-dof tracking and 3d reconstruction of unknown objects," *CVPR*, 2023.

[89] Y. Zhu, Z. Wang, J. Merel, A. Rusu, T. Erez, S. Cabi, S. Tunyasuvunakool, J. Kramár, R. Hadsell, N. de Freitas *et al.*, "Reinforcement and imitation learning for diverse visuomotor skills," *arXiv preprint arXiv:1802.09564*, 2018.

[90] A. Mandlekar, F. Ramos, B. Boots, S. Savarese, L. Fei-Fei, A. Garg, and D. Fox, "Iris: Implicit reinforcement without interaction at scale for learning control from offline robot manipulation data," in *IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 4414–4420.

[91] L. Chen, R. Paleja, and M. Gombolay, "Learning from suboptimal demonstration via self-supervised reward regression," in *Conference on robot learning*. PMLR, 2021, pp. 1262–1277.

[92] D. Brown, W. Goo, P. Nagarajan, and S. Niekum, "Extrapolating beyond suboptimal demonstrations via inverse reinforcement learning from observations," in *International conference on machine learning*. PMLR, 2019, pp. 783–792.

[93] R. Jeong, J. T. Springenberg, J. Kay, D. Zheng, Y. Zhou, A. Galashov, N. Heess, and F. Nori, "Learning dexterous manipulation from suboptimal experts," *arXiv preprint arXiv:2010.08587*, 2020.

[94] H. Xu, X. Zhan, H. Yin, and H. Qin, "Discriminator-weighted offline imitation learning from suboptimal demonstrations," in *International Conference on Machine Learning*. PMLR, 2022, pp. 24 725–24 742.

[95] M. Yang, S. Levine, and O. Nachum, "Trail: Near-optimal imitation learning with suboptimal data," *arXiv preprint arXiv:2110.14770*, 2021.

[96] A. S. Morgan, B. Wen, J. Liang, A. Boularias, A. M. Dollar, and K. Bekris, "Vision-driven compliant manipulation for reliable, high-precision assembly tasks," *RSS*, 2021.