AutoTrialGen: Automated Data Generation from Few Human Demonstrations via Trajectory Annotation and Simulation Trials

Huailiang Ma[1], Aiguo Song[1], Mutian He[1], Mingyu Li[1], Yibing Yan[1] and Linhu Wei[1]

***Abstract*—While imitation learning is a powerful paradigm for teaching robots complex manipulation skills, its effectiveness is often bottlenecked by the need for large-scale, human-collected datasets. This paper presents AutoTrialGen, an automated framework designed to generate large and diverse datasets of successful simulated demonstrations through trial-and-error, from only a few human demonstrations. Initially, AutoTrialGen employs a vision-language model to automatically decompose unprocessed human demonstrations into a library of object-centric and reusable skill primitives. Within a simulated environment, it then intelligently composes these primitives to solve new task instances, employing a novel weighted manipulability selection mechanism to ensure the quality and efficiency of the generated trajectories. Policies trained using the proposed sim-augmented data demonstrate improved performance and enhanced data efficiency across six diverse and challenging real-world manipulation tasks, compared with the baseline method.**

***Index Terms*—Learning from demonstration, imitation learning, robotic manipulation.**

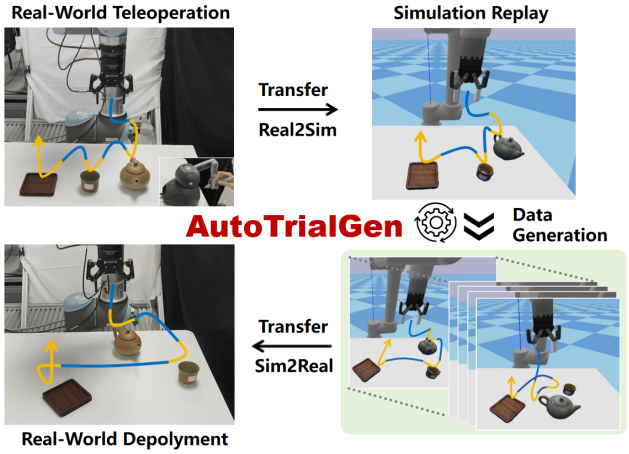

Fig. 1. **AutoTrialGen Overview.** AutoTrialGen provides an automated pipeline for data-efficient imitation learning. A human operator first collects a few task demonstrations via teleoperation. AutoTrialGen then automatically decomposes them into reusable, object-centric skill primitives and generates a large set of high-quality simulated demonstrations through trial-and-error. Finally, a policy is trained on the sim-augmented data and deployed in the real world.

I. Introduction

DEVELOPING robotic systems capable of performing long-horizon, multi-stage manipulation tasks remains a significant challenge in robotics. Such tasks, commonplace in human environments, require not only precise motor control but also the seamless sequencing of distinct skills while adapting to variations in the environment. Imitation learning (IL) [1]–[3] has emerged as a powerful and intuitive paradigm, enabling robots to acquire complex behaviors by learning from human-provided demonstrations. This approach bypasses the need for complex reward engineering and has shown considerable success in a variety of manipulation settings.

Despite its promise, the effectiveness of imitation learning is often constrained by a significant bottleneck, namely its heavy reliance on large amounts of data. To achieve robust performance and generalize to even minor variations in object poses or scene configurations, IL policies typically require large-scale, diverse datasets of expert demonstrations [4]–[7]. The process of collecting this data is notoriously resource-intensive, demanding substantial human effort, time, and specialized hardware. This creates a critical trade-off where researchers must either invest heavily in data collection or settle for smaller datasets that often yield policies with limited generalization capabilities.

To mitigate this data collection burden, recent works [8]–[15] have explored methods for automatically generating or augmenting demonstration data. A promising direction, exemplified by methods like MimicGen [8], involves adapting a small set of human demonstrations to new scenarios by decomposing them into segments and recombining them. While these approaches reduce the need for human intervention, they often suffer from a critical flaw: they rely on simplistic or random strategies for composing skill segments. This randomness can produce lower-quality or kinematically inefficient trajectories, and in some cases, combinations that are invalid or unstable, thereby degrading the quality of the generated dataset and hindering the performance of the final learned policy. In addition, they require extensive human annotation on task stages and the hand-crafting per-stage discriminator functions for each task.

To address these limitations, we propose AutoTrialGen, a framework for data-efficient imitation learning that facilitates the automated generation of large-scale, high-quality datasets from only a few human demonstrations. The proposed framework is built upon a fully automated pipeline that transfers a

Manuscript received October 28, 2025; Revised January 14, 2026; Accepted March 18, 2026.

This article was recommended for publication by Editor Tetsuya Ogata and Editor-in-Chief Tamim Asfour upon evaluation of the reviewers' comments. This work was supported by the National Key Research and Development Program of China under Grant No. 2024YFB4708801. *(Corresponding author: Aiguo Song.)*

[1]Huailiang Ma, Aiguo Song, Mutian He, Mingyu Li, Yibing Yan and Linhu Wei are with the School of Instrument Science and Engineering, Southeast University, Nanjing 210096, China (email: 230268559@seu.edu.cn; a.g.song@seu.edu.cn).

Digital Object Identifier (DOI): see top of this page.

small number of real-world demonstrations into a digital-twin simulation environment. It then automatically decomposes these demonstrations into a library of reusable, object-centric skill primitives using an off-the-shelf state-of-the-art vision-language model (VLM). Finally, it intelligently generates a large and diverse dataset in simulation, which is then used to train a robust policy for real-world deployment.

A key insight of our work is that the method used to compose skill primitives is paramount to the quality of the generated data. Unlike prior approaches that randomly combine segments [8]–[10], AutoTrialGen employs a novel weighted manipulability selection mechanism. This mechanism intelligently selects the most suitable skill primitive at each step by optimizing a composite score that jointly considers kinematics feasibility and transition efficiency. By prioritizing smooth transitions and stable configurations, AutoTrialGen ensures that the generated demonstrations are not only successful but are also of high quality, closely resembling the efficient and deliberate motions of a human expert.

In summary, the contributions of this paper are as follows.

- An automated data generation framework, AutoTrialGen, is proposed to generate large-scale, high-quality datasets of robotic behaviors from only a few human demonstrations. We construct a digital twin of the real-world environment, replay demonstrations, and generate new trajectories via trial-and-error for real-world policy training and deployment.
- An automatic skill decomposition module segments demonstrations into reusable, object-centric skill primitives. Building upon these primitives, a novel weighted manipulability selection mechanism is developed to compose them by jointly reasoning about feasibility and efficiency, generating stable and high-quality demonstrations compared with random composition strategies.
- Extensive experiments on challenging, multi-stage real-world manipulation tasks validate that policies trained on data from AutoTrialGen achieve substantially higher task success and better data efficiency than the baseline method, with average success rates rising from 49.1% to 70.3% in simulation and from 57.7% to 78.3% in real-world evaluations.

II. Related Works

A. Data-Efficient Imitation Learning

Behavioral Cloning (BC) [16] is a foundational method for learning control policies from expert demonstrations, with notable successes in domains like autonomous driving [17] and robotic manipulation [2], [3]. A primary limitation of standard BC, however, is its struggle with generalization; policies trained on small to moderately-sized datasets often fail to adapt to unseen variations in environmental layouts and visual appearances [18], [19]. While mitigating this issue through conventional means often necessitates large-scale data collection [1], [2], [20], this approach introduces significant scalability challenges. In contrast, this work tackles the problem of data efficiency. We apply state-of-the-art imitation learning methods to diverse datasets synthesized from only a few human demonstrations via our proposed data generation pipeline.

B. Data Generation and Augmentation for Robotic Manipulation.

Strategies for data generation and augmentation in robotic manipulation generally fall into two categories. One line of work leverages offline data augmentation techniques to synthetically expand the training dataset for policy learning [21]–[25]. Another recent approach focuses on generating entirely new demonstrations, often by leveraging Large Language Models (LLMs) for high-level task decomposition, followed by motion planning or reinforcement learning for subtask resolution [26]–[28]. A distinct but related strategy, exemplified by MimicGen [8] and its extensions [9]–[11], adapts existing human-collected demonstrations to novel object configurations rather than generating them from scratch. This is achieved by synthesizing new execution plans from segments of source demonstrations. However, this methodology suffers from critical limitations: it requires extensive manual effort, including multi-stage annotation for each task and the hand-crafting of discriminator functions for each stage. Furthermore, its random combination of segments from different source demonstrations can degrade the quality and kinematic consistency of the generated data [8]. The proposed AutoTrialGen enables a fully automated pipeline that first transfers real-world data from the physical environment to simulation, where a large-scale and diverse dataset is generated, and ultimately transfers the learned policy back to the real world.

III. Problem Formulation

Imitation Learning. Each manipulation task is formalized as a Partially Observable Markov Decision Process (POMDP). We are given a dataset of N demonstrations $\mathcal{D} = \left\{(s_0^i, o_0^i, a_0^i, s_1^i, o_1^i, a_1^i, \ldots, s_{H_i}^i)\right\}_{i=1}^{N}$, where each demonstration i is a time-ordered trajectory of horizon H_i. Here $s_t^i \in \mathcal{S}$ denotes the underlying system state at time step t, $o_t^i \in \mathcal{O}$ the observation available to the policy, and $a_t^i \in \mathcal{A}$ the executed action. Each initial state $s_0^i \sim D$ is sampled from the initial state distribution $D \subseteq \mathcal{S}$. The goal is to learn a policy $\pi : \mathcal{O} \to \mathcal{A}$ that maps observations to a distribution over the action space. We focus on Behavioral Cloning (BC) [16] methods that find a policy via the maximum-likelihood objective $\theta^\star = \arg\max_\theta \ \mathbb{E}_{(s,o,a)\sim\mathcal{D}}\big[\log \pi_\theta(a \mid o)\big]$. We train policies via BC with datasets generated via AutoTrialGen.

Assumptions. Like MimicGen [8], we make these assumptions. **(A1)** The action space $\mathcal{A}$ consists of continuous pose commands for an end effector controller along with a discrete gripper command (open/close). **(A2)** Each task involves a set of manipulable objects, and can be divided into object-centric subtasks (see Sec. IV-A). **(A3)** During data collection, the pose of objects can be observed or estimated prior to a robot arm making contact with that object.

IV. Method

The proposed AutoTrialGen, a framework for learning complex manipulation skills from a few human demonstrations by

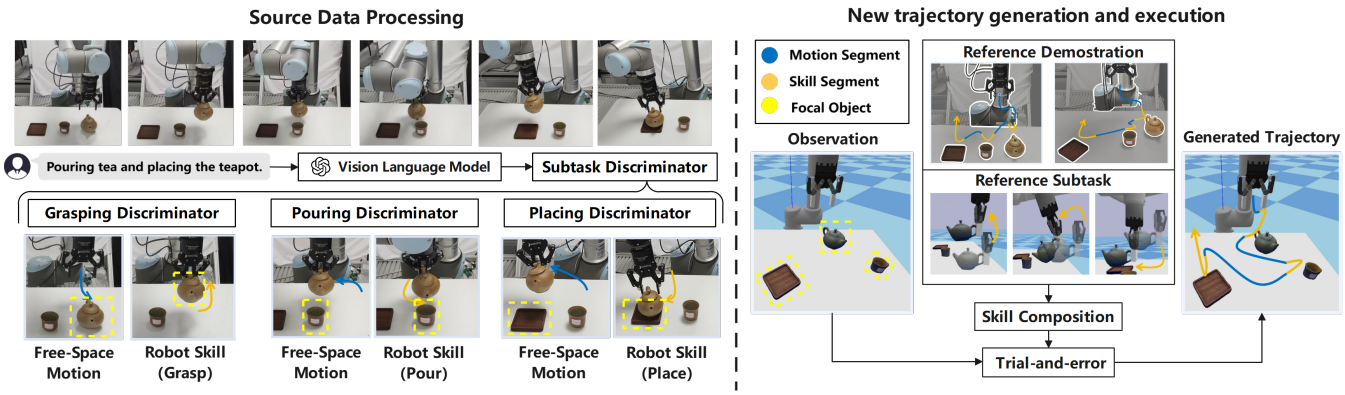

Fig. 2. Pipeline of AutoTrialGen: On the left, real-world teleoperation demonstrations are processed by a vision-language model to automatically generate subtask discriminator Python functions (e.g., grasping, pouring, placing) that segment trajectories into object-centric skill motions. On the right, under new object configurations, these skill primitives are composed using our skill composition strategy and replayed in simulation to generate new successful trajectory data for policy training.

generating a large-scale, sim-augmented dataset. An overview of proposed pipeline is illustrated in Fig. 1. AutoTrialGen comprises four key components (see Fig. 2): **(1) Automatic Subtask Decomposition:** Decomposing human demonstrations into reusable, object-centric skill primitives. **(2) Real-to-Sim Transfer:** Replicating real-world assets within a physically-randomized simulation. **(3) Demonstration Generation:** Synthesizing a large dataset by intelligently composing skill primitives in simulation. **(4) Policy Learning:** Training a visuomotor policy on the generated data for real-world deployment.

A. Automatic Subtasks Decomposition

Our method begins with a small source dataset of human demonstrations, $\mathcal{D}_{\text{src}}$, with the aim of automatically generating a large dataset $\mathcal{D}$ for the same task or a variant. We assume 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). Our pipeline divides each source demonstration $\tau \in \mathcal{D}_{\text{src}}$ into contiguous object-centric manipulation segments $\{\tau_i\}_{i=1}^{M}$, where each segment corresponds to a subtask $S_i(o_i)$. Each segment is a sequence of end-effector control poses $\tau_i = \left(T_W^{E_0}, T_W^{E_1}, \ldots, T_W^{E_K}\right)$,, where $T_W^{E_t} \in SE(3)$ denotes the 6-DoF pose of the robot end effector at time step t, and W denotes the world coordinate frame. To perform this decomposition automatically, we leverage a VLM through a structured prompt design rather than model fine-tuning. The VLM receives as input (i) a single RGB image of the scene to provide static scene and object context, and (ii) task-relevant object poses and (iii) robot end-effector poses across all time steps of the demonstration. A prompt specifies the task goal and object semantics, providing high-level prior information (including the involved objects and the expected number of object-centric skill segments) to guide the model in generating an executable Python discriminator function $d_i(s_t) : \mathcal{S} \to \{0, 1\}$ with NumPy operations, while avoiding fine-grained semantic rules or explicit segmentation criteria. This process decomposes each trajectory in $\mathcal{D}_{\text{src}}$ into free-space motion and object-centric manipulation segments, referred to as robot skills in Fig. 2. The object-centric segments form a skill library that is composed to generate trajectories for novel object configurations in simulation.

To ensure the resulting skill segments are generalizable, each end-effector pose action $T_W^{E_t}$ is stored in the frame of the relevant skill object O_i as $T_{O_i}^{E_t} \leftarrow \left(T_W^{O_i}\right)^{-1} T_W^{E_t}$, where $T_W^{O_i}$ denotes the observed pose of object O_i in the world frame prior to executing the skill. The first end-effector pose in the skill demonstration, $T_{O_i}^{E_0} \leftarrow \tau_i[0]$, is defined as the *initiation state*, specifying the entry point for the skill. The last pose in the demonstration, $T_{O_i}^{E_K}$, implicitly defines the *termination state*, with its detection determined by a task-specific discriminator function produced by a vision-language model.

B. Real-to-Sim Transfer

To enable scalable data generation, we construct a digital twin of the real-world environment in PyBullet [29]. First, we obtain high-fidelity textured meshes for all scene objects using an off-the-shelf multi-view 3D reconstruction method [30]. Then, to model the physical properties of these objects (e.g., density, friction), we avoid the complexities of explicit system identification, which typically requires extensive real-world trials [31]–[33]. Instead, we employ **domain randomization**: during the data generation process, we sample dynamics parameters from bounded distributions. This strategy helps bridge the reality gap and promotes the successful transfer of the learned policy to the real world.

C. Demonstration Generation

With the established library of skill primitives and the simulated environment, a large-scale dataset $\mathcal{D}$ is generated to enable data-efficient policy learning. The generation process is an automated trial-and-error loop where AutoTrialGen sequences skill primitives to solve the task under new, randomized initial conditions, and only save the successful trajectories. This process involves three main steps: adapting a selected skill to the current scene, intelligently composing skills to form a full trajectory, and generating smooth transitional motions.

1) Skill Adaptation: At each step of a task, a skill primitive s is chosen from the library (the selection process is detailed below). This primitive contains an object-relative trajectory $\tau_s = (T_{O_s}^{E_0}, \ldots, T_{O_s}^{E_K})$. To execute this in a new scene where the target object O_s has a pose $T_W^{O'_s}$, we transform the entire trajectory into the world frame: $\tau'_s = (T_W^{E_0}, \ldots, T_W^{E_K})$ where $T_W^{E_t} \leftarrow T_W^{O'_s} T_{O_s}^{E_t}$. This transformation preserves the learned end-effector motions relative to the target object, allowing the skill to be applied to novel object configurations. The resulting trajectory τ'_s is then executed by the end-effector controller in the simulator.

2) Skill Composition with Weighted Manipulability Selection: A key challenge arises when multiple skill primitives from the library can accomplish the next required subtask. A naive strategy, such as random selection, can degrade dataset quality by producing inefficient or unstable trajectories. To address this, AutoTrialGen employs a **Weighted Manipulability Selection** mechanism to guide the composition process. This method selects the optimal skill primitive s^* from a set of candidates $\mathcal{S}$ by minimizing a carefully designed penalty score that balances kinematic feasibility and motion efficiency.

The penalty is a weighted sum of two components: a manipulability penalty and a pose proximity penalty. To ensure these physically distinct measures are comparable, all penalty terms are normalized:

1. Kinematic Feasibility: We employ the Yoshikawa manipulability index [34] to penalize robot configurations near kinematic singularities. Given a candidate primitive's starting joint configuration q_s, the manipulability $w(s)$ is $w(s) = \sqrt{\det(J(q_s)J(q_s)^\top)}$, where $J(q_s)$ is the geometric Jacobian. The normalized manipulability penalty is:

$$\tilde{P}_{\text{manip}}(s) = \frac{1}{(w(s) + \varepsilon)\, P_{\text{manip}}^{\max}}, \tag{1}$$

where $\varepsilon > 0$ ensures numerical stability and $P_{\text{manip}}^{\max}$ depends on the robot's kinematic structure, representing the upper bound of its manipulability index.

2. Transition Efficiency: To encourage smooth motions between skills, we penalize large deviations between the current end-effector pose T_{current} and the candidate primitive's start pose T_{start}. We compute the normalized translational and rotational distances:

$$\tilde{d}_{\text{trans}} = \frac{\|\mathbf{p}_s - \mathbf{p}_{\text{current}}\|_2}{d_{\text{trans}}^{\max}}, \tag{2}$$

$$\tilde{d}_{\text{rot}} = \frac{\arccos\big(\frac{1}{2}\big(\text{tr}(R_s^\top R_{\text{current}}) - 1\big)\big)}{d_{\text{rot}}^{\max}}. \tag{3}$$

where $d_{\text{trans}}^{\max}$ and $d_{\text{rot}}^{\max}$, determined by the robot's kinematic structure and workspace, correspond to the maximum reachable displacement and the largest feasible rotational deviation of the end-effector, respectively. These constants normalize $\tilde{d}_{\text{trans}}$ and $\tilde{d}_{\text{rot}}$ to the range $[0, 1]$, yielding dimensionless and comparable penalty terms for weighted optimization.

Final Selection: The composite penalty score for a candidate skill s is a weighted combination of these terms:

$$P_{\text{weighted}}(s) = \lambda_1 \tilde{P}_{\text{manip}}(s) + \lambda_2(\alpha \tilde{d}_{\text{trans}} + \beta \tilde{d}_{\text{rot}}), \tag{4}$$

where the hyperparameters $\lambda_1, \lambda_2, \alpha, \beta$ balance the contributions of each factor. The primitive minimizing the composite penalty is selected for execution:

$$s^* = \arg\min_{s \in \mathcal{S}} P_{\text{weighted}}(s). \tag{5}$$

Furthermore, the selection rule in (5) depends only on the relative comparison of penalty magnitudes among candidates, whereas their absolute values are not physically meaningful. Hence, the critical factor is the relative scaling between λ_1 and λ_2. As described in Section V-A, both coefficients were set to equal proportions to balance the influence of the two penalties, and this configuration was empirically validated by the ablation results presented in Section V-C. The parameters α and β were kept fixed to maintain equal weights for translation and rotation, since the skill primitives varied across tasks.

Illustrative Example. For illustration, Fig. 3 visualizes our weighted manipulability selection on two representative tasks (cup hanging and tea pouring). From the current robot state (left), multiple candidate skill primitives are available (right), each with an associated penalty. This example is included here for clarity, while the complete evaluation of all tasks is presented in Sec. V.

3) Transitional Motion and Data Diversity: After selecting and adapting a skill primitive, a smooth transitional path is generated from the robot's current pose to the primitive's initiation state using Cartesian interpolation. Linear interpolation is applied to the translational component, while spherical linear interpolation (Slerp) is used for the rotational component. To further improve the diversity and generalization of the final dataset $\mathcal{D}$, the initial poses of all objects are randomized within a predefined range at the start of each new demonstration attempt. The entire process of selection, adaptation, and execution is repeated until the task is complete, and only successful trajectories are stored in the final dataset.

D. Policy Learning

The final stage of our framework is to learn a robust policy π from the large-scale dataset $\mathcal{D}$ generated by AutoTrialGen. For the policy architecture, we adopt Diffusion Policy [18], a state-of-the-art model for imitation learning. The policy takes the current robot proprioceptive state (e.g., joint positions) and the SE(3) poses of all task-relevant objects as input to predict a sequence of end-effector actions. While ground-truth object poses are readily available during training in simulation, a robust pose estimation system is required for real-world deployment. To this end, we leverage the textured 3D object models acquired during our Real-to-Sim transfer pipeline and employ FoundationPose [35] to accurately estimate object poses from sensor data at runtime.

V. Experiments

We designed the experiments to evaluate the quality of the augmented datasets generated by AutoTrialGen. Specifically, we aim to answer the following questions: 1) Does AutoTrialGen generate a higher-quality dataset, and how does this affect performance on complex, long-horizon tasks? 2) Does the proposed skill composition mechanism improve policy

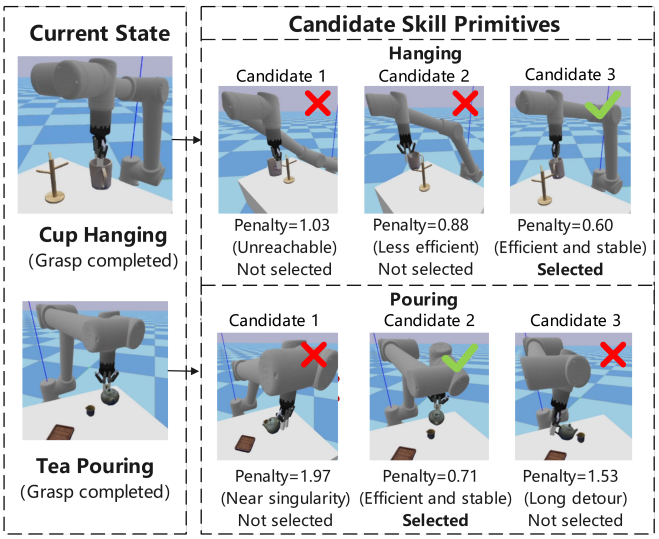

Fig. 3. **Illustration of the weighted manipulability selection mechanism.** Left: the current robot state for two representative tasks (cup hanging and tea pouring). Right: multiple candidate skill primitives, where the shown right-hand configurations correspond to segment start states rather than intermediate execution states, each associated with an estimated penalty value. Our method evaluates stability and efficiency, discarding infeasible or inefficient candidates and selecting the most suitable primitive for execution.

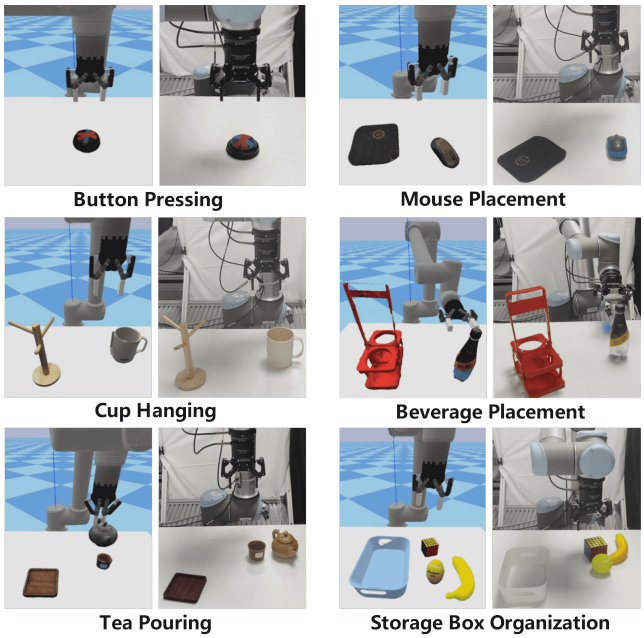

Fig. 4. **Experimental Tasks.** We evaluate across six manipulation tasks, covering both short-horizon (Button Pressing, Mouse Placement, Cup Hanging, Beverage Placement) and long-horizon scenarios (Storage Box Organization, Tea Pouring). Each task is shown in both the real-world setup (left) and the corresponding simulated environment (right).

performance compared to the baseline? 3) How do the key components of our selection mechanism—kinematic manipulability and pose proximity contribute to the final result? 4) Is AutoTrialGen practical and effective for real-world robotic deployment? 5) Can the datasets generated by AutoTrialGen serve as a substitute for human-collected teleoperation data, and how does their data efficiency compare?

A. *Experimental Setup*

We evaluated AutoTrialGen in both simulation and real-world. Simulation experiments were conducted in PyBullet [29]. The physical setup consisted a UR10 arm with a Robotiq parallel gripper; a global-view RealSense D435i camera provides object-centric observations. For each task, 1-4 teleoperated demonstrations were collected using a Geomagic Touch haptic device, and 500 synthetic demonstrations were generated in simulation. Visual prompting of the OpenAI O3 model [36] was used to synthesize Python-based discriminator functions for subtask termination. In simulation, we evaluate 50 episodes with 3 different random seeds per task under randomized object configurations. In the real world, we evaluate 50 episodes per task. The evaluation metrics include the task success rate, the data generation success rate, and trajectory smoothness, which is quantified using a joint-space angular velocity change metric defined as the mean of the top 5% largest velocity change magnitudes (lower is better).

Task Design. Evaluation of AutoTrialGen was conducted through six representative manipulation tasks: four short-horizon and two long-horizon tasks which involved diverse skills such as pressing, grasping, placing, and hanging in both simulation and the real world (see Fig. 4). During simulation-based data generation, each object's initial pose was randomly perturbed within a bounded planar region (typically 20–30 cm in translation and up to $\pm 180^\circ$ in rotation, depending on the task) to enhance variability across trials.

- **Button Pressing:** Pressing a button mounted on the table surface. We provided four demonstrations with distinct initial poses.
- **Mouse Placement:** Grasping a mouse and placing it onto a mousepad. We provided two demonstrations with distinct initial poses.
- **Cup Hanging:** Hanging a cup on a holder, demonstrated once for each of the three available hooks, which requires precise alignment of the cup handle with the narrow hook gap.
- **Beverage Placement:** Placing a beverage into a carrying rack, demonstrated twice, which requires precise alignment with limited tolerance.
- **Storage Box Organization (long-horizon):** Sequentially placing multiple objects into a storage box in cluttered scenes. Two demonstrations were provided, with task variants T0, T1, and T2 involving 4, 7, and 10 objects, respectively, where each target object induces a grasp-place subtask pair.
- **Tea Pouring (long-horizon):** Grasping the teapot, pouring tea into a cup, and returning the teapot to a coaster, which involves maintaining a stable grasp during pouring motions. Three demonstrations with distinct pouring motions were recorded.

Baseline. We implemented the dataset generation pipeline of MimicGen as our baseline [8]. In both AutoTrialGen and MimicGen, the same small set of expert demonstrations was segmented into subtasks using a vision-language model. The key difference lay in recombination: our approach employed manipulability-based selection, while MimicGen randomly

sampled segments. To ensure fairness, the same object pose perturbation scheme was applied to both methods.

Policy Training. The generated dataset was used to train a Diffusion Policy [18] for action prediction. The inputs of policy included object-centric observations (object pose, end-effector state, and gripper state), and outputs were absolute control commands for the end-effector pose and gripper. We adopted the DDPM [37] framework with a Conditional U-Net1D architecture.

B. Quantitative Results

Simulation Results. AutoTrialGen was first evaluated across six simulation tasks. As shown in Table I, the proposed **weighted manipulability mechanism** significantly outperforms the MimicGen baseline, achieving an average success rate of **70.3%**. Our analysis highlighted key advantages:

1) **Manipulability-Aware Selection Yields Higher-Quality Datasets.** Our method's core benefit stems from generating superior datasets. By prioritizing skill segments that avoid kinematic singularities, we achieved a higher data generation success rate than MimicGen (80.2% vs. 66.8%). Furthermore, incorporating a pose proximity term regularizes skill transitions and yields significantly smoother joint-space trajectories, reducing abrupt joint velocity changes by approximately 32.9% (49.4 vs. 73.6 deg/s). For instance, in the Mouse Placement task, two grasp poses were possible, differing by a 180° rotation. MimicGen's random selection often chose the kinematically challenging grasp near a singularity, resulting in a low 56.0% success rate. In contrast, AutoTrialGen consistently selected the more stable grasp, boosting the success rate to **90.7%**.
2) **Gains are Amplified in Long-Horizon Tasks.** The advantages of our systematic skill composition are particularly pronounced in multi-step, long-horizon tasks. In Storage Box Organization and Tea Pouring, AutoTrialGen improved success rates by over 20 and 16 percentage points, respectively (42.0% $\rightarrow$ 64.0% and 72.7% $\rightarrow$ 89.3%). Task difficulty is influenced by temporal horizon as well as the precision requirements of the underlying manipulation primitives, and some short-horizon tasks can be more challenging than longer-horizon ones, as seen in Beverage Placement versus Storage Box Organization (61.3% vs. 64.0%). Long-horizon performance is affected by subtask difficulty, with more challenging stages leading to lower overall success rates. As the number of subtasks increases, both policy performance and data generation success rates decrease across methods, with degradation becoming more pronounced from T0 to T2. Despite this, AutoTrialGen consistently outperforms MimicGen, demonstrating stronger scalability under increasing subtask complexity.
3) **Implicit Collision Reduction via Singularity Avoidance.** Although our method did not explicitly model obstacles, it implicitly reduced collisions. By favoring skill segments with high manipulability, it naturally steered the robot away from workspace boundaries where singularities and self-collisions were more likely. This focus on feasible configurations contributed to the **+8.0%** improvement in data generation success rate over MimicGen, as fewer trajectories were discarded due to collisions. This, in turn, leads to safer and more reliable policies.

TABLE I
TASK SUCCESS RATE AND DATA GENERATION RATE FOR MIMICGEN AND AUTOTRIALGEN (IN %), WITH TRAJECTORY SMOOTHNESS REPORTED AS JOINT-SPACE ANGULAR VELOCITY CHANGE (DEG/S).

Task	MimicGen		AutoTrialGen (ours)	
	Succ. Rate	Gen.	Succ. Rate	Gen.
Button Pressing	83.3 $\pm$ 3.4	95.3	86.0 $\pm$ 3.3	95.7
Mouse Placement	56.0 $\pm$ 1.6	92.4	90.7 $\pm$ 0.9	95.6
Cup Hanging	60.7 $\pm$ 2.5	58.2	72.7 $\pm$ 0.9	78.5
Beverage Placement	41.3 $\pm$ 2.5	76.4	61.3 $\pm$ 2.5	85.9
Storage Box Org. (T0)	42.0 $\pm$ 3.3	70.8	64.0 $\pm$ 1.6	87.5
Storage Box Org. (T1)	26.7 $\pm$ 2.3	39.6	56.7 $\pm$ 1.2	62.4
Storage Box Org. (T2)	10.0 $\pm$ 2.0	18.4	41.3 $\pm$ 1.2	44.0
Tea Pouring	72.7 $\pm$ 2.5	83.4	89.3 $\pm$ 1.9	92.3
Averaged	49.1	66.8	70.3	80.2
Trajectory Smoothness	73.6		49.4	

TABLE II
REAL-WORLD TASK SUCCESS RATE COMPARISON BETWEEN MIMICGEN AND AUTOTRIALGEN. EACH METHOD IS EVALUATED OVER 50 EPISODES PER TASK. VALUES DENOTE SUCCESS RATES (IN %).

Task	MimicGen	AutoTrialGen (ours)	Δ
Button Pressing	80.0	90.0	+10.0
Mouse Placement	54.0	88.0	+34.0
Cup Hanging	62.0	74.0	+12.0
Beverage Placement	40.0	64.0	+24.0
Storage Box Org. (T0)	42.0	68.0	+26.0
Tea Pouring	68.0	86.0	+18.0
Average	57.7	78.3	+20.6

Real-World Experiments. We further validated our approach on a physical UR10 robotic arm across six tasks. The real-world results, presented in Table II, corroborated our simulation findings. Our method again consistently outperformed the baseline, raising the average success rate from 57.7% to **78.3%**. The most significant improvements were observed in tasks with high kinematic redundancy, such as Mouse Placement (+34.0%) and Storage Box Organization (+26.0%).

1) **Effective Sim-to-Real Transfer.** The learned policies exhibited strong sim-to-real transferability. The high-fidelity simulation environment, combined with accurate object pose estimation from FoundationPose, ensured a minimal reality gap. The average success rate dropped by less than 5% from simulation (77.3%) to the real world (73.1%), validating the effectiveness of our training pipeline.
2) **System Robustness and Data Efficiency.** The overall success of the framework is underpinned by two key factors. First, a robust vision backbone allows the system to reliably perceive object states despite environmental variations like lighting changes. Second, our approach is highly data-efficient, generating large-scale, high-quality datasets in simulation from only a few human demonstra-

TABLE III
ABLATION STUDY ON DIFFERENT WEIGHTING RATIOS $\lambda_1 : \lambda_2$ FOR SHORT-HORIZON AND LONG-HORIZON TASKS. λ_1 AND λ_2 CORRESPOND TO THE WEIGHTS OF THE *MANIPULABILITY* AND *POSE PROXIMITY* TERMS, RESPECTIVELY. VALUES DENOTE TASK SUCCESS RATE (IN %).

Task	0:1	1:5	1:2	1:1	2:1	5:1	1:0
Short-horizon	69.5	71.2	76.7	77.7	76.0	71.0	71.0
Long-horizon	68.3	70.0	71.6	76.7	77.6	76.0	72.3
All tasks	69.1	70.8	75.0	77.3	76.5	72.6	71.4

tions. This efficient **real-to-sim-to-real pipeline** makes the system both scalable and practical for real-world applications.

3) **Data Source Efficiency.** Policies trained on AutoTrialGen datasets achieved comparable performance to those trained on human-collected demonstrations of equal size, and scaling to larger sim-generated datasets (e.g., 500 trials) even surpassed the 100-human baseline (82.0% vs. 78.0%). The results are shown in Fig. 5. Across three representative tasks (Button Pressing, Cup Hanging, Tea Pouring), AutoTrialGen data achieved performance comparable to, and in some cases surpassing, human-collected data. This demonstrates that AutoTrialGen can effectively substitute for costly teleoperation data while maintaining strong task performance.

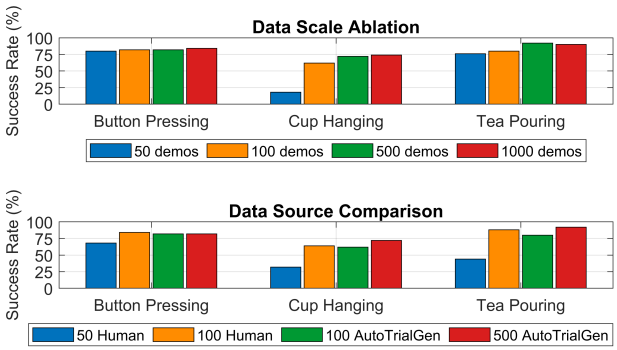

Fig. 5. Comparison of success rates (%) across different data conditions. (**Top: Data Scale Ablation**) Diffusion policies trained with datasets of different sizes (50, 100, 500, and 1000 demonstrations). (**Bottom: Data Source Comparison**) Policies trained on equal-scale datasets collected from human teleoperation or generated by AutoTrialGen. Each task was evaluated over 50 rollouts.

C. Ablation Studies

We conducted two sets of ablation studies to analyze the effects of data scale and the influence of the relative weighting between the manipulability and pose proximity terms in our weighted selection mechanism.

a) Data Scale Ablation: To understand the impact of dataset size, we trained policies with the number of demonstrations per task varying from 50 to 1000 in simulation and evaluated each configuration 50 episodes over 3 different random seeds. As shown in Figure 5, performance scales with data but exhibits diminishing returns. On average, the success rate grows significantly when scaling from 50 to 500 demonstrations (61.3% $\rightarrow$ 82.0%), but additional data yields marginal gains. Task complexity affects the saturation point: simple tasks saturate early, while more complex tasks (e.g., Cup Hanging) benefit from larger datasets.

This analysis shows that diffusion policies require larger datasets than conventional behavioral cloning to fully realize their generative capacity. For the experimental setup, 500 demonstrations per task yield a favorable balance between data efficiency and policy performance.

b) Component Ablation: To assess the contribution of each component in our weighted penalty formulation, we performed an ablation study on the two components of our weighted penalty design: the manipulability term and the pose proximity term. We adjusted the relative weighting ratio between λ_1 and λ_2, and evaluated each configuration in simulation for 50 episodes across 3 different random seeds. As shown in Table III, both components are essential for performance; removing the manipulability term causes the largest drop (77.3% $\rightarrow$ 69.1%), whereas removing the pose proximity term results in a smaller degradation (71.4%).

Notably, pose proximity benefits short-horizon tasks, while manipulability term benefits long-horizon tasks (1:2 vs. 5:1). The two components provide complementary benefits: manipulability steers the robot away from singular configurations, while pose proximity promotes smoother skill transitions, together enabling safe and efficient task execution.

VI. CONCLUSION

This letter presents AutoTrialGen, a framework that addresses the data-collection bottleneck in imitation learning through a complete **real-to-sim-to-real** pipeline. By first transferring a few expert demonstrations into simulation and then deploying the trained policy back to the physical robot, our approach maximizes data efficiency. Within this pipeline, we identify that the common practice of random skill composition is a critical flaw, often producing inefficient or unstable trajectories. Our core technical contribution is a novel weighted manipulability selection mechanism that intelligently assembles skill primitives by optimizing for kinematic manipulability and motion efficiency. Extensive experiments confirm the success of this entire workflow, demonstrating that policies trained on our high-quality generated data achieve superior performance in real-world execution compared to the baseline. Ultimately, our findings highlight that a robust real-to-sim-to-real framework, when coupled with an intelligent data generation strategy, is a powerful paradigm for teaching robots complex, long-horizon skills.

Limitations. Despite strong results, our approach has several limitations that suggest avenues for future work. First, our evaluation focuses on quasi-static manipulation tasks and does not cover highly dynamic or deformable-object scenarios. Second, the framework assumes a single central object per subtask and sufficiently accurate object pose estimates at deployment; in real-world settings, pose estimation errors may propagate to action execution, particularly for fine-grained manipulation tasks. Third, our demonstrations are generated

for a single object instance; extending to more geometrically diverse objects is another important step.

REFERENCES

[1] 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.

[2] 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," *arXiv preprint arXiv:2108.03298*, 2021.

[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] H. Shen, W. Wan, and H. Wang, "Learning category-level generalizable object manipulation policy via generative adversarial self-imitation learning from demonstrations," 2022. [Online]. Available: https://arxiv.org/abs/2203.02107

[5] H. Ravichandar, A. S. Polydoros, S. Chernova, and A. Billard, "Recent advances in robot learning from demonstration," *Annual review of control, robotics, and autonomous systems*, vol. 3, no. 1, pp. 297–330, 2020.

[6] A. O'Neill, A. Rehman, A. Maddukuri, A. Gupta, A. Padalkar, A. Lee, A. Pooley, A. Gupta, A. Mandlekar, A. Jain *et al.*, "Open x-embodiment: Robotic learning datasets and rt-x models: Open x-embodiment collaboration 0," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2024, pp. 6892–6903.

[7] W. Wan, Y. Zhu, R. Shah, and Y. Zhu, "Lotus: Continual imitation learning for robot manipulation through unsupervised skill discovery," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2024, pp. 537–544.

[8] 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 *7th Annual Conference on Robot Learning*, 2023.

[9] C. Garrett, A. Mandlekar, B. Wen, and D. Fox, "Skillmimicgen: Automated demonstration generation for efficient skill learning and deployment," *arXiv preprint arXiv:2410.18907*, 2024.

[10] Z. Jiang, Y. Xie, K. Lin, Z. Xu, W. Wan, A. Mandlekar, L. Fan, and Y. Zhu, "Dexmimicgen: Automated data generation for bimanual dexterous manipulation via imitation learning," *arXiv preprint arXiv:2410.24185*, 2024.

[11] Z. Xue, S. Deng, Z. Chen, Y. Wang, Z. Yuan, and H. Xu, "Demogen: Synthetic demonstration generation for data-efficient visuomotor policy learning," *arXiv preprint arXiv:2502.16932*, 2025.

[12] H. Geng, F. Wang, S. Wei, Y. Li, B. Wang, B. An, C. T. Cheng, H. Lou, P. Li, Y.-J. Wang *et al.*, "Roboverse: Towards a unified platform, dataset and benchmark for scalable and generalizable robot learning," *arXiv preprint arXiv:2504.18904*, 2025.

[13] W. Wan, J. Fu, X. Yuan, Y. Zhu, and H. Su, "Lodestar: Long-horizon dexterity via synthetic data augmentation from human demonstrations," in *9th Annual Conference on Robot Learning*, 2025.

[14] J. Wang, Y. Qin, K. Kuang, Y. Korkmaz, A. Gurumoorthy, H. Su, and X. Wang, "Cyberdemo: Augmenting simulated human demonstration for real-world dexterous manipulation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 17 952–17 963.

[15] T. Li, Y. Li, Z. Zhang, and N. Figueroa, "Flow with the force field: Learning 3d compliant flow matching policies from force and demonstration-guided simulation data," *arXiv preprint arXiv:2510.02738*, 2025.

[16] D. A. Pomerleau, "Alvinn: An autonomous land vehicle in a neural network," *Advances in neural information processing systems*, vol. 1, 1988.

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

[18] C. Chi, Z. Xu, S. Feng, E. Cousineau, Y. Du, B. Burchfiel, R. Tedrake, and S. Song, "Diffusion policy: Visuomotor policy learning via action diffusion," *The International Journal of Robotics Research*, p. 02783649241273668, 2023.

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

[20] S. Ross, G. Gordon, and D. Bagnell, "A reduction of imitation learning and structured prediction to no-regret online learning," in *Proceedings of the fourteenth international conference on artificial intelligence and statistics*. JMLR Workshop and Conference Proceedings, 2011, pp. 627–635.

[21] 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.

[22] 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.

[23] S. Young, D. Gandhi, S. Tulsiani, A. Gupta, P. Abbeel, and L. Pinto, "Visual imitation made easy," in *Conference on Robot learning*. PMLR, 2021, pp. 1992–2005.

[24] 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.

[25] 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 *2024 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2024, pp. 4788–4795.

[26] L. Wang, Y. Ling, Z. Yuan, M. Shridhar, C. Bao, Y. Qin, B. Wang, H. Xu, and X. Wang, "Gensim: Generating robotic simulation tasks via large language models," *arXiv preprint arXiv:2310.01361*, 2023.

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

[28] W. Huang, C. Wang, Y. Li, R. Zhang, and L. Fei-Fei, "Rekep: Spatio-temporal reasoning of relational keypoint constraints for robotic manipulation," *arXiv preprint arXiv:2409.01652*, 2024.

[29] E. Coumans and Y. Bai, "Pybullet, a python module for physics simulation for games, robotics and machine learning," 2016.

[30] AR Code, "AR Code," https://ar-code.com/, 2022.

[31] J. Bohg, K. Hausman, B. Sankaran, O. Brock, D. Kragic, S. Schaal, and G. S. Sukhatme, "Interactive perception: Leveraging action in perception and perception in action," *IEEE Transactions on Robotics*, vol. 33, no. 6, pp. 1273–1291, 2017.

[32] N. Pfaff, E. Fu, J. Binagia, P. Isola, and R. Tedrake, "Scalable real2sim: Physics-aware asset generation via robotic pick-and-place setups," *arXiv preprint arXiv:2503.00370*, 2025.

[33] M. Torne, A. Simeonov, Z. Li, A. Chan, T. Chen, A. Gupta, and P. Agrawal, "Reconciling reality through simulation: A real-to-sim-to-real approach for robust manipulation," *arXiv preprint arXiv:2403.03949*, 2024.

[34] T. Yoshikawa, "Manipulability of robotic mechanisms," *The international journal of Robotics Research*, vol. 4, no. 2, pp. 3–9, 1985.

[35] B. Wen, W. Yang, J. Kautz, and S. Birchfield, "Foundationpose: Unified 6d pose estimation and tracking of novel objects," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 17 868–17 879.

[36] OpenAI, "OpenAI o3: Introducing a new generation of reasoning models," https://openai.com/index/introducing-o3-and-o4-mini/, 2025.

[37] J. Ho, A. Jain, and P. Abbeel, "Denoising diffusion probabilistic models," *Advances in neural information processing systems*, vol. 33, pp. 6840–6851, 2020.