Local Policies Enable Zero-shot Long-horizon Manipulation

Anonymous Author(s)
Affiliation
Address
`email`

Figure 1: **Zero-shot Long-horizon Manipulation** Our approach trains a library of generalist manipulation skills in simulation and transfers them zero-shot to long-horizon manipulation tasks. We show a single, text-conditioned agent can manipulate unseen objects, in arbitrary poses and scene configurations, across long-horizons in the real world, solving challenging manipulation tasks with complex obstacles.

Abstract: Sim2real for robotic manipulation is difficult due to the challenges of simulating complex contacts and generating realistic task distributions. To tackle the latter problem, we introduce ManipGen, which leverages a new class of policies for sim2real transfer: local policies. Locality enables a variety of appealing properties including invariances to absolute robot and object pose, skill ordering, and global scene configuration. We combine these policies with foundation models for vision, language and motion planning and demonstrate SOTA zero-shot performance of our method to Robosuite benchmark tasks in simulation (97%). We transfer our local policies from simulation to reality and observe they can solve unseen long-horizon manipulation tasks with up to 8 stages with significant pose, object and scene configuration variation. ManipGen outperforms SOTA approaches such as SayCan, OpenVLA and LLMTrajGen across 50 real-world manipulation tasks by 36%, 76% and 62% respectively. All code, models and datasets will be released. Video results at `manipgen.github.io`

Keywords: Sim-to-Real Transfer, Long-horizon Manipulation

1 Introduction

How can we develop generalist robot systems that plan, reason, and interact with the world like humans? Tasks that humans solve during their daily lives, such as those shown in Figure 1, are incredibly challenging for existing robotics approaches. Cleaning the table, organizing the shelf, putting items away inside drawers, etc. are complex, long-horizon problems that require the robot to act capably and consistently over an extended period of time. Furthermore, such a generalist robot should be able to do so without requiring task-specific engineering effort or demonstrations. Although large-scale data-driven learning has produced generalists for vision and language [1], such models don't yet exist in robotics due to the challenges of scaling data collection. It often takes significant manual labor cost and years of effort to just collect datasets on the order of 100K-1M trajectories [2, 3, 4, 5]. Consequently, generalization is limited, often to within centimeters of an object's pose for complex tasks [6, 7].

Instead, we seek to use a large-scale approach via simulation-to-reality (sim2real) transfer, a cost-effective technique for generating vast datasets that has enabled training generalist policies for locomotion which can traverse complex, unstructured terrain [8, 9, 10, 11, 12, 13]. While sim2real transfer has shown success in industrial manipulation tasks [14, 15, 16], including with high-dimensional hands [17, 18, 19, 20], these efforts often involve training and testing on the same task in simulation. Can we extend sim2real to open-world manipulation, where robots need to solve any task from text instruction? The core bottlenecks are: 1) accurately simulating contact dynamics [21] - for which strategies such as domain randomization [17, 22], SDF contacts [23, 14, 15], and real world corrections [16] have shown promise, 2) generating all possible scene and task configurations to ensure trained policies generalize and 3) acquiring long-horizon behaviors themselves, which may require potentially intractable amounts of data for as the horizon grows.

To address points 2) and 3), our solution is to note that for many manipulation tasks of interest, the skill can be simplified to two steps: achieving a pose near a target object, then performing manipulation. The key idea is that of *locality of interaction*. Policies that observe and act in a region local to the target object of interest are by construction:

- **absolute pose invariant**: they reason over a smaller set of relative poses between objects and robot.
- **skill order invariant**: transition from the termination to initiation of policies via motion planning.
- **scene configuration invariant**: they solely observe the local region around the point of interaction.

We propose a novel approach that leverages the strong generalization capabilities of existing foundation models such as Visual Language Models (VLMs) for decomposing tasks into sub-problems [1], processing and understanding scenes [24] and planning collision-avoidant motions [25]. Specifically, given a text prompt, our approach outputs a plan to solve the task (using a VLM), estimates where to go and moves the robot accordingly (using motion planning) and deploys local policies to perform interaction. As a result, a simple scene generation approach can produce strong transfer results across many manipulation tasks (Fig. 1).

Our contribution is an approach to training agents at scale solely in simulation that are capable of solving a vast set of long-horizon manipulation tasks in the real world *zero-shot*. Our method generalizes to unseen objects, poses, receptacles and skill order configurations. To do so, our method, ManipGen, 1) introduces a novel policy class for sim2real transfer 2) proposes techniques for training policies at scale in simulation 3) and deploys policies via integration with VLMs and motion planners. We perform a thorough, real world evaluation of ManipGen on **50** long-horizon manipulation tasks in **five** environments with up to **8** stages, achieving a success rate of **76%**, outperforming SayCan, OpenVLA and LLMTrajGen by **36%**, **76%** and **62%**.

2 Related Work

Long-horizon Robotic Manipulation Sense-Plan-Act (SPA) has been explored extensively over the past 50 years [26, 27, 28, 29, 30, 31]. Traditionally, SPA assumes access to accurate state estimation, a well-defined model of the environment and low-level control primitives. SPA, while capable of generalizing to a broad set of tasks, can require manual engineering and systems effort to set up [32], struggles with contact-rich interactions [33, 34] and fails due to state-estimation errors [35]. By contrast, our method can be deployed to new tasks using generalist models which have minimal setup

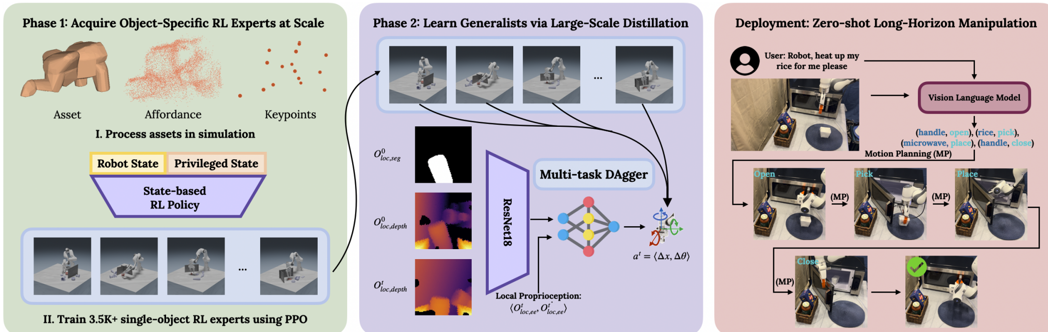

Figure 2: **ManipGen Method Overview** (*left*) Train 1000s of RL experts in simulation using PPO (*middle*) Distill single-task RL experts into generalist visuomotor policies via DAgger (*right*) Text-conditioned long-horizon manipulation via task decomposition (VLM), pose estimation and goal reaching (Motion Planning) and sim2real transfer of local policies

cost, train polices for contact-rich interactions and handle state-estimation issues by training with significant local randomization.

Zero/Few-shot Manipulation Using Foundation Models The robotics community has begun to investigate VLM's capabilities for controlling robots in a zero/few-shot manner [36, 37, 38, 39, 40, 41, 42, 43, 44]. Work such as SayCan [36] and TidyBot [39] are similar to our own. They behavior clone / design a library of skills and use LLMs to perform task planning over the set of skills. Our work focuses primarily on designing the structure of skills for low-level control, decomposing them into motion planning and sim2real local policies. On the other hand, works such as LLMTrajGen [45] and CoPa [46] directly prompt VLMs to output sequences of end-effector poses, but are limited to short horizon tasks. Finally, PSL [44] and Boss [42] use LLMs to accelerate the RL training process for long-horizon tasks, yet must train on the test task, unlike our method which can solve a wide array of manipulation tasks zero-shot.

Sim2real approaches in robotics Transfer of RL policies trained with procedural scene generation has produced generalist robot policies for locomotion [8, 9, 11, 10, 12]. However, the robot is often trained for a single skill, such as walking, or a limited set of similar skills, such as walking at different velocities or headings. Sim2real transfer has also been explored for transferring dexterous manipulation skills [17, 22, 18, 47, 19] and contact-rich manipulation [14, 15, 16]. In our work, we train a variety of skills for manipulation and demonstrate zero-shot capabilities on a large set of unseen tasks. We outperform methods that use end-to-end sim2real transfer [48] as well as real world corrections [16], ManipGen is orthogonal to human correction approaches, and can benefit from real-world data as well.

3 Methods

To build agents capable of generalizing to a wide class of long-horizon robotic manipulation tasks, we propose a novel approach (ManipGen) that hierarchically decomposes manipulation tasks, takes advantage of the generalization capabilities of foundation models for vision and language and uses large-scale learning with our proposed policy class to learn manipulation skills. We begin by describing our framework (Fig. 2) and formulate local policies. We then discuss how to train local policies for sim2real transfer. Finally, we outline deployment: integrating VLMs, Motion Planning and sim2real policy learning to foster broad generalization.

3.1 Framework

We can decompose any task the robot needs to complete into a problem of learning a set of temporally abstracted actions (skills) as well as a policy over those skills [49]. Given a language goal g, and observation O, we can select our policy over skills, $p_\theta(g_k|g, O)$ to be a pre-trained VLM, where g_k is skill k. State-of-the-art VLMs can decompose robotics tasks into language subgoals [36, 37, 38, 39] because they are trained using a vast corpus of internet-scale data and have captured powerful, visually grounded semantic priors for what various real world tasks look like.

Any policy class can be used to define the skills, denoted as $p_{\phi_k}(a^t|g_k, O^t)$, which take in the kth sub-goal g_k and current observation O^t. However, note that many manipulation skills (*e.g.* picking, pushing, turning, etc.) can be decomposed into a policy π_{reach} to achieve target poses near objects $X_{targ,k}$ followed by policy π_{loc} for contact-rich interaction. Accordingly, $p_{\phi_k}(a^t|g_k, O^t) = \pi_{reach}(\tau_{reach}|g_k, O^t)\pi_{loc}(a^t_{loc}|O^t_{loc})$. To implement π_{reach}, we need to interpret language sub-goals g_k to take the robot from its current configuration $q_{k,i}$ to some target configuration $q_{k,f}$ such that X_{ee} (the end-effector pose) is close to $X_{targ,k}$. Thus, we structure the VLM's sub-goal predictions, g_k, as tuples containing the following information (object, skill). We then interpret these plans into robot poses by pairing any language conditioned pose estimator or affordance model (to predict $X_{targ,k}$) with an inverse kinematics routine (to compute $q_{k,f}$). Motion planning can predict actions τ_{reach} to achieve the target configuration $q_{k,f}$ while avoiding collisions.

Finally, we instantiate local policies (π_{loc}) to be invariant to robot and object poses, order of skill execution and scene configurations with: 1) initialization region s_{init} near a target region/object of interest which has pose $X_{targ,k}$, 2) local observations O^t_{loc}, independent of the absolute configuration of the robot and scene and only observing the environment around the interaction region and 3) actions a^t_{loc}relative to the local observations. Overall: $\pi_{loc}(a^t_{loc}|O^t_{loc})$, $s_{init} = \{s \mid ||X_{ee} - X_{targ,k}||^2 < \epsilon\}$.

3.2 Training Local Policies for Sim2Real Manipulation

To train local policies, we adapt the standard two-phase training approach [47, 12, 11, 50, 19, 16] in which we first train state-based expert policies using RL, then distill them into visuomotor policies for transfer. Although local policies can generalize automatically across scene arrangements, robot configurations, and object poses, they must be trained across a wide array of objects to foster object-level generalization. To do so, we train a vast array of *single-object* state-based experts and then distill them into *generalist* visuomotor policies per skill.

While such local policies can cover a broad set of manipulation skills (pick and place, articulated/deformable object manipulation, assembly, etc.), in this work, we focus on training the following skills π_{loc}: **pick**, **place**, **grasp handle**, **open** and **close** as a minimal skill library to demonstrate generalist manipulation capabilities for a specific class of tasks. **Pick** grasps any free rigid objects. **Place** sets the object down near the initial pose. **Grasp Handle** grasps the handle of any door or drawer. **Open and Close** pull or push doors and drawers to open or close them.

To train robust local policies via RL, they require a diverse set of training environments, carefully designed observations and action spaces and well-defined reward functions enabling them to acquire behaviors in a manner that will transfer to the real world. We describe how to in this section.

Data Generation We need to first specify a set of objects to manipulate, an environment, and an initial local state distribution. For pick/place, we train on 3.5K objects from UnidexGrasp [51], randomly spawned on a table top. To ensure local policies can learn obstacle avoidance and constrained manipulation, we spawn clutter objects and obstacles in the scene. We sample initial poses in a half-sphere, with the gripper pointing toward the object (for picking) and near the placement location (for placing). For local articulated object manipulation, the region of interaction only contains the handle (2.6K objects of Partnet [52]) and door/drawer surface (designed as cuboids). We randomize the size, shape, position, orientation, joint range, friction and damping coefficients, covering a wide set of real world articulated objects. We sample initial poses in a half-sphere around the handle (for grasp handle) and a randomly sampled initial joint pose (open/close). Finally we collect valid pre-grasp poses (antipodal sampling [53]) for picking and grasping handles and rest poses (from UnidexGrasp) for learning placing.

Observations We use a single observation space for all RL experts, accelerating learning by incorporating significant amounts of privileged information. Blind local policies can struggle to learn to manipulate objects with complex geometries as it is often necessary to have some notion of object shape to know how to manipulate. Thus, we propose to use a low-dimensional representation of the object shape by performing Farthest Point Sampling (FPS) on the object mesh with a small set number of desired key-points K (16). Furthermore, to ease the burden of credit assignment and thereby accelerate learning, we incorporate the individual reward components $\{\mathbf{r}\}$ and an indicator for the final observation $\mathbb{1}\{t = T\}$. RL observations are $O^t = \langle X^t_{ee}, \dot{X_e e}^t, X^t_{obj}, \{FPS^t_{obj}\}^K_{k=1}, \{\mathbf{r}\}^t, \mathbb{1}\{t = T\}\rangle$

Actions We use the action space from Industreal [14] which has been shown to successfully transfer manipulation policies from sim2real for precise assembly tasks. Our policies predict delta pose targets for a Task Space Impedance (TSI) controller, where $a = [\Delta x; \Delta\theta]$, where Δx is a position error and $\Delta\theta$ is a axis-angle orientation error.

Rewards We train RL policies (π_{loc_k}) in simulation using reward functions we design to elicit the desired behavior per skill k. We propose a reward framework that encompasses our local skills: $\mathbf{r} = c_1 r_{ee} + c_2 r_{obj} + c_3 r_{ee,obj} + c_4 r_{action} + c_5 r_{succ}$. $\mathbf{r}$ specifies behavior for a broad range of manipulation tasks which involve moving the end-effector to specific poses (often right before contact) as well as a target object to desired poses and need to do so while maintaining certain constraints on the relative motion between the end-effector and the object as well as pruning out undesirable actions. r_{ee} encourages reaching/maintaining specific end-effector poses, r_{obj} restricts/encourages specific object poses or joint configurations, $r_{ee,obj}$ constrains the end-effector motion relative to the object(s) in the scene, r_{action} restricts or penalizes undesirable actions and r_{succ} is a binary success reward. Please see the website for detailed descriptions of the task specific reward functions.

3.3 Generalist Policies via Distillation

In order to convert single-object, privileged policies into real world deployable skills, we distill them into multi-object, generalist visuomotor policies using DAgger [54].

Multitask Online Imitation Learning Empirically the standard, off-policy version of DAgger with interleaved behavior cloning (to convergence) and large dataset collection does not perform well. The policy ends up modeling data from policies whose state visitation distributions deviate significantly from the current policy. On the other hand, on-policy variants of DAgger, which take a single gradient step per environment step [10, 47, 50, 19], can produce unstable results in the multi-task regime since the policy only gets data from a single object in a batch. We introduce a simple variant of DAgger which smoothly trades off between the two extremes by incorporating a replay buffer of size K that holds the last $K * B$ trajectories in memory. Training alternates between updating the agent for a single epoch on this buffer and collecting a batched set of trajectories (size B) from the environment for the current object.

Observation Space Design for Locality For local policies to transfer effectively to the real robot, the observation space and augmentations must be designed with transfer in mind. To imitate a privileged expert, our observation space must be expressive - providing as much information as possible to the agent. The observations must also be local to enable all of the properties of locality, and augmentations must ensure the policy is robust to noisy real world vision.

Local observations use wrist camera depth maps. Depth maps transfer well from sim2real for locomotion [10, 11, 12, 50], and wrist views are inherently local and improve manipulation performance [55, 56, 57]. To further enforce locality, we clamp depth values and normalize them. Since local wrist-views often get extremely close to the object during execution, it can become difficult for the agent to understand the overall object shape. Thus, we include the initial local observation $O^0_{loc,depth}$ at every step with a segmentation mask of the target object ($O^0_{loc,seg}$) so that the local policy is aware of which object to manipulate. We transform absolute proprioception into local by computing observations relative to the first time-step ($O_{loc,ee} = [X^0_{ee,t} - X^0_{ee}]$) and incorporate velocity information ($\dot{O}_{loc,ee,t}$), which improves transfer. Our observation space is $\mathbf{O^t_{loc}} = \langle O^t_{loc,depth}, O^0_{loc,seg}, O^0_{loc,depth}, O^t_{loc,ee}, \dot{O^t_{loc,ee}}\rangle$.

Augmentations To enable robustness to noisy real world observations, namely edge artifacts and irregular holes, we augment the clean depth maps we obtain in simulation. For edge artifacts, in which we observe dropped pixels and noisiness along edges, we use the correlated depth noise via bi-linear interpolation of shifted depth from [58] which tends to model this effect well. We also observe that real world depth maps tend to have randomly placed irregular holes (pixels with depth 0). As a result, we compute random pixel-level masks and Gaussian blur them to obtain irregularly shaped masks that we then apply to the depth image. We also use random camera cropping augmentations which has been shown to improve visuomotor learning performance [57]. Finally, we augment the proprioceptive observations to ensure robustness to exact measurements, adding uniformly random noise to the translation and rotation.

3.4 Zero-shot Text Conditioned Manipulation

Given our framework and trained local policies, how do we now deploy them in the real-world, to solve a wide array of manipulation tasks in a zero shot manner?

To enable our system to solve long-horizon tasks, $p_\theta(g_k|g, O)$, decomposes the task into a skill chain to execute given goal g. We implement p_θ as GPT-4o, a SOTA VLM. Given the task prompt g, descriptions of the pre-trained local skills and how they operate, and images of the scene O, we prompt GPT-4o to give a plan for the task structured as a list of (object, skill) tuples. For example, for the task shown in Fig. 2, GPT outputs ((handle, open), (rice pick), (microwave, place), (handle, close)). We then need a language conditioned pose estimator (to compute $X_{targ,k}$) that generalizes broadly; we opt to use Grounded SAM [24] due to its strong open-set segmentation capabilities. To estimate $X_{targ,k}$, we can segment the object pointcloud, average it to get a position and use its surface normals to select a collision-free orientation. One issue is that Grounding Dino [59], used in Grounded SAM, is very sensitive to the prompt. As a result, we pass its predictions back into GPT-4o to adjust the object prompts to capture the correct object.

For predicting τ_{reach}, while any motion planner can be used, we select Neural MP [25] due to its fast planning time (3s) and strong real-world planning performance. Given $X_{targ,k}$, we compute target joint state $q_{k,f}$, plan with Neural MP open-loop and execute the predicted τ_{reach} on the robot using a PID joint controller. We then execute the appropriate local policy (as predicted by the VLM) on the robot to perform manipulation. We alternate between motion planning and local policies until the task is complete. Finally, we note that the particular choice of models is orthogonal to our method.

4 Experimental Results

We pose the following experimental questions that guide our evaluation: 1) Can an autonomous agent control a robot to perform a wide array of *long-horizon* manipulation tasks zero-shot? 2) How does our approach compare to methods that learn from online interaction? 3) For direct sim2real transfer, how do Local Policies compare against end-to-end learning and other transfer techniques that leverage human correction data? 4) To what degree do the design decisions made in ManipGen affect the performance of the method?

4.1 Simulation Comparisons and Analysis

Robosuite Benchmark Results We first evaluate against the long-horizon manipulation tasks used in PSL [44] from the Robosuite benchmark [60] in simulation. We compare to end-to-end RL methods [61], hierarchical RL [62, 44], task and motion planning [63] and LLM planning [36]. In these experiments, we *zero-shot* transfer our trained policies to Robosuite and evaluate their performance against methods that use task specific data (Tab. 1). ManipGen outperforms or matches PSL, the SOTA method on these tasks, across the board, achieving an average success rate of 97.33% compared to 95.83%. These results demonstrate that ManipGen can outperform methods that are trained on the task of interest [44, 62, 61] as well as planning methods that have access to privileged state info [63, 36].

	Bread	Can	Milk	Cereal	CanBread	CerealMilk	Average
Stages	*2*	*2*	*2*	*2*	*4*	*4*	
DRQ-v2	52%	32%	2%	0%	0%	0%	14%
RAPS	0%	0%	0%	0%	0%	0%	0%
TAMP	90%	100%	85%	100%	72%	71%	86%
SayCan	93%	100%	90%	63%	63%	73%	80%
PSL	100%	100%	100%	100%	90%	85%	96%
Ours	100%	100%	99%	97%	97%	91%	**97%**

Table 1: **Robosuite Benchmark Results.** ManipGen zero-shot transfers to Robosuite, outperforming end-to-end and hierarchical RL methods as well as traditional and LLM planning methods.

Tasks	Ours	Transic	Direct Transfer	DR. & Data Aug. [48]	HG-Dagger [68]	IWR [69]	BC [65]
Stabilize	95%	**100%**	10%	35%	65%	65%	40%
Reach and Grasp	**95%**	**95%**	35%	60%	30%	40%	25%
Insert	**80%**	45%	0%	15%	35%	40%	10%
Avg	**90%**	80%	15%	36.7%	43.3%	48.3%	25%

Table 2: **Transic Benchmark Results** ManipGen achieves SOTA results on the Transic [16] benchmark in terms of task success rate without using any real world data, outperforming direct transfer, imitation learning and human-in-the-loop methods.

ManipGen Analysis and Ablations. We study design decisions proposed in our method by training single object pick policies on 5 objects (remote, can, bowl, bottle, camera) and testing on out held out poses. We begin with our observation space design choices: ManipGen achieves 97.44% success rate in comparison to (94.33%, 96.64%, 97.25%) for removing key-point observations, success observation and reward observations respectively. Incorporating key-point observations is the most impactful change, enabling the agent to perceive the shape of the target object. Next, we evaluate how the level of locality (the size of the region around the target object that we initialize over) affects learning performance. At convergence, we find that ManipGen (8cm max distance from target) achieves 97.44% success rate while performance diminishes with increasing distance (95.65%, 89.55%, 72.52%) for 16cm, 32cm and 64cm respectively.

For DAgger, we analyze our observation design choices and find that including velocity information, the first observation, and changing proprioception to be relative to the first frame are crucial to the success of our method. While ManipGen gets 94.3% success, removing velocity info and using absolute proprioception hurt significantly (89.92% and 90.94%) while removing the first observation drops performance to 93.13%. We also vary the DAgger buffer size, from 1 (on-policy), 10, 100, and 1000 (off-policy) for multitask training (with 3.5K objects, not 5). We find that 100 performs best, achieving 85% in simulation averaged across 100 held out objects, out performing (78%, 82% and 75%) for 1, 10 and 1000 respectively.

4.2 Real World Evaluation

FurnitureBench Results To evaluate the sim2real capabilities of local policies (Tab. 2), we deploy ManipGen on FurnitureBench [64], comparing against a wide array of direct-transfer [48], imitation learning [65, 66], offline RL [67] and human-in-the-loop methods [16, 68, 69] from Transic [16]. These tasks are single stage; we train local policies to perform pushing (**Stabilize**), picking (**Reach and Grasp**) and insertion (**Insert**). We predict a start pose to initialize the local policy from and deploy the simulation-trained policies. ManipGen matches or outperforms end-to-end direct transfer methods (75%, 53.3%), imitation methods (55%, 82.7%, 65%, 75%, 86.7%) and sim2real methods that leverage additional correction data [16]. For Insert, local policies are able to outperform Transic without using any real world data, achieving 80% while Transic achieves 45%. These experiments demonstrate ManipGen improves over end-to-end learning and is capable of handling challenging initial states, contact-rich interaction and precise motions.

Zero-shot Long-horizon Manipulation To test the generalization capabilities of our method, we propose 5 diverse long-horizon manipulation tasks (Fig. 1) which involve pick and place, obstacle

	Cook	Replace	CabinetStore	DrawerStore	Tidy	Avg
Stages	*2*	*4*	*4*	*6*	*8*	*4.8*
OpenVLA	0% (0.1)	0 (0.0)	0% (0.0)	0 (0.0)	0 (0.0)	0% (.02)
SayCan	80% (1.7)	10% (1.3)	70% (3.5)	20% (3.6)	20% (4.8)	40% (3.0)
LLMTrajGen	70% (1.5)	0% (0.6)	0% (0.6)	0% (1.0)	0% (2.6)	14% (1.3)
Ours	**90% (1.9)**	**80% (3.7)**	**90% (3.9)**	**60% (4.7)**	**60% (7.2)**	**76% (4.3)**

Table 3: **Zero-shot Long Horizon Manipulation** We report task success rate and average number of stages completed per real world task. ManipGen outperforms all methods on each task, achieving 76% with 4.28/4.8 stages completed on average.

avoidance and articulated object manipulation. **Cook**: put food into a pot on a stove (2 stages), **Replace**: take a pantry item out of the shelf, put it on a tray and take an object from the tray and put it in the shelf (4 stages), **CabinetStore**: open a drawer in the cabinet, put an object inside and close it (4 stages). **DrawerStore**: open a drawer, put two personal care items inside and close the drawer (6 stages) and **Tidy**: clean up the table by putting all the toys into a bin (8 stages). Each task has a unique object set (5 objects), receptacle (pot, shelf, etc.) and text description. We run 10 evaluations per task, randomizing which objects are present and their poses, receptacle poses, and target poses. All poses are randomized over the table and we select a diverse set of evaluation objects.

Comparisons We evaluate SOTA text-conditioned manipulation approaches: SayCan [36] and LLMTrajGen [45]. For SayCan, we use our VLM and motion planning system with engineered primitives for interaction; testing the importance of training local policies. We compare against a pre-trained model for manipulation, OpenVLA [70]. For each task, we collect 25 demonstrations on held out objects in held out poses and scene configurations and fine-tune OpenVLA per task. We pass in a text prompt specifying the task, recording the task success rate and number of stages completed.

Across all 5 tasks (Tab. 3), we find that ManipGen outperforms all methods, achieving 76% **zero-shot success rate** overall. Note that we have not trained our local policies on *any of these specific objects* or in *these specific configurations*; there is *no adaptation* in the real world. ManipGen is able to avoid obstacles while performing manipulation of unseen objects in arbitrary poses and configurations. Failure cases for our method resulted from 1) vision failures as open-set detection models such as Grounding Dino [59] detected the wrong object, 2) imperfect motion planning, resulting in collisions with the environment during execution which dropped objects sometimes and 3) local policies failing to manipulate from sub-optimal initial poses. In general, DrawerStore and Tidy are the most challenging tasks due to their horizon, and consequently all methods, including our own perform worse (60% for ours, 20% for best baseline).

SayCan is the strongest baseline (40% success), achieving non-zero success on every task by leveraging the generalization capabilities of vision-language foundation models in a structured manner. However, when initial poses are not ideal or the task requires contact-rich control, pre-defined primitives fall apart (10-20% success). LLMTrajGen, while capable of performing top-down unconstrained pick and place (Cook: 70%), only makes partial progress on tasks requiring obstacle avoidance (Replace) or articulated object manipulation (Store) as its prompts struggle to cover those cases well. Finally, OpenVLA failed to solve any task, failing to generalize to held out objects and poses even though it was the only method that was given few-shot data. We attempted to evaluate it on its training objects and it still performs poorly with strong pose randomization.

5 Discussion

We present ManipGen, a method for solving long-horizon manipulation tasks with unseen objects in unseen configurations by training generalist policies for sim2real transfer. We propose local policies, a novel policy class for sim2real transfer that is pose, skill order and scene configuration invariant, enabling broad generalization. For deployment, we take advantage of the generalization capabilities of foundation models for vision, language and motion planning to solve long-horizon manipulation tasks from text prompts. Across 50 real-world long-horizon manipulation tasks, our method achieves 76% *zero-shot* success, outperforming SOTA planning and imitation methods on every task.

References

[1] R. OpenAI. Gpt-4 technical report. *arXiv*, pages 2303–08774, 2023.

[2] O.-X. E. Collaboration, A. Padalkar, A. Pooley, A. Jain, A. Bewley, A. Herzog, A. Irpan, A. Khazatsky, A. Rai, A. Singh, et al. Open x-embodiment: Robotic learning datasets and rt-x models. *arXiv preprint arXiv:2310.08864*, 2023.

[3] A. Khazatsky, K. Pertsch, S. Nair, A. Balakrishna, S. Dasari, S. Karamcheti, S. Nasiriany, M. K. Srirama, L. Y. Chen, K. Ellis, P. D. Fagan, J. Hejna, M. Itkina, M. Lepert, Y. J. Ma, P. T. Miller, J. Wu, S. Belkhale, S. Dass, H. Ha, A. Jain, A. Lee, Y. Lee, M. Memmel, S. Park, I. Radosavovic, K. Wang, A. Zhan, K. Black, C. Chi, K. B. Hatch, S. Lin, J. Lu, J. Mercat, A. Rehman, P. R. Sanketi, A. Sharma, C. Simpson, Q. Vuong, H. R. Walke, B. Wulfe, T. Xiao, J. H. Yang, A. Yavary, T. Z. Zhao, C. Agia, R. Baijal, M. G. Castro, D. Chen, Q. Chen, T. Chung, J. Drake, E. P. Foster, J. Gao, D. A. Herrera, M. Heo, K. Hsu, J. Hu, D. Jackson, C. Le, Y. Li, K. Lin, R. Lin, Z. Ma, A. Maddukuri, S. Mirchandani, D. Morton, T. Nguyen, A. O'Neill, R. Scalise, D. Seale, V. Son, S. Tian, E. Tran, A. E. Wang, Y. Wu, A. Xie, J. Yang, P. Yin, Y. Zhang, O. Bastani, G. Berseth, J. Bohg, K. Goldberg, A. Gupta, A. Gupta, D. Jayaraman, J. J. Lim, J. Malik, R. Martín-Martín, S. Ramamoorthy, D. Sadigh, S. Song, J. Wu, M. C. Yip, Y. Zhu, T. Kollar, S. Levine, and C. Finn. Droid: A large-scale in-the-wild robot manipulation dataset. 2024.

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

[5] H. Walke, K. Black, A. Lee, M. J. Kim, M. Du, C. Zheng, T. Zhao, P. Hansen-Estruch, Q. Vuong, A. He, V. Myers, K. Fang, C. Finn, and S. Levine. Bridgedata v2: A dataset for robot learning at scale. In *Conference on Robot Learning (CoRL)*, 2023.

[6] 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.

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

[8] J. Lee, J. Hwangbo, L. Wellhausen, V. Koltun, and M. Hutter. Learning quadrupedal locomotion over challenging terrain. *Science robotics*, 5(47):eabc5986, 2020.

[9] A. Kumar, Z. Fu, D. Pathak, and J. Malik. Rma: Rapid motor adaptation for legged robots. *arXiv preprint arXiv:2107.04034*, 2021.

[10] A. Agarwal, A. Kumar, J. Malik, and D. Pathak. Legged locomotion in challenging terrains using egocentric vision. In *Conference on Robot Learning*, pages 403–415. PMLR, 2023.

[11] Z. Zhuang, Z. Fu, J. Wang, C. Atkeson, S. Schwertfeger, C. Finn, and H. Zhao. Robot parkour learning. *arXiv preprint arXiv:2309.05665*, 2023.

[12] X. Cheng, K. Shi, A. Agarwal, and D. Pathak. Extreme parkour with legged robots. *arXiv preprint arXiv:2309.14341*, 2023.

[13] D. Hoeller, N. Rudin, D. Sako, and M. Hutter. Anymal parkour: Learning agile navigation for quadrupedal robots. *Science Robotics*, 9(88):eadi7566, 2024.

[14] B. Tang, M. A. Lin, I. Akinola, A. Handa, G. S. Sukhatme, F. Ramos, D. Fox, and Y. S. Narang. Industreal: Transferring contact-rich assembly tasks from simulation to reality. In K. E. Bekris, K. Hauser, S. L. Herbert, and J. Yu, editors, *Robotics: Science and Systems XIX, Daegu, Republic of Korea, July 10-14, 2023*, 2023. doi:10.15607/RSS.2023.XIX.039. URL `https://doi.org/10.15607/RSS.2023.XIX.039`.

[15] B. Tang, I. Akinola, J. Xu, B. Wen, A. Handa, K. Van Wyk, D. Fox, G. S. Sukhatme, F. Ramos, and Y. Narang. Automate: Specialist and generalist assembly policies over diverse geometries. *arXiv preprint arXiv:2407.08028*, 2024.

[16] Y. Jiang, C. Wang, R. Zhang, J. Wu, and L. Fei-Fei. Transic: Sim-to-real policy transfer by learning from online correction. *arXiv preprint arXiv: Arxiv-2405.10315*, 2024.

[17] I. Akkaya, M. Andrychowicz, M. Chociej, M. Litwin, B. McGrew, A. Petron, A. Paino, M. Plappert, G. Powell, R. Ribas, et al. Solving rubik's cube with a robot hand. *arXiv preprint arXiv:1910.07113*, 2019.

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

[19] T. G. W. Lum, M. Matak, V. Makoviychuk, A. Handa, A. Allshire, T. Hermans, N. D. Ratliff, and K. Van Wyk. Dextrah-g: Pixels-to-action dexterous arm-hand grasping with geometric fabrics. *arXiv preprint arXiv:2407.02274*, 2024.

[20] T. Chen, M. Tippur, S. Wu, V. Kumar, E. Adelson, and P. Agrawal. Visual dexterity: In-hand dexterous manipulation from depth. *arXiv preprint arXiv:2211.11744*, 2022.

[21] E. Todorov, T. Erez, and Y. Tassa. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ international conference on intelligent robots and systems*, pages 5026–5033. IEEE, 2012.

[22] O. M. Andrychowicz, B. Baker, M. Chociej, R. Jozefowicz, B. McGrew, J. Pachocki, A. Petron, M. Plappert, G. Powell, A. Ray, et al. Learning dexterous in-hand manipulation. *The International Journal of Robotics Research*, 39(1):3–20, 2020.

[23] Y. S. Narang, K. Storey, I. Akinola, M. Macklin, P. Reist, L. Wawrzyniak, Y. Guo, Á. Moravánszky, G. State, M. Lu, A. Handa, and D. Fox. Factory: Fast contact for robotic assembly. In K. Hauser, D. A. Shell, and S. Huang, editors, *Robotics: Science and Systems XVIII, New York City, NY, USA, June 27 - July 1, 2022*, 2022. doi:10.15607/RSS.2022.XVIII.035. URL `https://doi.org/10.15607/RSS.2022.XVIII.035`.

[24] T. Ren, S. Liu, A. Zeng, J. Lin, K. Li, H. Cao, J. Chen, X. Huang, Y. Chen, F. Yan, Z. Zeng, H. Zhang, F. Li, J. Yang, H. Li, Q. Jiang, and L. Zhang. Grounded sam: Assembling open-world models for diverse visual tasks, 2024.

[25] M. Dalal, J. Yang, R. Mendonca, Y. Khaky, R. Salakhutdinov, and D. Pathak. Neural mp: A generalist neural motion planner. *arXiv preprint arXiv:2409.05864*, 2024.

[26] A. I. Center. Shakey the robot. 1984.

[27] R. P. Paul. *Robot manipulators: mathematics, programming, and control: the computer control of robot manipulators*. Richard Paul, 1981.

[28] D. E. Whitney. The mathematics of coordinated control of prosthetic arms and manipulators. 1972.

[29] M. Vukobratović and V. Potkonjak. *Dynamics of manipulation robots: theory and application*. Springer, 1982.

[30] D. Kappler, F. Meier, J. Issac, J. Mainprice, C. G. Cifuentes, M. Wüthrich, V. Berenz, S. Schaal, N. Ratliff, and J. Bohg. Real-time perception meets reactive motion generation. *IEEE Robotics and Automation Letters*, 3(3):1864–1871, 2018.

[31] R. R. Murphy. *Introduction to AI robotics*. MIT press, 2019.

[32] C. R. Garrett, C. Paxton, T. Lozano-Pérez, L. P. Kaelbling, and D. Fox. Online replanning in belief space for partially observable task and motion problems. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5678–5684. IEEE, 2020.

[33] M. T. Mason. *Mechanics of robotic manipulation*. MIT press, 2001.

[34] D. E. Whitney. *Mechanical assemblies: their design, manufacture, and role in product development*, volume 1. Oxford university press New York, 2004.

[35] L. P. Kaelbling and T. Lozano-Pérez. Integrated task and motion planning in belief space. *The International Journal of Robotics Research*, 32(9-10):1194–1227, 2013.

[36] M. Ahn, A. Brohan, N. Brown, Y. Chebotar, O. Cortes, B. David, C. Finn, K. Gopalakrishnan, K. Hausman, A. Herzog, D. Ho, J. Hsu, J. Ibarz, B. Ichter, A. Irpan, E. Jang, R. J. Ruano, K. Jeffrey, S. Jesmonth, N. J. Joshi, R. Julian, D. Kalashnikov, Y. Kuang, K.-H. Lee, S. Levine, Y. Lu, L. Luu, C. Parada, P. Pastor, J. Quiambao, K. Rao, J. Rettinghouse, D. Reyes, P. Sermanet, N. Sievers, C. Tan, A. Toshev, V. Vanhoucke, F. Xia, T. Xiao, P. Xu, S. Xu, and M. Yan. Do as i can, not as i say: Grounding language in robotic affordances. *arXiv preprint arXiv: Arxiv-2204.01691*, 2022.

[37] W. Huang, F. Xia, T. Xiao, H. Chan, J. Liang, P. Florence, A. Zeng, J. Tompson, I. Mordatch, Y. Chebotar, et al. Inner monologue: Embodied reasoning through planning with language models. *arXiv preprint arXiv:2207.05608*, 2022.

[38] W. Huang, P. Abbeel, D. Pathak, and I. Mordatch. Language models as zero-shot planners: Extracting actionable knowledge for embodied agents. In *International Conference on Machine Learning*, pages 9118–9147. PMLR, 2022.

[39] J. Wu, R. Antonova, A. Kan, M. Lepert, A. Zeng, S. Song, J. Bohg, S. Rusinkiewicz, and T. Funkhouser. Tidybot: Personalized robot assistance with large language models. *arXiv preprint arXiv:2305.05658*, 2023.

[40] K. Lin, C. Agia, T. Migimatsu, M. Pavone, and J. Bohg. Text2motion: From natural language instructions to feasible plans. *arXiv preprint arXiv:2303.12153*, 2023.

[41] W. Huang, C. Wang, R. Zhang, Y. Li, J. Wu, and L. Fei-Fei. Voxposer: Composable 3d value maps for robotic manipulation with language models. *arXiv preprint arXiv:2307.05973*, 2023.

[42] J. Zhang, J. Zhang, K. Pertsch, Z. Liu, X. Ren, M. Chang, S.-H. Sun, and J. J. Lim. Bootstrap your own skills: Learning to solve new tasks with large language model guidance. *Conference on Robot Learning*, 2023.

[43] B. Liu, Y. Jiang, X. Zhang, Q. Liu, S. Zhang, J. Biswas, and P. Stone. Llm+ p: Empowering large language models with optimal planning proficiency. *arXiv preprint arXiv:2304.11477*, 2023.

[44] M. Dalal, T. Chiruvolu, D. Chaplot, and R. Salakhutdinov. Plan-seq-learn: Language model guided rl for solving long horizon robotics tasks. In *International Conference on Learning Representations (ICLR)*, 2024.

[45] T. Kwon, N. Di Palo, and E. Johns. Language models as zero-shot trajectory generators. *IEEE Robotics and Automation Letters*, 2024.

[46] H. Huang, F. Lin, Y. Hu, S. Wang, and Y. Gao. Copa: General robotic manipulation through spatial constraints of parts with foundation models. *arXiv preprint arXiv:2403.08248*, 2024.

[47] A. Agarwal, S. Uppal, K. Shaw, and D. Pathak. Dexterous functional grasping. In *7th Annual Conference on Robot Learning*, 2023.

[48] X. B. Peng, M. Andrychowicz, W. Zaremba, and P. Abbeel. Sim-to-real transfer of robotic control with dynamics randomization. *arXiv preprint arXiv: Arxiv-1710.06537*, 2017.

[49] R. S. Sutton, D. Precup, and S. Singh. Between mdps and semi-mdps: A framework for temporal abstraction in reinforcement learning. *Artificial Intelligence*, 112(1):181–211, 1999. ISSN 0004-3702. doi:https://doi.org/10.1016/S0004-3702(99)00052-1. URL https://www.sciencedirect.com/science/article/pii/S0004370299000521.

[50] S. Uppal, A. Agarwal, H. Xiong, K. Shaw, and D. Pathak. Spin: Simultaneous perception interaction and navigation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18133–18142, 2024.

[51] Y. Xu, W. Wan, J. Zhang, H. Liu, Z. Shan, H. Shen, R. Wang, H. Geng, Y. Weng, J. Chen, et al. Unidexgrasp: Universal robotic dexterous grasping via learning diverse proposal generation and goal-conditioned policy. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4737–4746, 2023.

[52] K. Mo, S. Zhu, A. X. Chang, L. Yi, S. Tripathi, L. J. Guibas, and H. Su. Partnet: A large-scale benchmark for fine-grained and hierarchical part-level 3d object understanding. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 909–918, 2019.

[53] M. Sundermeyer, A. Mousavian, R. Triebel, and D. Fox. Contact-graspnet: Efficient 6-dof grasp generation in cluttered scenes. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 13438–13444. IEEE, 2021.

[54] 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*, pages 627–635. JMLR Workshop and Conference Proceedings, 2011.

[55] K. Hsu, M. J. Kim, R. Rafailov, J. Wu, and C. Finn. Vision-based manipulators need to also see from their hands. *arXiv preprint arXiv:2203.12677*, 2022.

[56] M. Dalal, A. Mandlekar, C. Garrett, A. Handa, R. Salakhutdinov, and D. Fox. Imitating task and motion planning with visuomotor transformers. 2023.

[57] 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 *arXiv preprint arXiv:2108.03298*, 2021.

[58] J. T. Barron and J. Malik. Intrinsic scene properties from a single rgb-d image. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 17–24, 2013.

[59] 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.

[60] Y. Zhu, J. Wong, A. Mandlekar, R. Martín-Martín, A. Joshi, S. Nasiriany, and Y. Zhu. robosuite: A modular simulation framework and benchmark for robot learning. *arXiv preprint arXiv: Arxiv-2009.12293*, 2020.

[61] D. Yarats, R. Fergus, A. Lazaric, and L. Pinto. Mastering visual continuous control: Improved data-augmented reinforcement learning. *arXiv preprint arXiv:2107.09645*, 2021.

[62] M. Dalal, D. Pathak, and R. R. Salakhutdinov. Accelerating robotic reinforcement learning via parameterized action primitives. *Advances in Neural Information Processing Systems*, 34: 21847–21859, 2021.

[63] C. R. Garrett, T. Lozano-Pérez, and L. P. Kaelbling. Pddlstream: Integrating symbolic planners and blackbox samplers via optimistic adaptive planning. In *Proceedings of the International Conference on Automated Planning and Scheduling*, volume 30, pages 440–448, 2020.

[64] M. Heo, Y. Lee, D. Lee, and J. J. Lim. Furniturebench: Reproducible real-world benchmark for long-horizon complex manipulation. In K. E. Bekris, K. Hauser, S. L. Herbert, and J. Yu, editors, *Robotics: Science and Systems XIX, Daegu, Republic of Korea, July 10-14, 2023*, 2023. doi:10.15607/RSS.2023.XIX.041. URL https://doi.org/10.15607/RSS.2023.XIX.041.

[65] D. A. Pomerleau. Alvinn: An autonomous land vehicle in a neural network. In D. Touretzky, editor, *Advances in Neural Information Processing Systems*, volume 1. Morgan-Kaufmann, 1988. URL https://proceedings.neurips.cc/paper_files/paper/1988/file/812b4ba287f5ee0bc9d43bbf5bbe87fb-Paper.pdf.

[66] 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: Arxiv-2108.03298*, 2021.

[67] I. Kostrikov, A. Nair, and S. Levine. Offline reinforcement learning with implicit q-learning. *arXiv preprint arXiv: Arxiv-2110.06169*, 2021.

[68] M. Kelly, C. Sidrane, K. Driggs-Campbell, and M. J. Kochenderfer. Hg-dagger: Interactive imitation learning with human experts. *arXiv preprint arXiv: Arxiv-1810.02890*, 2018.

[69] 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: Arxiv-2012.06733*, 2020.

[70] M. J. Kim, K. Pertsch, S. Karamcheti, T. Xiao, A. Balakrishna, S. Nair, R. Rafailov, E. Foster, G. Lam, P. Sanketi, et al. Openvla: An open-source vision-language-action model. *arXiv preprint arXiv:2406.09246*, 2024.

[71] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv: Arxiv-1707.06347*, 2017.

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

[73] 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: Arxiv-2403.04436*, 2024.

[74] C. Finn, X. Y. Tan, Y. Duan, T. Darrell, S. Levine, and P. Abbeel. Deep spatial autoencoders for visuomotor learning. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pages 512–519. IEEE, 2016.

[75] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural Computation MIT-Press*, 1997.

[76] A. Vaswani. Attention is all you need. *Advances in Neural Information Processing Systems*, 2017.

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

Appendix

A Experiment Details

A.1 Training and Deployment Details

Architecture and Training We train all RL policies at scale using PPO [71] in GPU-parallelized simulation [72] (Fig. 3). We train for 500 epochs, with an environment batch size of 8192 and max episode length of 120 steps per skill. To learn visuomotor policies to perform high-frequency (60 Hz) end-effector control, we pair Resnet-18 [73] and Spatial Soft-max [74] with a two layer MLP decoder (4096 hidden units). Finally, for training, minimizing Mean Squared Error loss is sufficient for learning multitask policies via DAgger. In early experiments, we found that our architecture performs comparably to using LSTMs [75], Transformers [76], and ACT [6] and is faster to train (5-10x) and deploy (2x).

Hardware Setup We use the Franka Panda robot arm with the UMI [77] gripper fingertips and a wrist-mounted Intel Realsense d405 camera for obtaining local observations (84x84 resolution). We perform hole-filling and smoothing to clean the depth maps. For real world control, we use a TSI end-effector controller at 60 Hz with (Leaky) Policy Level Action Integration (PLAI) [14]. We use Leaky PLAI with .001 position action scale, .05 rotation action scale for pick and .005 rotation action scale for all other skills. Finally, we use 4 calibrated Intel Realsense d455 cameras for global view observations (640x480).

Figure 3: **Training Environments** We train local policies (left to right) on picking, placing, handle grasping, opening and closing.