SCALING ROBOT LEARNING WITHOUT SCALING HUMAN EFFORT

A MASTER'S THESIS
SUBMITTED TO THE DEPARTMENT OF COMPUTER SCIENCE
OF STANFORD UNIVERSITY
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
MASTER OF SCIENCE WITH DISTINCTION IN RESEARCH

Olivia Y. Lee
May 2025

I certify that I have read this thesis and that, in my opinion, it is fully adequate in scope and quality as a thesis for the degree of Master of Science with Distinction in Research.

May 15th, 2025
Prof. Jeannette Bohg
Department of Computer Science

I certify that I have read this thesis and that, in my opinion, it is fully adequate in scope and quality as a thesis for the degree of Master of Science with Distinction in Research.

May 15th, 2025
Prof. C. Karen Liu
Department of Computer Science

Abstract

Deploying generalist robots that can learn and adapt to new tasks at scale without extensive human effort has been a longstanding goal in robotics. Early approaches in robotics relied heavily on manual engineering, which do not scale easily. This has motivated recent interest in learning-based methods for robotics, such as imitation learning (IL) and reinforcement learning (RL), that scale with data. While IL learns direct state-action correspondences, it requires high-quality action labels to learn successfully, thus requiring substantial human effort in collecting demonstrations. This process is costly to scale, especially as robot hardware and teleoperation systems become increasingly complex. In comparison, RL offers a scalable alternative by allowing robots to learn and explore by optimizing a reward signal, thereby learning robust behaviors through interaction without significantly scaling up human data collection. However, RL faces practical challenges in sample complexity and task-specific reward engineering, which is challenging in real-world settings.

This thesis focuses on the problem of scaling robot learning to new tasks, objects, and environments without proportionally scaling the amount of human effort required. We analyze two data-efficient RL approaches that tackle these challenges: (1) real-world autonomous RL shaped by VLM-generated rewards, and (2) sim-to-real RL from one human demonstration. In the first part of this thesis, we present a method that facilitates autonomous RL fine-tuning of policies pre-trained on diverse human teleoperated data, enabling robots to improve through interaction and self-practice. In the second part of this thesis, we propose a real-to-sim-to-real RL framework for learning robust dexterous manipulation skills from one human hand video demonstration, by extracting dense object-centric rewards and useful pregrasp priors from the video for training RL policies in simulation. We conclude by analyzing the opportunities and challenges of real-world RL, sim-to-real RL, and parallel developments in robotic foundation models, and discuss how we might unify these paradigms to most efficiently leverage data provided by humans to train robust robot policies.

Acknowledgments

First, I would like express my gratitude to my research advisor, Jeannette Bohg. Jeannette is one of the most genuine and supportive advisors I have met throughout my research journey. From the day I joined the lab, her unwavering dedication to guiding her students, from developing impactful research ideas to writing and presenting our work, was clear. Her empathy and encouragement have been invaluable in helping me navigate the setbacks and doubts that arose in my research work. Much of the growth I have experienced as a researcher is thanks to her incredible mentorship, and I am immensely grateful for the opportunity to work under her guidance.

Next, I would like to thank my secondary advisor, C. Karen Liu, for offering valuable feedback and posing thought-provoking questions about my work. Karen's insights into framing and presenting my research ideas are a testament to her systematic approach to identifying core research arguments, which has inspired me to strengthen the cohesion and clarity of my own scientific reasoning and presentation style. I would also like to thank Kuan Fang who supervised the work presented in the first half of this thesis, and has been a great supportive mentor throughout my research journey.

I am also thankful for the mentorship of several members of IPRL Lab over the past year. The strong community of researchers at IPRL has inspired many ideas and great conversations, while also fostering a welcoming and supportive research environment. I am grateful to have advanced my research in robotics, reinforcement learning, and computer vision with this group, and I am excited to follow their undoubtedly excellent contributions in the coming years.

Last but not least, this thesis is dedicated to my family, friends, and loved ones, near and far, who have supported me through the ups and downs of my research journey. It truly takes a village, and this work was only made possible by their unwavering love and encouragement.

Contents

List of Figures & Tables

Chapter 1

Introduction

Deploying generalist robots that can learn and adapt to new tasks at scale without extensive human effort has been a longstanding goal in robotics. Early approaches in robotics relied heavily on manual engineering, crafting task-specific controllers, designing bespoke perception pipelines, and hand-tuning reward functions for each new skill [59, 113]. While these methods can yield impressive performance in narrow domains, they do not scale easily: every new object, environment, or manipulation primitive demands additional human supervision.

These limitations of manual engineering for robotics have motivated recent interest in learning-based methods that scale with data, specifically imitation learning (IL) and reinforcement learning (RL). These paradigms have shown promising results that point towards generalizable and scalable robotics by enabling robots to acquire manipulation skills by learning from data [5, 49, 69]. To that end, large-scale robot data collection efforts [19, 28, 126] in the form of demonstrations provided by human experts have been key sources of data for training these models.

However, several technical challenges remain. While IL learns direct state-action correspondences and scales well with diverse expert demonstrations, this often requires substantial human effort in the form of large-scale demonstration collection via teleoperated demonstrations [3, 5, 19, 46, 56, 69] or motion capture [11, 119, 128]. Even after training on large amounts of data, the resultant policies often struggle with robust recovery from failures or out-of-distribution states, unless such behaviors are demonstrated. Adapting or fine-tuning policies for new tasks also requires a moderate number of demonstrations, representing a non-trivial cost to scaling to new tasks [56, 57]. Furthermore, most large-scale robotic data collection efforts have used simple robotic arms with parallel-jaw grippers [19, 28, 55, 126]. Scaling up human demonstration collection can therefore be expensive and impractical, especially as robot hardware becomes more complex than the platforms traditionally used for robot learning, for instance in dexterous manipulation or whole-body control.

For scalable deployment in diverse real-world settings, robotic systems should allow human users to specify goals easily and flexibly with high-level instructions, without having to dramatically scale

teleoperated data collection and human effort. RL hence offers a promising alternative by allowing robotic agents to learn new tasks by optimizing a reward signal, demonstrating potential for scalable learning. Through exploration, RL policies can also generalize beyond the behaviors demonstrated in expert-provided trajectories and learn more robust behaviors, addressing a key limitation of IL. Furthermore, by specifying a reward function, RL can autonomously improve with minimal human intervention, enabling autonomous learning at scale. However, RL faces significant challenges in sample complexity: the high cost of real-world robot interaction makes conventional RL approaches impractical as they typically require thousands or millions of environment interactions to learn effective policies [27, 148]. Furthermore, the challenge of tedious task-specific reward engineering, especially in unstructured, real-world settings, can introduce bias and make performance brittle.

Despite these challenges, we believe RL offers a promising path toward scalable robot learning with minimal human effort as it *enables robots to learn through interaction in their own embodiments*. This opens up several key advantages: it enables robots to discover solutions more optimal for their own embodiments than merely imitating human behaviors; it can learn robust recovery behaviors through exploration; it can adapt to changing dynamics and unforeseen circumstances; and it allows for continuous improvement with minimal additional human demonstrations. By focusing on RL as our primary learning paradigm, we can leverage its inherent ability to discover novel solutions through trial and error, optimize behavior beyond human demonstrations, and adapt to dynamic conditions. The challenge is addressing RL's limitations, namely sample efficiency and reward specification, through strategic integration with other techniques. The approaches presented in this thesis aim to harness RL's advantages while dramatically reducing the human effort required by (1) leveraging foundation models to automatically generate reward signals, and (2) extracting useful abstractions from minimal human demonstrations for task specification and guidance.

Modern vision–language models (VLMs) represent a promising solution to RL's reward specification challenge. The rapid development of foundation models [29] trained on broad, Internet-scale data demonstrate remarkable adaptability to downstream tasks via in-context learning or parameter-efficient fine-tuning. In robotics, researchers have begun to harness the commensense understanding in pre-trained foundation models for robotic tasks [1, 25, 43, 68]. While some works derive rewards implicitly via visual representations of large pre-trained models [77, 78, 89], we explore explicit approaches to eliciting useful object representations embedded in VLMs zero-shot via mark-based visual prompting [30, 138]. By prompting VLMs to identify task-relevant affordances, object states, and waypoints, we can derive dense reward signals that guide RL exploration effectively without task-specific reward engineering. This approach addresses a critical bottleneck in policy fine-tuning with RL, while maintaining flexibility to specify diverse tasks through natural language. We demonstrate that VLM-generated dense rewards derived from affordance representations significantly improve sample efficiency of fine-tuning multi-task policies with RL, enabling robots to learn through interaction and self-practice in an autonomous, data-efficient manner.

Complementary to leveraging VLMs for reward specification, we also investigate formulating useful task abstractions to guide RL policy training using minimal human demonstrations. Traditional IL approaches typically require hundreds or thousands of demonstrations to learn robust policies, which is impractical and often suboptimal, especially for complex embodiments like multi-fingered dexterous hands. Instead of using demonstrations as direct examples to imitate, we propose using them as sources of task specification and guidance for exploration via RL. By extracting object pose trajectories and human pre-grasp poses from as little as one demonstration, we can define embodiment-agnostic reward functions and provide advantageous initializations for exploration. This perspective shifts the role of human data from being the primary learning signal to being a strategic resource that bootstraps and guides RL. Through RL in simulation, policies can learn behaviors optimal for the robot's specific embodiment through online interaction, potentially discovering solutions that differ from but outperform direct imitation of human demonstrations. This approach is particularly valuable for dexterous manipulation, where morphological differences between human hands and robotic end-effectors make direct imitation challenging. Our work demonstrates the importance of dense, object-centric rewards, useful priors from human-provided object grasps, and domain-randomized simulation to train robust RL policies that can achieve zero-shot sim-to-real transfer from a single human demonstration. This highlights RL's potential to dramatically reduce the amount human effort required for learning complex robotic manipulation skills.

This thesis focuses on the problem of scaling robot learning to new tasks, objects, and environments, without proportionally scaling the amount of human effort required. We present several ideas to address a critical challenge in the field: how can we most efficiently leverage data provided by humans to train robust robot policies? To explore this question, we present two data-efficient RL approaches: (1) real-world autonomous RL shaped by VLM-generated rewards, and (2) sim-to-real RL from one human demonstration. In Section 2, we present Keypoint-based Affordance Guidance for Improvements (KAGI), a method for facilitating autonomous RL fine-tuning of policies pre-trained on diverse human teleoperated data using VLM-generated shaping rewards. In Section 3, we propose HUMAN2SIM2ROBOT, a real-to-sim-to-real RL framework for training robust dexterous manipulation policies from a single human hand RGB-D video demonstration. Finally, in Section 4, we conclude our discussion by connecting the works presented with parallel research developments that facilitate scalable robot learning with minimal human effort.

This thesis covers works submitted for publication from [65, 73]. In Lee et al. [2024], I led project ideation and scoping, end-to-end methodology development and iteration, experiment design and implementation, paper writing and presentation, mentored by Annie Xie, Profs. Kuan Fang and Chelsea Finn. In Lum et al. [2025], which I co-led with Tyler Lum, I contributed to project ideation and scoping, methodology development (particularly real-to-sim, human demonstration processing, RL formulation, inference-time deployment), experiment design and implementation, paper writing and presentation, mentored by Profs. Jeannette Bohg and C. Karen Liu.

Chapter 2

Autonomous Reinforcement Learning via Visual Affordances

Much of recent robot learning research has been dedicated to developing generalist robots that can learn and adapt to new tasks. Typically, this is achieved by collecting a large amount of demonstration data, often in the same domain and manually teleoperated by a human expert, which is costly and limits generalization. Ideally, robotic systems should allow human users to specify reward functions easily and flexibly with high-level instructions. Reinforcement learning (RL) holds great promise of enabling robots to learn new tasks autonomously by optimizing a reward signal. Prior work has sought to improve the sample efficiency of these algorithms by pre-training on large offline datasets [60, 61, 62, 84, 88, 139]. However, learning robust reward functions still requires careful engineering or large amounts of data [31, 32, 38, 115, 133].

Recent advances in large language models (LLMs) and vision-language models (VLMs) trained on Internet-scale data show promising results in breaking down complex language instructions into task plans [1, 4, 8, 25, 42, 43], performing visual reasoning in varied contexts [9, 92, 138], and generating coarse robot motions for simple manipulation tasks [4, 42, 47, 86, 92, 129]. However, fine-tuning large pre-trained models to perform such tasks typically requires extensive human supervision, such as teleoperated demonstrations or feedback on whether the task was successfully completed. A group of prior work uses the generalization capabilities of LLMs/VLMs to solve open sets of tasks through prompt engineering and in-context learning [30, 44, 68], but these works use external motion planners and skill primitives to generate low-level actions, which require significant engineering efforts and are not robust. Another line of work prompts LLMs/VLMs to generate rewards for policy training via code or plan motions to solve the task [79, 109, 143], though existing approaches are only applied to high-level state spaces, navigation, simulated tasks, or coarse manipulation, and can fall short on real-world manipulation tasks requiring sophisticated spatial reasoning over visual inputs.

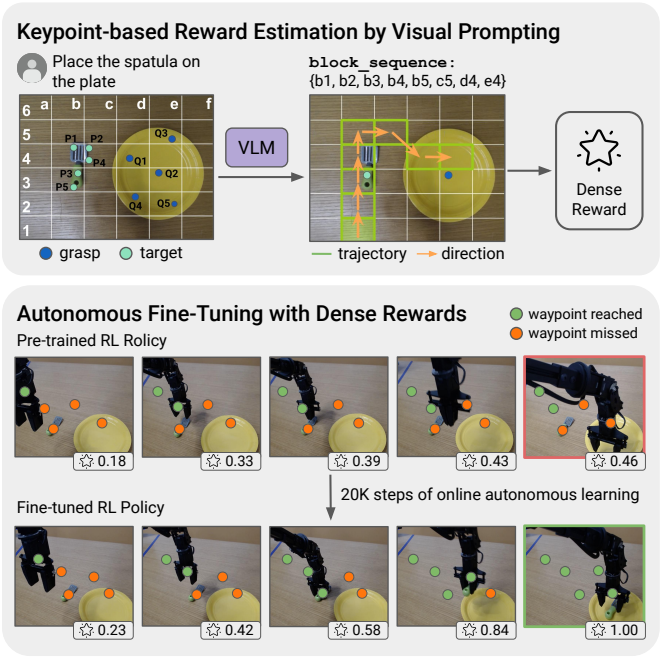


Figure 2.1: **Keypoint-based Affordance Guidance for Improvements (KAGI)** computes dense rewards defined using affordance keypoints and waypoint trajectories inferred by a VLM. Our dense rewards help to shape learned behaviors, facilitating efficient online fine-tuning across diverse real-world tasks.

Defining explicit rewards using pre-trained VLMs is hence an attractive approach for learning manipulation tasks. Thus far, VLMs have mainly been used to generate sparse rewards [80, 124, 139], which often leads to less efficient learning. VLMs hold much richer geometric and semantic representations that we can elicit, such as reasoning about the affordances of objects and environments. In this work, we leverage insights on effective visual prompting techniques [30, 138] to generate affordance keypoints and waypoint trajectories, from which we directly derive dense rewards to improve the efficiency of online fine-tuning with RL.

We present **K**eypoint-based **A**ffordance **G**uidance for **I**mprovements (KAGI), a method for fine-tuning policies pre-trained on diverse data using shaping rewards generated by VLMs. KAGI enables robots to learn autonomously through interaction and self-practice: our method uses open-vocabulary visual prompting to extract affordance keypoint representations zero-shot from VLMs and compute waypoint-based dense rewards (Figure 2.1), which we integrate into an autonomous RL system that uses sparse rewards from a fine-tuned VLM [139]. KAGI demonstrates improved success rates on complex manipulation tasks while being robust to reduced in-domain expert demonstrations, which systems using only sparse rewards or open-loop primitives struggle to generalize to.

2.1 Related Work

2.1.1 Leveraging Foundation Models for Robotics

Planning and high-level task reasoning. The rapid development of foundation models has drawn significant attention [29], demonstrating that models trained on Internet-scale data are highly adaptable to a wide range of downstream tasks, including robotics. Foundation models have enabled high-level planning and reasoning for embodied agents [1, 42, 43, 70, 104, 114, 145], generated subgoals [24, 40, 109], and generated coarse motions for manipulation [3, 42, 47, 86, 92, 129]. These impressive results show great potential in harnessing the rich knowledge and reasoning capabilities of foundation models for tackling robotic manipulation tasks.

Despite the rapid development of LLMs for robotic control, recent works have investigated using multi-modal models to jointly provide visual and language prompts to robotic agents [25, 34, 47, 92]. These approaches recognize that converting visual observations into language descriptions and planning solely with language loses rich information from the visual modality for scene understanding. Though LLMs can provide general priors for spatial planning and reasoning, they must be properly grounded to generate accurate responses about real-world environments [39, 41, 44, 131, 137]. Our work harnesses state-of-the-art VLMs to facilitate reasoning in both language and image domains. We leverage the image domain by computing rewards in image space to bridge high-level LLM plans with low-level actions for embodied tasks.

Spatial reasoning with VLMs. Language provides a highly natural interface for providing task instructions and specifying goals. That said, robotics relies heavily on accurately perceiving and interacting with the environment. Modern VLMs, such as GPT-4o [96] and Gemini [33], demonstrate promising capabilities combining reasoning with environment perception via visual inputs. Recent approaches have explored using VLMs for generating shaping rewards via code [79, 143], reward estimation [89, 109], or preference-based learning [130]. Our work focuses on directly obtaining rewards *zero-shot* from VLMs by leveraging the rich geometric and semantic representations that VLMs trained on Internet-scale data possess, such that VLM-generated dense rewards can be used effectively in closed-loop autonomous RL systems. Our approach is inspired by works that use generalizable and accurate keypoint-based reasoning of VLMs for zero-shot manipulation [30, 45, 92, 138], translating high-level plans into low-level robot actions. Beyond zero-shot open-loop manipulation, our work presents a novel application of open-vocabulary visual prompting to improve policies pre-trained on diverse data via online RL fine-tuning with VLM-defined dense rewards.

Modern VLMs can derive rewards in image space for learning state-action mappings via RL, enabling robots to learn through trial-and-error without training skill policies via imitation learning, which are in comparison more costly and difficult to scale. A key contribution of our work is using VLMs to determine task completion (sparse rewards) and specify intermediate waypoints as goals (dense rewards), where prior works focus mainly on the former [80, 124, 139]. Our work explores

using mark-based visual prompting (e.g., grid lines and object segmentations [105]) to augment raw image observations and service as guidance to facilitate semantic reasoning in VLMs, leveraging pre-trained representations in VLMs as zero-shot reward predictors.

2.1.2 Autonomous Reinforcement Learning

Online RL is the paradigm by which an agent gathers data through interaction with the environment, then stores this experience in a replay buffer to update its policy. This contrasts with offline RL, where an agent updates its policy using previously collected data without itself interacting with the environment. In autonomous RL, an agent autonomously practices skills and gathers real-world experience online. This approach holds great promise for scalable robot learning as agents learn through their own experience and do not require manual environment resets between trials [110].

Autonomous RL is difficult to implement in the real world, with the primary challenges being sample complexity, providing well-shaped rewards, and continual reset-free training. Several works have developed reset-free systems that reduce human interventions [2, 35, 117, 139], but reward shaping is an open problem. As manually specified rewards are difficult to engineer and easy to exploit, there is great potential to learn rewards from offline data or large pre-trained models. The large bank of image and video datasets, plus the fast inference speed and accessibility of large pre-trained models, could provide more precise and informative shaping rewards.

RoboFuME [139] is a recent work leveraging a reset-free pre-train fine-tune paradigm to train manipulation policies via autonomous RL. RoboFuME pre-trains offline RL policies on diverse offline data [28, 126] and in-domain demonstrations, then fine-tunes a sparse reward classifier [147] for online policy fine-tuning. However, online RL suffers when reward signals are too sparse, and fine-tuning the reward classifier requires a substantial number of in-domain demonstrations per task, which is costly but also makes the system more brittle and less robust to generalization. Our work similarly focuses on the pre-train fine-tune paradigm, specifically on improving the efficiency and robustness of online fine-tuning, enabling effective adaptation of RL policies pre-trained on diverse datasets. To do so, inspired by recent work extracting rewards from VLMs to guide zero-shot robotic manipulation [30, 80, 124] and online adaptation [134, 139], we leverage affordance representations extracted zero-shot from VLMs to tackle the dense reward shaping problem for online RL.

2.2 Methodology

Problem Statement. We propose Keypoint-based Affordance Guidance for Improvements (KAGI) to facilitate autonomous fine-tuning on unseen tasks as shown in Figure 2.2. We consider problems that can be formulated as a partially observable Markov Decision Process (POMDP) tuple $(\mathcal{S}, \mathcal{A}, \mathcal{O}, \gamma, f, p, r, d_0)$ where $\mathcal{S}$ is the state space, $\mathcal{A}$ is the action space, $\mathcal{O}$ is the observation space,

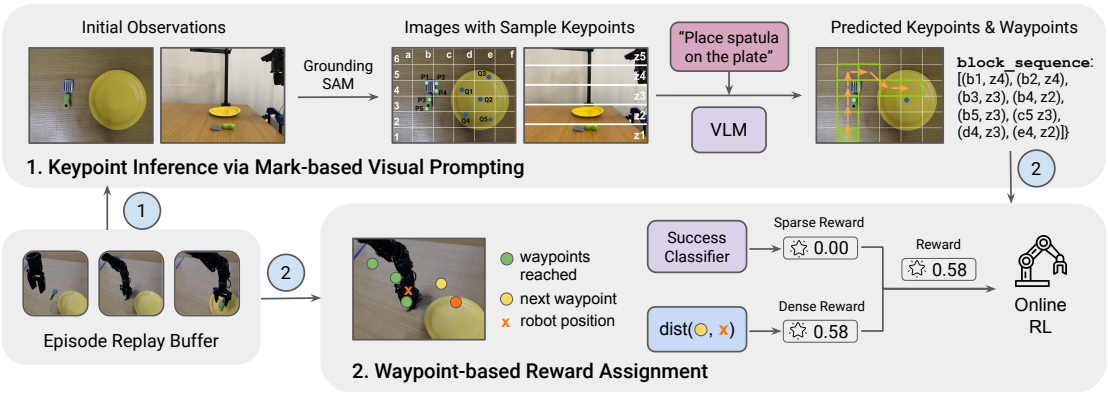


Figure 2.2: **Diagrammatic illustration of KAGI.** KAGI consists of two components. The first component leverages a VLM to select from a set of affordance keypoints, then generate a waypoint sequence. The second component involves per timestep reward computation for each frame in the episode replay buffer, computing dense reward with respect to the waypoint sequence and a sparse reward derived from a success classifier. The dense reward is used for online RL if the sparse reward is 0, else the sparse reward is used.

$\gamma \in (0, 1)$ is the discount factor, $r(s, a)$ is the reward function and $d_0(s)$ is the initial state distribution. The dynamics are governed by a transition function $p(s'|s, a)$. The observations are generated by an observation function $f(o|s)$. The goal of RL is to maximize the expected sum of discounted rewards $\mathbb{E}_\pi[\sum_{t=1}^{\infty} \gamma^t r(s_t, a_t)]$. In this work, we use RGB image-based observations. The reward function is typically hand-engineered or learned, for instance via examples of task success and failure [31, 32, 38, 115, 133, 139]. We also use a sparse task completion reward (i.e., $r(s, a) \in \{0, 1\}$), acquired with systems like RoboFuME [139].

Our system consists of an offline pre-training phase and an online fine-tuning phase, where the latter phase requires a reward function to label successes and failures. A sparse reward is typically easier to specify but, with it, RL algorithms require more samples to learn a successful policy, because the agent must encounter success through its own exploration. In comparison, a dense reward provides continuous feedback that guides the agent towards success. KAGI aims to provide the latter type of feedback by augmenting sparse task completion rewards with dense shaping rewards. Specifically, dense rewards are calculated with respect to intermediate waypoints marking trajectory points towards the goal. We find that such guidance can facilitate more efficient and generalizable online learning than sparse rewards alone.

KAGI: Keypoint-Based Reward Estimation via Visual Prompting. We employ an vision language model (VLM) to estimate dense rewards to facilitate fine-tuning of a pre-trained policy. We prompt the VLM to select appropriate grasp and target keypoints for the task, inspired by recent visual prompting techniques [30]. Then, the estimated rewards are used to generate a coarse trajectory of how the robot should complete the task. We then assign rewards to each timestep of an RL episode based on how well it follows this trajectory.

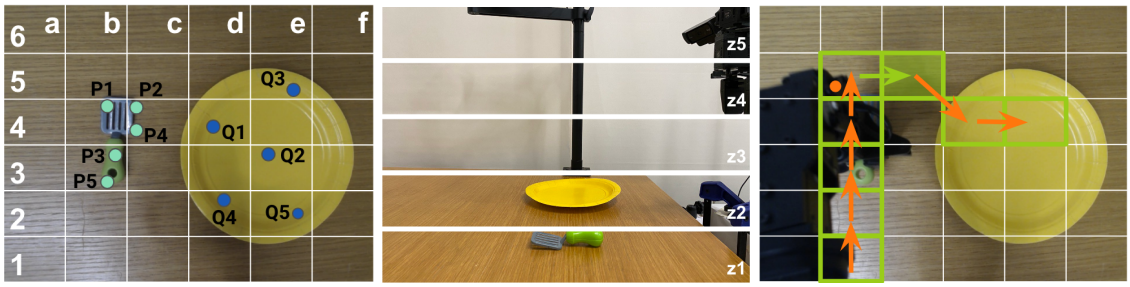
Our method thus consists of two main stages:

1. **Keypoint inference via mark-based visual prompting.** We take the first observation in the episode o_0, consisting of a top-down image o_0^d and a side view image o_0^s. Our goal is to create a candidate set of grasp and target keypoints that GPT-4o can select from. We preprocess these images by (1) passing o_0^d through SAM 2 [105] to get segmentations of relevant objects and sample five points from their masks, and (2) overlaying a 6×6 grid and the sampled keypoints on top of o_0^d to get $\tilde{o}^d$. For the side-view image, we augment o_0^s with evenly-spaced labeled horizontal lines to get $\tilde{o}^s$, providing depth information that is not available in the top-down view. See Figures 2.3a and 2.3b for sample processed inputs. We use this density of grid lines as [30] and preliminary tests found that this was most suitable for GPT-4o to reason with. We pass $\tilde{o}^d$ and $\tilde{o}^s$ with a language instruction and metaprompt to GPT-4o, which generates `block_sequence`. This sequence is a list of tuples (x, y, z), where (x, y) is a grid point chosen from $\tilde{o}^d$ representing a position in the xy-plane from top-down and z is chosen from $\tilde{o}^s$ representing height in the z-axis from the side view. Dense rewards are calculated with respect to `block_sequence` in the next step.

2. **Waypoint-based reward assignment.** For each frame in the episode, we compute the robot position in image space. We use two fitted RANSAC regressors, one for each camera view, to compute the robot position $(x^t_{\text{rob}}, y^t_{\text{rob}}, z^t_{\text{rob}})$, where $(x^t_{\text{rob}}, y^t_{\text{rob}})$ is from the top-down and z^t_{rob} is from the side view. Optionally, we can use pixel trackers [50] to track a specific point on the object $(x^t_{\text{obj}}, y^t_{\text{obj}}, z^t_{\text{obj}})$ to additionally define rewards based on object poses. Using these coordinates, we compute the nearest block to the robot position in `block_sequence`, B^i_{rob} (see Figure 2.3c for an example). We compute a reward r^t_{rob} is based on the L2 distance from the robot position to the *block after the closest block in* `block_sequence`, B^{i+1}_{rob}. We use the next block to encourage progression towards the goal and avoid stagnating at the current position. We transform the distance such that reward is between 0 and 1: $r_t = 0.5 \cdot (1 - tanh(\text{dist}))$. The reward is 1 when the sparse reward is 1, otherwise it is our dense reward. This reward is then optimized using an online RL algorithm [111].

To optimize this reward, we build on RoboFuME [139], an autonomous RL pipeline that learns from image observations. RoboFuME requires a set of forward and backward task demonstrations, and pre-trains a policy on this data and the Bridge dataset [28, 126]. For the reward model, it fine-tunes MiniGPT-4 [147] as a success classifier using the in-domain demonstrations and a few additional failure examples. We augment these sparse rewards with our waypoint-based dense rewards to combine the benefits of both modes of feedback.

From the VLM-generated waypoint trajectory, we use the centroid of each grid tile and height from the side-view to create a 3D coordinate trajectory. We calculate the L2 distance between the current robot position and target waypoint in image space, and pass each distance through a modified



(a) Top-down view (b) Side view (c) VLM Trajectory

Figure 2.3: **Example of annotated images to VLM (2.3a, 2.3b) and VLM-generated trajectory (2.3c).** In 2.3a, teal points labeled P1-5 denote grasp keypoints, blue points labeled Q1-5 denote target keypoints for the VLM to select from. In 2.3c, orange point is robot position, green tiles denote the generated trajectory, and arrows denote motion direction. KAGI's dense reward formulation encourages the robot to move to the next block, following the green arrow.

tanh function, so $r_{\text{dense}} = 0.5(1 - \tanh(\lambda(d_t - \varphi))$, where d_t is the L2 distance between the robot position and target waypoint at timestep t. The scaling factor λ and offset φ are hyperparameters; $\lambda = 0.1$, $\varphi = 15$ for the simulation experiments in Section 2.3.3, and $\lambda = 0.02$, $\varphi = 100$ for the real-robot experiments in Section 2.3.4. This ensures the dense reward stays between 0 and 1, with the dense reward values being closer to 1 when the robot trajectory is closer to the VLM-generated waypoint trajectory. What distinguishes trajectories that stay close to the VLM-generated trajectories and truly successful trajectories is the sparse reward. We set the final reward for each timestep $r = r_{\text{sparse}}$ if $r_{\text{sparse}} = 1$ else $r = r_{\text{dense}}$.

2.3 Experimental Results & Analysis

2.3.1 Key Research Questions

Our experiments aim to answer the following questions:

1. **Importance of Dense Rewards:** Does our VLM-based dense reward formulation improve the efficiency of online RL? (Section 2.3.4)
2. **Robustness of Demonstration Reductions:** Is our autonomous RL system robust to fewer in-domain demonstrations? (Section 2.3.5)
3. **Robustness to Environment Perturbations:** Is our closed-loop system more robust to environment perturbations than methods using open-loop action primitives? (Section 2.3.6)

We study tasks in both simulation and on a real robot to answer these questions.

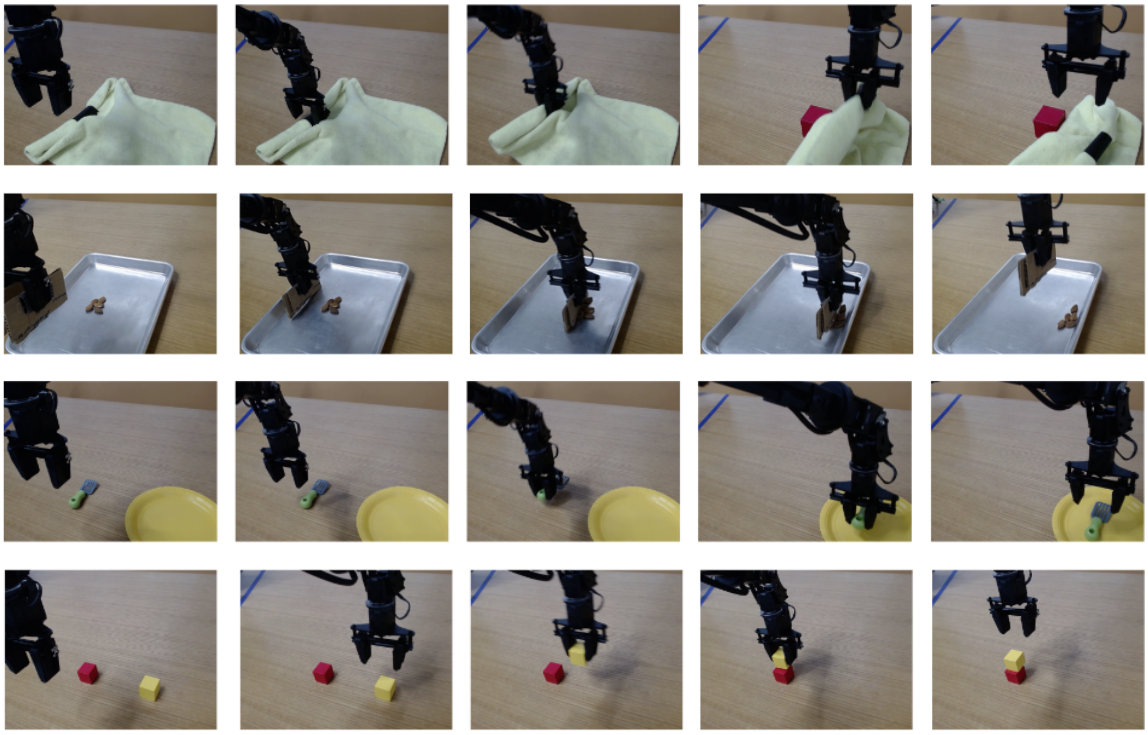


Figure 2.4: **Visualization of real-world tasks.** Evaluation is conducted on four real-world tasks spanning different manipulation types: Cloth Covering (deformables manipulation), Almond Sweeping (non-prehensile manipulation), Spatula Pick-Place (functional grasping), and Cube Stacking (precise manipulation).

2.3.2 Experimental Setup

Tasks. In PyBullet simulation, we study a *Bin-Sorting* task [62, 139], where a WidowX 250 arm needs to sort objects into the correct bin specified by the language instruction. On a physical WidowX 250 robot arm, we study *Cloth Covering*, *Almond Sweeping*, *Spatula Pick-Place*, and *Cube Stacking* (visualized in Figure 2.4). Our tasks span a range of complex skills involving manipulation of deformables, non-prehensile manipulation, functional grasping of objects handles, and precise grasping and placement of 3cm small cubes designed to challenge our system. Each of our tasks consists of a forward component (e.g., uncovering the block from under the cloth) and backward component (e.g., covering the block with the cloth), and therefore can be autonomously practiced by alternating between these two task components.

Comparisons. To assess performance without any online fine-tuning, we compare KAGI to language-conditioned behavior cloning and a pre-trained RL baseline, *CalQL* [90]. We also evaluate *RoboFuME* [139], which fine-tunes the pre-trained RL policy online with sparse rewards derived from a VLM success classifier (see Appendix A.2 for details on verifying successful reproduction of RoboFuME). This comparison allows us to understand the benefits of the proposed dense rewards.

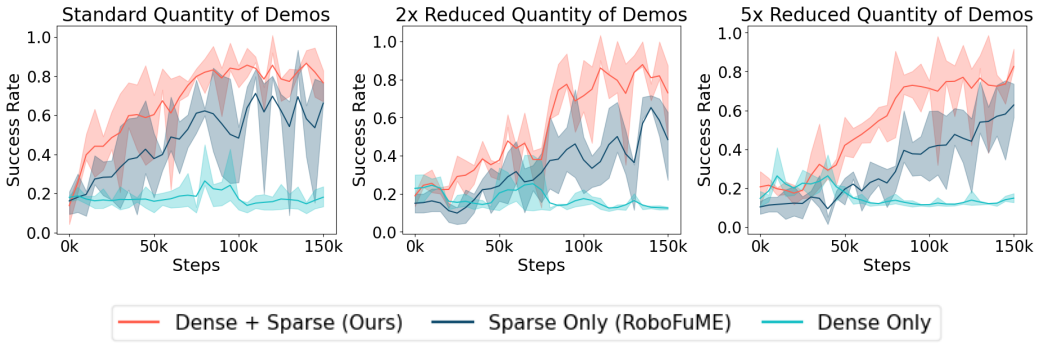


Figure 2.5: **Average success across 3 seeds on simulated Bin-Sorting.** We evaluate each reward formulation under: the standard number of demos (left), 2× reduction (middle), and 5× reduction (right).

Finally, we evaluate *MOKA* [30], which executes the trajectory defined by the VLM-generated waypoints. This last comparison does not perform any additional learning or fine-tuning.

Datasets. All methods except for MOKA are pre-trained on a combination of trajectories selected from the Bridge dataset [28, 126] and a set of in-domain demonstrations. The in-domain demonstrations consist of 50 forward trajectories and 50 backward trajectories. Following RoboFuME [139], we additionally collect 20 failure trajectories, which are used to train the VLM-based success classifier used by RoboFuME and KAGI. In Section 2.3.5, we test these methods on a 5× reduction in the quantity of in-domain demonstrations.

Evaluation. In simulation and real-world evaluations, we roll out each forward task policy for 20 trials. For the real-world tasks, success is evaluated as follows. For Cube Covering, the entire cube must be uncovered from the camera perspective shown in Figure 2.4; for Almond Sweeping, all five almonds must reach the right side of the tray; for Spatula Pick-Place, the spatula must be on the plate; for Cube Stacking, the yellow cube must be stable atop the red cube.

2.3.3 Simulation Experiments

We conduct a preliminary test of our dense reward formulation in a simulated Bin-Sorting task. Following the experimental setup in RoboFuME [139], each approach is provided 10 forward and backward demonstrations, 30 failure demonstrations, and 200 total demonstrations of other pick-place tasks of diverse objects. In simulation, rewards are computed in 100×100 pixel image space. We compare three reward formulations: dense waypoint-based only, sparse success-based only (RoboFuME), and dense + sparse (KAGI).

As shown in Figure 2.5, **our reward formulation achieves higher success rates than the sparse-only reward formulation**. The dense-only reward formulation performs worse than the other two. This indicates that both types of rewards are necessary for efficient and successful learning: while dense rewards are useful in shaping learned behaviors, sparse rewards are also crucial

Task	BC	Offline RL	RoboFuME	KAGI (Ours)
Cloth Covering	35%	55%	**80%**	**80%**
Almond Sweeping	25%	45%	70%	**80%**
Spatula Pick-Place	30%	25%	45%	**65%**
Cube Stacking	10%	15%	25%	**45%**

Table 2.1: **Success rates over 20 trials for each method on four real-world tasks.** We compare KAGI (fine-tuned for 30K steps) to offline-only methods and RoboFuME (fine-tuned for 30K steps).

to distinguish truly successful trajectories. We further test our reward formulation succeeds with a reduced number of demonstrations of the task. We test the effect of a 2× reduction in demonstrations (10 forward and 10 backward demonstrations) and a 5× reduction in demonstrations (4 forward and 4 backward demostrations). In Figure 2.5, we see that **our reward formulation is robust to these reductions in quantity of task demonstrations**.

2.3.4 Real-World Fine-Tuning with Dense Rewards

In our real-world experiments, we evaluate RoboFuME and KAGI after fine-tuning for 30K online environment training steps, with rewards computed in 640 × 480 pixel image space. The results are shown in Table 2.1. For each task, RoboFuME fine-tuning improves over the offline RL policies by 10-25%, which is consistent with the performance of RoboFuME in Yang et al. [2024b] and confirms that further online fine-tuning is beneficial. Notably, we see lower success rates on Cube Stacking, which requires high precision and is considerably harder than all the pick-place tasks in RoboFuME [139]. With KAGI, task performance is similar or further increases across the board.

While the increase in success rates achieved by KAGI over RoboFuME is modest using the standard number of in-domain demosntrations for pre-training, we notice qualitatively different behaviors learned by each policy. As an example, Figure 2.6 shows a representative trajectory demonstrating policy performance for Spatula Pick-Place. The policy fine-tuned with RoboFuME, while fairly successful, commonly demonstrated the behavior of dropping the spatula onto the plate from a height, rather than lowering and placing the spatula like the demonstrations. The robot gripper also continues moving rightward rather than staying above the placement point, indicating some coincidental successes. In comparison, KAGI's policy fine-tuned with dense shaping rewards showed behavior that matched the demonstrations more closely. For more examples comparing qualitative behavior across tasks, please see the project website.

2.3.5 Robustness to Reduced In-Domain Demonstrations

Both RoboFuME and KAGI pre-train on a set of 100 in-domain demonstrations for each new task, which is a non-trivial cost. We investigate the effect of reducing the number of in-domain

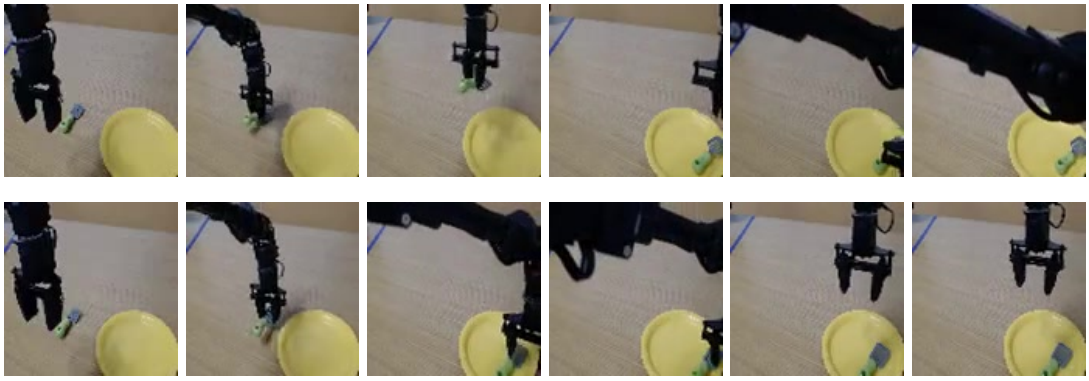


Figure 2.6: **Qualitative examples of policies fine-tuned with RoboFuME (top) and KAGI (bottom).** RoboFuME drops the spatula from a higher height and unnecessarily moves right after. KAGI places the spatula down more gently and moves to a neutral position after.

Task	Offline RL	RoboFuME	KAGI (Ours)
Cloth Covering	45%	50%	**75%**
Almond Sweeping	40%	55%	**80%**
Spatula Pick-Place	10%	30%	**60%**
Cube Stacking	5%	15%	**40%**

Table 2.2: **Success rates over 20 trials for each method with 5× fewer in-domain demonstrations.** KAGI achieves similar performance to the setting with the standard amount of demonstrations, while RoboFuME performance drops with fewer demonstrations.

demonstrations for both methods. We pre-train an offline RL policy on 5× fewer than the standard quantity of in-domain demonstrations (10 forward tasks, 10 backward tasks, and 4 failures). We then fine-tune this policy online for 45K steps with RoboFuME and with KAGI. The results are shown in Table 2.2.

Impact of reduced demos on the policy. Unsurprisingly, the offline RL policy performs worse than when trained on the standard quantity of in-domain demonstrations. Fine-tuning for 45K steps using RoboFuME somewhat improves success rate over offline RL policies, but we see a significant gap with 5× less demos compared to its original performance with the standard number of demonstrations. Therefore, the performance of RoboFuME depends heavily on the amount of in-domain demonstrations provided. However, across all tasks, KAGI reaches close to its original performance even with this reduction. This small drop indicates that, **even with fewer in-domain demonstrations, KAGI can recover comparable performance with more fine-tuning**, as it uses dense shaping rewards to guide its behavior. Since KAGI can tolerate fewer task-specific demonstrations, it can be applied to new tasks more easily, with a smaller data burden.

Task	MOKA without perturbations	MOKA with perturbations	KAGI (Ours)
Cloth Covering	90%	50%	80%
Almond Sweeping	100%	65%	80%
Spatula Pick-Place	70%	30%	65%
Cube Stacking	60%	15%	45%

Table 2.3: **Success rate of MOKA [30] without perturbations of the objects and with slight perturbations to the object positions.** With perturbations, MOKA's performance drops significantly.

Impact of reduced demos on the success classifier. The reduction in in-domain data crucially impacts the behavior of the VLM-based (MiniGPT-4) success classifier used by both RoboFuME and KAGI. We observed the sparse reward predictions of MiniGPT-4 were significantly worse, indicating fine-tuning the success predictor is reliant on substantial high-quality in-domain data to generate accurate sparse rewards. To mitigate this issue and eliminate the confounding factor of inaccurate sparse rewards, we modify RoboFuME's [139] original sparse reward computation: the reward predictor was provided four task-completion prompts as input (e.g., 'Is the spatula on the plate?', 'Has the spatula been moved to the plate?', etc.), and must reach a consensus across all prompts to generate a sparse reward of 1. This reduces the number of false positives, which RL algorithms often exploit. The decreased accuracy of the sparse reward predictor is a key factor behind the minimal improvements of RoboFuME over offline RL, revealing another source of fragility in the system and underscoring the importance of dense shaping rewards for data-efficient fine-tuning.

2.3.6 Importance of Online Fine-tuning

Since the VLM prompting component of KAGI is inspired by MOKA [30], we additionally evaluate MOKA to understand if online fine-tuning offers any advantages. MOKA [30] leverages GPT-4V to predict affordance keypoints and plan a sequence of action primitives (e.g., 'lift', 'reach grasp', 'grasp') to directly execute by computing the actions required to reach each point. We evaluate two versions of MOKA: the first with precise resets that closely match the inputs to the VLM and no environmental perturbations, and the second with slight perturbations to these initial conditions during policy rollouts, that is representative of environment resets after backward policy rollouts. The comparison seeks to highlight the benefits of closed-loop RL policies over open-loop primitives.

In Table 2.3, we see that MOKA with precise resets succeeds between 60% to 100% of the time depending on the task, which actually outperforms KAGI. However, without these precise resets, the performance of MOKA drops noticeably, as MOKA uses pre-defined motion primitives that are not robust to even slight environment perturbations, since actions are computed directly with respect to specific VLM-generated keypoints on the objects.

However, KAGI can recover most of this performance loss with online fine-tuning using VLM-generated waypoint trajectories. KAGI leverages the benefits of closed-loop systems that are more generalizable, robust to environment changes, and exhibit retry behavior, as opposed to open-loop systems that use hand-designed action primitives. This makes KAGI more suitable for the autonomous RL paradigm, which involves potentially imperfect policy rollouts on a real robot.

2.4 Discussion

In this work, we present KAGI, a method for facilitating autonomous improvements of policies pre-trained on diverse human teleoperated data using shaping rewards generated by VLMs. We leverage affordance keypoints and waypoints extracted zero-shot from VLMs as dense rewards for online RL, enabling robots to learn autonomously through interaction and self-practice. We highlight three key findings for making fine-tuning of pre-trained multi-task policies faster and more data-efficient:

1. **Dense shaping rewards extracted zero-shot from VLMs can help speed up online RL**. Providing dense shaping rewards that are easily generated from VLMs can facilitate generalization to new tasks where relying only on sparse rewards is less efficient.
2. **Dense shaping rewards improve robustness to fewer in-domain demonstrations**. Leveraging both dense and sparse rewards improves autonomous learning, with better robustness to reduced in-domain demonstrations than sparse rewards alone.
3. **Closed-loop RL systems are more robust to environmental perturbations than methods using open-loop action primitives.** While action primitives may be easier to engineer than policy training, handling dynamic changes during policy rollouts is challenging.

Overall, our reward formulation can modify existing fine-tuning methods, thereby making the process of fine-tuning policies pre-trained on diverse data more robust and data-efficient. Our framework is also agnostic to the VLM that is used for keypoint and waypoint generation, and theoretically should be substitutable with future VLMs that are developed (for a qualitative analysis of VLM quality as of late 2024, see Appendix A.1). Our work demonstrates the benefits of VLM-based dense shaping rewards, and opens up new exploration avenues for harnessing the generalization capabilities of VLMs to improve robustness of robot learning systems.

Our experiments present several insights into the opportunities and challenges of autonomous RL pipelines for scalable robot learning. Approaches using only sparse rewards for online fine-tuning are slower to learn how to complete tasks, and approaches relying on in-domain demonstrations with low multimodality are much more brittle. To address this, we leverage mark-based visual prompting to improve generalization capabilities of robots equipped with RL. In service of the broader discussion around scaling robot learning without scaling human effort, we analyze three findings from this work:

2.4.1 Brittleness of Real-World RL with Images

Reproducing RoboFuME [139] (see Appendix A.2) was initially a struggle as the system is fairly brittle. RoboFuME [139] requires 120 in-domain demonstrations per task to be collected, and trajectories must have very low multimodality. Collecting >100 demonstrations per task is not practically feasible, especially given the relative simplicity of the tasks in comparison to what we ideally want robotic systems to be capable of doing. Furthermore, requiring low multimodality across all demonstrations is challenging if using methods such as crowdsourcing [87] or data collection by users who are unfamiliar with robot teleoperation. While the objective of this work was to explore an efficient pre-train fine-tune paradigm for learning from diverse data, one could argue that a more efficient approach for the single-task setting is using the 120 demonstrations to train an imitation learning policy (e.g., diffusion policy [16]) without requiring highly unimodal trajectories, potentially reaching similar task performance as RoboFuME [139] without pre-training on Bridge data.

RoboFuME [139] also requires very similar environmental conditions (e.g. lighting, camera angle, background) between demonstration collection and during test-time. Anecdotally, even slight lighting changes (e.g., at a different time of day from demonstration collection) could be highly damaging to the policy performance, thus our environment was set up away from a window in artificially lit room. This is a consequence of using RGB images as the observation modality for policy training. Realistically, this system cannot be implemented in a highly dynamic environment (e.g., manufacturing and warehouse settings), where lighting and background is constantly changing.

Consequently, KAGI, which augments RoboFuME with dense rewards, inherited some of these limitations. Our work demonstrated that with dense rewards from VLMs, we could reduce the number of in-domain demonstrations to 24 in-domain demonstrations, which is much better in terms of scalability. It refutes the above argument of using an IL method instead, as most IL policies would not be successful with so few demonstrations and also suffer from their own limitations which we explore elsewhere in this thesis (Sections 3.1.1, 3.1.2). However, the brittleness to environmental changes between demonstration collection and test-time is still a crucial limitation. In the next work, we explore (1) whether we can further reduce the number of demonstrations provided, and (2) whether alternative visual inputs could enable better policy robustness to environmental changes.

2.4.2 Feasibility of Real-World RL

While KAGI is an autonomous RL system, it is worth considering the level of human supervision that is required. Autonomous RL indeed requires less supervision than, for instance, an IL or traditional offline RL pipeline that requires human supervision for collecting on the order of 100-1000 demonstrations (which can take hours), plus additional supervision to perform environment resets at test-time. This is because of the alternate self-practicing of the forward and backward policy, which is an improvement from traditional approaches and has the additional benefit of enabling robots to learn from autonomously collected data. However, even in our regime, the human still

needs to collect 24 demonstrations per task (which takes ∼30 minutes if done efficiently). This level of human supervision is still non-trivial, especially considering the relative simplicity of the task. In practice, the human also needs to periodically check on the autonomous RL system, which takes a wall clock time of 3-4 hours. An ideal, scalable system would require just minutes of human input to train a performant policy, and in the next work we explore a framework that trains RL policies in simulation to bring human effort down to <10 minutes for complex, dexterous manipulation tasks.

Assessing the feasibility of real-world RL for fine-tuning pre-trained policies also leads to another key insight: real world fine-tuning with RL should only be used as a *last-mile solution*. That is to say, the base pre-trained policy should be *generally very performant* for real-world fine-tuning to yield policies capable of real-world deployment in a reasonable amount of time. A significant contributor to KAGI's autonomous RL pipeline taking 3-4 hours to reach a reasonable success rate was that offline RL policies pre-trained on diverse data performed very poorly (5-40% in the reduced demonstration setting), leaving a large gap for online RL fine-tuning to fill. Realistically, for a robotics system that can be deployed at scale in real-world environments, real-world RL fine-tuning should elevate the performance of a pre-trained policy that succeeds ∼70-90% of the time to ∼100%. Further research into pre-training policies by improving the efficacy of offline RL [66, 90] or more recently, robotic foundation models [53, 54, 56, 94, 95], can address this limitation. We explore this insight further after presenting the next work, but is an important takeaway from this work worth mentioning here.

2.4.3 Scalability to More Complex Scenarios

Related to the feasibility of real-world RL, the entire autonomous RL fine-tuning pipeline was generally quite time-consuming for the four tasks in Figure 2.4, which are relatively simple tasks performed by a robot with a parallel-jaw gripper. It is challenging to envision this exact framework scaling up to more complex scenarios such as dexterous manipulation or mobile manipulation, though the key takeaways from this work still apply. Works like HuDOR [36] attempt real-world RL for dexterous manipulation using object-centric rewards via object pixel tracking. However, they use a relatively static hand and assume minimal occlusion of the object during interaction to enable pixel-based object-centric rewards, which can be fairly restrictive assumptions. Using RGB images as input also makes the system subject to the brittleness problems discussed above in Section 2.4.1, in terms of environmental changes and occlusion of objects. Finally, their system still takes ∼1 hour of online fine-tuning, which is much better than KAGI but still has room for improvement, especially considering the ideal robotics system scales to many more tasks and operates over a larger action space. In the next work, we investigate a method applied to complex dexterous manipulation and trains policies operating over a full arm-and-hand action space, by similarly leveraging object-centric rewards for RL. We also explore training RL policies in domain-randomized, parallelizable simulation environments in light of challenges in real-world RL.

Chapter 3

Sim-to-Real Reinforcement Learning from One Human Demonstration

We now turn our focus to the dexterous manipulation setting involving multi-fingered robotic hands, rather than parallel-jaw grippers. Human-like dexterous hands have the potential to significantly advance robotic manipulation [14, 36, 102]. However, the complexity of dexterous robot hands introduces substantial challenges for many existing robot learning methods. For example, imitation learning (IL) from human demonstrations has shown success using simpler end-effectors with large amounts of training data [3, 5, 19, 46, 56], but collecting high-quality demonstrations for dexterous hands is far more difficult. Capturing high-quality 3D human hand motion typically relies on wearable sensors and teleoperation systems [11, 128], which are expensive and difficult to scale.

In contrast, videos of humans interacting with objects using their own hands are inexpensive to collect, offering a scalable alternative to traditional demonstration collection. However, leveraging human videos directly for robotic IL is challenging as they lack explicit robot action labels [141]. One common approach to derive robot action labels is by obtaining per-timestep human hand pose estimates and converting them to robot actions via fingertip retargeting and inverse kinematics (IK) [11, 128]. However, this approach is often unreliable as hand pose reconstruction methods [97] are susceptible to occlusion and sensor noise. Even with perfect pose estimates, this simple retargeting strategy often results in suboptimal robot trajectories due to morphological differences between the robot and human. These challenges are particularly punishing for contact-rich, dexterous manipulation [10, 76, 93]. Existing IL methods are ill-equipped to address this issue as they rely on accurate correspondences between demonstrated and learned behaviors.

Reinforcement learning (RL) offers a promising alternative to overcome these limitations by

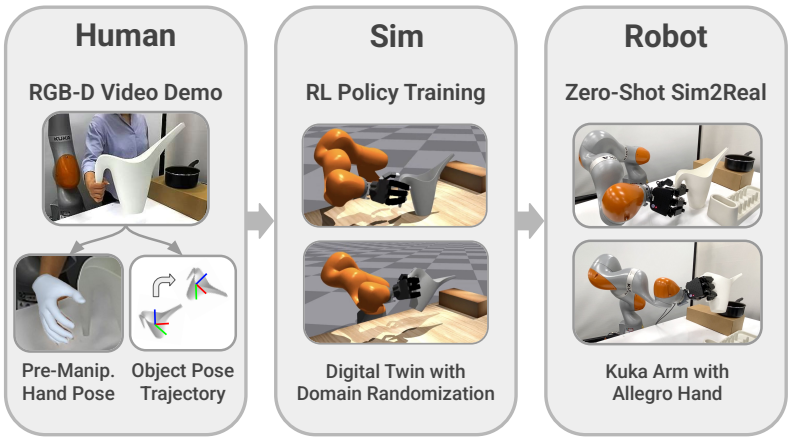


Figure 3.1: **Our Framework.** HUMAN2SIM2ROBOT learns dexterous manipulation policies from one human RGB-D video using object pose trajectories and pre-manipulation poses. These policies are trained with RL in simulation and transfer zero-shot to a real robot.

enabling robots to directly learn manipulation tasks using their own embodiment. However, RL has several limitations such as tedious, task-specific reward engineering and unfavorable sample complexity, making real-world policy training infeasible [121, 148].

In this paper, we propose HUMAN2SIM2ROBOT, a real-to-sim-to-real RL framework that addresses the limitations of existing methods and combines the best of both worlds: it only requires a single human RGB-D video demonstration and does not require any task-specific reward engineering. Crucially, we found that high-fidelity 3D human motion data is not necessary to learn robust dexterous manipulation policies. Instead, training an RL dexterous manipulation policy only requires two task-specific components that can be reliably extracted from the human video: (1) the object 6D pose trajectory, and (2) a single pre-manipulation hand pose.

We use (1) to define an embodiment-agnostic, object-centric reward that specifies the desired task, and (2) to provide advantageous initialization for RL training and facilitate more efficient exploration. Formalizing an expert demonstration with these two components facilitates RL policy training with no task-specific reward tuning. Instead of directly learning state-action mappings, we use the demonstration for *task specification* and *guidance*, encouraging human-like behavior while allowing deviations when the human strategy is unsuitable for the robot's embodiment. This enables HUMAN2SIM2ROBOT policies to achieve zero-shot sim-to-real transfer on a real-world dexterous robot, without requiring wearables, teleoperation, or large-scale data collection.

To the best of our knowledge, HUMAN2SIM2ROBOT is the first system that learns a robust real-world dexterous manipulation policy from only one human RGB-D video demonstration, bridging the human-robot embodiment gap across grasping, non-prehensile manipulation, and complex multi-step tasks. We achieve this with just *a few minutes* of human effort end-to-end, from demonstration collection to digital twin construction. Our extensive ablation studies demonstrate the importance

of our system's design decisions; while individual components have precedents, these works are often limited to simulation [102, 116], are not reactive closed-loop policies [14, 52, 140], require significantly more demonstrations [63, 102, 116, 144], or only perform prehensile manipulation [52, 102, 116, 140].

HUMAN2SIM2ROBOT policies can execute diverse real-world dexterous manipulation tasks, such as pouring from a pitcher, pivoting a box against a wall, and inserting a plate into a dishrack, without any task-specific reward tuning. In the single human demo regime, our method outperforms object-aware trajectory replay by >55% and imitation learning by >68% across all real-world tasks.

3.1 Related Work

3.1.1 Visuomotor Imitation Learning for Robotics

Visuomotor IL for robotic manipulation has shown success in learning from a large number of expert demonstrations [3, 5, 16, 46, 56, 146] collected through teleoperation or specialized wearable equipment [17, 69, 128], which makes scaling data collection efforts expensive. In contrast, human videos are inexpensive and more intuitive to collect. Per-timestep human hand pose estimates can then be converted into robot action labels through IK-based retargeting [11, 128]. However, hand pose estimation noise and the human-robot embodiment gap often result in infeasible or suboptimal IK solutions for the robot embodiment, a challenge for visuomotor IL methods that directly rely on high-quality action labels. While human demonstrations provide useful guiding strategies for task completion, certain actions may not be suitable for robots given substantial embodiment differences. HUMAN2SIM2ROBOT performs RL in simulation guided by a single human video demonstration. It encourages human-like behavior when beneficial while allowing deviations when the human strategy is unsuitable for the robot's embodiment.

3.1.2 One-Shot Imitation Learning (OSIL)

OSIL methods parallel our approach as a single demonstration is provided. Past work has performed object-aware retargeting to transfer the demonstrated trajectory to novel scenes [37, 67, 125], leveraged object segmentation and visual servoing to adapt the single demonstration to a new scene [123], or augmented teleoperated demonstrations by retargeting and success filtering in a digital twin simulation [48]. Though more data-efficient than visuomotor IL policies, OSIL methods suffer from limited generalization beyond the demonstrated actions. Simply replaying modifications of the single demonstration is unlikely to succeed in contact-rich settings requiring closed-loop, reactive behavior (e.g., variations in contact interactions or perturbations during policy rollout). Our insight is that using the human video to provide task specification and guidance for RL leverages this data source for robot learning more effectively. This allows robots to develop effective strategies with their own embodiment, rather than rigidly imitating human behaviors.

3.1.3 Reinforcement Learning for Robotics

RL enables robots to learn complex behaviors through interaction with the environment. Real-world RL is often impractical due to slow training, safety concerns, manual environment resets, and difficult reward tuning [121, 148]. Sim-to-real RL circumvents these challenges and has led to breakthroughs in other robotic domains [15, 51, 83, 85, 101]. However, it remains underexplored for full arm-and-hand dexterous manipulation, as most prior work relies on simulation with non-physical, floating-hand models [13, 103, 127, 136]. Among works that have demonstrated sim-to-real transfer, Torne et al. [2024] use demo-augmented RL, which requires many demonstrations collected with the same robot embodiment. Chen et al. [2024c] learn residual actions on top of an open-loop base trajectory learned from human data, but the resulting policy lacks the flexibility for error recovery and struggles under a large embodiment gap. In contrast, HUMAN2SIM2ROBOT trains robust dexterous RL policies in simulation from minimal human input over a full arm-and-hand action space, which successfully transfer to the real world. Other works use inverse RL on human videos [63, 144], but inferring a reward function typically requires ∼100 demos. In contrast, our explicit object-centric reward works with a single demo.

Prior works corroborate the observation that pre-grasp poses can accelerate policy learning and result in human-like grasps [18, 22, 74]. However, these methods only focus on using pre-grasps from very similar embodiments for grasping tasks in simulation. We focus on training RL policies that transfer to the real world, overcome the human-robot embodiment gap, and perform prehensile and non-prehensile manipulation. These policies can be deployed zero-shot in real-world environments, as we build on recent work in sim-to-real transfer [72, 121].

3.2 Methodology

We present HUMAN2SIM2ROBOT, a real-to-sim-to-real RL framework for learning robust, dexterous manipulation policies from a single human-hand RGB-D video demonstration. Figure 3.1 shows an overview of our framework, and the following sections detail the key design decisions of our framework.

3.2.1 Real-to-Sim & Human Demo

We first create a digital twin of the robot's real-world environment and the target object to act as a policy training ground in simulation. The

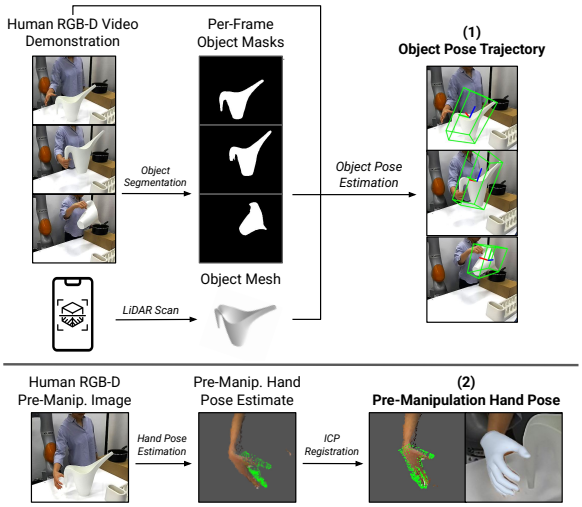


Figure 3.2: **Human Demo Processing.** (1) The object pose trajectory defines an object-centric, embodiment-agnostic reward. (2) The pre-manipulation hand pose provides advantageous initialization for RL training.

construction of the digital twin only takes ∼10 minutes of human effort (see Appendix B.1 for a detailed time breakdown). We use off-the-shelf mobile applications [58, 64] to capture a high-fidelity object mesh $\mathcal{O}$ and scene mesh $\mathcal{S}$.

Next, we record a single monocular RGB-D video demonstration $\{\boldsymbol{I}_t\}_{t=1}^{T}$ using a camera with known intrinsics and extrinsics, where each frame $\boldsymbol{I}_t \in \mathbb{R}^{H\times W\times 4}$ contains RGB-D data, and T is the total number of timesteps. Figure 3.2 visualizes how we process the human demonstration to obtain (1) an object pose target trajectory $\{\boldsymbol{T}_t^{\text{target}}\}_{t=1}^{T}$, and (2) a human hand pose trajectory represented as MANO [107] parameters $\{(\boldsymbol{\theta}_t, \boldsymbol{\beta}_t)\}_{t=1}^{T}$. At each timestep t, $\boldsymbol{T}_t^{\text{target}} \in \text{SE}(3)$ is the object pose from the demonstration's object trajectory, $\boldsymbol{\theta_t} \in \mathbb{R}^{48}$ is the MANO hand pose parameter, and $\boldsymbol{\beta_t} \in \mathbb{R}^{10}$ is the MANO hand shape parameter. In our system, we extract the object pose trajectory using FoundationPose [132], an open-source object pose detection model, which requires the scanned object mesh $\mathcal{O}$ and per-timestep object masks generated using Segment Anything Model 2 (SAM 2) [105], an open-source segmentation model. We extract per-timestep human hand poses using HaMeR [97], an open-source hand pose detection model, which takes RGB images as input. Using our depth images, we perform ICP registration to align the hand point clouds for obtaining accurate hand poses (see Appendix B.1 for details on aligning HaMeR predictions with depth images).

We then determine the pre-manipulation hand pose at timestep $\tau = t_0 - t_{\text{offset}}$, where t_0 is the first timestep in which the object's velocity exceeds a threshold $v_{\text{min}} = 5cm/s$, and $t_{\text{offset}} = 10$ represents a fixed number of timesteps prior to the object's motion. We use $\boldsymbol{\theta}_\tau$ and $\boldsymbol{\beta}_\tau$ to compute fingertip positions and middle finger base knuckle pose as the human pre-manipulation hand pose.

Lastly, we retarget the human pre-manipulation hand pose to a robot hand pose through a two-step IK procedure using cuRobo [118]. Figure 3.3 visualizes how we perform human-to-robot hand retargeting. In Step 1, the robot arm's joint angles are adjusted to align the base position and orientation of the robot hand's middle knuckle with that of the human hand (with a small offset, see Appendix B.2 for details). In Step 2, the robot hand's joint angles are adjusted to align the robot's fingertip positions with the corresponding human fingertip positions. This generates a pre-manipulation robot hand pose $(\boldsymbol{T}^{\text{wrist}}, \boldsymbol{q}^{\text{hand}})$ for the object pose represented as $\boldsymbol{T}_\tau^{\text{target}}$, where $\boldsymbol{T}^{\text{wrist}} \in \text{SE}(3)$ is the robot wrist pose, and $\boldsymbol{q}^{\text{hand}} \in \mathbb{R}^{N^{\text{hand-joints}}}$ is the robot hand joint configuration. This faithfully retargets the human pre-manipulation hand pose, while maintaining kinematic feasibility and alignment between the human and robot hand.

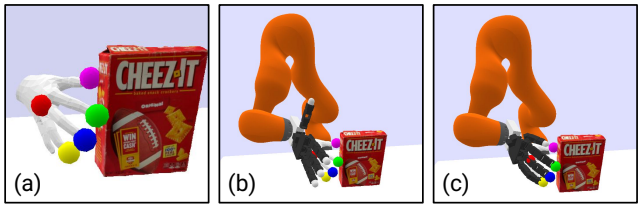


Figure 3.3: **Human to Robot Hand Retargeting.** (a) Estimated MANO hand pose. Middle knuckle: red. Fingertips: pink, green, blue, yellow. (b) IK Step 1 (Arm): Align middle knuckle. (c) IK Step 2 (Hand): Align fingertips.

The object pose trajectory provides *task specification* by defining an object-centric trajectory-tracking reward for policy training. The pre-manipulation pose offers *task guidance* by defining a good state initialization for exploration [22]. The pre-manipulation pose retargeting does not need to be extremely precise, as it is just a prior to facilitate exploration. Together, these abstractions guide RL policy training in simulation via reward guidance and advantageous state initializations.

3.2.2 Simulation-based Policy Learning

We create a training environment in the IsaacGym simulator [82] that matches the real-world environment, consisting of the robot, scene mesh $\mathcal{S}$, and object mesh $\mathcal{O}$, which takes only ∼10 minutes of human effort. We then train a policy using Proximal Policy Optimization [108] which outputs robot actions that move the object along the target trajectory, guided by the provided pre-manipulation pose. We emphasize that we primarily care about *how the object moves*, rather than imitating the actions of the human demonstrator; the pre-manipulation pose serves as a rough prior, but the policy will learn to use the robot embodiment to achieve the desired object motion.

The reward given at timestep t is an object-tracking reward, $r_t = r_t^{\text{obj}}$ defined as

$$r_t^{\text{obj}} = \exp\big(-\alpha\, d(\boldsymbol{T}_{\tau+t}^{\text{target}}, \boldsymbol{T}_t^{\text{obj}})\big), \quad \text{where} \quad d(\boldsymbol{T}_1, \boldsymbol{T}_2) = \sum_{i=1}^{N^{\text{anchor}}} \big\| \boldsymbol{T}_1 \boldsymbol{k}_i - \boldsymbol{T}_2 \boldsymbol{k}_i \big\|, \tag{3.1}$$

where d is the relative pose distance function and $\alpha = 10$. For most objects, we select $N^{\text{anchor}} = 3$, with $\boldsymbol{k}_1 = [L, 0, 0]$, $\boldsymbol{k}_2 = [0, L, 0]$, and $\boldsymbol{k}_3 = [0, 0, L]$ in the local object frame, where $L = 0.2m$ is a distance parameter for orientation. Figure 3.4 visualizes the object pose tracking reward. This formulation integrates position and orientation naturally: larger L emphasizes orientation by placing anchor points farther from the object's origin. See Appendix B.3 for reward function details.

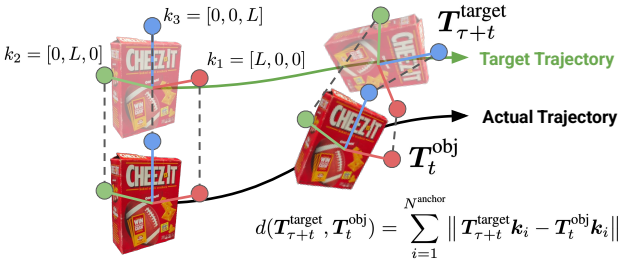


Figure 3.4: **Object Pose Tracking Reward.** The agent is rewarded for minimizing distance between current pose and target object pose $d(\boldsymbol{T}_{\tau+t}^{\text{target}}, \boldsymbol{T}_t^{\text{obj}})$ using anchor points $\boldsymbol{k}_i$.

The observation at timestep t is $\boldsymbol{o}_t = [\boldsymbol{q}_t, \dot{\boldsymbol{q}}_t, \boldsymbol{X}_t^{\text{fingertips}}, \boldsymbol{x}_t^{\text{palm}}, \boldsymbol{X}_t^{\text{obj}}, \boldsymbol{X}_{\tau+t}^{\text{target}}]$, where $\boldsymbol{q}_t, \dot{\boldsymbol{q}}_t \in \mathbb{R}^{N^{\text{joints}}}$ are the robot's joint angles and velocities, $\boldsymbol{X}_t^{\text{fingertips}} \in \mathbb{R}^{N^{\text{fingers}} \times 3}$ are the fingertip positions, $\boldsymbol{x}_t^{\text{palm}} \in \mathbb{R}^3$ is the palm position, and $\boldsymbol{X}_t^{\text{obj}}, \boldsymbol{X}_{\tau+t}^{\text{target}} \in \mathbb{R}^{N^{\text{anchor}} \times 3}$ are anchor point positions for the object and target poses. $N^{\text{joints}}, N^{\text{fingers}} \in \mathbb{N}$ are the number of robot joints and fingers, respectively.

The action at timestep t is $\boldsymbol{a}_t = [\boldsymbol{x}_t^{\text{palm-target}}, \boldsymbol{r}_t^{\text{palm-target}}, \boldsymbol{x}_t^{\text{pca-target}}]$, where $\boldsymbol{x}_t^{\text{palm-target}} \in \mathbb{R}^3$ is the target palm center position, $\boldsymbol{r}_t^{\text{palm-target}} \in \mathbb{R}^3$ is the target palm orientation expressed as Euler

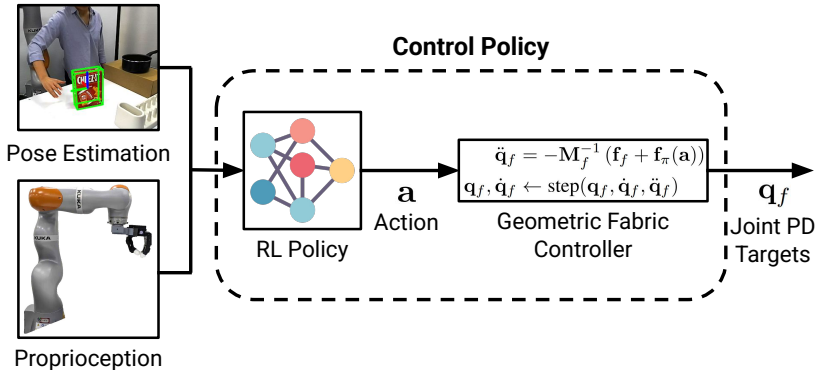


Figure 3.5: **Inference-Time Diagram.** Our RL policy takes current object 6D pose and robot proprioception as input. It outputs an action consisting of a palm position and orientation target that is sent to a geometric fabric controller [72], which generates joint PD targets.

angles, and $\boldsymbol{x}_t^{\text{pca-target}} \in \mathbb{R}^{N_{\text{pca}}}$ is the vector of target PCA values used to control the hand joints, where $N_{\text{pca}} = 5$ is the number of principal components. We use a geometric fabric controller and PCA-based hand action space, enabling human-like hand motions (see [72] for details).

We use an initial state distribution, guided by the human pre-manipulation hand pose, to simplify exploration and bias the policy toward human-like behavior. We construct this distribution by sampling object pose around the trajectory's initial pose, then set the robot configuration to match the pre-manipulation hand pose with slight perturbation (see Appendix B.4). Given this perturbed pose, we compute a feasible arm joint configuration with IK [1]. The environment is reset if the object is too far from the current target $d(\boldsymbol{T}_{\tau+t}^{\text{target}}, \boldsymbol{T}_t^{\text{obj}}) > D_{\max}$, the robot palm is too far from the object $||\boldsymbol{x}_t^{\text{palm}} - \boldsymbol{x}_t^{\text{obj}}|| > D_{\max}$, or the target trajectory is complete $\tau + t > T$, where $D_{\max} = 0.25m$.

Real-world dynamics parameters (e.g., object mass, inertia, friction) are often unknown. To handle this uncertainty, we use domain randomization during simulation, enabling the policy to adapt to diverse dynamics and transfer to the real world. We train an LSTM-based policy that leverages a history of observations to handle this partial observability and noisy data. We train the policy using object poses instead of images to accelerate training and enhance robustness to visual variation. To improve resilience to pose errors and calibration noise, we add noise to pose observations. We apply random object forces to increase robustness to unexpected contacts, disturbances, and dynamics variation, enabling zero-shot sim-to-real transfer. See Appendix B.5 for training details.

3.2.3 Sim-to-Real Policy Transfer

After training, we deploy the policy on a real robot without additional fine-tuning (i.e., zero-shot). Figure 3.5 illustrates the control pipeline at deployment, highlighting the inputs and outputs of our

[1] We use cuRobo [118] to perform parallelized, collision-free IK to ensure RL training is not bottlenecked by IK computations

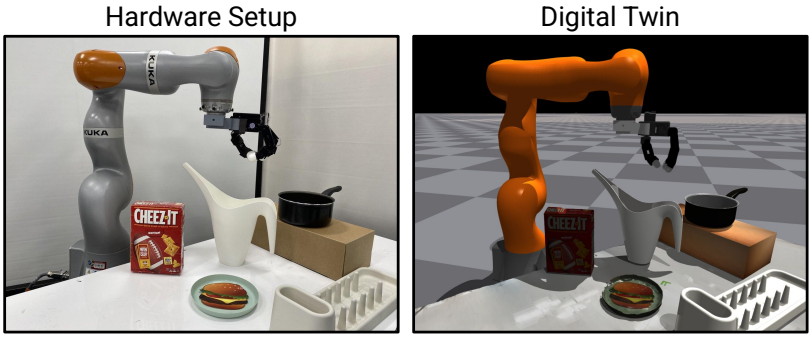


Figure 3.6: **Hardware Setup & Digital Twin.** The hardware setup includes an Allegro hand mounted on a KUKA LBR iiwa 14 arm with a ZED 1 stereo camera mounted on the table (images from the camera's perspective). Experiments are conducted in a tabletop setting with a static box, saucepan, and dishrack, involving manipulation tasks with three objects: `snackbox`, `pitcher`, and `plate`. The right image illustrates our digital twin.

policy at test time. We track object 6D poses at 30Hz using FoundationPose [132] (for details, see Appendix B.6). The RL policy processes real-time observations and outputs actions at 15Hz, which are then passed to a geometric fabric controller [72] running at 60Hz. Finally, this controller produces robot joint PD targets, which are executed by a low-level PD controller at 200Hz.

3.3 Experimental Results & Analysis

3.3.1 Key Research Questions

Our experiments aim to answer the following questions:

1. **Importance of Embodiment-Specific RL**: Do RL policies trained via HUMAN2SIM2ROBOT outperform baselines on dexterous manipulation tasks? (Sec 3.3.3)
2. **Importance of Object Pose Trajectory**: How effective is using the object pose trajectory from a human demonstration as a dense reward for RL policy training, compared to other reward formulations? (Sec 3.3.4)
3. **Importance of Pre-Manipulation Pose Initialization**: Does a pre-manipulation hand pose from a human demonstration provide more effective initialization for learning manipulation skills than generic initializations? (Sec 3.3.5)
4. **Sufficiency of Pre-Manipulation Hand Pose**: How effective is a single pre-manipulation pose in guiding RL policy training, compared to alternatives that require full human hand trajectories? (Sec 3.3.6)

We evaluate HUMAN2SIM2ROBOT in simulation and on a real robot across a diverse set of tasks and objects to answer these questions.

3.3.2 Experimental Setup

Hardware Setup. Our robot consists of a 16-DoF dexterous Allegro hand mounted on a 7-DoF KUKA LBR iiwa 14 arm. We used a ZED 1 stereo camera mounted on the table for recording both the human demonstration and real-time object pose estimation for policy input at test time. Our experiments are conducted in a tabletop setting with three static objects: a box, a large saucepan placed atop the box, and a dishrack. The tabletop and its static objects are captured in scene scan $\mathcal{S}$. Figure 3.6 shows our hardware setup and objects, as well as the digital twin.

Tasks & Objects. We perform experiments with three objects: `snackbox`, `pitcher`, and `plate`. Figure 3.7 visualizes our tasks using these objects, which include grasping, non-prehensile manipulation, and extrinsic manipulation. We also explore multi-step tasks that compose sequences of these skills, such as pivoting the plate, lifting it, and placing it in a dishrack (see Appendix B.7 for details).

Simulation Ablation Setup. For our ablation experiments, we evaluate all methods in simulation on the `plate-pivot-lift-rack` task, which is our most complex multi-step task. We train policies with three random seeds for all simulation results, and we compare them on their speed and stability of learning, their final achieved reward, and their qualitative behavior.

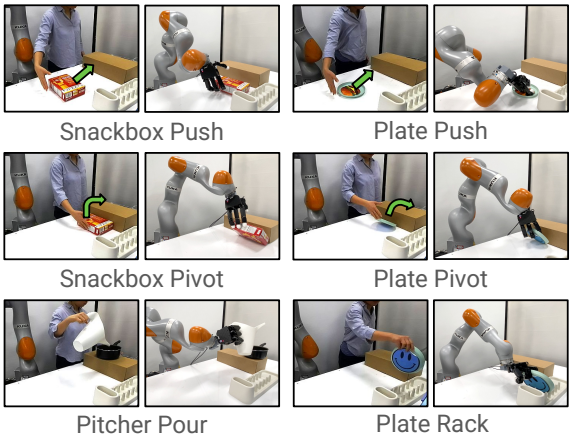


Figure 3.7: **Task Visualization.** Our real-world tasks span grasping, non-prehensile manipulation, and extrinsic manipulation. We show the human demo and the robot behavior side-by-side for each task. We also include three multi-step tasks that compose multiple skill types listed above.

3.3.3 Importance of Embodiment-Specific RL

In real-world experiments, we compare HUMAN2SIM2ROBOT policies to non-RL baselines to evaluate the impact of closed-loop, embodiment-specific RL. These baselines require robot action labels for the entire task, which are obtained by performing hand pose estimation and human-to-robot retargeting for *every* frame of the video.

We evaluate against three baselines across seven real-world tasks (see Appendix B.8 for baseline details): (1) **Replay**: Replays the retargeted trajectory open-loop by setting PD targets to these positions; (2) **Object-Aware (OA) Replay**: Like **Replay**, but warps the trajectory by the relative transform between initial object pose in the human demo and at test time (similar to [125, 67]); and (3) **Behavior Cloning (BC)**: Trains a closed-loop diffusion policy [16] on 30 demos (same number as in [36]), generated from our one demo by sampling object poses (same range as our RL training), performing OA Replay, and using these trajectories as demo data.

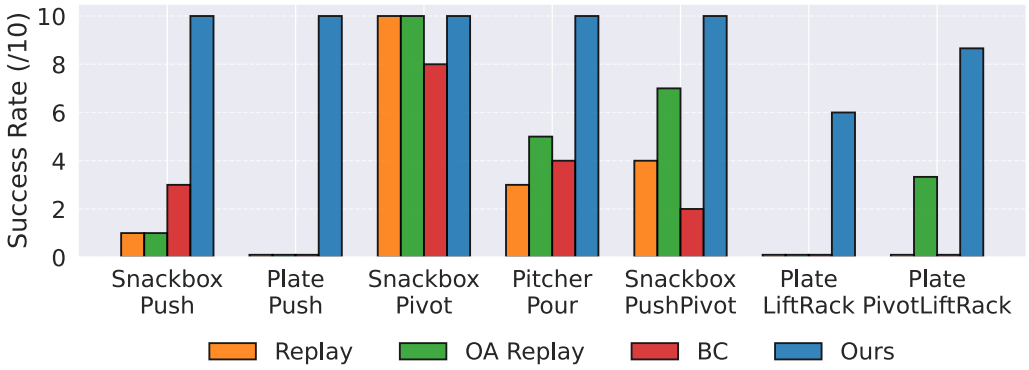


Figure 3.8: **Real-World Success Rates.** HUMAN2SIM2ROBOT policies outperform Replay by 67%, Object-Aware (OA) Replay by 55%, and Behavior Cloning (BC) by 68% across all tasks.

For each task, we evaluate the success rate of the task-specific RL policy and the baselines across 10 policy rollouts (Figure 3.8). In all tasks, HUMAN2SIM2ROBOT policies substantially outperform the baselines. **Replay** was unsuccessful on most tasks, but performed well on tasks requiring low precision like `snackbox-pivot`. **OA Replay** performed better than **Replay** as it accounts for randomizations in initial pose, but still had many failures due to (a) hand pose estimation errors, (b) morphological differences between the robot and human, and (c) non-reactive open-loop control. **BC** performed similarly to **Replay**, which can be attributed to the low-quality dataset (actions computed from noisy hand pose estimations) and compounding errors throughout policy rollouts. While retargeted robot demos can succeed on simpler tasks, they often fail on harder multi-step tasks. **Ours** does not simply imitate human behaviors, but adapts the behavior for the robot embodiment, resulting in much higher success rates across all tasks.

Qualitatively, for more intricate tasks like `plate-pivot-lift-rack`, small differences in the pre-manipulation hand pose can result in very different learned strategies due to the differences in human and robot morphologies. While the human hand used the pinky and ring fingers to lift the plate before transitioning to a grasp, the Allegro hand, which is much larger, used its ring finger to pivot the plate and clipped it between the middle and index finger once the plate was off the table (see Figure 3.9). This further underscores our hypothesis that significant morphological differences may lead to strategies that are guided by the human motion but ultimately converge on a different strategy that is more suitable for the robot's embodiment after learning through trial-and-error.

Regarding HUMAN2SIM2ROBOT failure modes, failures typically arose from converging on policies that exploited simulation inaccuracies or from significant pose estimation error from occlusion. Examples of simulation inaccuracies include imperfect friction modeling of the tabletop and static objects, such that policies converged on behavior that leveraged these inaccurate parameters. The most challenging task was `plate-pivot-lift-rack` as it required high-precision object handling: the plate is very thin, and is hard to manipulate and slips out of the large Allegro hand easily.

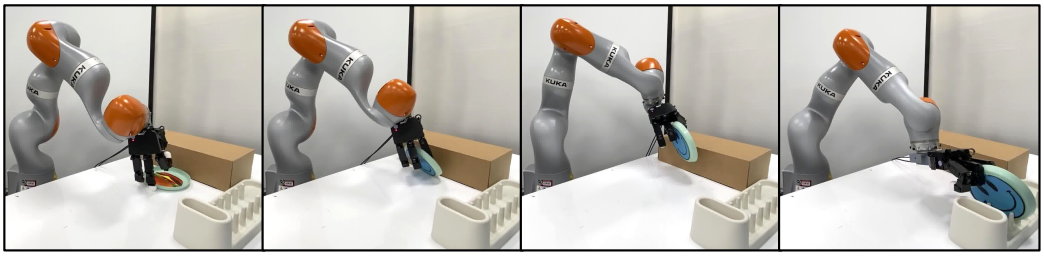


Figure 3.9: **Plate Pivot Lift Rack.** Robot converges on a strategy guided by the human demonstration, but adapted to its morphological differences. See our website for more qualitative videos.

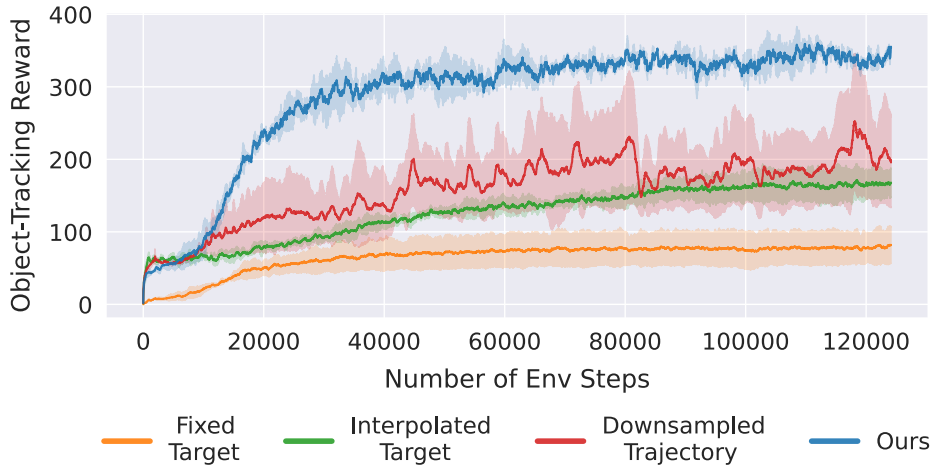


Figure 3.10: **Object Pose Tracking Reward Ablation.** Reward curves comparing different object rewards. **Ours** achieves the highest reward with fewer environment steps compared to ablations that use much sparser reward formulations.

3.3.4 Importance of the Object Pose Trajectory

We run ablation experiments in simulation to study the importance of the object pose tracking reward, comparing against: (1) **Fixed Target**: In r_t^{obj} (Eq. 3.1), we replace the current target object pose $\boldsymbol{T}_{\tau+t}^{\text{target}}$ with a final one $\boldsymbol{T}_{T}^{\text{target}}$; (2) **Interpolated Target**: In r_t^{obj} (Eq. 3.1), we replace the current target pose $\boldsymbol{T}_{\tau+t}^{\text{target}}$ with an interpolated pose between the initial and final target pose: $\textsc{interp}\left(\boldsymbol{T}_{\tau}^{\text{target}}, \boldsymbol{T}_{T}^{\text{target}}, t/(T-\tau)\right)$, where $\textsc{interp} : \text{SE}(3) \times \text{SE}(3) \times [0,1] \to \text{SE}(3)$ linearly interpolates position and uses slerp for orientation; (3) **Downsampled Trajectory**: In r_t^{obj} (Eq. 3.1), we replace the current target object pose $\boldsymbol{T}_{\tau+t}^{\text{target}}$ with the downsampled pose $\boldsymbol{T}_{\tau+t_{\text{down}}}^{\text{target}}$, where $t_{\text{down}} = \lfloor t/D \rfloor \cdot D$ and D is the downsampling factor. This reduces the temporal resolution of the human demonstration trajectory into a series of key poses.

Figure 3.10 shows that HUMAN2SIM2ROBOT achieves a substantially higher average reward than the other methods. The plate lying flat on the table is too large to be directly grasped, therefore the optimal strategy is to use extrinsic manipulation leveraging the wall to pivot the plate into

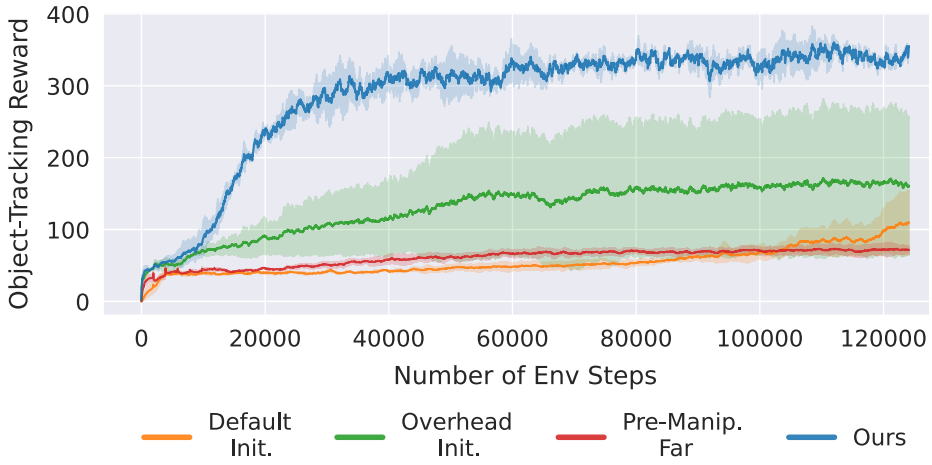


Figure 3.11: **Pre-Manipulation Hand Pose Ablations.** Reward curves comparing different initialization strategies. **Ours** leverages a good pregrasp close to the object as a prior that improves exploration efficiency compared to ablations that do not use a pregrasp or are initialized far away from the object.

a graspable position, as demonstrated in the human video. **Fixed Target** and **Interpolated Target** encourage the policy to greedily move the plate directly to the target in the air, but they struggle to pick up the plate and get stuck in a local minimum. The policy does not explore extrinsic manipulation because this requires navigating to low reward regions for a long period. **Downsampled Trajectory** is able to learn the task, but it takes longer to learn due to the weaker learning signal. It produces jerky motions in hopes of maximizing reward by tracking the waypoints that jump suddenly. **Ours** uses the full dense object pose trajectory, and it achieves the strongest performance and converges the fastest. This underscores the effectiveness of our dense, object-centric, and embodiment-agnostic reward function.

3.3.5 Importance of Pre-Manipulation Pose Initialization

We run ablation experiments in simulation to study the importance of pre-manipulation pose initialization, comparing against: (1) **Default Initialization**: We initialize the robot configuration at a default rest pose that is not close to the object; (2) **Overhead Initialization**: We compute a joint configuration with IK setting the robot palm 5cm above the object. We set the hand to a default open hand pose; (3) **Pre-Manipulation Far**: We initialize the robot with the pre-manipulation hand pose, but adjust the arm joint angles to move the robot palm 20cm away from the object.

Figure 3.11 shows that HUMAN2SIM2ROBOT achieves a substantially higher average reward than the other methods. **Default Initialization** and **Pre-Manipulation Far** perform the worst due to exploration challenges from the hand starting too far from the object. **Overhead Initialization** performs slightly better because it starts closer to the object, but fails to converge on a successful

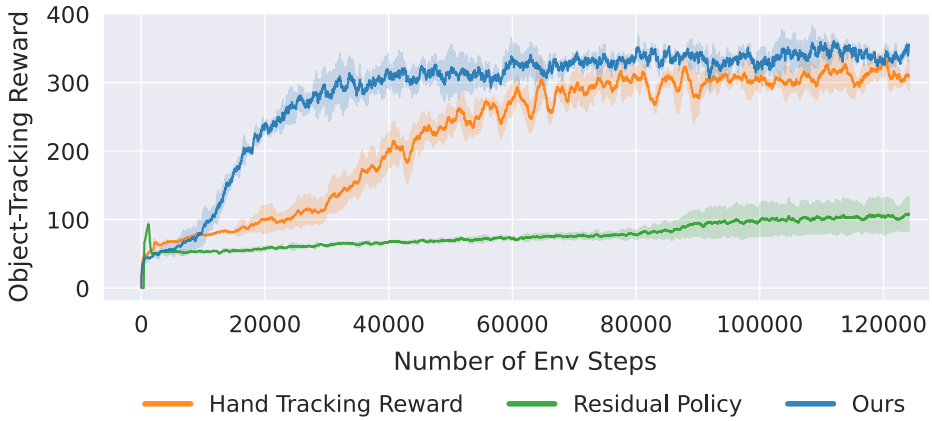


Figure 3.12: **Full Hand Trajectory Ablation.** Reward curves comparing our method with methods that require the full human hand trajectory. **Ours** achieves higher rewards in a shorter amount of time than ablations that use the full hand trajectory as hand-tracking rewards (achieves comparable reward eventually but is slower to converge) and to train a residual policy (much lower rewards).

policy because the overhead grasp provides a disadvantageous prior, as it is not the optimal approach for performing the task. **Ours** achieves the highest reward by providing an advantageous initialization, which minimizes exploration challenges.

Overall, we show that for efficient exploration, the policy should be initialized in the rough region of the pre-manipulation pose. To eliminate the need to initialize the robot hand in close proximity to the object, methods such as collision-free motion planning or an initial approach stage reward formulation (like having a reward for minimizing the L2 distance to the pre-manipulation pose) have potential to overcome this challenge. These techniques can be seamlessly integrated into our system to facilitate navigation from a default rest pose to the pre-manipulation pose. Our experiments presented here highlight the significant effectiveness of the pre-manipulation pose in guiding the RL policy toward learning human-like behaviors for contact-rich manipulation tasks.

3.3.6 Sufficiency of Single Pre-Manipulation Pose

We run ablation experiments in simulation to compare pre-manipulation pose initialization to methods that require the full human hand trajectory: (1) **Hand Tracking Reward**: We add $r_t^{\text{hand}} = \exp\left(-\alpha \| \boldsymbol{X}^{\text{fingertips}} - \boldsymbol{X}^{\text{desired-fingertips}} \|\right)$ to encourage tracking the human hand trajectory. The total reward is $r_t = r_t^{\text{obj}} + r_t^{\text{hand}}$; (2) **Residual Policy**: Using the same object-centric reward, we replay the retargeted robot trajectory open-loop while learning delta PD joint targets [14].

Figure 3.12 shows the reward curves of these methods. **Hand Tracking Reward** is able to learn an effective policy, but it learns more slowly because it initially focuses on improving its hand-tracking reward, which is not always conducive to policy performance. When trained to convergence, **Hand Tracking Reward** does not show any substantial improvement over our method, despite

requiring additional data and supervision. **Residual Policy**'s performance is substantially worse because inaccurate hand pose estimation results in a poor base motion that is difficult to learn an effective residual policy for. In contrast, **Ours** is able to effectively learn the task without requiring the full human hand trajectory as additional supervision.

3.4 Discussion

In this work, we present HUMAN2SIM2ROBOT, a real-to-sim-to-real RL framework for learning robust dexterous manipulation policies from a single human hand RGB-D video demonstration. Our method facilitates training RL policies in simulation for dexterous manipulation by formalizing tasks through object-centric rewards and a pre-manipulation hand pose. We present the following system design decisions in HUMAN2SIM2ROBOT for addressing several key challenges in training general dexterous manipulation robot policies:

1. **Eliminating reward engineering effort**: We automatically extract rewards from the human demonstration via an object pose trajectory, creating a dense, embodiment-agnostic, object-centric reward for effectively training RL policies without any task-specific reward tuning.

2. **Facilitating efficient exploration**: We use a pregrasp pose derived directly from the human demonstration as a prior to initialize RL training, biasing the policy towards human-like grasps.

3. **Bridging the human-robot embodiment gap**: By allowing the agent to learn and explore through interactions in its own embodiment, the agent was able to learn manipulation strategies optimal for its own embodiment rather than rigidly imitating human behaviors, thereby successfully bridging the human-robot embodiment gap.

4. **Robust sim-to-real transfer**: By leveraging techniques such as domain randomization of system parameters, random perturbations, and pose noise, we facilitated robust sim-to-real transfer of policies trained in simulation to a real-world robot arm with a dexterous hand.

Our policies show significant improvements over existing methods across grasping, non-prehensile manipulation, and extrinsic manipulation tasks, representing a step forward in facilitating scalable and robust training of real-world dexterous manipulation policies with <10 minutes of human effort.

Our experiments present new insights into the potential avenues for leveraging RL in simulation to cross the embodiment gap in a low demonstration regime. We explore ways of formalizing task specification and guidance using a single human demonstration, and leveraging this to efficiently train RL policies in simulation, which we use sim-to-real policy transfer to deploy in the real world. This prompts several avenues for future research as well as broader implications for exploring how to most efficiently leverage a small amount of human data for generalization robot learning at scale:

3.4.1 Scaling to New Embodiments

Scaling to Other Arms and End-Effectors. We evaluate HUMAN2SIM2ROBOT's ability to bridge the human-robot embodiment gap using a Kuka arm and Allegro hand. While we have not conducted extensive quantitative evaluations on other embodiments, we expect our framework to be similarly effective for other embodiments, as our framework was deliberately designed to be *embodiment-agnostic*. Specifically, our object-centric reward function and training pipeline do not rely on embodiment-specific assumptions. As a preliminary test, we present initial experiments on the LEAP Hand [112] and UMI gripper [17], which achieved strong performance with minimal modifications (see Appendix B.9 for details). We believe these early results are promising, and future work can further validate our approach on a broader range of robot hands and arms.

Scaling to More Complex Embodiments. Beyond different robots and arms, an exciting area of future work is to explore the applicability of this framework to hardware with more complex morphologies and kinematics, such as bimanual or whole-body manipulation and mobile manipulation. Some refinements to the framework for such settings might include:

- Exploring a broader range of more complex tasks. For example, handover tasks from one hand to the other, reorientation of large objects that require two hands, and coordination tasks (e.g., pouring from a pitcher in one hand to a cup in another hand).
- Robust collision avoidance for bimanual robot arms or whole-body manipulators.
- Determining pregrasp(s) for multiple end-effectors. For example, determining the necessary pregrap(s) for an end-effector waiting to receive an object in a handover task that is not actively manipulation an object at the start. In such cases, perhaps a single pregrasp does not provide sufficient priors for multiple manipulators that are coordinating object interactions.
- Modifying our object-centric reward function to encode temporal consistencies between trajectories of two objects. For example, a right arm should only begin to pour from the pitcher when left arm has correctly positioned the cup.
- Exploring dynamic cameras, egocentric camera viewpoints, or multi-camera perception to minimize occlusions of manipulators by objects from a fixed camera viewpoint.

The above are considerations for more complex systems that our framework currently does not explicitly address a single-arm, single-object setting. In these settings, the range and complexity of tasks and action space increases dramatically, and seeing what aspects of the framework transfer and what aspects need to be modified is an interesting area of exploration.

3.4.2 Feasibility of Real-to-Sim-to-Real at Scale

Training RL policies in simulation was crucial to our pipeline, and enabling this required a robust real-to-sim and sim-to-real pipeline. Recent advancements in scanning applications for real-to-sim transfer [58, 64] made this process very efficient, requiring <10 minutes of human effort per task. Obtaining high-quality scene and object scans was very achievable in a lab setting; however, it is unclear how feasible real-to-sim will be for complex industrial or home settings. There is good reason to believe that obtaining high-fidelity scene scans of complex environments will become easier over time: 3D scanning applications on mobile devices equipped with LiDAR makes this procedure accessible, and with further engineering developments, there is good reason to be optimistic that fidelity of these scans will only continue improving.

A key challenge involves accurately modeling articulated objects with joints at scale: current approaches such as Torne et al. [2024] add prismatic and revolute joints manually to a scene scan using a GUI, but this is hard to do at scale for every object in the home, office, or warehouse. Another challenge is modeling the interaction dynamics or collision models of very irregular objects: for example, the dish rack we used in `plate-pivot-lift-rack` had spokes which were complicated to model even with a good quality object mesh, thus the plate was hard to insert fully in simulation even though it was achievable in the real environment. The primary challenge for future research, therefore, is less so the photorealism of 3D scans, but instead modeling the physical interaction dynamics of objects in the scene, which becomes exponentially more complex as the environment scale increases beyond a tabletop and features a greater diversity of objects.

For sim-to-real transfer, proposing novel approaches to minimize the sim-to-real gap was orthogonal to our focus on learning-based methods for dexterous manipulation. However, it was critical for the success of our system in real-world deployment. A large body of work has investigated techniques such as domain randomization [12], simulation parameter inference and system identification methods for accurate physics simulation [120], adding observation noise to simulate sensor noise [122], adding random perturbation forces [6], and augmenting datasets to simulate partial observability (e.g., by intentionally adding occlusions). Continued research into refining these techniques will help improve the photorealism and physical realism of simulators and further narrow the sim-to-real gap.

That said, a key observation of our work is the importance of a good observation bridge between simulated and real environments that are visually dissimilar. We used 6D object pose that works well despite visually dissimilar environments; other works use natural language as a bridge [142], and another option is object segmentation masks, which are can be accurately obtained with modern segmentation and tracking models. We found that SAM 2 [105] was generally more robust than FoundationPose [132] for reliably tracking objects over long periods despite occlusions; while our system used SAM 2 masks to correct FoundationPose predictions at test time, future work can explore using 2D or 3D object segmentations or point trajectory methods as an observation bridge.

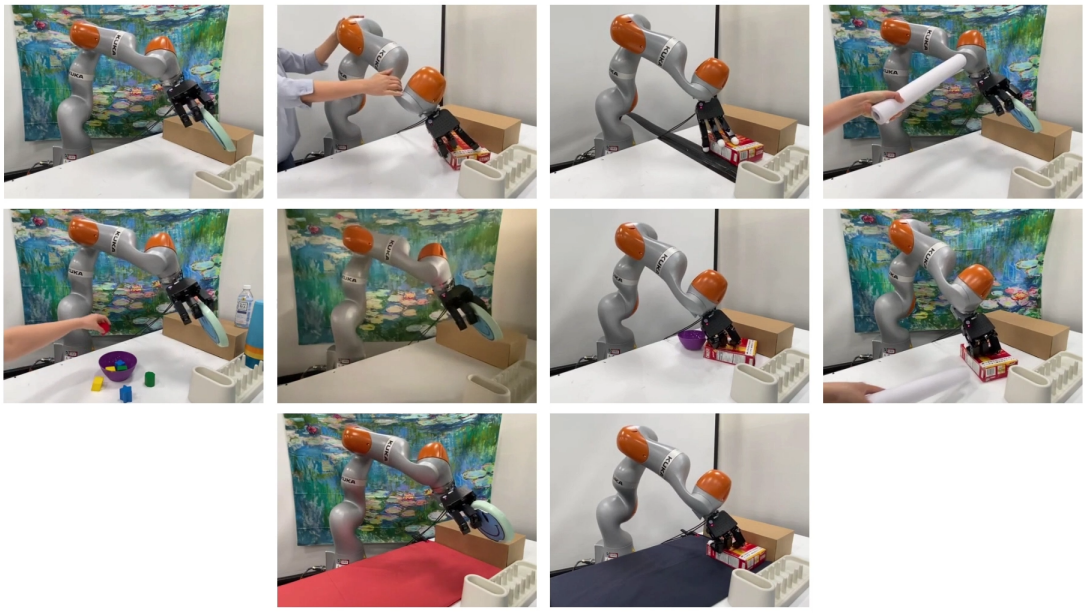


Figure 3.13: **Robustness Analysis:** Demonstrating HUMAN2SIM2ROBOT's robustness to visual distractors, background and lighting changes, and robot and object perturbations during policy rollouts, a benefit of training with low-dim pose observations and random object forces in simulation.

3.4.3 6D Pose vs. Point-based Observation Modality.

HUMAN2SIM2ROBOT demonstrates several advantages of using 6D object pose as an observation modality for policy training: strong robustness to visual distractors, background changes, and lighting changes during policy rolloust (see Figure 3.13), faster and simplified policy training due to low-dimensional observation inputs, a dense object-centric reward function, and a good bridge for sim-to-real policy transfer that sidesteps the photorealism gap, a challenge that affects training image-based policies in simulation with rendered images.

However, there are limitations to using object pose for low-dimensional observations. Frameworks using this modality assume access to a high-quality object tracker (in our case, an object pose estimator) and simulator. Our tasks can therefore only feature rigid-body objects and environments, which can be efficiently tracked with existing pose estimators [132] and simulated using existing rigid-body simulators [82]. Extending the framework to articulated or deformable objects would require adapting HUMAN2SIM2ROBOT to different approaches for state estimation and simulation modeling, which is dependent on advancements in these areas beyond rigid objects.

Beyond generalizing to non-rigid objects, our pose estimator also struggles with pose ambiguity of symmetric objects and sensor noise from reflective objects, or prolonged periods of significant occlusion of objects (especially when in grasp). Invariance to pose ambiguity of symmetric objects

can be achieved by modifying the anchor points used to determine object-centric rewards, for instance by automatically determining the axis of symmetry given the object mesh [98] and removing points orthogonal to this axis of rotation (see Appendix B.3). Reflective objects can be managed by distilling the pose-based policies into image-based policies, similar to prior work [72, 121].

Even if more generalizable and robust object tracking methods are developed, a broader question is whether it is feasible to assume that such a system can be implemented at scale, for instance in homes or in industrial settings. Our object pose estimator requires the object 3D model as input, and the model-free pose estimator using multiple RGB images of the object was much less reliable. While obtaining a high-quality object 3D model from a LiDAR scan was not prohibitively difficult for our framework, it might not be realistic to assume access to all object 3D models in large-scale industrial or home settings. In the broader literature, training image-based visuomotor policies [3, 5, 16, 46, 56, 146] has been dominant in robot learning due to the vast availability of Internet-scale images and videos [23] and image-based robot datasets [19, 28, 55, 126]. Crucially, such methods do not require prior knowledge of the object's 3D geometry at test-time, and only requires RGB cameras rather than depending on RGB-D or stereo cameras for accurate depth and shape information.

Hence, a key advantage of using point-based observation modalities (e.g., RGB images or point clouds) is their ability to relax the assumption of access to high-quality object 3D models, making them more feasible to obtain at test time. While they are more scalable, image-based policies tend to be more sensitive to visual changes such as background and lighting, as we observed in Section 2.4.1. Alternatively, if using simulation-based methods, the photorealism gap between real and rendered images or point clouds often causes policy deterioration during sim-to-real transfer, and training time is longer due to image rendering in simulation. This can be addressed with significant visual augmentation or pose-to-image distillation. In addition, obtaining object-centric representations for policy training require using object segmentations [105] or 2D point-trajectories [106]. These representations can serve as good bridges for sim-to-real transfer, though current pixel tracking methods [50] are generally less robust to occlusions and noise than pose estimators.

Overall, determining the ideal observation modality for a particular use case depends on whether certain assumptions (e.g., access to object 3D models, depth information) can be made. In either case, a combination of data augmentation or policy distillation [72, 121] between modalities can be leveraged to achieve the best of both worlds and facilitate more robust policies, and future research can investigate performing such pipelines at scale in diverse environments.

3.4.4 Test-time Adaptation to New Objects and Trajectories

HUMAN2SIM2ROBOT currently trains robust single-object, single-task policies for tasks specified by the pose trajectory of a single object. To achieve truly generalizable robotic manipulation, robotic agents should be able to leverage experience from prior object interactions to generalize to novel objects and trajectories at test-time, as humans often do. Future work can explore training

generalist multi-task, multi-object policies that are conditioned on object geometry (e.g., basis point set of point cloud features [100]) and desired object trajectory. This can be achieved using multi-task RL or teacher-student distillation [21, 135]. A potential pipeline for such a system could be: leveraging HUMAN2SIM2ROBOT to train multiple single-task single-object policies spanning several objects and trajectories, then distill those policies into a single multi-task policy conditioned on trajectory and object geometry. At test time, given an unseen object with novel geometry and a trajectory specified by a human demonstrator, the policy can execute the trajectory zero-shot.

To implement this at scale with diverse objects and trajectories that sufficiently cover the state space, a more automated method than collecting many real-world human videos is to train single-task single-object policies and perform policy distillation purely in simulation. By using highly diverse simulated assets and automatic trajectory generation (e.g., RoboCasa [91] or other text-to-3D asset generative models), it may be easier to scale up object and trajectory diversity to distill into a robust multi-task policy (inspired by Luo et al. [2024b]). This approach would instead be a sim-to-real pipeline that eliminates the need to collect lots of human videos with real-world objects, and just use a single human demonstration at test-time for the novel object and trajectory.

3.4.5 Accounting for Execution Mismatch

HUMAN2SIM2ROBOT dense object pose trajectory-tracking reward encourages the robot to follow the same object trajectory specified by the human, which in our case was successful for most tasks using a dexterous robotic hand. However, for highly dissimilar embodiments to human hands such as parallel-jaw grippers, it may be impossible for the robot to follow the human-provided trajectory for very complex tasks. For example, for `plate-pivot-lift-rack`, a parallel-jaw gripper robot may not be able to accomplish the human trajectory. An example optimal strategy for the robot could be pushing the plate to the edge of the table until part of the plate is off the table, then grasping and placing it into the dishrack. Our current reward formulation would not directly reward such behavior as it deviates fairly significantly from the human-provided trajectory, even if it is more optimal for the robot, and whether the robot explores this strategy is up to random chance.

This *execution mismatch* is not accounted for in our current reward formulation. Future work could investigate leveraging more diverse data to address this issue; for example, collecting multi-modal trajectories for completing the task (from one or multiple demonstrators), predicting feasiblity scores for object poses in the human-provided trajectoryies given the robot embodiment [7], and stitching together trajectory segments with high feasibility to complete the task.

Chapter 4

Conclusions & Future Work

In this thesis, we focused on the problem of scaling robot learning to new tasks, objects, and environments, without proportionally scaling the amount of human effort required. We discussed several ideas to address a critical challenge in robot learning: efficiently leveraging human data to train robust robot policies. To explore this question, we compared two RL-based approaches: (1) real-world autonomous RL shaped by VLM-generated rewards, and (2) sim-to-real RL from one human-demonstration. In Section 2, we presented Keypoint-based Affordance Guidance for Improvements (KAGI), a method for facilitating autonomous improvements of policies pre-trained on diverse human teleoperated data using VLM-generated shaping rewards, enabling robots to learn through interaction and self-practice in an autonomous, data-efficient manner. In Section 3, we proposed HUMAN2SIM2ROBOT, a real-to-sim-to-real RL framework for learning robust dexterous manipulation policies from a single human hand RGB-D video demonstration, leveraging task abstractions from the human video and domain-randomized simulation to train robust real-world RL policies with only a few minutes of human effort. We now conclude with a commentary comparing the approaches presented in this work, and by connecting the works presented with parallel research developments that facilitate scalable robot learning with minimal human effort.

Real-World vs. Sim-to-Real RL. Assuming a specific target environment, there are two primary benefits of real-world RL: first, the input observations will be in a more similar data distribution between train-time and test-time (especially for image-based policies), and second, agents learn directly from interaction with the physical environment subject to the environment's true dynamics, avoiding the sim-to-real dynamics gap. However, the primary drawbacks are that collecting a large amount of interaction data with environment resets is time-consuming, costly, and potentially unsafe, therefore it typically requires constant human supervision. The high sample complexity of RL often makes real-world training impractical. Therefore, current approaches still require substantial human effort and supervision for collecting in-domain demonstrations of the target task, optionally providing expert corrections during policy rollouts, and either performing environment

resets or supervising systems that autonomously self-practice. This may seem prohibitive for scaling robot learning; however, significant improvements in the performance of robotic foundation models could dramatically reduce the amount of human effort required for fine-tuning these models for specific tasks and environments with RL, which we explore in the next section.

In simulation, trajectory rollouts can be generated cheaply and at high speed, especially in fast, parallelized simulation environments. State-of-the-art techniques like domain randomization and random perturbations have helped narrow the sim-to-real gap dramatically. Only a small amount of real-world data (or none at all) is required for task specification or to adapt to or validate real-world dynamics; in this regard, sim-to-real RL is a promising approach in scaling robot learning without significantly increasing human effort. That said, the primary limitation of this method is quickly and accurately building high-fidelity 3D models in simulation, both in terms of photorealism and physical realism, of diverse environments at scale. For policies trained in simulation to successfully and safely transfer from to the real world, accurate collision and dynamics modeling at scale is of paramount importance. A potential hybrid approach could involve generating highly diverse interaction data for pre-training policies in simulation, followed by a short phase of real-world RL fine-tuning to adapt to novel tasks and the real-world environment dynamics, as proposed in Section 3.4.4.

Importance of a Performant Pre-Trained Policy. A key takeaway from our discussion on the efficacy of autonomous real-world RL is that real world RL fine-tuning should only be used as a *last-mile solution*. It is important that the pre-trained policy is generally very performant, such that real-world fine-tuning with RL does not have such a large performance gap to make up for, and can yield policies capable of real-world deployment within a reasonable amount of time. As a general, high-level goal, a sound pipeline for deploying a robotics system at scale in real-world environments would use real-world RL fine-tuning to elevate the performance of a pre-trained policy that succeeds ∼70-90% of the time to ∼100% in ∼1 hour or less. KAGI's pre-trained offline RL policies were much below this threshold, creating a large performance gap for RL fine-tuning to improve, and elevating the performance of pre-trained models across general tasks is an important research endeavor.

Beyond directly leveraging LLMs and VLMs to reason over robotic manipulation tasks as we explored in KAGI, parallel efforts seek to apply the foundation model paradigm to robot data by training vision-language-action models (VLAs): models that take as input raw visual observations and high-level language instructions, and outputs robot actions [3, 5, 19, 53, 54, 56, 94, 95]. By leveraging language model and vision transformer backbones, as well as pre-training on large-scale human and robotic datasets, VLAs formalize robotic manipulation as next-token prediction, generating action tokens that can be decoded into robot actions and directly executed on robot hardware. While VLAs require broad, large-scale training datasets, generalist robot policies that transfer zero- or few-shot to novel settings can drastically reduce the amount of human effort required at test time.

Training highly performant and general pre-trained VLAs could make autonomous RL fine-tuning pipelines like KAGI more practically useful. Kim et al. [2025] explores fine-tuning VLAs to adapt to

new tasks using teleoperated demonstrations. However, allowing pre-trained VLAs to autonomously improve and refine their behavior, guided by shaping rewards from VLMs, could further reduce the amount of human effort needed to collect task-specific in-domain demonstrations, allowing VLAs to generalize to novel tasks by interacting with the environment online. The benefit of real-world RL in this scenario is that most VLAs are trained with real-world RGB images. With real-world RL fine-tuning, the observations collected as interaction data are more in-distribution with the pre-training data, making it a viable method for elevating the performance of pre-trained VLAs. Fine-tuning VLAs in simulation has not yet been explored, potentially due to the significant distribution shift in observation space that might not result in improvements or even worsen the performance of the pre-trained model. An interesting avenue of research could explore unifying the paradigms of real-world RL, sim-to-real RL, and robotic foundation models by leveraging the advantages of simulation to fine-tune VLAs for adaptation to a specific environment.

Evaluating the Purpose of Human Demonstrations. A broader question involves re-evaluating the purpose of human demonstrations. Rethinking the role of human data is essential for scaling up robot learning and deployment without proportionally increasing human effort. Traditionally, data-hungry learning regimes like IL require humans to collect on the order of hundreds or thousands of trajectories as input for model training via teleoperation. While these efforts are important for training generalist robotic foundation models, the sheer diversity of tasks, objects, and environments that we want robots to be useful for makes it extremely challenging to collect data that sufficiently covers the space of potential interactions. This thesis proposes a shift in how we conceptualize human data in robot learning: rather than relying on large-scale demonstrations as direct inputs for training robotic policies, we propose a more strategic use of human data as a means for task specification and adaptation through online interaction.

Our work shows that human demonstrations need not serve as large-scale training datasets to be imitated, but rather as rich sources of task specification and task completion strategies. By extracting essential task abstractions and goal representations from minimal human input, we enable robot learning systems to autonomously explore and refine their own behaviors. The work presented in this thesis points towards an approach that inverts the traditional paradigm: instead of scaling human effort proportionally with the complexity of robot capabilities, we utilize sparse human guidance to bootstrap self-directed learning processes. Our results across diverse manipulation tasks, using autonomous RL with VLM-shaped rewards and sim-to-real transfer from a single demonstration, suggest that the future of robot learning lies not in relying primarily on humans to collect millions of demonstrations, but in developing frameworks that maximize the informational value extracted from each human interaction. A small set of human demonstrations can provide useful priors, guiding robots toward efficient fine-tuning and rapid generalization to new tasks. This approach shifts the focus from data quantity to data utility, unlocking scalable and adaptive robot learning systems while reducing reliance on manual human effort.

Bibliography

Michael Ahn, Anthony Brohan, Noah Brown, Yevgen Chebotar, Omar Cortes, Byron David, Chelsea Finn, Chuyuan Fu, Keerthana Gopalakrishnan, Karol Hausman, et al. Do as i can, not as i say: Grounding language in robotic affordances. In *Conference on Robot Learning*, 2022.

Max Balsells, Marcel Torne, Zihan Wang, Samedh Desai, Pulkit Agrawal, and Abhishek Gupta. Autonomous robotic reinforcement learning with asynchronous human feedback. In *Conference on Robot Learning*, 2023.

Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Xi Chen, Krzysztof Choromanski, Tianli Ding, Danny Driess, Avinava Dubey, Chelsea Finn, Pete Florence, Chuyuan Fu, Montse Gonzalez Arenas, Keerthana Gopalakrishnan, Kehang Han, Karol Hausman, Alex Herzog, Jasmine Hsu, Brian Ichter, Alex Irpan, Nikhil Joshi, Ryan Julian, Dmitry Kalashnikov, Yuheng Kuang, Isabel Leal, Lisa Lee, Tsang-Wei Edward Lee, Sergey Levine, Yao Lu, Henryk Michalewski, Igor Mordatch, Karl Pertsch, Kanishka Rao, Krista Reymann, Michael Ryoo, Grecia Salazar, Pannag Sanketi, Pierre Sermanet, Jaspiar Singh, Anikait Singh, Radu Soricut, Huong Tran, Vincent Vanhoucke, Quan Vuong, Ayzaan Wahid, Stefan Welker, Paul Wohlhart, Jialin Wu, Fei Xia, Ted Xiao, Peng Xu, Sichun Xu, Tianhe Yu, and Brianna Zitkovich. Rt-2: Vision-language-action models transfer web knowledge to robotic control. In *arXiv preprint arXiv:2307.15818*, 2023a.

Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Xi Chen, Krzysztof Choromanski, Tianli Ding, Danny Driess, Avinava Dubey, Chelsea Finn, et al. Rt-2: Vision-language-action models transfer web knowledge to robotic control. *arXiv preprint arXiv:2307.15818*, 2023b.

Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Joseph Dabis, Chelsea Finn, Keerthana Gopalakrishnan, Karol Hausman, Alex Herzog, Jasmine Hsu, Julian Ibarz, Brian Ichter, Alex Irpan, Tomas Jackson, Sally Jesmonth, Nikhil Joshi, Ryan Julian, Dmitry Kalashnikov, Yuheng Kuang, Isabel Leal, Kuang-Huei Lee, Sergey Levine, Yao Lu, Utsav Malla, Deeksha Manjunath, Igor Mordatch, Ofir Nachum, Carolina Parada, Jodilyn Peralta, Emily Perez, Karl Pertsch, Jornell Quiambao, Kanishka Rao, Michael Ryoo, Grecia Salazar, Pannag Sanketi, Kevin Sayed, Jaspiar Singh, Sumedh Sontakke, Austin Stone, Clayton Tan, Huong Tran, Vincent Vanhoucke,

Steve Vega, Quan Vuong, Fei Xia, Ted Xiao, Peng Xu, Sichun Xu, Tianhe Yu, and Brianna Zitkovich. Rt-1: Robotics transformer for real-world control at scale. In *Robotics: Science and Systems*, 2023c.

Luigi Campanaro, Siddhant Gangapurwala, Wolfgang Merkt, and Ioannis Havoutis. Learning and deploying robust locomotion policies with minimal dynamics randomization. In Alessandro Abate, Mark Cannon, Kostas Margellos, and Antonis Papachristodoulou, editors, *Proceedings of the 6th Annual Learning for Dynamics amp; Control Conference*, volume 242 of *Proceedings of Machine Learning Research*, pages 578–590. PMLR, 15–17 Jul 2024. URL https://proceedings.mlr.press/v242/campanaro24a.html.

Annie S. Chen, Alec M. Lessing, Yuejiang Liu, and Chelsea Finn. Curating demonstrations using online experience, 2025a. URL https://arxiv.org/abs/2503.03707.

Boyuan Chen, Fei Xia, Brian Ichter, Kanishka Rao, Keerthana Gopalakrishnan, Michael S Ryoo, Austin Stone, and Daniel Kappler. Open-vocabulary queryable scene representations for real world planning. In *IEEE International Conference on Robotics and Automation*, pages 11509–11522. IEEE, 2023.

Boyuan Chen, Zhuo Xu, Sean Kirmani, Brian Ichter, Danny Driess, Pete Florence, Dorsa Sadigh, Leonidas Guibas, and Fei Xia. Spatialvlm: Endowing vision-language models with spatial reasoning capabilities. In *Conference on Computer Vision and Pattern Recognition*, 2024a.

Claire Chen, Zhongchun Yu, Hojung Choi, Mark Cutkosky, and Jeannette Bohg. Dexforce: Extracting force-informed actions from kinesthetic demonstrations for dexterous manipulation, 2025b. URL https://arxiv.org/abs/2501.10356.

Sirui Chen, Chen Wang, Kaden Nguyen, Li Fei-Fei, and C Karen Liu. Arcap: Collecting high-quality human demonstrations for robot learning with augmented reality feedback. *arXiv preprint arXiv:2410.08464*, 2024b.

Xiaoyu Chen, Jiachen Hu, Chi Jin, Lihong Li, and Liwei Wang. Understanding domain randomization for sim-to-real transfer, 2022a. URL https://arxiv.org/abs/2110.03239.

Yuanpei Chen, Yaodong Yang, Tianhao Wu, Shengjie Wang, Xidong Feng, Jiechuan Jiang, Zongqing Lu, Stephen Marcus McAleer, Hao Dong, and Song-Chun Zhu. Towards human-level bimanual dexterous manipulation with reinforcement learning. In *Thirty-sixth Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2022b. URL https://openreview.net/forum?id=D29JbExncTP.

Yuanpei Chen, Chen Wang, Yaodong Yang, and Karen Liu. Object-centric dexterous manipulation from human motion data. In *8th Annual Conference on Robot Learning*, 2024c.

Xuxin Cheng, Kexin Shi, Ananye Agarwal, and Deepak Pathak. Extreme parkour with legged robots. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, 2024.

Cheng Chi, Siyuan Feng, Yilun Du, Zhenjia Xu, Eric Cousineau, Benjamin Burchfiel, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion. In *Proceedings of Robotics: Science and Systems (RSS)*, 2023.

Cheng Chi, Zhenjia Xu, Chuer Pan, Eric Cousineau, Benjamin Burchfiel, Siyuan Feng, Russ Tedrake, and Shuran Song. Universal manipulation interface: In-the-wild robot teaching without in-the-wild robots. In *Proceedings of Robotics: Science and Systems (RSS)*, 2024.

Matei T. Ciocarlie, Corey Goldfeder, and Peter K. Allen. Dexterous grasping via eigengrasps : A low-dimensional approach to a high-complexity problem. 2007. URL `https://api.semanticscholar.org/CorpusID:6853822`.

Open X-Embodiment Collaboration. Open X-Embodiment: Robotic learning datasets and RT-X models. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6892–6903, 2024. doi: 10.1109/ICRA57147.2024.10611477.

Erwin Coumans and Yunfei Bai. Pybullet, a python module for physics simulation for games, robotics and machine learning. `http://pybullet.org`, 2016–2019.

Murtaza Dalal, Min Liu, Walter Talbott, Chen Chen, Deepak Pathak, Jian Zhang, and Ruslan Salakhutdinov. Local policies enable zero-shot long-horizon manipulation. *International Conference of Robotics and Automation*, 2025.

Sudeep Dasari, Abhinav Gupta, and Vikash Kumar. Learning dexterous manipulation from exemplar object trajectories and pre-grasps. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3889–3896. IEEE, 2023.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, 2009. doi: 10.1109/CVPR.2009.5206848.

Vishnu Sashank Dorbala, Gunnar Sigurdsson, Robinson Piramuthu, Jesse Thomason, and Gaurav S. Sukhatme. Clip-nav: Using clip for zero-shot vision-and-language navigation. In *arXiv preprint arXiv:2211.16649*, 2022.

Danny Driess, Fei Xia, Mehdi S. M. Sajjadi, Corey Lynch, Aakanksha Chowdhery, Brian Ichter, Ayzaan Wahid, Jonathan Tompson, Quan Vuong, Tianhe Yu, Wenlong Huang, Yevgen Chebotar, Pierre Sermanet, Daniel Duckworth, Sergey Levine, Vincent Vanhoucke, Karol Hausman, Marc Toussaint, Klaus Greff, Andy Zeng, Igor Mordatch, and Pete Florence. Palm-e: An embodied multimodal language model. In *arXiv preprint arXiv:2303.03378*, 2023.

Michael Drolet, Simon Stepputtis, Siva Kailas, Ajinkya Jain, Jan Peters, Stefan Schaal, and Heni Ben Amor. A comparison of imitation learning algorithms for bimanual manipulation. *IEEE Robotics and Automation Letters (RA-L)*, 2024.

Gabriel Dulac-Arnold, Nir Levine, Daniel J. Mankowitz, Jerry Li, Cosmin Paduraru, Sven Gowal, and Todd Hester. Challenges of real-world reinforcement learning: definitions, benchmarks and analysis. *Mach. Learn.*, 110(9):2419–2468, September 2021. ISSN 0885-6125. doi: 10.1007/s10994-021-05961-4. URL https://doi.org/10.1007/s10994-021-05961-4.

Frederik Ebert, Yanlai Yang, Karl Schmeckpeper, Bernadette Bucher, Georgios Georgakis, Kostas Daniilidis, Chelsea Finn, and Sergey Levine. Bridge data: Boosting generalization of robotic skills with cross-domain datasets. In *Robotics: Science and Systems*, 2022.

Rishi Bommasani et al. On the opportunities and risks of foundation models. In *arXiv preprint arXiv:2108.07258*, 2022.

Kuan Fang, Fangchen Liu, Pieter Abbeel, and Sergey Levine. Moka: Open-world robotic manipulation through mark-based visual prompting. In *Robotics: Science and Systems*, 2024.

Justin Fu, Katie Luo, and Sergey Levine. Learning robust rewards with adverserial inverse reinforcement learnin. In *International Conference on Learning Representations*, 2018a.

Justin Fu, Avi Singh, Dibya Ghosh, Larry Yang, and Sergey Levine. Variational inverse control with events: A general framework for data-driven reward definition. *Advances in Neural Information Processing Systems*, 31, 2018b.

Gemini Team Google. Gemini: A family of highly capable multimodal models. In *arXiv preprint arXiv:2312.11805*, 2024.

Jiayuan Gu, Sean Kirmani, Paul Wohlhart, Yao Lu, Montserrat Gonzalez Arenas, Kanishka Rao, Wenhao Yu, Chuyuan Fu, Keerthana Gopalakrishnan, Zhuo Xu, Priya Sundaresan, Peng Xu, Hao Su, Karol Hausman, Chelsea Finn, Quan Vuong, and Ted Xiao. Rt-trajectory: Robotic task generalization via hindsight trajectory sketches, 2023.

Abhishek Gupta, Justin Yu, Tony Z. Zhao, Vikash Kumar, Aaron Rovinsky, Kelvin Xu, Thomas Devlin, and Sergey Levine. Reset-free reinforcement learning via multi-task learning: Learning dexterous manipulation behaviors without human intervention. In *2021 IEEE International Conference on Robotics and Automation*, pages 6664–6671, 2021.

Irmak Guzey, Yinlong Dai, Georgy Savva, Raunaq Bhirangi, and Lerrel Pinto. Bridging the human to robot dexterity gap through object-oriented rewards. In *arXiv preprint arXiv:2410.23289*, 2024. URL https://arxiv.org/abs/2410.23289.

Nick Heppert, Max Argus, Tim Welschehold, Thomas Brox, and Abhinav Valada. Ditto: Demonstration imitation by trajectory transformation. In *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2024.

Jonathan Ho and Stefano Ermon. Generative adversarial imitation learning. *Advances in Neural Information Processing Systems*, 29, 2016.

Yining Hong, Haoyu Zhen, Peihao Chen, Shuhong Zheng, Yilun Du, Zhenfang Chen, and Chuang Gan. 3d-llm: Injecting the 3d world into large language models. In *Advances in Neural Information Processing Systems*, 2023.

Chenguang Huang, Oier Mees, Andy Zeng, and Wolfram Burgard. Visual language maps for robot navigation. In *Proceedings of the IEEE International Conference on Robotics and Automation*, London, UK, 2023a.

Haifeng Huang, Zehan Wang, Rongjie Huang, Luping Liu, Xize Cheng, Yang Zhao, Tao Jin, and Zhou Zhao. Chat-3d v2: Bridging 3d scene and large language models with object identifiers. *arXiv preprint arXiv:2312.08168*, 2023b.

Wenlong Huang, Pieter Abbeel, Deepak Pathak, and Igor Mordatch. Language models as zero-shot planners: Extracting actionable knowledge for embodied agents. *arXiv preprint arXiv:2201.07207*, 2022a.

Wenlong Huang, Fei Xia, Ted Xiao, Harris Chan, Jacky Liang, Pete Florence, Andy Zeng, Jonathan Tompson, Igor Mordatch, Yevgen Chebotar, Pierre Sermanet, Noah Brown, Tomas Jackson, Linda Luu, Sergey Levine, Karol Hausman, and Brian Ichter. Inner monologue: Embodied reasoning through planning with language models. In *Conference on Robot Learning*, 2022b.

Wenlong Huang, Chen Wang, Ruohan Zhang, Yunzhu Li, Jiajun Wu, and Li Fei-Fei. Voxposer: Composable 3d value maps for robotic manipulation with language models. *arXiv preprint arXiv:2307.05973*, 2023c.

Wenlong Huang, Chen Wang, Yunzhu Li, Ruohan Zhang, and Li Fei-Fei. Rekep: Spatio-temporal reasoning of relational keypoint constraints for robotic manipulation. In *Conference on Robot Learning*, 2024.

Eric Jang, Alex Irpan, Mohi Khansari, Daniel Kappler, Frederik Ebert, Corey Lynch, Sergey Levine, and Chelsea Finn. BC-z: Zero-shot task generalization with robotic imitation learning. In *5th Annual Conference on Robot Learning*, 2021. URL https://openreview.net/forum?id=8kbp23tSGYv.

Yunfan Jiang, Agrim Gupta, Zichen Zhang, Guanzhi Wang, Yongqiang Dou, Yanjun Chen, Li Fei-Fei, Anima Anandkumar, Yuke Zhu, and Linxi Fan. Vima: General robot manipulation with multimodal prompts. In *Fortieth International Conference on Machine Learning*, 2023.

Zhenyu Jiang, Yuqi Xie, Kevin Lin, Zhenjia Xu, Weikang Wan, Ajay Mandlekar, Linxi Fan, and Yuke Zhu. Dexmimicgen: Automated data generation for bimanual dexterous manipulation via imitation learning. In *arXiv preprint arXiv:2410.24185*, 2024.

Dmitry Kalashnikov, Alex Irpan, Peter Pastor, Julian Ibarz, Alexander Herzog, Eric Jang, Deirdre Quillen, Ethan Holly, Mrinal Kalakrishnan, Vincent Vanhoucke, and Sergey Levine. Scalable deep reinforcement learning for vision-based robotic manipulation. In Aude Billard, Anca Dragan, Jan Peters, and Jun Morimoto, editors, *Proceedings of The 2nd Conference on Robot Learning*, volume 87 of *Proceedings of Machine Learning Research*, pages 651–673. PMLR, 29–31 Oct 2018. URL https://proceedings.mlr.press/v87/kalashnikov18a.html.

Nikita Karaev, Ignacio Rocco, Benjamin Graham, Natalia Neverova, Andrea Vedaldi, and Christian Rupprecht. Cotracker: It is better to track together. In *European Conference on Computer Vision*, 2024.

Elia Kaufmann, Leonard Bauersfeld, Antonio Loquercio, Matthias Mueller, Vladlen Koltun, and Davide Scaramuzza. Champion-level drone racing using deep reinforcement learning. In *Nature*, volume 620, pages 982–987, 2023.

Justin Kerr, Chung Min Kim, Mingxuan Wu, Brent Yi, Qianqian Wang, Ken Goldberg, and Angjoo Kanazawa. Robot see robot do: Imitating articulated object manipulation with monocular 4d reconstruction. In *8th Annual Conference on Robot Learning*, 2024. URL https://openreview.net/forum?id=2LLu3gavF1.

Danny Driess Adnan Esmail Michael Equi Chelsea Finn Niccolo Fusai Lachy Groom Karol Hausman Brian Ichter Szymon Jakubczak Tim Jones Liyiming Ke Sergey Levine Adrian Li-Bell Mohith Mothukuri Suraj Nair Karl Pertsch Lucy Xiaoyang Shi James Tanner Quan Vuong Anna Walling Haohuan Wang Ury Zhilinsky Kevin Black, Noah Brown. 0: Our first generalist policy, October 2024. URL https://www.physicalintelligence.company/blog/pi0.

James Darpinian Karan Dhabalia Danny Driess Adnan Esmail Michael Equi Chelsea Finn Niccolo Fusai Manuel Y. Galliker Dibya Ghosh Lachy Groom Karol Hausman Brian Ichter Szymon Jakubczak Tim Jones Liyiming Ke Devin LeBlanc Sergey Levine Adrian Li-Bell Mohith Mothukuri Suraj Nair Karl Pertsch Allen Z. Ren Lucy Xiaoyang Shi Laura Smith Jost Tobias Springenberg Kyle Stachowicz James Tanner Quan Vuong Homer Walke Anna Walling Haohuan Wang Lili Yu Ury Zhilinsky Kevin Black, Noah Brown. 0.5: a vision-language-action model with open-world generalization, April 2025. URL https://www.pi.website/blog/pi05.

Alexander Khazatsky, Karl Pertsch, Suraj Nair, Ashwin Balakrishna, Sudeep Dasari, Siddharth Karamcheti, Soroush Nasiriany, Mohan Kumar Srirama, Lawrence Yunliang Chen, Kirsty Ellis, Peter David Fagan, Joey Hejna, Masha Itkina, Marion Lepert, Yecheng Jason Ma, Patrick Tree

Miller, Jimmy Wu, Suneel Belkhale, Shivin Dass, Huy Ha, Arhan Jain, Abraham Lee, Youngwoon Lee, Marius Memmel, Sungjae Park, Ilija Radosavovic, Kaiyuan Wang, Albert Zhan, Kevin Black, Cheng Chi, Kyle Beltran Hatch, Shan Lin, Jingpei Lu, Jean Mercat, Abdul Rehman, Pannag R. Sanketi, Archit Sharma, Cody Simpson, Quan Vuong, Homer Rich Walke, Blake Wulfe, Ted Xiao, Jonathan Heewon Yang, Arefeh Yavary, Tony Z. Zhao, Christopher Agia, Rohan Baijal, Mateo Guaman Castro, Daphne Chen, Qiuyu Chen, Trinity Chung, Jaimyn Drake, Ethan Paul Foster, Jensen Gao, David Antonio Herrera, Minho Heo, Kyle Hsu, Jiaheng Hu, Donovon Jackson, Charlotte Le, Yunshuang Li, Roy Lin, Zehan Ma, Abhiram Maddukuri, Suvir Mirchandani, Daniel Morton, Tony Nguyen, Abigail O'Neill, Rosario Scalise, Derick Seale, Victor Son, Stephen Tian, Emi Tran, Andrew E. Wang, Yilin Wu, Annie Xie, Jingyun Yang, Patrick Yin, Yunchu Zhang, Osbert Bastani, Glen Berseth, Jeannette Bohg, Ken Goldberg, Abhinav Gupta, Abhishek Gupta, Dinesh Jayaraman, Joseph J. Lim, Jitendra Malik, Roberto Martín-Martín, Subramanian Ramamoorthy, Dorsa Sadigh, Shuran Song, Jiajun Wu, Michael C. Yip, Yuke Zhu, Thomas Kollar, Sergey Levine, and Chelsea Finn. Droid: A large-scale in-the-wild robot manipulation dataset. In *Robotics: Science and Systems*, 2024. URL https://doi.org/10.15607/RSS.2024.XX.120.

Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan Foster, Grace Lam, Pannag Sanketi, Quan Vuong, Thomas Kollar, Benjamin Burchfiel, Russ Tedrake, Dorsa Sadigh, Sergey Levine, Percy Liang, and Chelsea Finn. Openvla: An open-source vision-language-action model. *8th Annual Conference on Robot Learning*, 2024.

Moo Jin Kim, Chelsea Finn, and Percy Liang. Fine-tuning vision-language-action models: Optimizing speed and success. *arXiv preprint arXiv:2502.19645*, 2025.

KIRI Innovation (Hongkong) Limited. Kiri Engine: 3D Scanner App, 2024. URL https://www.kiriengine.app. Available for Android, iOS, and Web.

Jens Kober, J. Bagnell, and Jan Peters. Reinforcement learning in robotics: A survey. *The International Journal of Robotics Research*, 32:1238–1274, 09 2013. doi: 10.1177/0278364913495721.

Ilya Kostrikov, Ashvin Nair, and Sergey Levine. Offline reinforcement learning with implicit q-learning. *arXiv preprint arXiv:2110.06169*, 2021.

Aviral Kumar, Aurick Zhou, George Tucker, and Sergey Levine. Conservative q-learning for offline reinforcement learning. *Advances in Neural Information Processing Systems*, 33:1179–1191, 2020.

Aviral Kumar, Anikait Singh, Frederik Ebert, Mitsuhiko Nakamoto, Yanlai Yang, Chelsea Finn, and Sergey Levine. Pre-training for robots: Offline rl enables learning new tasks in a handful of trials. In *Robotics: Science and Systems*, 2022a.

Sateesh Kumar, Jonathan Zamora, Nicklas Hansen, Rishabh Jangir, and Xiaolong Wang. Graph inverse reinforcement learning from diverse videos. *Conference on Robot Learning (CoRL)*, 2022b.

Laan Labs. 3D Scanner App: LiDAR Scanner for iPad and iPhone Pro, 2024. URL https://3dscannerapp.com. Available for iOS devices.

Olivia Y. Lee, Annie Xie, Kuan Fang, Karl Pertsch, and Chelsea Finn. Affordance-guided reinforcement learning via visual prompting. *Robotics: Science Systems 2024, Task Specification Lifelong Robot Learning Workshops*, 2024.

Sergey Levine, Aviral Kumar, George Tucker, and Justin Fu. Offline reinforcement learning: Tutorial, review, and perspectives on open problems, 2020. URL https://arxiv.org/abs/2005.01643.

Jinhan Li, Yifeng Zhu, Yuqi Xie, Zhenyu Jiang, Mingyo Seo, Georgios Pavlakos, and Yuke Zhu. Okami: Teaching humanoid robots manipulation skills through single video imitation. In *8th Annual Conference on Robot Learning (CoRL)*, 2024.

Jacky Liang, Wenlong Huang, Fei Xia, Peng Xu, Karol Hausman, Brian Ichter, Pete Florence, and Andy Zeng. Code as policies: Language model programs for embodied control. In *arXiv preprint arXiv:2209.07753*, 2022.

Fanqi Lin, Yingdong Hu, Pingyue Sheng, Chuan Wen, Jiacheng You, and Yang Gao. Data scaling laws in imitation learning for robotic manipulation, 2024. URL https://arxiv.org/abs/2410.18647.

Kevin Lin, Christopher Agia, Toki Migimatsu, Marco Pavone, and Jeannette Bohg. Text2motion: from natural language instructions to feasible plans. *Autonomous Robots*, Nov 2023. ISSN 1573-7527. doi: 10.1007/s10514-023-10131-7.

Shilong Liu, Zhaoyang Zeng, Tianhe Ren, Feng Li, Hao Zhang, Jie Yang, Chunyuan Li, Jianwei Yang, Hang Su, Jun Zhu, et al. Grounding dino: Marrying dino with grounded pre-training for open-set object detection. *arXiv preprint arXiv:2303.05499*, 2023.

Tyler Ga Wei Lum, Martin Matak, Viktor Makoviychuk, Ankur Handa, Arthur Allshire, Tucker Hermans, Nathan D. Ratliff, and Karl Van Wyk. DextrAH-g: Pixels-to-action dexterous arm-hand grasping with geometric fabrics. In *8th Annual Conference on Robot Learning*, 2024. URL https://openreview.net/forum?id=S2Jwb0i7HN.

Tyler Ga Wei Lum, Olivia Y. Lee, C. Karen Liu, and Jeannette Bohg. Crossing the human-robot embodiment gap with sim-to-real rl using one human demonstration, 2025. URL https://arxiv.org/abs/2504.12609.

Zhengyi Luo, Jinkun Cao, Sammy Christen, Alexander Winkler, Kris Kitani, and Weipeng Xu. Grasping diverse objects with simulated humanoids, 2024a. URL https://arxiv.org/abs/2407.11385.

Zhengyi Luo, Jinkun Cao, Sammy Christen, Alexander Winkler, Kris M. Kitani, and Weipeng Xu. Omnigrasp: Simulated humanoid grasping on diverse objects. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024b. URL https://openreview.net/forum?id=Glt37xoU7e.

Raymond R. Ma and Aaron M. Dollar. On dexterity and dexterous manipulation. In *2011 15th International Conference on Advanced Robotics (ICAR)*, pages 1–7, 2011. doi: 10.1109/ICAR.2011.6088576.

Yecheng Jason Ma, William Liang, Vaidehi Som, Vikash Kumar, Amy Zhang, Osbert Bastani, and Dinesh Jayaraman. Liv: Language-image representations and rewards for robotic control. *arXiv preprint arXiv:2306.00958*, 2023a.

Yecheng Jason Ma, Shagun Sodhani, Dinesh Jayaraman, Osbert Bastani, Vikash Kumar, and Amy Zhang. Vip: Towards universal visual reward and representation via value-implicit pre-training. In *International Conference on Learning Representations*, 2023b.

Yecheng Jason Ma, William Liang, Guanzhi Wang, De-An Huang, Osbert Bastani, Dinesh Jayaraman, Yuke Zhu, Linxi Fan, and Anima Anandkumar. Eureka: Human-level reward design via coding large language models. In *International Conference on Learning Representations*, 2024.

Parsa Mahmoudieh, Deepak Pathak, and Trevor Darrell. Zero-shot reward specification via grounded natural language. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162, pages 14743–14752. PMLR, 2022.

Denys Makoviichuk and Viktor Makoviychuk. rl-games: A high-performance framework for reinforcement learning. https://github.com/Denys88/rl_games, May 2021.

Viktor Makoviychuk, Lukasz Wawrzyniak, Yunrong Guo, Michelle Lu, Kier Storey, Miles Macklin, David Hoeller, Nikita Rudin, Arthur Allshire, Ankur Handa, and Gavriel State. Isaac gym: High performance gpu-based physics simulation for robot learning, 2021. URL https://arxiv.org/abs/2108.10470.

Gabriel Margolis, Ge Yang, Kartik Paigwar, Tao Chen, and Pulkit Agrawal. Rapid locomotion via reinforcement learning. In *Robotics: Science and Systems*, 2022.

Max Sobol Mark, Ali Ghadirzadeh, Xi Chen, and Chelsea Finn. Fine-tuning offline policies with optimistic action selection. In *Deep Reinforcement Learning Workshop NeurIPS 2022*, 2022.

Takahiro Miki, Joonho Lee, Jemin Hwangbo, Lorenz Wellhausen, Vladlen Koltun, and Marco Hutter. Learning robust perceptive locomotion for quadrupedal robots in the wild. *Science Robotics*, 7 (62):eabk2822, 2022. doi: 10.1126/scirobotics.abk2822. URL https://www.science.org/doi/abs/10.1126/scirobotics.abk2822.

Suvir Mirchandani, Fei Xia, Pete Florence, Brian Ichter, Danny Driess, Montserrat Gonzalez Arenas, Kanishka Rao, Dorsa Sadigh, and Andy Zeng. Large language models as general pattern machines. In *Conference on Robot Learning*, 2023.

Suvir Mirchandani, David D. Yuan, Kaylee Burns, Md Sazzad Islam, Tony Z. Zhao, Chelsea Finn, and Dorsa Sadigh. Robocrowd: Scaling robot data collection through crowdsourcing, 2024. URL https://arxiv.org/abs/2411.01915.

Ashvin Nair, Abhishek Gupta, Murtaza Dalal, and Sergey Levine. Awac: Accelerating online reinforcement learning with offline datasets. In *arXiv preprint arXiv:2006.09359*, 2021.

Suraj Nair, Aravind Rajeswaran, Vikash Kumar, Chelsea Finn, and Abhinav Gupta. R3m: A universal visual representation for robot manipulation. In *Conference on Robot Learning*, 2022.

Mitsuhiko Nakamoto, Yuexiang Zhai, Anikait Singh, Max Sobol Mark, Yi Ma, Chelsea Finn, Aviral Kumar, and Sergey Levine. Cal-ql: Calibrated offline rl pre-training for efficient online fine-tuning. In *Advances in Neural Information Processing Systems*, 2023.

Soroush Nasiriany, Abhiram Maddukuri, Lance Zhang, Adeet Parikh, Aaron Lo, Abhishek Joshi, Ajay Mandlekar, and Yuke Zhu. Robocasa: Large-scale simulation of everyday tasks for generalist robots. In *Robotics: Science and Systems (RSS)*, 2024a.

Soroush Nasiriany, Fei Xia, Wenhao Yu, Ted Xiao, Jacky Liang, Ishita Dasgupta, Annie Xie, Danny Driess, Ayzaan Wahid, Zhuo Xu, et al. Pivot: Iterative visual prompting elicits actionable knowledge for vlms. *arXiv preprint arXiv:2402.07872*, 2024b.

William Noll, Yen-Hsun Wu, and Marco Santello. Dexterous manipulation: Differential sensitivity of manipulation and grasp forces to task requirements. *Journal of neurophysiology*, 132, 06 2024. doi: 10.1152/jn.00034.2024.

NVIDIA, :, Johan Bjorck, Fernando Castañeda, Nikita Cherniadev, Xingye Da, Runyu Ding, Linxi "Jim" Fan, Yu Fang, Dieter Fox, Fengyuan Hu, Spencer Huang, Joel Jang, Zhenyu Jiang, Jan Kautz, Kaushil Kundalia, Lawrence Lao, Zhiqi Li, Zongyu Lin, Kevin Lin, Guilin Liu, Edith Llontop, Loic Magne, Ajay Mandlekar, Avnish Narayan, Soroush Nasiriany, Scott Reed, You Liang Tan, Guanzhi Wang, Zu Wang, Jing Wang, Qi Wang, Jiannan Xiang, Yuqi Xie, Yinzhen Xu, Zhenjia Xu, Seonghyeon Ye, Zhiding Yu, Ao Zhang, Hao Zhang, Yizhou Zhao, Ruijie Zheng, and Yuke Zhu. Gr00t n1: An open foundation model for generalist humanoid robots, 2025. URL https://arxiv.org/abs/2503.14734.

Octo Model Team, Dibya Ghosh, Homer Walke, Karl Pertsch, Kevin Black, Oier Mees, Sudeep Dasari, Joey Hejna, Charles Xu, Jianlan Luo, Tobias Kreiman, You Liang Tan, Lawrence Yunliang Chen, Pannag Sanketi, Quan Vuong, Ted Xiao, Dorsa Sadigh, Chelsea Finn, and Sergey Levine. Octo: An open-source generalist robot policy. In *Proceedings of Robotics: Science and Systems*, Delft, Netherlands, 2024.

OpenAI. Gpt-4o system card. In *arXiv preprint arXiv:2410.21276*, 2024.

Georgios Pavlakos, Dandan Shan, Ilija Radosavovic, Angjoo Kanazawa, David Fouhey, and Jitendra Malik. Reconstructing hands in 3D with transformers. In *CVPR*, 2024.

Soo-Chang Pei and Lin-Gwo Liou. Automatic symmetry determination and normalization for rotationally symmetric 2d shapes and 3d solid objects. *Pattern Recognition*, 27(9):1193–1208, 1994. ISSN 0031-3203. doi: https://doi.org/10.1016/0031-3203(94)90005-1. URL https://www.sciencedirect.com/science/article/pii/0031320394900051.

Lerrel Pinto, Marcin Andrychowicz, Peter Welinder, Wojciech Zaremba, and Pieter Abbeel. Asymmetric actor critic for image-based robot learning. In Hadas Kress-Gazit, Siddhartha S. Srinivasa, Tom Howard, and Nikolay Atanasov, editors, *Robotics: Science and Systems XIV, Carnegie Mellon University, Pittsburgh, Pennsylvania, USA, June 26-30, 2018*, 2018. doi: 10.15607/RSS.2018.XIV.008. URL http://www.roboticsproceedings.org/rss14/p08.html.

Sergey Prokudin, Christoph Lassner, and Javier Romero. Efficient learning on point clouds with basis point sets. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4332–4341, 2019.

Haozhi Qi, Ashish Kumar, Roberto Calandra, Yi Ma, and Jitendra Malik. In-Hand Object Rotation via Rapid Motor Adaptation. In *Conference on Robot Learning (CoRL)*, 2022.

Yuzhe Qin, Yueh-Hua Wu, Shaowei Liu, Hanwen Jiang, Ruihan Yang, Yang Fu, and Xiaolong Wang. Dexmv: Imitation learning for dexterous manipulation from human videos, 2021.

Aravind Rajeswaran, Vikash Kumar, Abhishek Gupta, Giulia Vezzani, John Schulman, Emanuel Todorov, and Sergey Levine. Learning Complex Dexterous Manipulation with Deep Reinforcement Learning and Demonstrations. In *Proceedings of Robotics: Science and Systems (RSS)*, 2018.

Shreyas Sundara Raman, Vanya Cohen, Eric Rosen, Ifrah Idrees, David Paulius, and Stefanie Tellex. Planning with large language models via corrective re-prompting. In *Foundation Models for Decision Making Workshop at NeurIPS 2022*, 2022.

Nikhila Ravi, Valentin Gabeur, Yuan-Ting Hu, Ronghang Hu, Chaitanya Ryali, Tengyu Ma, Haitham Khedr, Roman Rädle, Chloe Rolland, Laura Gustafson, Eric Mintun, Junting Pan, Kalyan Vasudev Alwala, Nicolas Carion, Chao-Yuan Wu, Ross Girshick, Piotr Dollár, and

Christoph Feichtenhofer. Sam 2: Segment anything in images and videos. *arXiv preprint arXiv:2408.00714*, 2024. URL https://arxiv.org/abs/2408.00714.

Juntao Ren, Priya Sundaresan, Dorsa Sadigh, Sanjiban Choudhury, and Jeannette Bohg. Motion tracks: A unified representation for human-robot transfer in few-shot imitation learning, 2025. URL https://arxiv.org/abs/2501.06994.

Javier Romero, Dimitrios Tzionas, and Michael J. Black. Embodied hands: Modeling and capturing hands and bodies together. *ACM Transactions on Graphics, (Proc. SIGGRAPH Asia)*, 36(6), November 2017.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

Dhruv Shah, Blazej Osinski, Brian Ichter, and Sergey Levine. LM-nav: Robotic navigation with large pre-trained models of language, vision, and action. In *6th Annual Conference on Robot Learning*, 2022.

Archit Sharma, Kelvin Xu, Nikhil Sardana, Abhishek Gupta, Karol Hausman, Sergey Levine, and Chelsea Finn. Autonomous reinforcement learning: Formalism and benchmarking. In *International Conference on Learning Representations*, 2022.

Archit Sharma, Ahmed M. Ahmed, Rehaan Ahmad, and Chelsea Finn. Self-improving robots: End-to-end autonomous visuomotor reinforcement learning. In *7th Annual Conference on Robot Learning*, 2023.

Kenneth Shaw, Ananye Agarwal, and Deepak Pathak. Leap hand: Low-cost, efficient, and anthropomorphic hand for robot learning. *Robotics: Science and Systems (RSS)*, 2023.

Bruno Siciliano, Lorenzo Sciavicco, Luigi Villani, and Giuseppe Oriolo. *Robotics: Modelling, Planning and Control*. Springer Publishing Company, Incorporated, 2010. ISBN 1849966346.

Tom Silver, Soham Dan, Kavitha Srinivas, Joshua B. Tenenbaum, Leslie Pack Kaelbling, and Michael Katz. Generalized planning in pddl domains with pretrained large language models. In *AAAI Conference on Artificial Intelligence*, 2023.

Avi Singh, Larry Yang, Kristian Hartikainen, Chelsea Finn, and Sergey Levine. End-to-end robotic reinforcement learning without reward engineering. In *Robotics: Science and Systems*, 2019.

Himanshu Gaurav Singh, Antonio Loquercio, Carmelo Sferrazza, Jane Wu, Haozhi Qi, Pieter Abbeel, and Jitendra Malik. Hand-object interaction pretraining from videos, 2024. URL https://arxiv.org/abs/2409.08273.

Charles Sun, Jędrzej Orbik, Coline Manon Devin, Brian H. Yang, Abhishek Gupta, Glen Berseth, and Sergey Levine. Fully autonomous real-world reinforcement learning with applications to mobile manipulation. In *Proceedings of the 5th Conference on Robot Learning*, volume 164 of *Proceedings of Machine Learning Research*, pages 308–319. PMLR, 2022.

Balakumar Sundaralingam, Siva Kumar Sastry Hari, Adam Fishman, Caelan Garrett, Karl Van Wyk, Valts Blukis, Alexander Millane, Helen Oleynikova, Ankur Handa, Fabio Ramos, Nathan Ratliff, and Dieter Fox. curobo: Parallelized collision-free minimum-jerk robot motion generation, 2023.

Tony Tao, Mohan Kumar Srirama, Jason Jingzhou Liu, Kenneth Shaw, and Deepak Pathak. Dexwild: Dexterous human interactions for in-the-wild robot policies, 2025. URL https://arxiv.org/abs/2505.07813.

Gabriele Tiboni, Karol Arndt, and Ville Kyrki. Dropo: Sim-to-real transfer with offline domain randomization, 2023. URL https://arxiv.org/abs/2201.08434.

Marcel Torne, Anthony Simeonov, Zechu Li, April Chan, Tao Chen, Abhishek Gupta, and Pulkit Agrawal. Reconciling reality through simulation: A real-to-sim-to-real approach for robust manipulation. *Arxiv*, 2024.

Eugene Valassakis, Zihan Ding, and Edward Johns. Crossing the gap: A deep dive into zero-shot sim-to-real transfer for dynamics. In *International Conference on Intelligent Robots and Systems (IROS)*, 2020.

Eugene Valassakis, Georgios Papagiannis, Norman Di Palo, and Edward Johns. Demonstrate once, imitate immediately (dome): Learning visual servoing for one-shot imitation learning. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2022.

Andrés Villa, Juan León Alcázar, Motasem Alfarra, Kumail Alhamoud, Julio Hurtado, Fabian Caba Heilbron, Alvaro Soto, and Bernard Ghanem. Pivot: Prompting for video continual learning. In *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 24214–24223, 2023. doi: 10.1109/CVPR52729.2023.02319.

Pietro Vitiello, Kamil Dreczkowski, and Edward Johns. One-shot imitation learning: A pose estimation perspective. In *Conference on Robot Learning*, 2023.

Homer Walke, Kevin Black, Abraham Lee, Moo Jin Kim, Max Du, Chongyi Zheng, Tony Zhao, Philippe Hansen-Estruch, Quan Vuong, Andre He, Vivek Myers, Kuan Fang, Chelsea Finn, and Sergey Levine. Bridgedata v2: A dataset for robot learning at scale. In *Conference on Robot Learning*, 2023.

Weikang Wan, Haoran Geng, Yun Liu, Zikang Shan, Yaodong Yang, Li Yi, and He Wang. Unidexgrasp++: Improving dexterous grasping policy learning via geometry-aware curriculum and iterative generalist-specialist learning. *arXiv preprint arXiv:2304.00464*, 2023.

Chen Wang, Haochen Shi, Weizhuo Wang, Ruohan Zhang, Li Fei-Fei, and C. Karen Liu. Dexcap: Scalable and portable mocap data collection system for dexterous manipulation. *Robotics: Science and Systems*, 2024a.

Yen-Jen Wang, Bike Zhang, Jianyu Chen, and Koushil Sreenath. Prompt a robot to walk with large language models. *arXiv preprint arXiv:2309.09969*, 2023a.

Yufei Wang, Zhanyi Sun, Jesse Zhang, Zhou Xian, Erdem Biyik, David Held, and Zackory Erickson. Rl-vlm-f: Reinforcement learning from vision language foundation model feedback. In *Proceedings of the 41th International Conference on Machine Learning*, 2024b.

Zehan Wang, Haifeng Huang, Yang Zhao, Ziang Zhang, and Zhou Zhao. Chat-3d: Data-efficiently tuning large language model for universal dialogue of 3d scenes. *arXiv preprint arXiv:2308.08769*, 2023b.

Bowen Wen, Wei Yang, Jan Kautz, and Stan Birchfield. Foundationpose: Unified 6d pose estimation and tracking of novel objects. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 17868–17879, June 2024.

Annie Xie, Avi Singh, Sergey Levine, and Chelsea Finn. Few-shot goal inference for visuomotor learning and planning. In *Conference on Robot Learning*, pages 40–52, 2018.

Haoyu Xiong, Russell Mendonca, Kenneth Shaw, and Deepak Pathak. Adaptive mobile manipulation for articulated objects in the open world. *arXiv preprint arXiv:2401.14403*, 2024.

Charles Xu, Qiyang Li, Jianlan Luo, and Sergey Levine. Rldg: Robotic generalist policy distillation via reinforcement learning, 2024. URL https://arxiv.org/abs/2412.09858.

Yinzhen Xu, Weikang Wan, Jialiang Zhang, Haoran Liu, Zikang Shan, Hao Shen, Ruicheng Wang, Haoran Geng, Yijia Weng, Jiayi Chen, et al. Unidexgrasp: Universal robotic dexterous grasping via learning diverse proposal generation and goal-conditioned policy. *arXiv preprint arXiv:2303.00938*, 2023.

Jianing Yang, Xuweiyi Chen, Nikhil Madaan, Madhavan Iyengar, Shengyi Qian, David F. Fouhey, and Joyce Chai. 3d-grand: A million-scale dataset for 3d-llms with better grounding and less hallucination. In *arXiv preprint arXiv:2406.05132*, 2024a.

Jianwei Yang, Hao Zhang, Feng Li, Xueyan Zou, Chunyuan Li, and Jianfeng Gao. Set-of-mark prompting unleashes extraordinary visual grounding in gpt-4v. *arXiv preprint arXiv:2310.11441*, 2023.

Jingyun Yang, Max Sobol Mark, Brandon Vu, Archit Sharma, Jeannette Bohg, and Chelsea Finn. Robot fine-tuning made easy: Pre-training rewards and policies for autonomous real-world reinforcement learning. In *2024 IEEE International Conference on Robotics and Automation*. IEEE, 2024b.

Jianglong Ye, Jiashun Wang, Binghao Huang, Yuzhe Qin, and Xiaolong Wang. Learning continuous grasping function with a dexterous hand from human demonstrations. *IEEE Robotics and Automation Letters*, 2023.

Seonghyeon Ye, Joel Jang, Byeongguk Jeon, Sejune Joo, Jianwei Yang, Baolin Peng, Ajay Mandlekar, Reuben Tan, Yu-Wei Chao, Bill Yuchen Lin, Lars Liden, Kimin Lee, Jianfeng Gao, Luke Zettlemoyer, Dieter Fox, and Minjoon Seo. Latent action pretraining from videos, 2024. URL https://arxiv.org/abs/2410.11758.

Albert Yu, Adeline Foote, Raymond Mooney, and Roberto Martín-Martín. Natural language can help bridge the sim2real gap. In *Robotics: Science and Systems (RSS), 2024*, 2024.

Wenhao Yu, Nimrod Gileadi, Chuyuan Fu, Sean Kirmani, Kuang-Huei Lee, Montse Gonzalez Arenas, Hao-Tien Lewis Chiang, Tom Erez, Leonard Hasenclever, Jan Humplik, Brian Ichter, Ted Xiao, Peng Xu, Andy Zeng, Tingnan Zhang, Nicolas Heess, Dorsa Sadigh, Jie Tan, Yuval Tassa, and Fei Xia. Language to rewards for robotic skill synthesis. In *Conference on Robot Learning*, 2023.

Kevin Zakka, Andy Zeng, Pete Florence, Jonathan Tompson, Jeannette Bohg, and Debidatta Dwibedi. Xirl: Cross-embodiment inverse reinforcement learning. In *Conference on Robot Learning*, 2021.

Andy Zeng, Maria Attarian, Brian Ichter, Krzysztof Choromanski, Adrian Wong, Stefan Welker, Federico Tombari, Aveek Purohit, Michael Ryoo, Vikas Sindhwani, Johnny Lee, Vincent Vanhoucke, and Pete Florence. Socratic models: Composing zero-shot multimodal reasoning with language. In *International Conference on Learning Representations*, 2023.

Tony Z. Zhao, Jonathan Tompson, Danny Driess, Pete Florence, Seyed Kamyar Seyed Ghasemipour, Chelsea Finn, and Ayzaan Wahid. ALOHA unleashed: A simple recipe for robot dexterity. In *8th Annual Conference on Robot Learning*, 2024. URL https://openreview.net/forum?id=gvdXE7ikHI.

Deyao Zhu, Jun Chen, Xiaoqian Shen, Xiang Li, and Mohamed Elhoseiny. Minigpt-4: Enhancing vision-language understanding with advanced large language models. In *International Conference on Learning Representations*, 2024.

Henry Zhu, Justin Yu, Abhishek Gupta, Dhruv Shah, Kristian Hartikainen, Avi Singh, Vikash Kumar, and Sergey Levine. The ingredients of real-world robotic reinforcement learning, 2020. URL https://arxiv.org/abs/2004.12570.

Appendix A

Chapter 2 Appendix

A.1 Qualitative Analysis of VLM Performance.

The performance of our online RL fine-tuning system is influenced by the accuracy of VLM keypoint and waypoint predictions. While MOKA [30] verifies that keypoint-based reasoning of VLMs are reliably accurate for tabletop manipulation and generalizable to new objects and tasks, the paper only provides a single top-down viewpoint to the VLM, since MOKA's open-loop primitives compute actions based on pre-determined depth information specific to the evaluation setup, which requires manual engineering and tuning.

We observe that for closed-loop RL systems, providing dual-angle viewpoints are important to accurately generate waypoint trajectories in 3D space for dense rewards. We modify MOKA's [30] metaprompt to accommodate this additional dimension of input, so as to generate 3D waypoints instead of 2D waypoints for KAGI's dense rewards. We keep this metaprompt the same across all tasks. To test the robustness of our prompting structure, in addition to GPT-4o, we analyze the outputs generated from our prompts when provided to GPT-4V (used by MOKA [30], larger but slower than GPT-4o) and Gemini 1.5. Qualitative analysis of each VLM's outputs on our tasks determines that Gemini performs similarly and can replace GPT-4o in our framework. GPT-4V, while generally reliable, sometimes struggles to generate sensible depth movements using the side-angle view to facilitate successful completion of the task. Overall, KAGI is designed to flexibly incorporate existing and future VLMs of similar or better reasoning capability compared to GPT-4o, and we leave extensive qualitative comparisons and benchmarking frameworks for VLMs on 3D spatial reasoning tasks to future work.

A.2 Verifying Reproduction of RoboFuME.

Since our proposed dense shaping rewards to improve online policy fine-tuning are integrated into RoboFuME [139], and RoboFuME is a key baseline, we performed preliminary experiments on

two tasks featured in [139] – Cloth Folding and Cloth Covering – and verified their successful reproduction. We also performed additional experiments verifying RoboFuME's reliance on high-quality in-domain demonstrations with low multimodality for both pre-training RL policies and fine-tuning the sparse reward classifier, showing that eliminating in-domain demonstrations entirely was catastrophic for the system's performance. Finally, we explored RoboFuME's selection of Bridge data subsets [28, 126] for pre-training their offline RL policies for tabletop manipulation tasks, which we use in our experiments.

Overall, our preliminary experiments revealed that there are robustness challenges for pipelines relying on a large number of high-quality in-domain demonstrations, and incorporating task-relevant offline data facilitates transfer to the desired task. These key findings motivated the methodological details of KAGI for improving the efficiency of online fine-tuning with VLM-generated dense shaping rewards.

Please see the Additional Information section of our project page for more details on these areas of analysis and other preliminary experiments.

Appendix B

Chapter 3 Appendix

B.1 Real-to-Sim & Human Demo Processing Details

B.1.1 Digital Twin Construction Details

In this section, we describe the construction of the digital twin simulation environment (i.e., real-to-sim) and how we process the human demonstration. We start by taking a scan of the real-world environment for transfer to simulation. We used the off-the-shelf LiDAR scanning app called 3D Scanner App [64] to perform a detailed scene scan of the workspace (including static objects on the tabletop) and the robot. We then used this scene scan to align the robot, tabletop, and static objects in our digital twin simulation. The scene scan took ∼3 minutes and alignment of the assets in simulation took another ∼2 minutes. This process was performed once and the same simulation environment was used for all tasks. Next, we need to take an object scan to obtain the object mesh used for object pose tracking. We use another off-the-shelf LiDAR scanning app called Kiri Engine [58] to do this, as we find it is better for scanning small objects than the 3D Scanner App (conversely, 3D Scanner App seems better for larger scene scans than Kiri Engine). This takes ∼2 minutes per object, and we only need to do this once per object used in our experiments. All relevant code for the real-to-sim pipeline can be found here.

B.1.2 Human Demo Processing Details

For processing the human demonstration, each task takes 0.5 minutes to obtain an RGB-D video demonstration. The human demonstration is then processed in three steps: (i) generating the per-frame object and hand masks using SAM 2 [105]; (ii) generating the object pose trajectory by passing in the RGB-D video frames, object segmentation mask frames, and object mesh into FoundationPose [132]; (iii) obtaining the pre-manipulation hand pose by passing the corresponding RGB frame into HaMeR [97], then performing depth alignment with the segmented hand depth values. In total, demonstration processing takes ∼5-10 minutes, with only a few seconds of human effort. All relevant code for processing human video demonstrations can be found here.

B.1.3 Depth Alignment of HaMeR Hand Pose Estimates

HaMeR [97] predicts MANO hand pose and shape parameters from a single RGB image. While its 2D projections are generally reliable, the absence of depth information leads to significant 3D errors, which can be as large as 20–30 cm, particularly when the hand is far from the camera. Such discrepancies result in suboptimal pre-manipulation hand poses, adversely affecting reinforcement learning (RL) initialization and methods that require a full human hand trajectory.

To address this issue, we incorporate depth information to refine HaMeR's hand pose predictions. First, we obtain the initial MANO hand estimate from HaMeR. Next, we generate a hand segmentation mask using SAM 2 [105] and extract the corresponding 3D hand points from the depth image using the camera parameters. Next, we compute the 3D positions of the predicted MANO hand vertices visible to the camera and use a clustering algorithm to filter out erroneous depth points. Specifically, we construct a KD-tree from the extracted 3D points and identify pairs of points within 5 cm using nearest-neighbor queries. We then represent these points as a graph, where edges connect nearby points. We assume that the hand point cloud is the largest connected component, so we only retain the points in this connected graph, which mitigates the impact of depth noise and segmentation errors from the arm or background. Finally, we apply Iterative Closest Point (ICP) registration to align the MANO prediction with the segmented depth-based point cloud. This approach effectively improves alignment in most frames. However, accuracy degrades when significant occlusions occur, such as when fingers grasp around an object and face away from the depth camera.

B.2 Human-to-Robot Retargeting Details

To perform arm IK, we use the parallelized IK solver provided by cuRobo [118]. The solver generates 100 unique solutions, each seeded with 20 joint configurations. These configurations consist of a selected joint configuration combined with random noise sampled from a normal distribution with a standard deviation of 15 degrees. From these solutions, we select the one closest to the selected joint configuration that meets the desired target within 5 cm for position and 3 degrees for orientation. The pose target is the position of middle knuckle (called "middle_0") with a 3cm offset in the negative direction of the palm normal and a 3cm offset in the negative direction of the wrist to middle knuckle, and the orientation of the wrist (called "global_orient") with a rotation offset accounting for the difference in orientation convention of the wrist and middle knuckle.

For hand IK, we use PyBullet's IK solver [20]. The Allegro hand is moved to the pose from the previous arm IK solution, after which we solve IK for each finger to reach the specified fingertip targets. A default hand pose is used as the rest pose. Since achieving precise alignment with all fingertip targets is not always possible, PyBullet's solver provides reasonable solutions in these cases, whereas cuRobo's solver may yield poor results when it fails. We use position targets at the human hand fingertips (called "index_3", "middle_3", "ring_3", and "thumb_3") with no adjustments.

To retarget the pre-manipulation hand pose, we execute the above process once, using the default upright arm configuration as the selected joint configuration for seeding the solver and selecting the best solution. During this step, we apply collision-free arm IK to ensure the robot avoids contact with the environment or objects, as this joint configuration should not be in collision.

For retargeting the full hand pose trajectory, we iteratively solve for each hand pose, using the previously computed solution as the selected joint configuration to seed the solver. This approach ensures consistency in joint angles between frames, resulting in smoother motion. While solving, we enforce collision-free arm IK to avoid contact with the environment. However, we do not enforce object collision avoidance, as contact with the object is often necessary and both hand and object pose estimates are imperfect. If the IK process fails for a particular pose, we skip that index and proceed to the next one, keeping track of the corresponding timesteps.

When evaluating solutions, we select the one closest to the selected joint configuration based on the infinity norm, $||\boldsymbol{q}_{\text{solution}} - \boldsymbol{q}_{\text{selected}}||_{\infty}$.

After retargeting the full hand pose trajectory, we modify the trajectory to ensure smoothness. At each timestep, we compute the arm joint velocity with first-order finite-differencing. If any arm joint velocity exceeds its velocity limit (indicating a discontinuity or sudden jump in joint positions within a short time period), we increase the time interval between those points and interpolate the intermediate values. This ensures that the joint velocity limits are respected, resulting in a smooth trajectory that can be safely executed on the robot.

B.3 Reward Function Details

Our reward function formulation is flexible enough to handle rotational symmetries or axis invariances by removing or repositioning anchor points as needed. Apart from the adjustments for rotationally symmetric objects, we use the same value for hyperparameter L for all tasks in the experiments. This highlights the generality and robustness of the proposed reward specification, making it broadly applicable across various tasks and objects.

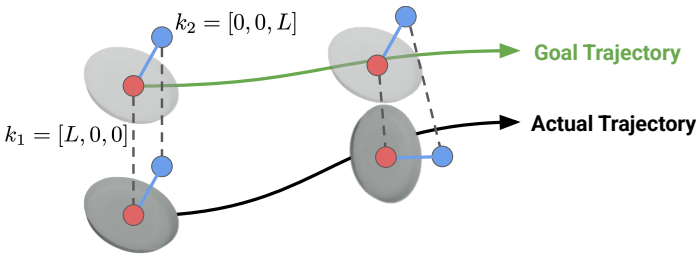


Figure B.1: **Modifying Anchor Points.** For rotationally symmetric objects, we remove anchor points orthogonal to the axis of symmetry (automatically determined [98]). This modified object pose representation is used for both reward computation and policy observation.

Figure B.1 shows how anchor points can be modified to accommodate rotational invariance for rotationally symmetric objects. Our use of anchor points is a very flexible representation to parameterize a pose tracking reward function.

B.4 Initial State Distribution Sampling

In this section, we describe sampling the initial state distribution for RL training in simulation. We sample the object pose as $\boldsymbol{T}_1^{\text{obj}} = \boldsymbol{T}_\tau^{\text{target}} \boldsymbol{T}^{\text{random}}$, where $\boldsymbol{T}^{\text{random}} \in \text{SE}(3)$ is random pose noise. The rotation component of $\boldsymbol{T}^{\text{random}}$ is sampled as roll, pitch, and yaw angles from $r, p, y \sim \mathcal{U}(-\theta_{\max}, \theta_{\max})$, where $\theta_{\max} = 20°$. The translation component of $\boldsymbol{T}^{\text{random}}$ is sampled as $\boldsymbol{t}_x, \boldsymbol{t}_y, \boldsymbol{t}_z \sim \mathcal{U}(-t_{\max}, t_{\max})$, where $t_{\max} = 0.1m$ is the maximum displacement. Next, we compute the relative transformation $\boldsymbol{T}^{\text{relative}} = \left(\boldsymbol{T}^{\text{obj}}\right)^{-1} \boldsymbol{T}^{\text{wrist}}$ to determine the new pre-manipulation wrist pose as $\boldsymbol{T}_1^{\text{obj}} \boldsymbol{T}^{\text{relative}}$. A small amount of noise is added to the new wrist pose and hand joint angles.

B.5 Simulation Training Details

We train our policy using Isaac Gym [82], a high-performance GPU-accelerated simulator that enables the parallel simulation of 4096 robots per GPU. Each policy is trained on a single NVIDIA A100 GPU with 40 GB of VRAM. This configuration allows us to achieve a simulation speed of approximately 7k frames per second (FPS). Each frame corresponds to a single action step with a control timestep of 66.7 ms (15 Hz), subdivided into 8 simulation timesteps of 8.33 ms (120 Hz).

Our training duration ranges from approximately 5 to 24 hours wall-clock time depending on the task, amounting to around 0.6 billion frames, which corresponds to roughly one year of simulated experience (0.6B / 15 / 3600 / 24 / 365).

We train our policies using Proximal Policy Optimization (PPO) [108] with rl-games [81], a highly optimized GPU-based implementation that employs vectorized observations and actions for efficient training. The policy is trained with learning rate 5×10^{-4}, discount factor $\gamma = 0.998$, entropy coefficient of 0, and PPO clipping parameter $\epsilon_{\text{clip}} = 0.2$. Additionally, we normalize observations, value estimates, and advantages, and train the policy using four mini-epochs per policy update.

We use $t_{\text{offset}} = 30$ (at 30Hz, 1 second) by default, though we slightly vary t_{offset} depending on how quickly the human demonstration was performed.

Although our control policies will not have access to privileged simulation state information when deployed in the real world, we can still use privileged information to accelerate training in simulation. We use Asymmetric Actor Critic training [99], in which our critic $V(\boldsymbol{s})$ is given all privileged state information $\boldsymbol{s}$ and our policy $\pi(\boldsymbol{o})$ is provided an observation $\boldsymbol{o}$, which is a limited subset of this privileged state information. With this method, the policy learns to perform the task using only observations we can capture in the real world, but the critic can leverage privileged state information to provide more accurate value estimates, improving the speed and quality of policy training.

The state at timestep t is $\boldsymbol{s}_t = [\boldsymbol{o}_t, \boldsymbol{v}_t, \boldsymbol{\omega}_t, t, \boldsymbol{f}_t^{\text{dof}}, \boldsymbol{F}_t^{\text{fingers}}]$, where $\boldsymbol{o}_t$ is the observation, $\boldsymbol{v}_t \in \mathbb{R}^3$ is the object linear velocity, $\boldsymbol{\omega}_t \in \mathbb{R}^3$ is the object angular velocity, $t \in \mathbb{R}$ is current timestep, $\boldsymbol{f}_t^{\text{dof}} \in \mathbb{R}^{N^{\text{joints}}}$ is the vector of robot joint forces, $\boldsymbol{F}_t^{\text{fingers}} \in \mathbb{R}^{N^{\text{fingers}}}$ contains fingertip contact forces.

The training process utilizes a horizon length of 16 (i.e., the number of timesteps between updates

for each robot, with all robots running in parallel) and 4096 parallel agents. The policy architecture consists of a multi-layer perceptron (MLP) with hidden layers of size [512, 512], an LSTM module with 1024 hidden units, and a critic network with hidden layers of size [1024, 512].

To improve the robustness and generalization of our policy, we apply extensive domain randomization during training. Randomizations are applied every 720 simulation steps and include variations in observations, actions, physics parameters, and object properties. Gaussian noise with a standard deviation of 0.01 is added to both observations and actions. Gravity is perturbed additively using Gaussian noise with a standard deviation of 0.3. The scale, mass, and friction of the object and table are randomized with a scaling parameter sampled from $[0.7, 1.3]$. The robot's scale, damping, stiffness, friction, and mass are also randomized with a scaling parameter sampled from $[0.7, 1.3]$.

We also introduce random force perturbations to the object. At each timestep, there is a 5% probability of applying a force with a magnitude equal to 50 times the object's mass, directed along a randomly sampled unit vector. These perturbations serve two key purposes. First, they can displace the object before the robot makes contact, simulating real-world uncertainties such as unexpected disturbances or pose estimation errors. This encourages the policy to actively track and reach for objects that may not be precisely where they were initially observed. Second, if the object is already grasped, these perturbations can destabilize the grasp, promoting the development of robust and stable grasping strategies that minimize the likelihood of dropping the object.

All relevant code for PPO can be found here, and for simulation training found here

B.6 Real-Time Perception Details

In this section, we describe our real-time perception pipeline, which enables pose estimation at 30Hz using FoundationPose [132]. The process begins with pose registration to determine the object's initial pose, which takes approximately 1 second. This step requires a textured object mesh and a segmented object mask. To generate the mask, we pass the image and a text prompt into Grounding DINO [71] to obtain an object bounding box. The image and bounding box are then passed to SAM2 [105], which produces a high-quality segmented object mask. The text prompt can either be manually provided or automatically generated by rendering the textured object mesh into an image and using GPT-4o to produce a descriptive prompt.

Once the initial pose is established, FoundationPose performs real-time object pose tracking by generating pose hypotheses near the previous estimate and selecting the best match, achieving a consistent 30Hz rate. Although tracking is generally robust, pose estimates can degrade when the object moves rapidly or becomes heavily occluded. To address this, we implement a separate pose evaluation process that monitors the pose estimate quality and triggers re-registration if needed.

This evaluation is performed by running SAM 2 at 1Hz to produce high-quality segmented

object masks, which are treated as ground truth due to SAM 2's reliability, even under challenging conditions. The predicted object mask, generated using FoundationPose's pose estimate and the known camera parameters, is compared to the ground-truth mask using the intersection over union (IoU) metric. If the IoU falls below 0.1, FoundationPose reinitializes the pose registration process.

B.7 Full Task List

We perform experiments on the following tasks:

- `snackbox-push`: The objective is to push the snackbox across the table until it makes contact with a static box. The snackbox is initialized face-down, with its position randomized within a 4 cm $\times$ 4 cm region. The task is considered successful if the snackbox contacts the static box.
- `plate-push`: Similar to `snackbox-push`, this task requires pushing a plate across the table until it contacts the static box. The plate starts in a flat orientation, with its position randomized within a 4 cm $\times$ 4 cm region. Success is achieved when the plate contacts the static box.
- `snackbox-pivot`: The goal is to pivot the snackbox from a face-down orientation to a sideways orientation using the static box for support. The snackbox is initialized in a face-down orientation, with its position randomized within a 1 cm $\times$ 4 cm region. Due to the robot's initial position, the snackbox cannot be substantially moved in one direction. The task is successful if the snackbox is pivoted against the static box into a stable sideways orientation.
- `pitcher-pour`: This task involves grasping a pitcher by its handle, lifting it off the table, and reorienting it so that its spout is positioned above a static saucepan, simulating a pouring motion. The pitcher starts in an upright orientation, with its position randomized within a 4 cm $\times$ 4 cm region. The task is successful if the pitcher is lifted by its handle and correctly positioned with its spout above the saucepan.
- `snackbox-push-pivot`: This task combines the `snackbox-push` and `snackbox-pivot` tasks. The snackbox starts in a face-down orientation, with its position randomized within a 4 cm $\times$ 4 cm region. The task consists of two sequential steps. The push task is successful if the snackbox is first pushed to make contact with the static box, and the pivot task is successful if the snackbox is pivoted against the box into a sideways orientation. Success is graded on a three-level scale. The score is 0 if the push fails, 0.5 if the push is successful but the pivot fails, and 1 if both the push and pivot are successful.
- `plate-lift-rack`: The objective is to lift a plate and place it into a dishrack. The plate starts in an upright orientation, leaning against the static box, with its position randomized within a 0.5 cm $\times$ 4 cm region. Since the plate must remain leaning against the box, its movement

is primarily constrained along the box length. The task consists of two sequential steps. The lift task is successful if the plate is lifted off of the table, and the rack task is successful if the plate is placed into the dishrack while maintaining an upright orientation. Success is graded on a three-level scale. The score is 0 if the lift fails, 0.5 if the lift is successful but the rack fails, and 1 if both the lift and rack are successful.

- `plate-pivot-lift-rack`: This task requires a sequence of actions: pivoting a plate against the static box, lifting it off the table, and placing it into a dishrack. The plate starts in a flat orientation next to the static box, with its position randomized within a 0.5 cm $\times$ 4 cm region.

  The task consists of three sequential steps. The pivot task is successful if the plate is pivoted against the static box to an upright orientation, the lift task is successful if the plate is lifted off of the table, and the rack task is successful if the plate is placed into the dishrack while maintaining an upright orientation. Success is graded on a four-level scale. The score is 0 if the pivot fails, 0.33 if the pivot is successful but the lift fails, 0.66 if the pivot and lift are successful but the rack fails, and 1 if the pivot, lift, and rack are successful.

B.8 Baseline Details

B.8.1 Full Human Hand Trajectory Estimation

Our baseline methods typically require robot action labels for every step of the task. When working with only a single human video demonstration, this can be done by estimating the human hand pose in each frame and retargeting it to the robot. Specifically, we perform hand pose estimation using HaMeR, followed by a depth alignment step (Appendix B.1). The resulting hand poses are mapped to robot joint configurations using an inverse kinematics (IK) procedure similar to that in Section 3.2.1. However, simply solving the IK for each frame independently and stitching the results together often fails due to three main issues: errors in hand pose estimation, a lack of consistency/smoothness over time, and unreachable target poses. We address these issues as follows:

1. **Mitigating Hand Pose Errors.** Hand pose estimation can suffer from large errors when fingers are occluded (Appendix B.1). To address this, we compare each newly estimated pose with the previous pose. If their distance exceeds a threshold, we skip the current frame's pose rather than attempting to retarget an unreliable estimate.

2. **Ensuring Joint Consistency.** To maintain smooth transitions between consecutive poses, we iteratively solve IK using the previous solution as both a default and a seed configuration. We employ cuRobo [118] to generate 100 parallel solutions, each initialized by adding Gaussian noise (15° standard deviation) to the previous IK solution. We then select the solution whose joint

angles have the smallest ℓ_∞ difference from the previous timestep's configuration. If even this "best" solution differs excessively, we skip that frame.

3. **Handling Unreachable Targets.** If the IK target is unreachable, we skip the corresponding frame.

After applying these checks, we downsample the resulting trajectory and verify that all joint velocities remain below the robot's limits. If any exceed the limit, we stretch the time between successive waypoints to reduce velocity. Although this procedure generally yields a plausible trajectory, inaccuracies can still arise in cases of severe finger occlusion or when the hand is far from the camera. In contrast, obtaining a reliable pre-manipulation hand pose is generally easier, as it requires only a single frame with accurate hand pose estimation. Such a frame is easier to find because the hand is typically not heavily occluded when it approaches the object, while the full demonstration can include much more occlusion of the hand.

B.8.2 Replay Details

In this section, we provide additional details on the implementation of replay. First, we need to clearly define the frames we care about. We define $\boldsymbol{T}^{A\to B}$ as the relative transformation of frame B with respect to frame A. This means $\boldsymbol{T}^{A\to C} = \boldsymbol{T}^{A\to B}\boldsymbol{T}^{B\to C}$.

Let R be the robot's base frame, M be the robot's middle-finger frame, C be the camera frame, and O be the object frame. Given these four frames, we need three independent relative transforms to fully specify this system. The camera extrinsics calibration gives us $\boldsymbol{T}^{R\to C}$. FoundationPose object pose estimates give us $\boldsymbol{T}^{C\to O}$ at each timestep. HaMeR with depth refinement gives us hand pose estimates that allow us to compute the desired pose of the robot's middle-finger frame $\boldsymbol{T}^{C\to M}$. This fully specifies the demonstration. This allows us to compute the object trajectory $\{\boldsymbol{T}_t^{R\to O}\}_{t=1}^T$ and the robot's middle-finger trajectory $\{\boldsymbol{T}_t^{R\to M}\}_{t=1}^T$ for this demonstration. This is used to perform the inverse kinematics procedure described above to generate a robot configuration trajectory. To execute replay in the real world, we can directly track this joint trajectory with joint PD control.

B.8.3 Object-Aware Replay Details

For object-aware replay, at timestep $t = 1$, we use FoundationPose to estimate the new object pose $\boldsymbol{T}_1^{R\to O^{\text{new}}}$, which will be similar but not identical to the initial object pose during demonstration collection $\boldsymbol{T}_1^{R\to O}$. Our goal is to compute a new robot middle-finger trajectory $\{\boldsymbol{T}_t^{R\to M^{\text{new}}}\}_{t=1}^T$ that keeps the same relative pose between the object and the middle-finger as in the demonstration.

Let $\boldsymbol{T}^{O\to M}$ be the relative pose between the object and the middle-finger at each timestep of the demonstration. This can be computed as $\boldsymbol{T}^{O\to M} = (\boldsymbol{T}^{R\to O})^{-1}\boldsymbol{T}^{R\to M}$. Our goal is to maintain this same relative relationship during replay, such that $\boldsymbol{T}^{O\to M} = \boldsymbol{T}^{O^{\text{new}}\to M^{\text{new}}}$.

The new middle-finger trajectory can then be computed as:

$$\boldsymbol{T}_t^{R\to M^{\text{new}}} = \boldsymbol{T}_t^{R\to O^{\text{new}}}\boldsymbol{T}_t^{O^{\text{new}}\to M^{\text{new}}} \tag{B.1}$$
$$= \boldsymbol{T}_t^{R\to O^{\text{new}}}\boldsymbol{T}_t^{O\to M} \tag{B.2}$$
$$= \boldsymbol{T}_t^{R\to O^{\text{new}}}(\boldsymbol{T}_t^{R\to O})^{-1}\boldsymbol{T}_t^{R\to M} \tag{B.3}$$

To compute $\boldsymbol{T}_t^{R\to O^{\text{new}}}$ for $t > 0$, we assume the object follows the same relative motion as in the demonstration, but starting from the new initial pose. This can be computed as:

$$\boldsymbol{T}_t^{R\to O^{\text{new}}} = \boldsymbol{T}_1^{R\to O^{\text{new}}}(\boldsymbol{T}_1^{R\to O})^{-1}\boldsymbol{T}_t^{R\to O} \tag{B.4}$$

Substituting this into our equation for the new middle-finger trajectory:

$$\boldsymbol{T}_t^{R\to M^{\text{new}}} = \boldsymbol{T}_1^{R\to O^{\text{new}}}(\boldsymbol{T}_1^{R\to O})^{-1}\boldsymbol{T}_t^{R\to O}(\boldsymbol{T}_t^{R\to O})^{-1}\boldsymbol{T}_t^{R\to M} \tag{B.5}$$
$$= \boldsymbol{T}_1^{R\to O^{\text{new}}}(\boldsymbol{T}_1^{R\to O})^{-1}\boldsymbol{T}_t^{R\to M} \tag{B.6}$$

This simplifies to:

$$\boldsymbol{T}_t^{R\to M^{\text{new}}} = \boldsymbol{T}_{\text{RELATIVE}}\boldsymbol{T}_t^{R\to M} \tag{B.7}$$

where $\boldsymbol{T}_{\text{RELATIVE}} = \boldsymbol{T}_1^{R\to O^{\text{new}}}(\boldsymbol{T}_1^{R\to O})^{-1} = \boldsymbol{T}_1^{R\to O}\boldsymbol{T}_1^{O\to O^{\text{new}}}(\boldsymbol{T}_1^{R\to O})^{-1}$ is the transformation that accounts for the change in the initial object pose. This transformation is applied to the entire middle-finger trajectory, effectively adjusting the demonstration to the new initial object pose while preserving the relative motion between the object and the middle-finger.

To execute object-aware replay in the real world, we can directly track this new joint trajectory with joint PD control.

B.8.4 Behavior Cloning Details

To train a Diffusion Policy, we need to collect a dataset of observation and action pairs. Because we only have one human demonstration, we can generate additional demonstration data by introducing small amounts of transformation noise to the object's pose and then use the process above to compute robot joint configuration trajectories with adjusted IK targets that account for this noise. Specifically, we sample $\boldsymbol{T}_1^{O\to O^{\text{new}}}$ with the same translation and rotation noise as used in RL training and then run the object-aware replay computation above to get new object pose trajectories and robot joint configuration trajectories. The observation consists of the vector of robot joints $\boldsymbol{q}_t$, the palm pose $\boldsymbol{p}_t^{\text{palm}}$, and the object pose $\boldsymbol{p}_t^{\text{object}}$. The action consists of joint position targets relative to the current robot position. We train with batch size 128, 50 diffusion iterations, learning rate 1e-4, and weight decay 1e-6, using Drolet et al. [2024]'s state-based diffusion policy implementation.

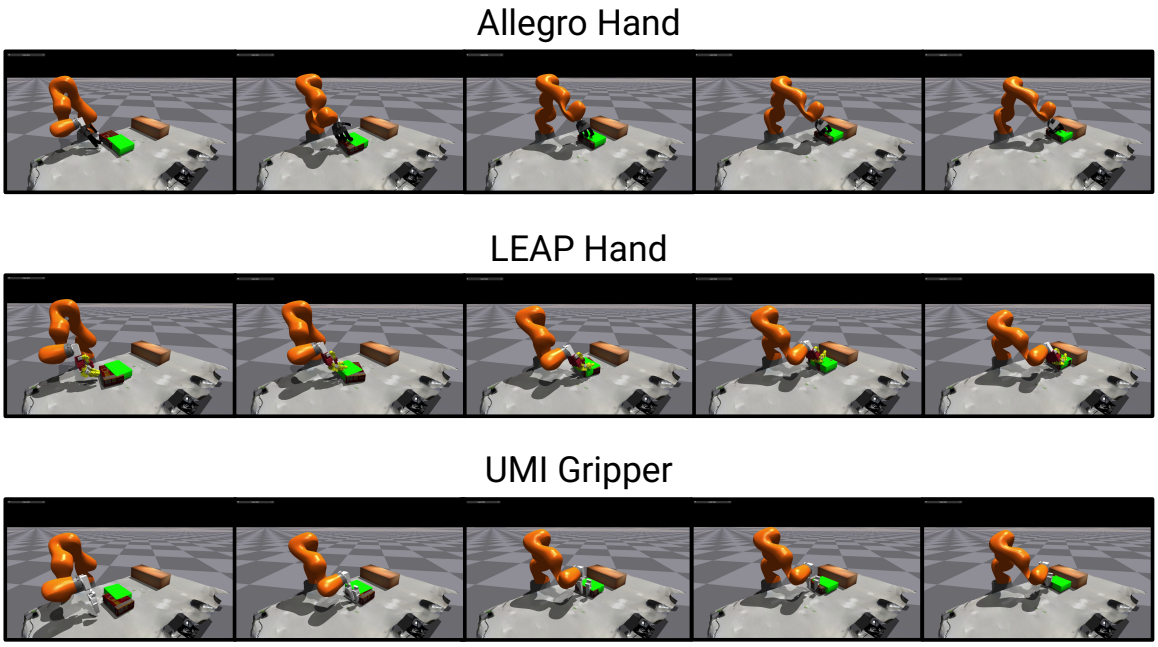


Figure B.2: **Other Embodiments.** HUMAN2SIM2ROBOT can be applied to different robot embodiments, including an Allegro Hand, a LEAP Hand [112] and a UMI gripper [17]. This is demonstrated with preliminary simulation experiments for the `snackbox-push` task. The green box represents the target pose of the snackbox.

B.9 Other Embodiments

Although our experiments are primarily tested on a Kuka arm and Allegro hand, we expect HUMAN2SIM2ROBOT to work on other robot embodiments, as there are no aspects of the framework that are specific to this embodiment. To validate this, we perform initial experiments demonstrating HUMAN2SIM2ROBOT on a LEAP Hand [112] and UMI gripper [17] on the `snackbox-push` task. Figure B.2 shows qualitative results using these embodiments, which shows that this was successful without any reward tuning. The only modifications required to make this work were changing the robot URDF and the robot configuration for retargeting via inverse kinematics (changing link names, relative orientations, default joint configuration, collision spheres), geometric fabric controller (link names, default joint configuration, collision spheres), and simulation environment (link names, action/observation dimensions).