DexMimicGen: Automated Data Generation for Bimanual Dexterous Manipulation via Imitation Learning

Zhenyu Jiang[*1,2] **Yuqi Xie**[*1,2] **Kevin Lin**[*1,2] **Zhenjia Xu**[1] **Weikang Wan**[3]
Ajay Mandlekar[†1] Linxi "Jim" Fan[†1] Yuke Zhu[†1,2] ‡

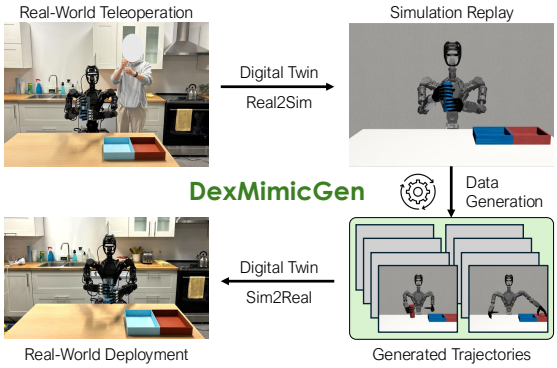

Figure 1: DexMimicGen Overview.

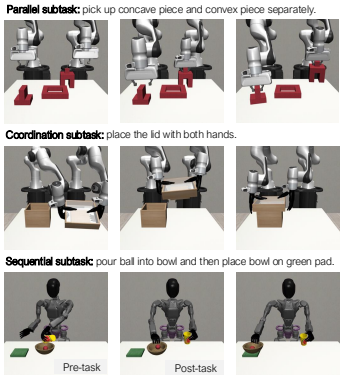

Figure 2: Subtask Types.

1 Introduction

Imitation learning from human demonstrations is an effective means to teach robots manipulation skills [1, 2]. However, data acquisition is a key bottleneck in applying this paradigm more broadly. Prior efforts for data collection in the single robot arm setting required multiple human operators, robots, and months of human effort [3, 2, 4, 5, 6]. Unfortunately, scaling data collection for humanoids can be even more difficult, owing to the challenges of controlling multiple arms and multi-fingered dexterous hands simultaneously. Enabling real-time teleoperation for humanoids has required the development of special-purpose teleoperation interfaces [7, 8, 9, 10, 11, 12], but these pipelines can be costly and difficult to scale. Furthermore, the increase in operator burden due to multi-arm and multi-finger hand control makes collecting demonstrations in this setting more challenging compared to the single-arm setting, further limiting the rate of data collection. The data acquisition burden is further compounded by the fact that data requirements in the humanoid setting can be higher due to the increased task complexity.

Instead, leveraging automated data generation in simulation is a compelling alternative that has proved effective for the single-arm robot manipulation setting [13, 14, 15]. Inspired by this success, **we introduce DexMimicGen (DexMG), a large-scale automated data generation system for bimanual robots with dexterous hands, such as humanoids (Fig. 1)**. The core idea is to leverage a small set of human demonstrations and use demonstration transformation and replay in physical simulation to automatically generate large amounts of training data suitable for imitation learning in the bimanual dexterous manipulation setting. This system builds on top of MimicGen [15], which proposed a similar pipeline for the single-arm with parallel-jaw gripper setting, but there are several technical challenges that DexMimicGen overcomes to apply the same principles to our setting.

MimicGen relies on decomposing each task into a sequence of subtasks, to generate trajectories for each subtask separately and then stitch them together. Bimanual dexterous manipulation involves

[*]*Equal Contributions, †Project Leads
[†1]NVIDIA Research, [2]UT Austin, [3]UC San Diego

8th Conference on Robot Learning (CoRL 2024), Munich, Germany.

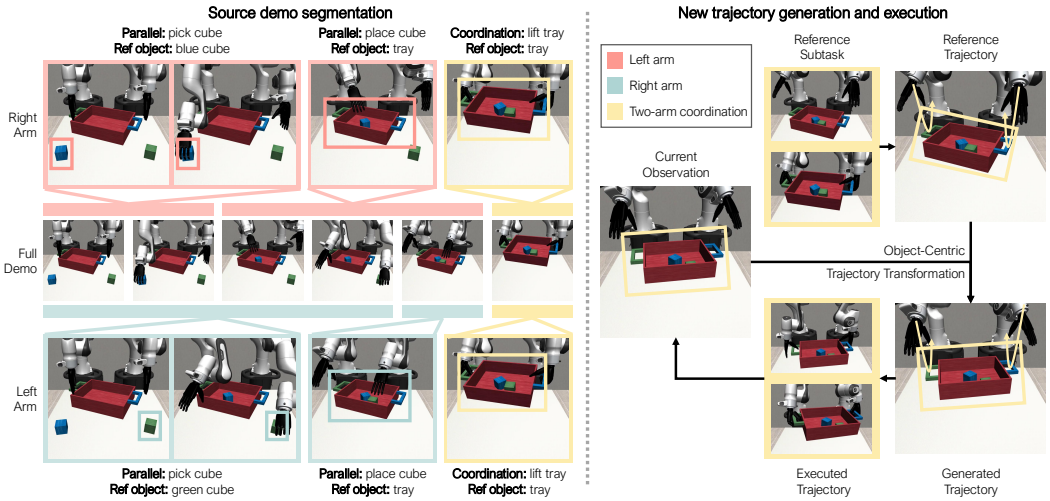

Figure 3: **DexMimicGen Workflow.** Left: DexMimicGen segments source demonstrations for each arm and records the poses of the reference objects. Right: In a new simulation environment, we generate trajectories by transforming source trajectories with reference object poses and executing them.

three types of subtasks where the two arms need to achieve sub-goals independently, with coordination, and following a specific order. MimicGen, which relies on a single subtask segmentation, struggles to handle the independent and interdependent actions required in bimanual tasks. To address these challenges, DexMimicGen incorporates a flexible per-arm subtask segmentation strategy, allowing each arm to execute its subtasks independently while still accommodating the necessary coordination phases. DexMG employs a synchronization strategy to ensure precise alignment of actions during coordination subtasks and an ordering constraint mechanism to enforce the correct order of actions during sequential subtasks. More videos are at `https://dexmimicgen.github.io/`. **Please check the full paper in Appendix**.

2 DexMimicGen Method

DexMimicGen generates data for bimanual and dexterous manipulation — this involves handling three key challenges compared to MimicGen (Fig. 2). First, each arm must be able to operate independently of the other arm to achieve different goals. Next, arms must be able to coordinate to achieve a shared goal. Finally, certain arm objectives must be completed before other objectives can be attempted. DexMimicGen handles these challenges by introducing a taxonomy of subtask types — parallel (Sec. 2.1), coordination (Sec. 2.2), and sequential (Sec. 2.3), and making changes to the data generation process to accommodate them. Sec. 2.4 provides an overview of the entire data generation process.

2.1 Parallel Subtasks

In the bimanual setting, each arm must be able to operate independently of the other arm. This makes the single fixed sequence of subtasks from MimicGen unsuitable. To enable a flexible order of completion for **parallel subtasks** involving two arms, we consider each task to consist of a sequence of subtasks for each arm. Each source demonstration is split into object-centric manipulation segments as in MimicGen, but now each arm has its own set of segments.

However, since arm subtasks are defined independently, their execution can start and end at different times that are not aligned. To accommodate this, DexMimicGen employs an asynchronous execution strategy, where an action queue is maintained for each arm. Actions are dequeued for each arm one by one in parallel. Whenever an arm's queue is empty, it is populated with the transformed subtask segment for the next subtask (using the same transformation from MimicGen). This approach allows for the execution of actions for both arms without requiring alignment between subtasks.

Task	Source Demo DP	BC-RNN-GMM	BC-RNN	DP
Piece Assembly	3.3 ± 0.9	74.0 ± 2.8	74.7 ± 2.5	$\mathbf{80.7 \pm 0.9}$
Threading	1.3 ± 0.9	54.0 ± 4.3	55.3 ± 5.0	$\mathbf{69.3 \pm 1.9}$
Transport	52.7 ± 7.7	64.0 ± 3.3	60.0 ± 0.0	$\mathbf{83.3 \pm 0.9}$
Box Cleanup	62.0 ± 1.6	64.0 ± 10.0	$\mathbf{96.7 \pm 2.5}$	92.0 ± 4.3
Drawer Cleanup	0.7 ± 0.9	30.7 ± 5.0	56.7 ± 4.1	$\mathbf{76.0 \pm 0.0}$
Tray Lift	3.3 ± 0.9	66.0 ± 8.2	75.3 ± 7.5	$\mathbf{88.7 \pm 0.9}$
Pouring	0.7 ± 0.9	74.0 ± 8.6	$\mathbf{86.0 \pm 3.3}$	79.3 ± 0.9
Coffee	14.7 ± 0.9	12.0 ± 1.6	73.3 ± 12.3	$\mathbf{77.3 \pm 0.9}$
Can Sorting	0.7 ± 0.9	75.3 ± 1.9	96.0 ± 1.6	$\mathbf{97.3 \pm 0.9}$

Table 1: Success rates (3 seeds) of image-based policies trained with BC on the source demos and DexMimicGen datasets of 1000 trajectories.

2.2 Coordination Subtasks

Some tasks require precise coordination. In these **coordination subtasks**, the relative poses between the two end-effectors during execution must be aligned with the corresponding relative poses in the source demonstration. To achieve this, we ensure that 1) both arms execute their trajectories in a synchronized manner and 2) the trajectories for both arms are generated with the same transformation. To achieve this temporal alignment, we enforce that coordination subtasks end at the same timestep during source demo segmentation. During execution, we implement a synchronization strategy in which each arm waits for the other until both have the same number of remaining steps in the coordination subtask, aligning the end of subtask execution with the subtask segmentation.

2.3 Sequential Subtasks

Some tasks require subtasks to be completed in a specific order. To handle these **sequential subtasks**, we implement an ordering constraint mechanism. We specify a pre-subtask (pouring the ball) and a post-subtask (picking the bowl) based on the task requirement. This mechanism ensures that the arm executing the post-subtask waits until the pre-subtask of the other arm is completed before continuing with the post-subtask.

2.4 Data Generation for Bimanual Manipulation

We outline the overall DexMimicGen data generation workflow. First, source demos are segmented into per-arm subtasks using heuristics or human annotation. The final subtask for each arm requires coordination (they must lift the tray together), so it is annotated as a coordination subtask for synchronization during data generation (Sec. 2.2). At the start of data generation, the scene is randomized and a source demonstration is selected (as in MimicGen). We then iteratively generate and execute trajectories for each subtask of each arm in parallel. In this example, given the pose of the reference object (the tray), we compute the relative transformation between the current tray pose and the tray pose in the source segment. We use this transformation to transform the source trajectories of both arms because these are coordination subtasks. Then we use the synchronization execution strategy described in Sec. 2.2 to execute the generated trajectory. Note that we generate finger motion by replaying the finger joint actions in the source demo because the finger movement is always relative to the end-effector movement. Each generated demonstration is only kept if the task is successful, and this process repeats until a sufficient amount of data is generated.

3 Experiments

3.1 DexMimicGen Features

DexMimicGen significantly boosts the policies' success rates over using the source demonstrations only. Robots trained on DexMimicGen's datasets outperform those trained only on the small

Task	Policy	DemoNoise	DexMimicGen
Piece Assembly	BCRNN + GMM	12.7 ± 3.4	$\mathbf{74.0 \pm 2.8}$
Tray Lift	BCRNN	16.7 ± 2.5	$\mathbf{75.3 \pm 7.5}$
Pouring	DP	26.7 ± 2.5	$\mathbf{79.3 \pm 0.9}$

Figure 4: Comparison with demo-noise baseline.

Task	Policy	D0	D1	D2
Piece Assembly	BC-RNN-GMM	74.0 ± 2.8	67.3 ± 0.9	44.0 ± 3.3
Box Cleanup	BCRNN	96.7 ± 2.5	88.0 ± 5.9	78.7 ± 2.5
Pouring	DP	79.3 ± 0.9	82.0 ± 2.0	71.3 ± 2.5

Figure 5: Success rates on broader initial distributions.

source datasets across all tasks (see Table 1). Notable improvements include policy performance on Drawer Cleanup (0.7% to 76.0% success), Threading (1.3% to 69.3%), and Piece Assembly (3.3% to 80.7%).

DexMimicGen produces capable policies across diverse initial state distributions. DexMimicGen generates datasets with broader initial state distributions (D_1, D_2) from source demos in D_0. As shown in Table 5, policies trained on these datasets are performant in the evaluation with the same broader initial state distributions, showing that DexMimicGen generates valuable datasets on new initial state distributions.

DexMimicGen generates data across different benchmarks. We apply DexMimicGen to BiGym [16], a new simulation benchmark for humanoid robots involving bimanual mobile manipulation tasks. We generate 1000 demonstrations for each of the three tasks, FlipCup, DishwasherLoadPlates, and CupBoardsCloseAll, and achieve data generation success rates of 29.1%, 43.6%, and 76.4%. The visualizations of generated demonstrations can be found on the project website.

3.2 DexMimicGen Analysis

How does DexMimicGen data generation compare to alternatives? We compare DexMimicGen with a Demo-Noise data generation baseline, which takes the same source demonstrations as DexMimicGen, but generates data by replaying the source demos with action noise during execution. In Table 4, we train policies on datasets of 1000 demos generated by both DexMimicGen and the Demo-Noise baseline. We can see that the policies trained using DexMimicGen outperform those trained on the Demo-Noise baseline by more than 58% across all tasks. Furthermore, unlike DexMimicGen, the Demo-Noise baseline cannot generate results on D_1 and D_2, as it can only replay the same initial configurations in the source demos.

How do different policy architecture choices affect success rates? In Table 1, we also compare the performance of different policy architectures (Diffusion Policy [17], BC-RNN-GMM [1], BC-RNN [1] with no GMM action head) on the datasets generated by DexMimicGen. We found that Diffusion Policy [17] generally outperforms the other architectures. Interestingly, we also found that BC-RNN-GMM generally underperformed BC-RNN and Diffusion Policy, especially on tasks that involve dexterous hands, in contrast to the RoboMimic study [1] which found the use of a GMM head to be beneficial. We believe DexMimicGen datasets will make it easier for future work to study further how imitation learning choices might differ in the bimanual dexterous manipulation setting.

3.3 Real-World Evaluation

We showcase how DexMimicGen enables real-world deployment using the pipeline illustrated in Fig. 1. We generate 40 real-world demonstrations from 4 source demonstrations with DexMimicGen using a digital twin [18] as safety insurance. We use a Fourier GR1 robot equipped with two 6-DoF Inspire dexterous hands. We compare visuomotor policies trained using Diffusion Policy [17] on the 40 DexMimicGen demos with one trained on the 4 source demos. The policy trained on DexMimicGen data achieves 90% success, while the model trained on the source data achieves 0%; DexMimicGen thus offers an efficient pipeline for training real-world robots through the use of a digital twin.

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] 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. In *Proceedings of Robotics: Science and Systems*, New York City, NY, USA, 6 2022. doi:10.15607/RSS.2022.XVIII.063.

[4] A. Brohan, Y. Chebotar, C. Finn, K. Hausman, A. Herzog, D. Ho, J. Ibarz, A. Irpan, E. Jang, R. Julian, et al. Do as i can, not as i say: Grounding language in robotic affordances. In *Conference on Robot Learning*, pages 287–318. PMLR, 2023.

[5] 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*, pages 991–1002. PMLR, 2022.

[6] 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. *IEEE Robotics and Automation Letters*, 2023.

[7] K. Darvish, L. Penco, J. Ramos, R. Cisneros, J. Pratt, E. Yoshida, S. Ivaldi, and D. Pucci. Teleoperation of humanoid robots: A survey. *IEEE Transactions on Robotics*, 39(3):1706–1727, 2023.

[8] R. Ding, Y. Qin, J. Zhu, C. Jia, S. Yang, R. Yang, X. Qi, and X. Wang. Bunny-visionpro: Real-time bimanual dexterous teleoperation for imitation learning. *arXiv preprint arXiv:2407.03162*, 2024.

[9] X. Cheng, J. Li, S. Yang, G. Yang, and X. Wang. Open-television: teleoperation with immersive active visual feedback. *arXiv preprint arXiv:2407.01512*, 2024.

[10] T. He, Z. Luo, W. Xiao, C. Zhang, K. Kitani, C. Liu, and G. Shi. Learning human-to-humanoid real-time whole-body teleoperation. *arXiv preprint arXiv:2403.04436*, 2024.

[11] T. He, Z. Luo, X. He, W. Xiao, C. Zhang, W. Zhang, K. Kitani, C. Liu, and G. Shi. Omnih2o: Universal and dexterous human-to-humanoid whole-body teleoperation and learning. *arXiv preprint arXiv:2406.08858*, 2024.

[12] Z. Fu, Q. Zhao, Q. Wu, G. Wetzstein, and C. Finn. Humanplus: Humanoid shadowing and imitation from humans. *arXiv preprint arXiv:2406.10454*, 2024.

[13] M. Dalal, A. Mandlekar, C. R. Garrett, A. Handa, R. Salakhutdinov, and D. Fox. Imitating task and motion planning with visuomotor transformers. In *Conference on Robot Learning*, pages 2565–2593. PMLR, 2023.

[14] 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. In *Forty-first International Conference on Machine Learning*, 2023.

[15] 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. In *Conference on Robot Learning*, pages 1820–1864. PMLR, 2023.

[16] N. Chernyadev, N. Backshall, X. Ma, Y. Lu, Y. Seo, and S. James. Bigym: A demo-driven mobile bi-manual manipulation benchmark. *arXiv preprint arXiv:2407.07788*, 2024.

[17] C. Chi, S. Feng, Y. Du, Z. Xu, E. Cousineau, B. Burchfiel, and S. Song. Diffusion policy: Visuomotor policy learning via action diffusion. In *Proceedings of Robotics: Science and Systems (RSS)*, 2023.

[18] Z. Jiang, C.-C. Hsu, and Y. Zhu. Ditto: Building digital twins of articulated objects from interaction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5616–5626, 2022.

DexMimicGen: Automated Data Generation for Bimanual Dexterous Manipulation via Imitation Learning

***Abstract*— Imitation learning from human demonstrations is an effective means to teach robots manipulation skills. But data acquisition is a major bottleneck in applying this paradigm more broadly, due to the amount of cost and human effort involved. There has been significant interest in imitation learning for bimanual dexterous robots, like humanoids. Unfortunately, data collection is even more challenging here due to the challenges of simultaneously controlling multiple arms and multi-fingered hands. Automated data generation in simulation is a compelling, scalable alternative to fuel this need for data. To this end, we introduce DexMimicGen, a large-scale automated data generation system that synthesizes trajectories from a handful of human demonstrations for humanoid robots with dexterous hands. We present a collection of simulation environments in the setting of bimanual dexterous manipulation, spanning a range of manipulation behaviors and different requirements for coordination among the two arms. We generate 21K demos across these tasks from just 60 source human demos and study the effect of several data generation and policy learning decisions on agent performance. Finally, we present a real-to-sim-to-real pipeline and deploy it on a real-world humanoid can sorting task.**

I. Introduction

Imitation learning from human demonstrations is an effective means to teach robots manipulation skills [1,2]. One popular approach is teleoperation, where human operators control robot arms to collect data, and the data is used to train the robots for autonomous operation [3,4]. Recent efforts have scaled this approach to collect large diverse datasets through teams of human operators, and shown that robots trained on this data can achieve impressive performance and even generalize to different settings [2,5–8]. There has also been recent interest in applying this paradigm to humanoid robot embodiments [9–14].

However, data acquisition is a key bottleneck in applying this paradigm more broadly. Prior efforts for data collection in the single robot arm setting required multiple human operators, robots, and months of human effort [2,5–8]. Unfortunately, scaling data collection for humanoids can be even more difficult, owing to the challenges of controlling multiple arms and multi-fingered dexterous hands simultaneously. Enabling real-time teleoperation for humanoids has required the development of special-purpose teleoperation interfaces [9–14], but these pipelines can be costly and difficult to scale. Furthermore, the increase in operator burden due to multi-arm and multi-finger hand control makes collecting demonstrations in this setting more challenging compared to the single-arm setting, further limiting the rate of data collection. The data acquisition burden is further compounded by the fact that data requirements in the humanoid setting can be higher due to the increased task complexity.

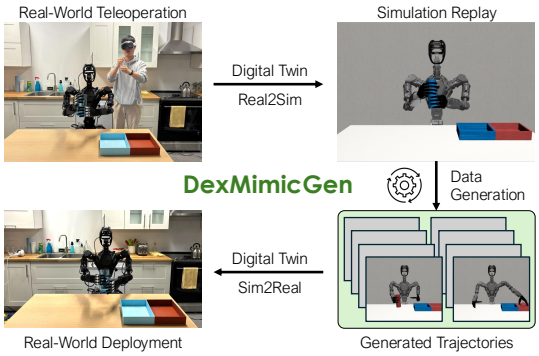

Fig. 1: **DexMimicGen Overview.** DexMimicGen offers an efficient pipeline to train capable bimanual dexterous robots. (left) First, a human operator collects around five demonstrations of a task using a teleoperation device. (middle) Next, DexMimicGen automatically generates a large set of demonstration trajectories in simulation. (right) Finally, a policy is trained with imitation learning and deployed in the real world.

Instead, leveraging automated data generation in simulation is a compelling alternative that has proved effective for the single-arm robot manipulation setting [15–17]. Inspired by this success, **we introduce DexMimicGen (DexMG), a large-scale automated data generation system for bimanual robots with dexterous hands, such as humanoids**. The core idea is to leverage a small set of human demonstrations and use demonstration transformation and replay in physical simulation to automatically generate large amounts of training data suitable for imitation learning in the bimanual dexterous manipulation setting. This system builds on top of MimicGen [17], which proposed a similar pipeline for the single-arm with parallel-jaw gripper setting, but there are several technical challenges that DexMimicGen overcomes to apply the same principles to our setting.

MimicGen relies on decomposing each task into a sequence of subtasks, to generate trajectories for each subtask separately and then stitch them together. Bimanual dexterous manipulation involves three types of subtasks where the two arms need to achieve sub-goals independently, with coordination, and following a specific order. MimicGen, which relies on a single subtask segmentation, struggles to handle the independent and interdependent actions required in bimanual tasks. To address these challenges, DexMimicGen incorporates a flexible per-arm subtask segmentation strategy, allowing each arm to execute its subtasks independently while still accommodating the necessary coordination phases. DexMG employs a synchronization strategy to ensure precise alignment of actions during coordination subtasks and an ordering constraint mechanism to enforce the correct order

of actions during sequential subtasks.

We make the following contributions:

• We introduce DexMimicGen (DexMG), a data generation system that automatically synthesizes trajectories from a small number of human demonstrations for bimanual and dexterous robot manipulation. We introduce several key design features, including an asynchronous per-arm execution strategy, synchronization, and sequential constraints that enable handling multi-arm coordination.

• We introduce a suite of nine simulation environments across three different embodiment types requiring different coordination behaviors between the two arms. We apply DexMimicGen to generate 21K demos across these tasks from merely 60 source human demos and study the effect of several data generation and policy learning decisions on agent performance. **We will release the simulations and datasets to facilitate future study into the bimanual and dexterous manipulation setting.**

• We create a simulated digital twin of a real-world can-sorting task, replay real-world human demonstrations in the simulation, synthesize trajectories with DexMimicGen, and then transfer the generated trajectories back into the real world, producing a visuomotor policy of 90% success rate, as opposed to 0% from just using the human demos (Fig. 1).

II. Related Work

Data Collection through Teleoperation. Teleoperation is a prevalent approach to gather task demonstrations for robotics [3,4,18–23]. Human operators use an interface to control a robot in real time, and sensor data and robot control commands are logged to a dataset. Some systems allow data collection for multiple robot arms [24–26] and humanoids [9–14,27], and some also enable robot-free data collection using specialized hardware [28–30]. However, all these methods require significant human time and resources to collect large datasets. Some other efforts use pre-programmed experts to automate data generation in simulation [15,16,31–35], but applying these methods to challenging scenarios involving multi-arm coordination can be difficult. By contrast, DexMimicGen builds on MimicGen [17,36,37] to automate data generation using a handful of human demonstrations, greatly reducing the human effort involved in collecting large datasets.

Imitation Learning and Data Augmentation. Behavioral Cloning [38] is an established framework for learning robot control policies from demonstrations and has been used extensively in prior work [32,39–49], including for bimanual manipulators [24–26,50] and humanoid robots [10,11,27,51,52]. In this work, we apply existing imitation learning methods [1,53] to datasets generated by DexMimicGen, but DexMimicGen can play a significant role in facilitating algorithm development for bimanual manipulation by making simulation-based bimanual manipulation datasets more widely accessible and providing easy to reproduce results. Some works leverage offline data augmentation to increase the size of demonstration datasets [1,54–68]. By contrast, DexMimicGen collects datasets online, ensuring the generated trajectories are physically valid.

III. Prerequisites

Imitation Learning. We formalize each manipulation task as a Partially Observable Markov Decision Process (POMDP). We are given 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 episode starts in a state $s_0^i \sim D$ sampled from the initial state distribution $D \subseteq \mathcal{S}$. The goal is to learn a policy $\pi : \mathcal{O} \rightarrow \mathcal{A}$ that maps observations to a distribution over the action space. We focus on Behavioral Cloning (BC) [38] methods that find a policy via the maximum likelihood objective $\arg\max_\theta \mathbb{E}_{(s,o,a)\sim\mathcal{D}}[\log \pi_\theta(a \mid o)]$. In this work, we train our policies with datasets generated via DexMimicGen.

Assumptions. Like MimicGen [17], we make these assumptions. (**A1**): the action space $\mathcal{A}$ consists of the following components for each robot arm: a pose command for an end effector controller and an actuation command for the hand (1-D open/close for parallel-jaw gripper, 6-D joint commands for dexterous hand). (**A2**): Each task can be divided into object-centric subtasks (see Sec. IV-A). (**A3**): During data collection, an object's pose can be observed or estimated prior to a robot arm making contact with that object.

MimicGen. MimicGen [17] uses a small number of source human demonstrations $\mathcal{D}_{\text{src}}$ to generate a large dataset $\mathcal{D}$. It assumes that every task consists of a sequence of object-centric subtasks ($S_1(o_1)$, $S_2(o_2)$, $\ldots$, $S_M(o_M)$) where the manipulation in each subtask $S_i(o_i)$ is relative to a single object's coordinate frame ($o_i \in \mathcal{O}$, where $\mathcal{O}$ is the set of objects in the task). It divides each source demo $\tau \in D_{src}$ into contiguous object-centric manipulation segments $\{\tau_i\}_{i=1}^M$, each of which corresponds to a subtask $S_i(o_i)$. Each segment is a sequence of end effector control poses $\tau_i = (T_W^{C_0}, T_W^{C_1}, ..., T_W^{C_K})$ where W is the world reference frame. This segmentation can be done with human annotation or using heuristics. To generate a new demonstration in a novel scene, it observes the pose of the object for the current subtask $T_W^{o_i'}$, and transforms the poses in a source human segment (with a constant SE(3) transform $T_W^{o_i'}(T_W^{o_i})^{-1}$) such that relative poses between the end effector and object frame are preserved between the source segment and the new scene. It then adds poses to the start of the segment to interpolate between the robot's current state and the start of the transformed segment. Then, it executes the entire sequence of poses open-loop using the robot end effector controller and repeats this process for the next subtask. It checks for task success after executing all subtasks and only keeps the demonstration if it was successful.

IV. DexMimicGen Method

DexMimicGen generates data for bimanual and dexterous manipulation — this involves handling three key challenges compared to MimicGen. First, each arm must be able to operate independently of the other arm to achieve different goals. Next, arms must be able to coordinate to achieve a

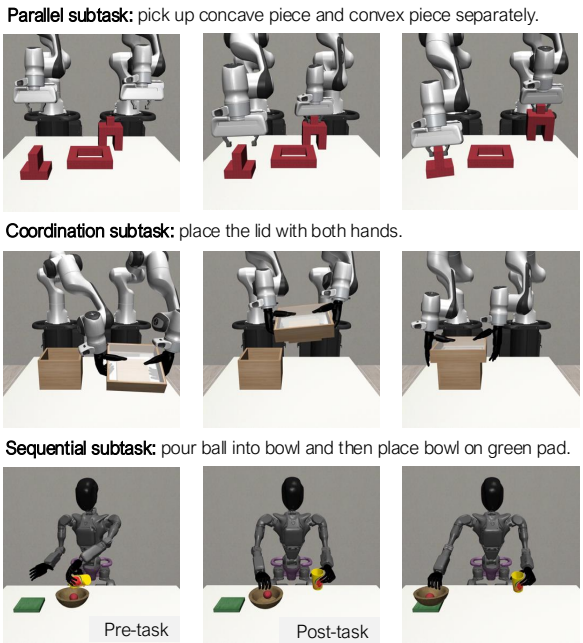

Fig. 2: **Subtask Types.** We categorize the subtasks into parallel, coordination, and sequential subtasks, where two arms achieve subgoals independently, with coordination, and following a specific order.

shared goal. Finally, certain arm objectives must be completed before other objectives can be attempted. DexMimicGen handles these challenges by introducing a taxonomy of subtask types (Fig. 2) — parallel (Sec. IV-A), coordination (Sec. IV-B), and sequential (Sec. IV-C), and making changes to the data generation process to accommodate them. Sec. IV-D provides an overview of the entire data generation process.

A. Parallel Subtasks

In the bimanual setting, each arm must be able to operate independently of the other arm. For example, at the start of the Piece Assembly task (Fig. 2 top), each arm needs to grasp a separate object and might finish grasping the object at different points in time. This makes the single fixed sequence of subtasks from MimicGen unsuitable. To enable a flexible order of completion for **parallel subtasks** involving two arms, we consider each task to consist of a sequence of subtasks for each arm: $S_1^{a_1}(o_1)$, ..., $S_{M_1}^{a_1}(o_{M_1})$ and $S_1^{a_2}(o_1)$, ..., $S_{M_2}^{a_2}(o_{M_2})$. Each source demonstration is split into object-centric manipulation segments as in MimicGen, but now each arm has its own set of segments ($\{\tau_i^n\}_{i=1}^{M_n}$, $n \in \{1, 2\}$).

However, since arm subtasks are defined independently, their execution can start and end at different times that are not aligned. To accommodate this, DexMimicGen employs an asynchronous execution strategy, where an action queue is maintained for each arm. Actions are dequeued for each arm one by one in parallel. Whenever an arm's queue is empty, it is populated with the transformed subtask segment for the next subtask (using the same transformation from MimicGen). This approach allows for the execution of actions for both arms without requiring alignment between subtasks.

B. Coordination Subtasks

Some tasks require precise coordination, such as placing the lid in the Box Cleanup task (Fig. 2 middle). In these **coordination subtasks**, the relative poses between the two end-effectors during execution must be aligned with the corresponding relative poses in the source demonstration. To achieve this, we ensure that 1) both arms execute their trajectories in a synchronized manner and 2) the trajectories for both arms are generated with the same transformation. To achieve this temporal alignment, we enforce that coordination subtasks end at the same timestep during source demo segmentation. During execution, we implement a synchronization strategy in which each arm waits for the other until both have the same number of remaining steps in the coordination subtask, aligning the end of subtask execution with the subtask segmentation.

We provide different source demonstration transformation schemes to acquire the common transformation matrix for both arms in coordination subtasks. These include the Transform and Replay schemes. The Transform scheme utilizes the transformation matrix $T_W^{o_i'}(T_W^{o_i})^{-1}$ computed from the object pose at the moment the first arm begins the coordination subtask $T_W^{o_i'}$ and the object pose in the corresponding source segment $T_W^{o_i}$. In contrast, the replay scheme directly uses the source trajectories without applying any transformation. The replay scheme can be beneficial for coordination subtasks like handover because it ensures the trajectory remains within kinematic limits and is fully executable.

C. Sequential Subtasks

Some tasks require subtasks to be completed in a specific order. For example, in the Pouring task (Fig. 2 bottom), the robot must pour the ball into the bowl with one hand before moving the bowl to the pad with the other hand. To handle these **sequential subtasks**, we implement an ordering constraint mechanism. We specify a pre-subtask (pouring the ball) and a post-subtask (picking the bowl) based on the task requirement. This mechanism ensures that the arm executing the post-subtask waits until the pre-subtask of the other arm is completed before continuing with the post-subtask.

D. Data Generation for Bimanual Manipulation

We outline the overall DexMimicGen data generation workflow using the Tray Lift task as an example. First, source demos are segmented into per-arm subtasks using heuristics or human annotation (Fig. 3 left). The final subtask for each arm requires coordination (they must lift the tray together), so it is annotated as a coordination subtask for synchronization during data generation (Sec. IV-B).

At the start of data generation, the scene is randomized and a source demonstration is selected (as in MimicGen). We then iteratively generate and execute trajectories for each subtask of each arm in parallel (see Fig. 3 right). In this example, given the pose of the reference object (the tray), we compute the relative transformation between the current tray pose and the tray pose in the source segment. We use this transformation to transform the source trajectories of both

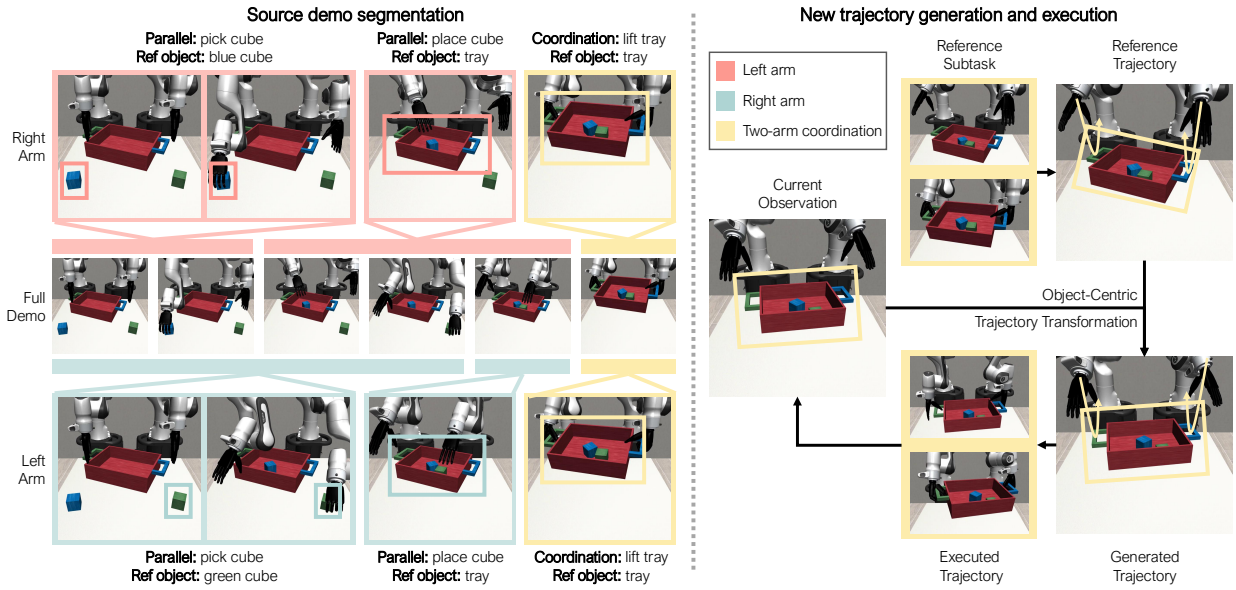

Fig. 3: **DexMimicGen Workflow.** Left: DexMimicGen segments source demonstrations for each arm and records the poses of the reference objects. Right: In a new simulation environment, we generate trajectories by transforming source trajectories with reference object poses and executing them.

arms because these are coordination subtasks. Then we use the synchronization execution strategy described in Sec. IV-B to execute the generated trajectory. Note that we generate finger motion by replaying the finger joint actions in the source demo because the finger movement is always relative to the end-effector movement. Each generated demonstration is only kept if the task is successful, and this process repeats until a sufficient amount of data is generated.

V. System Design

In order to instantiate DexMimicGen, we build a large collection of simulation environments and a teleoperation system allowing for source human demonstration collection in both simulation and the real world.

Simulation Environments. We introduce a diverse range of setups and tasks to demonstrate the capability of DexMimicGen to generate data across different embodiments and manipulation behaviors. The tasks are developed in RoboSuite [69] and use MuJoCo [70] for physics simulation. We focus on three embodiments: (1) bimanual Panda arms equipped with parallel-jaw grippers, (2) bimanual Panda arms with dexterous hands, and (3) a GR-1 humanoid equipped with dexterous hands. We apply different controllers for different embodiments. For the Panda arms, we leverage the Operational Space Control (OSC) [71] framework, which converts the delta end-effector pose into joint torque commands. For the humanoid, we implemented an Inverse Kinematics (IK) controller based on mink [72,73]. We found this to be an effective approach to deal with the complexity of the humanoid kinematic tree, where both arms are linked to a single torso. The IK controller translates global target end-effector poses into robot joint positions. For finger control, we directly use joint position control.

For each embodiment, we introduce three tasks, resulting in a total of nine tasks, as depicted in Fig. 4. These tasks involve high-precision manipulation (Threading, Piece Assembly, Box Packing, Coffee), manipulation of articulated objects (Drawer), and are long-horizon (Transport). The tasks also require overcoming key challenges in multi-arm interaction. Several of these tasks contain coordination subtasks, where both arms need to cooperate to finish the subtask (Threading, Transport, Box Packing, Tray Lift, Can Sorting). Other tasks necessitate sequential subtask execution (Piece Assembly, Drawer Cleanup, Pouring, Coffee). We also introduce task variants that broaden the default reset distribution D_0 for certain tasks, as in MimicGen. For instance, in the Pouring task, D_1 represents a variant where objects have a larger initial reset distribution, while in D_2, the reset positions of the bowl and the green pad are swapped. These simulation environments along with the datasets generated by DexMimicGen provide a valuable platform to analyze various factors that influence the performance of imitation learning in the bimanual and dexterous manipulation setting.

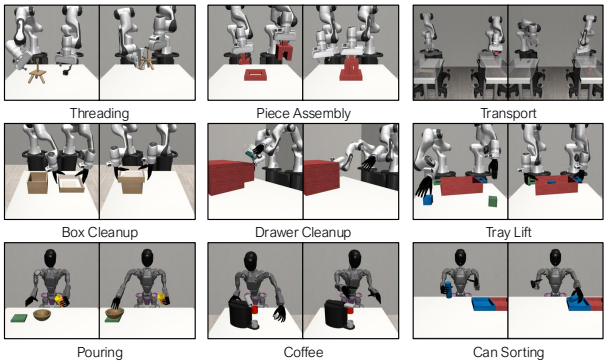

Fig. 4: **Simulation Tasks.** We deploy DexMimicGen on nine simulation tasks across three embodiments — two arms with parallel-jaw grippers (top), two arms with dexterous hands (middle), and a humanoid (bottom)

Teleoperation System. To collect source demonstrations for the tasks, we employ different teleoperation methods tailored to each embodiment. For bimanual Panda arms equipped with parallel-jaw grippers, we use an iPhone-based

Task	Source Demo DP	BC-RNN-GMM	BC-RNN	DP
Piece Assembly	3.3 ± 0.9	74.0 ± 2.8	74.7 ± 2.5	$\mathbf{80.7 \pm 0.9}$
Threading	1.3 ± 0.9	54.0 ± 4.3	55.3 ± 5.0	$\mathbf{69.3 \pm 1.9}$
Transport	52.7 ± 7.7	64.0 ± 3.3	60.0 ± 0.0	$\mathbf{83.3 \pm 0.9}$
Box Cleanup	62.0 ± 1.6	64.0 ± 10.0	$\mathbf{96.7 \pm 2.5}$	92.0 ± 4.3
Drawer Cleanup	0.7 ± 0.9	30.7 ± 5.0	56.7 ± 4.1	$\mathbf{76.0 \pm 0.0}$
Tray Lift	3.3 ± 0.9	66.0 ± 8.2	75.3 ± 7.5	$\mathbf{88.7 \pm 0.9}$
Pouring	0.7 ± 0.9	74.0 ± 8.6	$\mathbf{86.0 \pm 3.3}$	79.3 ± 0.9
Coffee	14.7 ± 0.9	12.0 ± 1.6	73.3 ± 12.3	$\mathbf{77.3 \pm 0.9}$
Can Sorting	0.7 ± 0.9	75.3 ± 1.9	96.0 ± 1.6	$\mathbf{97.3 \pm 0.9}$

TABLE I: Success rates (3 seeds) of image-based policies trained with BC on the source demos and DexMimicGen datasets of 1000 trajectories.

Task	Policy	D0	D1	D2
Piece Assembly	BC-RNN-GMM	74.0 ± 2.8	67.3 ± 0.9	44.0 ± 3.3
Box Cleanup	BCRNN	96.7 ± 2.5	88.0 ± 5.9	78.7 ± 2.5
Pouring	DP	79.3 ± 0.9	82.0 ± 2.0	71.3 ± 2.5

TABLE II: Success rates of policy trained on data generated with broader initial distributions, evaluated with same broader initial distributions.

teleoperation interface, as introduced in RoboTurk [4,24], to capture human wrist and gripper actions. For robots equipped with dexterous hands, we implemented an Apple Vision Pro-based teleoperation system. Specifically, we employ the VisionProTeleop software [74] to collect wrist and finger poses via Apple Vision Pro. We first need to align the human and the robot to convert the raw human end effector poses to robot poses. We design a human-to-robot calibration process asking the human teleoperator to start with a fixed pose, and we automatically compute the relative transformation matrices that map the human poses to robot targets. This calibration process adapts to both bimanual panda arms with dexterous hands and the GR-1 humanoid. We use the retargeting method provided by OmniH2O [13] to retarget human finger pose to robot finger joint positions. This teleoperation system converts human actions to robot action targets, allowing us to collect demonstrations intuitively.

VI. Experiments

In this section, we provide empirical evidence showcasing the efficacy of DexMimicGen. We first provide details on experiment setup (Sec. VI-A), then highlight DexMimicGen features and applications (Sec. VI-B), then analyze how data generation and policy learning choices impact policy performance (Sec. VI-C), and finally present a real-world application of the DexMimicGen system (Sec. VI-D).

A. Experimental Setup

We collect ten source human demonstrations for each task with parallel-jaw grippers, but only five demonstrations for those involving dexterous hands due to the additional operator burden and time cost of collecting demonstrations for dexterous hands. DexMimicGen is subsequently used to generate 1000 demonstrations per task. Each dataset was used to train visuomotor policies through Behavioral Cloning with an RNN [1], an RNN-GMM [1], and a Diffusion Policy [53]. For evaluation, we follow the procedure in [1,17]: we run each experiment across 3 different seeds, and take the maximum policy success rate for each seed.

B. DexMimicGen Features

DexMimicGen significantly boosts the policies' success rates over using the source demonstrations only. Robots

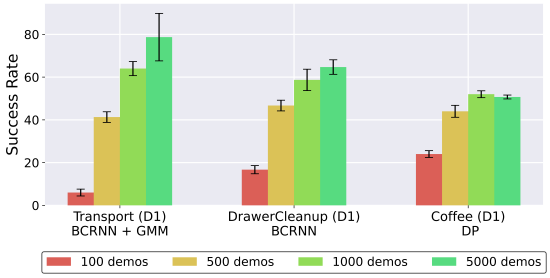

Fig. 5: **Dataset Size Comparison.** Success rates of policies trained on datasets with different sizes

Task	Policy	DemoNoise	DexMimicGen
Piece Assembly	BCRNN + GMM	12.7 ± 3.4	$\mathbf{74.0 \pm 2.8}$
Tray Lift	BCRNN	16.7 ± 2.5	$\mathbf{75.3 \pm 7.5}$
Pouring	DP	26.7 ± 2.5	$\mathbf{79.3 \pm 0.9}$

TABLE III: Success rates of policies trained on data generated with DexMimicGen and Demo-noise baseline.

trained on DexMimicGen's datasets outperform those trained only on the small source datasets across all tasks (see Table I). Notable improvements include policy performance on Drawer Cleanup (0.7% to 76.0% success), Threading (1.3% to 69.3%), and Piece Assembly (3.3% to 80.7%).

DexMimicGen produces capable policies across diverse initial state distributions. DexMimicGen generates datasets with broader initial state distributions (D_1, D_2) from source demos in D_0. As shown in Table II, policies trained on these datasets are performant in the evaluation with the same broader initial state distributions, showing that DexMimicGen generates valuable datasets on new initial state distributions.

DexMimicGen generates data across different benchmarks. We apply DexMimicGen to BiGym [75], a new simulation benchmark for humanoid robots involving bimanual mobile manipulation tasks. We generate 1000 demonstrations for each of the three tasks, FlipCup, DishwasherLoadPlates, and CupBoardsCloseAll, and achieve data generation success rates of 29.1%, 43.6%, and 76.4%. The visualizations of generated demonstrations can be found on the project website.

C. DexMimicGen Analysis

How does DexMimicGen data generation compare to alternatives? We compare DexMimicGen with a Demo-Noise data generation baseline, which takes the same source demonstrations as DexMimicGen, but generates data by replaying the source demos with action noise during execution. In Table III, we train policies on datasets of 1000 demos generated by both DexMimicGen and the Demo-Noise baseline. We can see that the policies trained using DexMimicGen outperform those trained on the Demo-Noise baseline by more than 58% across all tasks. Furthermore, unlike DexMimicGen, the Demo-Noise baseline cannot generate results on D_1 and D_2, as it can only replay the same initial configurations in the source demos.

Do larger datasets boost policy performance? We train policies on 100, 500, 1000, and 5000 demos generated by DexMimicGen across several tasks (Fig. 5). We observe large

boosts in performance from 100 to 500 and 1000, showing that increasing dataset size boosts performance in this data regime; however, the success rate does not always increase from 1000 to 5000, suggesting that there can be diminishing returns depending on the task.

How do different DexMimicGen data generation strategies impact results? First, we compare the Replay and Transform schemes in the coordination subtask (Sec. IV-B). Specifically, we evaluate two tasks involving the handover subtask with two distinct policies: Transport using BCRNN+GMM, and Can Sorting using a diffusion policy. Replay demonstrates better policy performance (63.3% vs. 46.0%) in the Transport task and achieves comparable outcomes (97.3% vs. 98.6%) in the Can Sorting task. Thus, Replay is our default choice for subtasks that involve handover.

Next, we assess the effectiveness of ordering constraints in sequential subtasks (Sec. IV-C). When using the same source demonstration for both arms, subtask ordering requirements are typically satisfied automatically. In contrast, employing different source demonstrations for each arm requires an ordering constraint but also increases data diversity. We also evaluate two tasks involving the sequential subtasks with two distinct policies: Drawer Cleanup with BCRNN, and Pouring with diffusion policy. We found training on data generated with ordering constraints consistently outperforms training without them (50.7% vs. 48.0% in Drawer Cleanup and 88.7% vs. 76.7% in Pouring). Directly using the same source demo yields the policy success rates of 56.7% in the Drawer Cleanup and 79.3% in Pouring.

How do different policy architecture choices affect success rates? In Table I, we also compare the performance of different policy architectures (Diffusion Policy [53], BC-RNN-GMM [1], BC-RNN [1] with no GMM action head) on the datasets generated by DexMimicGen. We found that Diffusion Policy [53] generally outperforms the other architectures. Interestingly, we also found that BC-RNN-GMM generally underperformed BC-RNN and Diffusion Policy, especially on tasks that involve dexterous hands, in contrast to the RoboMimic study [1] which found the use of a GMM head to be beneficial. We believe DexMimicGen datasets will make it easier for future work to study further how imitation learning choices might differ in the bimanual dexterous manipulation setting.

D. Real-World Evaluation

We showcase how DexMimicGen enables real-world deployment using the pipeline illustrated in Fig. 1. We generate real-world demonstrations from source demonstrations using a digital twin [76] as safety insurance.

Hardware Setup. We use a Fourier GR1 robot equipped with two 6-DoF Inspire dexterous hands. For vision, we use two Intel RealSense D435i cameras: one head-mounted camera provides a first-person view and one camera in front of the robot as a third-person view.

Digital Twin Setup. We perform our experiment on the Can Sorting task (Fig. 6), with digital twin assets in simulation that align with the real-world setup. To ensure accurate alignment between the real-world and simulated environments, we perform pose estimation on the objects prior to data collection. Using the head-mounted camera, we capture an initial RGB-D frame and apply GroundingDINO [77] to segment an RGB mask of the object. We use the real world object's center point (determined by averaging the depth values within the RGB mask) to initialize the object's $x-$ and $y-$coordinates in simulation.

Fig. 6: **Real-World DexMimicGen Deployment.** Rollouts of real-world visuomotor policy trained with DexMimicGen data and digital twin.

Data Collection Pipeline. Using the teleoperation pipeline described in Sec. V, we collect 4 source human demonstrations for the Can Sorting task. These demonstrations are replayed in simulation, and are used as source demonstrations for DexMimicGen in the digital twin. Next, new real-world demonstrations are collected by synchronizing the initial object state from real to sim, and then attempting to generate a new demonstration in sim with DexMimicGen. If the demonstration is successful in simulation, the sequence of robot control actions is sent to the real-world for execution. In this way, the digital twin functions to ensure safety during real-world data generation, while DexMimicGen mitigates human effort for data collection, which is autonomous apart from the environment resets. We generate 40 successful demonstrations with the approach described above.

Results. We compare visuomotor policies trained using Diffusion Policy [53] on the 40 DexMimicGen demos with one trained on the 4 source demos. We evaluated both models by running 10 trials each for the red and blue cups. The policy trained on DexMimicGen data achieves 90% success, while the model trained on the source data achieves 0%; DexMimicGen thus offers an efficient pipeline for training real-world robots through the use of a digital twin.

VII. Conclusion

We introduce DexMimicGen, a large-scale automated data generation system that synthesizes trajectories from a small number of human demonstrations for bimanual and dexterous robots, and a collection of simulation environments across 3 embodiments requiring different coordination behaviors. Our findings from applying DexMimicGen to these tasks show that there is great value in further investigating policy learning in this setting. We hope the release of our DexMimicGen datasets and environments will facilitate future research.

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, X. Chen, K. Goldberg, and P. Abbeel, "Deep imitation learning for complex manipulation tasks from virtual reality teleoperation," in *2018 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2018, pp. 5628–5635.

[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," in *Proceedings of Robotics: Science and Systems*, New York City, NY, USA, 6 2022.

[6] A. Brohan, Y. Chebotar, C. Finn, K. Hausman, A. Herzog, D. Ho, J. Ibarz, A. Irpan, E. Jang, R. Julian, *et al.*, "Do as i can, not as i say: Grounding language in robotic affordances," in *Conference on Robot Learning*. PMLR, 2023, pp. 287–318.

[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," *IEEE Robotics and Automation Letters*, 2023.

[9] K. Darvish, L. Penco, J. Ramos, R. Cisneros, J. Pratt, E. Yoshida, S. Ivaldi, and D. Pucci, "Teleoperation of humanoid robots: A survey," *IEEE Transactions on Robotics*, vol. 39, no. 3, pp. 1706–1727, 2023.

[10] R. Ding, Y. Qin, J. Zhu, C. Jia, S. Yang, R. Yang, X. Qi, and X. Wang, "Bunny-visionpro: Real-time bimanual dexterous teleoperation for imitation learning," *arXiv preprint arXiv:2407.03162*, 2024.

[11] X. Cheng, J. Li, S. Yang, G. Yang, and X. Wang, "Open-television: teleoperation with immersive active visual feedback," *arXiv preprint arXiv:2407.01512*, 2024.

[12] T. He, Z. Luo, W. Xiao, C. Zhang, K. Kitani, C. Liu, and G. Shi, "Learning human-to-humanoid real-time whole-body teleoperation," *arXiv preprint arXiv:2403.04436*, 2024.

[13] T. He, Z. Luo, X. He, W. Xiao, C. Zhang, W. Zhang, K. Kitani, C. Liu, and G. Shi, "Omnih2o: Universal and dexterous human-to-humanoid whole-body teleoperation and learning," *arXiv preprint arXiv:2406.08858*, 2024.

[14] Z. Fu, Q. Zhao, Q. Wu, G. Wetzstein, and C. Finn, "Humanplus: Humanoid shadowing and imitation from humans," *arXiv preprint arXiv:2406.10454*, 2024.

[15] M. Dalal, A. Mandlekar, C. R. Garrett, A. Handa, R. Salakhutdinov, and D. Fox, "Imitating task and motion planning with visuomotor transformers," in *Conference on Robot Learning*. PMLR, 2023, pp. 2565–2593.

[16] 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," in *Forty-first International Conference on Machine Learning*, 2023.

[17] 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," in *Conference on Robot Learning*. PMLR, 2023, pp. 1820–1864.

[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," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2019, pp. 1048–1055.

[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] 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.

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

[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. Martín-Martín, "Telemoma: A modular and versatile teleoperation system for mobile manipulation," in *2nd Workshop on Mobile Manipulation and Embodied Intelligence at ICRA 2024*, 2024.

[24] A. Tung, J. Wong, A. Mandlekar, R. Martín-Martín, Y. Zhu, L. Fei-Fei, and S. Savarese, "Learning multi-arm manipulation through collaborative teleoperation," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 9212–9219.

[25] T. Z. Zhao, V. Kumar, S. Levine, and C. Finn, "Learning Fine-Grained Bimanual Manipulation with Low-Cost Hardware," in *Proceedings of Robotics: Science and Systems*, Daegu, Republic of Korea, 7 2023.

[26] 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.

[27] S. Schaal, "Is imitation learning the route to humanoid robots?" *Trends in cognitive sciences*, vol. 3, no. 6, pp. 233–242, 1999.

[28] 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," in *Towards Generalist Robots: Learning Paradigms for Scalable Skill Acquisition@ CoRL2023*, 2023.

[29] 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," in *Proceedings of Robotics: Science and Systems (RSS)*, 2024.

[30] H. Etukuru, N. Naka, Z. Hu, S. Lee, J. Mehu, A. Edsinger, C. Paxton, S. Chintala, L. Pinto, and N. M. M. Shafiullah, "Robot utility models: General policies for zero-shot deployment in new environments," 2024.

[31] 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.

[32] 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," in *Conference on Robot Learning*. PMLR, 2021, pp. 726–747.

[33] 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," in *Fortieth International Conference on Machine Learning*, 2023.

[34] 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," in *The Eleventh International Conference on Learning Representations*, 2023.

[35] H. Ha, P. Florence, and S. Song, "Scaling up and distilling down: Language-guided robot skill acquisition," in *Conference on Robot Learning*. PMLR, 2023, pp. 3766–3777.

[36] 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

[37] S. Nasiriany, A. Maddukuri, L. Zhang, A. Parikh, A. Lo, A. Joshi, A. Mandlekar, and Y. Zhu, "Robocasa: Large-scale simulation of everyday tasks for generalist robots," in *Robotics: Science and Systems (RSS)*, 2024.

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

[39] 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.

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

[41] 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.

[42] A. Mandlekar, D. Xu, R. Martín-Martín, S. Savarese, and L. Fei-Fei, "GTI: Learning to Generalize across Long-Horizon Tasks from Human Demonstrations," in *Proceedings of Robotics: Science and Systems*, Corvalis, Oregon, USA, 7 2020.

[43] C. Wang, R. Wang, A. Mandlekar, L. Fei-Fei, S. Savarese, and D. Xu, "Generalization through hand-eye coordination: An action space for learning spatially-invariant visuomotor control," in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2021, pp. 8913–8920.

[44] 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.

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

[46] 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.

[47] 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.

[48] 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.

[49] 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.

[50] M. Drolet, S. Stepputtis, S. Kailas, A. Jain, J. Peters, S. Schaal, and H. Ben Amor, "A comparison of imitation learning algorithms for bimanual manipulation," *IEEE Robotics and Automation Letters (RA-L)*, 2024.

[51] 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.

[52] M. Seo, S. Han, K. Sim, S. H. Bang, C. Gonzalez, L. Sentis, and Y. Zhu, "Deep imitation learning for humanoid loco-manipulation through human teleoperation," in *2023 IEEE-RAS 22nd International Conference on Humanoid Robots (Humanoids)*. IEEE, 2023, pp. 1–8.

[53] C. Chi, S. Feng, Y. Du, Z. Xu, E. Cousineau, B. Burchfiel, and S. Song, "Diffusion policy: Visuomotor policy learning via action diffusion," in *Proceedings of Robotics: Science and Systems (RSS)*, 2023.

[54] P. Mitrano and D. Berenson, "Data Augmentation for Manipulation," in *Proceedings of Robotics: Science and Systems*, New York City, NY, USA, 6 2022.

[55] M. Laskin, K. Lee, A. Stooke, L. Pinto, P. Abbeel, and A. Srinivas, "Reinforcement learning with augmented data," *Advances in neural information processing systems*, vol. 33, pp. 19 884–19 895, 2020.

[56] D. Yarats, I. Kostrikov, and R. Fergus, "Image augmentation is all you need: Regularizing deep reinforcement learning from pixels," in *International conference on learning representations*, 2021.

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

[58] A. Zhan, R. Zhao, L. Pinto, P. Abbeel, and M. Laskin, "A framework for efficient robotic manipulation," in *Deep RL Workshop NeurIPS 2021*, 2021.

[59] 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.

[60] 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.

[61] S. Pitis, E. Creager, A. Mandlekar, and A. Garg, "Mocoda: model-based counterfactual data augmentation," in *Proceedings of the 36th International Conference on Neural Information Processing Systems*, 2022, pp. 18 143–18 156.

[62] 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," in *CoRL 2022 Workshop on Pre-training Robot Learning*, 2022.

[63] 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.

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

[65] 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," in *First Workshop on Out-of-Distribution Generalization in Robotics at CoRL 2023*, 2023.

[66] X. Zhang, M. Chang, P. Kumar, and S. Gupta, "Diffusion meets dagger: Supercharging eye-in-hand imitation learning," *arXiv preprint arXiv:2402.17768*, 2024.

[67] S. Tian, B. Wulfe, K. Sargent, K. Liu, S. Zakharov, V. Guizilini, and J. Wu, "View-invariant policy learning via zero-shot novel view synthesis," in *Conference on Robot Learning (CoRL)*, Munich, Germany, 2024.

[68] L. Y. Chen, C. Xu, K. Dharmarajan, M. Z. Irshad, R. Cheng, K. Keutzer, M. Tomizuka, Q. Vuong, and K. Goldberg, "Rovi-aug: Robot and viewpoint augmentation for cross-embodiment robot learning," in *Conference on Robot Learning (CoRL)*, Munich, Germany, 2024.

[69] 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.

[70] 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.

[71] 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.

[72] K. Zakka, "mink," 2024. [Online]. Available: https://github.com/kevinzakka/mink

[73] S. Caron, Y. De Mont-Marin, R. Budhiraja, S. H. Bang, I. Domrachev, and S. Nedelchev, "Pink: Python inverse kinematics based on Pinocchio," 2024. [Online]. Available: https://github.com/stephane-caron/pink

[74] Y. Park and P. Agrawal, "Using apple vision pro to train and control robots," 2024. [Online]. Available: https://github.com/Improbable-AI/VisionProTeleop

[75] N. Chernyadev, N. Backshall, X. Ma, Y. Lu, Y. Seo, and S. James, "Bigym: A demo-driven mobile bi-manual manipulation benchmark," *arXiv preprint arXiv:2407.07788*, 2024.

[76] Z. Jiang, C.-C. Hsu, and Y. Zhu, "Ditto: Building digital twins of articulated objects from interaction," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 5616–5626.

[77] S. Liu, Z. Zeng, T. Ren, F. Li, H. Zhang, J. Yang, C. Li, J. Yang, H. Su, J. Zhu, *et al.*, "Grounding dino: Marrying dino with grounded pre-training for open-set object detection," *arXiv preprint arXiv:2303.05499*, 2023.