A Real-to-Sim-to-Real Approach to Robotic Manipulation with VLM-Generated Iterative Keypoint Rewards

Shivansh Patel[1*], Xinchen Yin[1*], Wenlong Huang[2], Shubham Garg[3], Hooshang Nayyeri[3], Li Fei-Fei[2], Svetlana Lazebnik[1], Yunzhu Li[4]

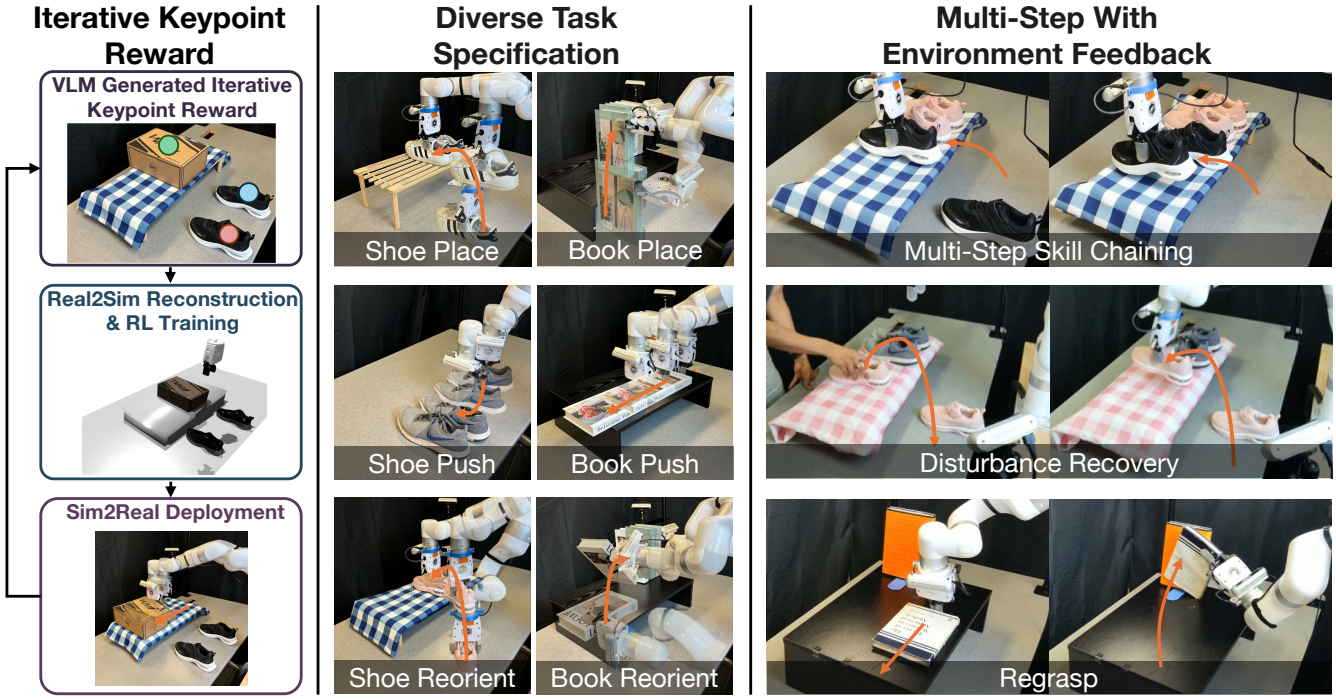

Fig. 1: **Iterative Keypoint Reward (IKER)** is a visually grounded reward generated by Vision-Language Models (VLMs) as task specification. The framework reconstructs the real-world scene in simulation, and the generated reward is used to train RL policies. We evaluate our framework on diverse tasks in the real world and demonstrate several notable capabilities.

***Abstract*— Task specification for robotic manipulation in open-world environments is challenging. Importantly, this process requires flexible and adaptive objectives that align with human intentions and can evolve through iterative feedback. We introduce Iterative Keypoint Reward (IKER), a framework that leverages VLMs to generate and refine visually grounded reward functions serving as dynamic task specifications for multi-step manipulation tasks. Given RGB-D observations and free-form language instructions, IKER samples keypoints from the scene and utilizes VLMs to generate Python-based reward functions conditioned on these keypoints. These functions operate on the spatial relationships between keypoints, enabling precise SE(3) control and leveraging VLMs as proxies to encode human priors about robotic behaviors. We reconstruct real-world scenes in simulation and use the generated rewards to train RL policies, which are then deployed into the real world—forming a real-to-sim-to-real loop. Our approach demonstrates notable capabilities across diverse scenarios, including both prehensile and non-prehensile tasks, showcasing multi-step task execution, spontaneous error recovery, and on-the-fly strategy adjustments. The results highlight IKER's effectiveness in enabling robots to perform multi-step tasks in dynamic environments through iterative reward shaping.**

*indicates equal contribution. [1]University of Illinois at Urbana-Champaign, [2]Stanford University, [3]Amazon, [4]Columbia University

Project Page: https://iker-robot.github.io/

I. Introduction

Task specification is a critical capability for robots operating in unstructured, open-world environments, where predefined, rigid instructions are insufficient to capture the complexity of real-world interactions. In such settings, task specifications must not only define an optimizable objective but also incorporate human priors about the robot's intended behaviors. This process often evolves iteratively, as humans refine the objective by observing the robot's execution and making adjustments. For example, in practical RL deployments, reward functions used to guide the robot's actions are shaped through multiple iterations, with humans revising them based on the outcomes observed during training. This iterative shaping encodes human expectations and behavioral biases, aligning the robot's behavior more closely with real-world needs. For instance, in multi-stage tasks, the reward function can be designed to encourage intermediate steps that mirror human strategies, such as regrasping an object to enable a feasible grasp.

Recent works have demonstrated that vision-language models (VLMs) encode rich world knowledge through pre-training on Internet-scale data [1–8]. Prior works have explored various task specifications using LLMs and VLMs [9–12], but typically operate on predefined objects or environments which are challenging to deploy in open-world scenarios. Additionally, much of this prior work focuses on tasks like positional rearrangement, where the primary challenge lies in specifying only coarse-grained positional information (e.g., object location) [8–11, 13], rather than handling more complex, dynamic tasks (e.g., non-prehensile object pushing). Given these limitations, a natural question arises: How can we effectively ground the visual understanding of VLMs to open-world tasks that require more nuanced control and adaptability? We identify three key challenges for VLMs: 1) *Fine-grained SE(3) pose control*, requiring precise manipulation of objects in 3D space; 2) *Human-like behavioral priors*, enabling VLMs to break down complex, multi-step tasks similarly to how humans approach them; and 3) *Iterative reward shaping*, where VLMs refine their predictions through embodied feedback—the environment changes as a result of the robot's actions.

In this work, we introduce **Iterative Keypoint Reward (IKER)**, a framework that enables Vision-Language Models (VLMs) to generate and refine visually-grounded reward functions, which serve as task specifications for open-world manipulation tasks. Rather than relying on rigid, predefined instructions, our approach allows VLMs to dynamically synthesize task objectives based on real-time observations and language inputs. Specifically, given an RGB-D observation and a free-form language instruction, we first sample keypoints in the scene. Then, leveraging the visual code-generation capabilities of VLMs, we generate Python-based reward functions conditioned on these sampled keypoints. These reward functions, which operate on the spatial relationships between keypoints, define the desired outcomes of the manipulation task, thus acting as flexible and adaptable task specifications. Building upon recent work [14], we make the crucial observation that keypoints capture rich geometric structures inherent to many manipulation tasks. Multiple keypoints can collectively encode the full SE(3) rotations of objects. By using keypoint-based rewards to specify tasks, we enable VLMs to leverage their rich understanding of the world to generate reward functions that reflect human strategies for manipulation. This means the robot is guided to perform actions in ways that humans naturally would, ensuring that the task objectives align closely with human expectations and approaches. For example, in a task requiring regrasping, the reward function can be designed to guide intermediate actions that mirror human problem-solving approaches, such as repositioning objects to facilitate a successful grasp. Furthermore, through iterative refinement based on embodied feedback, VLMs can continuously update and improve the task specification, ensuring that the robot's behavior remains adaptive to the complexities of real-world environments. In this way, reward functions in our framework effectively serve as evolving task specifications.

A key advantage of our approach is its ability to bridge the gap between simulation and reality while maintaining flexible, visually grounded task specifications. We first transfer the 3D meshes of real-world objects into a simulation. Since the reward function depends on VLMs, which perform better on real-world data than simulation data, we generate the reward using real-world observations to ensure contextually grounded task specifications. Then we transfer it to a simulated environment for policy training. This approach leverages the precision of VLMS for real-world observations, while utilizing the scalability of simulation for training. Once optimized, the policy is deployed back into the real world. This real-to-sim-to-real loop enables seamless adaptation from simulation to physical execution. Leveraging this framework, we demonstrate the efficacy of IKER across diverse scenarios involving real-world objects like shoes and books, spanning both prehensile tasks—such as grasping and placing shoes on racks—and non-prehensile tasks like pushing or sliding books to designated locations. We perform quantitative and qualitative evaluations to assess the system's ability to autonomously perform complex, long-horizon tasks. The results showcase human-like capabilities, including multi-step chaining, spontaneous error recovery and updating strategies on-the-fly.

In summary, our contributions are as follows:

- We introduce a visually grounded reward representation IKER, that serves as flexible task specification, enabling robots to tackle complex open-world tasks.
- We demonstrate that IKER possesses the unique advantage of incorporating human-like behavioral priors and an iterative reward shaping process.
- We integrate IKER with a real-to-sim-to-real framework to perform both prehensile and non-prehensile tasks, which is robust in challenging long-horizon tasks.

II. Related Work

VLMs in Robotics. VLMs have appeared as a prominent tool in robotics [9–11, 15–25]. Broadly, existing work in this space can be categorized into two groups: (1) those that leverage VLMs for task specification [11, 15–17], and (2) those that focus on low-level control [16, 18, 19, 26]. Our work aligns with the former, emphasizing task specification.

Several studies have explored task specification by decomposing complex tasks into actionable steps. For example, Ahn *et al.* [15] proposes a method to break down long-horizon tasks into subtasks using LLMs. Belkhale *et al.* [27] introduces language motions as intermediaries between high-level instructions and robotic actions, enabling the policy to learn shared low-level structures. Building on this, Rana *et al.* [26] extends task decomposition to expansive environments like floor plans. Other works, such as [28], employ LLMs to generate task targets that guide RL agents in achieving long-horizon tasks. Unlike most of these works, which require pre-defined models or prior knowledge of the environment and objects, our approach reconstructs the 3D meshes of the scene and objects from RGB-D observations. By not relying

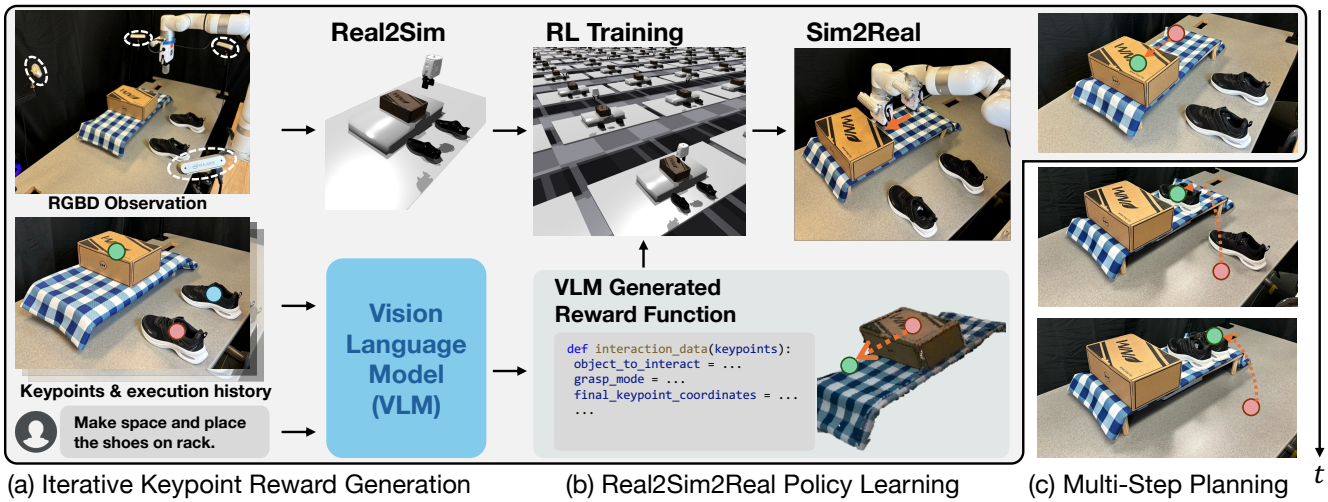

Fig. 2: **Framework Overview.** We first obtain keypoints in the scene. These, combined with a human command and execution history, are processed by a Vision-Language Model (VLM) to generate code that defines the reward function to move the box (a). We transfer the scene into simulation, and the reward function is used to train a policy, which is subsequently executed in the real world (b). This process is repeated across multiple steps to enable long-horizon task execution (c).

on pre-existing models, we increase the adaptability of our approach to unseen scenarios.

In addition to task decomposition, alternative task specification methods such as affordances and value maps have been investigated. For instance, Huang *et al.* [17] generates 3D affordance and constraint maps as objective functions for motion planners, while [14] introduces visually grounded representations for constraints. Likewise, Lit *et al.* [11] leverages VLMs to predict point-based affordances for zero-shot manipulation tasks. Moreover, Zhoe *et al.* [29] integrates VLMs into model predictive control by sampling candidate action sequences and using VLMs to generate video predictions, which are then evaluated to determine optimal actions.

Similar to our work, some approaches explore task specification through reward function generation. Works such as [30–32] employ LLMs to generate reward functions that train robotic policies, but their practical applicability is often limited. In contrast, we demonstrate the practical applicability and robustness of our method by successfully addressing challenging manipulation tasks in real-world settings, underscoring its effectiveness in multi-step tasks.

Real-Sim-Real. Real-to-sim has gained significant attention for its ability to facilitate efficient agent training. Once a scene is transferred to simulation, it can be used for a wide range of tasks, including RL training. Several approaches focus on reconstructing rigid bodies for use in simulation [33–35]. For instance, Kappler et al. [33] introduced a method for reconstructing rigid objects to facilitate grasping. Some works rather focus on reconstructing articulated objects [36–40]. Huang et al.[36] presented methods for reconstructing the occluded shapes of articulated objects. Jiang *et al.* [40] introduced a framework, DITTO, to generate digital twins of articulated objects from real-world interactions. In our work, we utilize BundleSDF [41] to generate object meshes that are transferred to the simulation.

Sim-to-real transfers have shown great performance in a variety of skills, including tabletop manipulation [42, 43], mobile manipulation [44, 45], dynamic manipulation [46], dexterous manipulation [47, 48], and locomotion [49, 50]. However, directly deploying learned policies to physical robots cannot guarantee successful performance due to sim-to-real gap. To bridge sim-to-real gap, researchers has developed many techniques, such as system identification [51–54], domain adaptation [55–57], and domain randomization [49, 58–60]. In our work, we use domain randomization as it does not require any interaction data from real-world during training. It relies entirely on simulation and makes policies robust by exposing them to a wide variety of randomized conditions within simulation. Recently, Torne *et al.* [61] propose RialTo, a complete real-to-sim-to-real loop system.

III. Method

Herein we first formally define Iterative Keypoint Reward (IKER) and discuss how they are automatically synthesized and refined by VLMs by continuously taking in environmental feedback. Then we discuss how IKER serves as a versatile task specification that bridges simulation and reality for manipulation. Our method overview is shown in Figure 2.

A. Iterative Keypoint Reward (IKER)

Given RGB-D observation of the environment and an open-vocabulary language instruction I for a multi-step task, our objective is to obtain N policies $\pi_{i=1}^{N}$ that complete the task. Importantly, the number of policies N is not predefined and can adapt dynamically based on the environment and task progression. For example, Figure 2 considers a multi-step shoe-placing task where the robot first needs to clear space on a rack before sequentially placing and aligning two shoes side by side. In this scenario, the first policy, π_1, must be learned to move a shoe box to make room for the shoes. In this work, we focus on the task specification challenge of how one can *automatically* formulate the reward functions required to train the policies (e.g., in RL framework).

For step i, we denote the RGB observation as O_i. We assume a set of K keypoints $k_{j=1}^{K}$ is given (discussed later in

Sec. III-B), each specifying a 3D position in the task space. Using the keypoints, we aim to obtain a reward function, termed IKER, that maps keypoint positions to a scalar reward $f^{(i)} : \mathbb{R}^{K\times 3} \to \mathbb{R}$. To automatically generate $f^{(i)}$, we use a VLM, GPT-4o [1], to write $f^{(i)}$ as a Python function which may contain arithmetic operations on the keypoints. In order for the VLM to refer to the keypoints, we use visual prompting by overlaying the keypoints on the observation O_i with numerical marks $\{1, \ldots, K\}$ as shown in Figure 2.

Notably, the reward generation process is conditioned on all previous interactions to allow the VLM to either create a new reward function that progresses to the next stage of the task or refine previous ones in case of failures. Namely, the context C_i provided to the VLM at step i includes the original instruction, the RGB observations from previous steps, and the current observation:

$$C_i = \left\{ I, O_1, f^{(1)}, \ldots, O_{i-1}, f^{(i-1)}, O_i \right\}.$$

B. Transferring real-world scene to simulation

After formulating the reward function, we transfer the real-world scene within the workspace boundary to simulation. Poses of manipulable objects are estimated using FoundationPose [62], which allows us to position the corresponding object meshes in the simulation. For example, the shoe box and shoes depicted in Figure 2 are manipulable objects. FoundationPose requires CAD models of the objects, which we generate using BundleSDF [41]. This process involves capturing a video of the object from a set of angles to create a 3D model. For static workspace elements, like the workspace table and shoe rack in Figure 2, we capture a point cloud which is used to create the meshes in the simulation.

We leverage the generated meshes to identify candidate keypoints. For manipulable objects like shoes and books, keypoints are positioned at the object's extremities along its axes and are defined with respect to the corresponding object's center. Keypoints that are too close in the image projection are removed. Conversely, for static objects like shoe racks, which serve as environment, we uniformly distribute keypoints across their surfaces.

C. Train policy in Simulation

We directly control the robot in the end-effector space, which has six degrees of freedom: three prismatic joints for movement along the x, y, and z axes, and three revolute joints for rotation. The gripper fingers remain closed by default, opening only when grasping objects. In simulation, we employ a heuristics-based grasp for faster training.

State Space: The state space for our policy captures the essential information to execute the task. The input is a vector s_t consisting of the gripper's end-effector pose $(\mathbf{p}_e, \mathbf{q}_e) \in \mathbb{R}^7$, the pose of object currently being manipulated $(\mathbf{p}_o, \mathbf{q}_o) \in \mathbb{R}^7$, a set of object keypoints $\mathcal{K}_o = \{\mathbf{k}_{o_1}, \ldots, \mathbf{k}_{o_n}\} \in \mathbb{R}^{3n}$, and their corresponding target positions $\mathcal{K}_t = \{\mathbf{k}_{t_1}, \ldots, \mathbf{k}_{t_n}\} \in \mathbb{R}^{3n}$. $\mathcal{K}_t$ is derived from the reward function f generated by the VLM. Rotations $\mathbf{q}_e$ and $\mathbf{q}_o$ are represented as quaternions. This state space $s_t = (\mathbf{p}_e, \mathbf{q}_e, \mathbf{p}_o, \mathbf{q}_o, \mathcal{K}_o, \mathcal{K}_t)$ captures essential information on objects of interest as well as the goal of the policy.

Action Space: The action space is defined relative to the gripper's current position and orientation. The policy outputs actions $a_t = (\Delta\mathbf{p}_e, \Delta\mathbf{r}_e)$, where $\Delta\mathbf{p}_e \in \mathbb{R}^3$ and $\Delta\mathbf{r}_e \in \mathbb{R}^3$ specifies the changes in translation and rotation respectively.

Domain Randomization: Recognizing the challenges inherent in transferring policies between simulation and the real world, we employ domain randomization to bridge the real-to-sim-to-real gaps. Domain randomization is applied to object properties like friction, mass, center of mass, restitution, compliance, and geometry. We further randomize the object position, the gripper location, as well as the grasp within a range. We found these to be especially crucial for non-prehensile tasks like pushing.

D. Deploy Trained Policy in Real-World

The trained RL policy π_i is then deployed directly in the real-world. Since the policy outputs end-effector pose, we employ inverse kinematics to compute the joint angles at each timestep. The RL policy operates at 10Hz frequency, producing action commands that are then clipped to ensure the end effector remains within the workspace limits. For keypoint tracking, we utilize FoundationPose [62] to estimate the object's pose. These pose estimates are subsequently used to compute the keypoint locations that are defined relative to the objects. We use AnyGrasp [63] to detect grasps in the real-world. The VLM predicts the object to interact with, and the optimal grasp is selected from AnyGrasp detection using back and top views of the object.

IV. Experiments and Analysis

We aim to answer the following research questions: (1) Why is Iterative Keypoint Reward (IKER) a versatile reward representation for training diverse manipulation skills, specifically in the context of harnessing the world knowledge from VLMs? And can we construct a pipeline leveraging IKER for real-to-sim-to-real transfer? (2) Leveraging task-level feedback for replanning, can the pipeline perform multi-step tasks in dynamic environments?

A. Metrics and Baselines

We compare to a human-annotated variant of IKER, where reward functions are human-specified, allowing evaluation without VLM influence.

We also compare our method with using object pose (represented by coordinates for position and quaternions for orientation) as the state representation, which is the conventional approach in RL training [64–68]. In the pose automatic method, the VLM generates a function f that maps the initial poses of the objects to their final poses.

We evaluate our approach across four scenarios: Shoe Place, Shoe Push, Book Push, and Book Reorient. In Shoe Place, the robot picks a shoe from the ground and places it on a rack. In Shoe Push, it pushes one shoe to form a matching pair. In Book Push, it aligns or pushes a book to the table's edge, and in Book Reorient, it repositions a book on a shelf.

Task	Annotated (Human labeled reward)				Automatic (VLM-generated reward)			
	Simulation		Real-World		Simulation		Real-World	
	IKER (Ours)	Pose	IKER (Ours)	Pose	IKER (Ours)	Pose	IKER (Ours)	Pose
Shoe Place	0.945	0.938	0.8	0.9	**0.778**	0.353	**0.7**	0.3
Shoe Push	0.871	0.850	0.7	0.7	**0.716**	0.289	**0.6**	0.2
Stowing Push	0.901	0.914	0.8	0.7	**0.679**	0.374	**0.6**	0.3
Stowing Reorient	0.848	0.859	0.8	0.7	**0.858**	0.265	**0.7**	0.2

TABLE I: **Performance of IKER in simulation and real-world**. IKER, which makes use of visual keypoints, significantly outperforms the conventional pose-based approach, especially when using VLMs to automatically generate reward functions.

Each scenario has 10 start/end configurations. In simulation, we report success rates averaged over 128 environments, with trials considered successful if the average keypoint distance to target is within 5 cm.

B. Simulation Environment, Real-World System and Experiment Objects

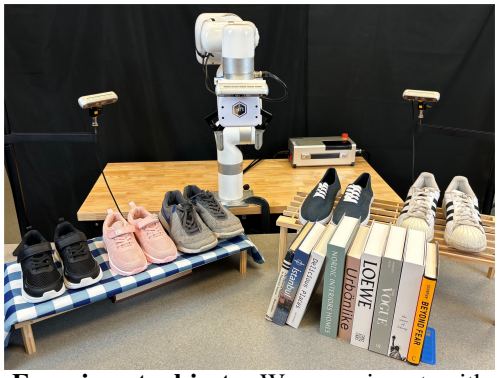

Fig. 3: **Experiment objects.** We experiment with 5 shoe pairs and 2 shoe racks for tasks involving shoe scenarios. We experiment with 9 different books for stowing tasks.

We use IsaacGym [69] to train our policies. In real-world, we conduct experiments on XArm7 with four stationary RealSense cameras. These cameras capture the point clouds, which are used to construct the simulation environment and to provide data for AnyGrasp to predict grasp. Additionally, a wrist-mounted camera is used to capture images, which are then used to query the VLM. Figure 3 shows the objects used in our experiments.

C. Policy Training with Iterative Keypoint Reward

Iterative Keypoint Reward offers a flexible and effective way to represent rewards for training manipulation policies using RL. To validate this, we conduct experiments comparing RL training with keypoints and object pose. Our experiments span four representative tasks, with quantitative results summarized in Table I.

In the annotated method, success rates for shoe placement using IKER and object pose are 0.945 and 0.938, respectively. A similar trend is seen in the shoe push, stowing push, and reorient tasks, where performance differences are minimal. These results suggest that IKER effectively captures diverse behaviors, comparable to object poses, making it a promising alternative for RL policy training.

In the automated method, IKER significantly outperforms object pose representations. For example, in shoe placement, IKER achieves a 0.7 success rate, while object poses reach only 0.3. Similar results are seen across other tasks. Object pose success is limited to simpler scenarios with no pose changes, as VLMs struggle with rotations in SO(3) space. In contrast, keypoints simplify the challenge by requiring VLMs to reason only in Cartesian space, eliminating the need to handle object poses in SE(3) space.

As shown in Table I, there is a slight reduction in success rate from simulation to the real world. For shoe placement, IKER achieves success of 0.945 in simulation and 0.8 in the real world. For shoe push, the success rate drops from 0.871 to 0.850. These results suggest that domain randomization techniques in Section III-C help bridge the sim-to-real gap, but factors like inaccuracies in environment reconstruction, real-world perception errors, and the inability to simulate extreme object dynamics still affect performance.

D. Iterative Replanning for Multi-Step Tasks

We demonstrate the robot's iterative chaining ability with a task involving three sequential actions: pushing a shoe box to create space, then placing a pair of shoes on a rack. Failure in any task leads to failure in the next. This task is more challenging for the VLM, using around 16 keypoints compared to 8 in earlier tasks due to background elements. We test 10 different start and end configurations, iterating through each to assess overall performance.

We compare our method with VoxPoser [17], which employs LLMs to generate code that produces potential fields for motion planning. VoxPoser serves as an ideal baseline because it excels at synthesizing motion plans for a diverse range of manipulation tasks from free-form language instructions. Notably, their plans are open-loop and lack feedback to the VLM for refining specifications at each step. To adapt it to our tasks, we enhanced VoxPoser with two major modifications: (1) VoxPoser used OWL-ViT to find object bounding boxes, but it struggled to distinguish between left and right shoes, so we provided ground-truth object locations. (2) We gave VoxPoser the entire plan, as the original planner struggled with multi-step tasks. This gave VoxPoser an advantage over our method due to access to privileged information.

Figure 5 shows the iterative chaining results. Across the three tasks, our method consistently outperformed VoxPoser. In the first task, we succeeded 6 out of 10 times compared to VoxPoser's 5 successes. For the second task, we had 2 successes while VoxPoser had 1. In the final task, our method succeeded once, whereas VoxPoser failed in all attempts. VoxPoser's failures can be attributed to several factors, such

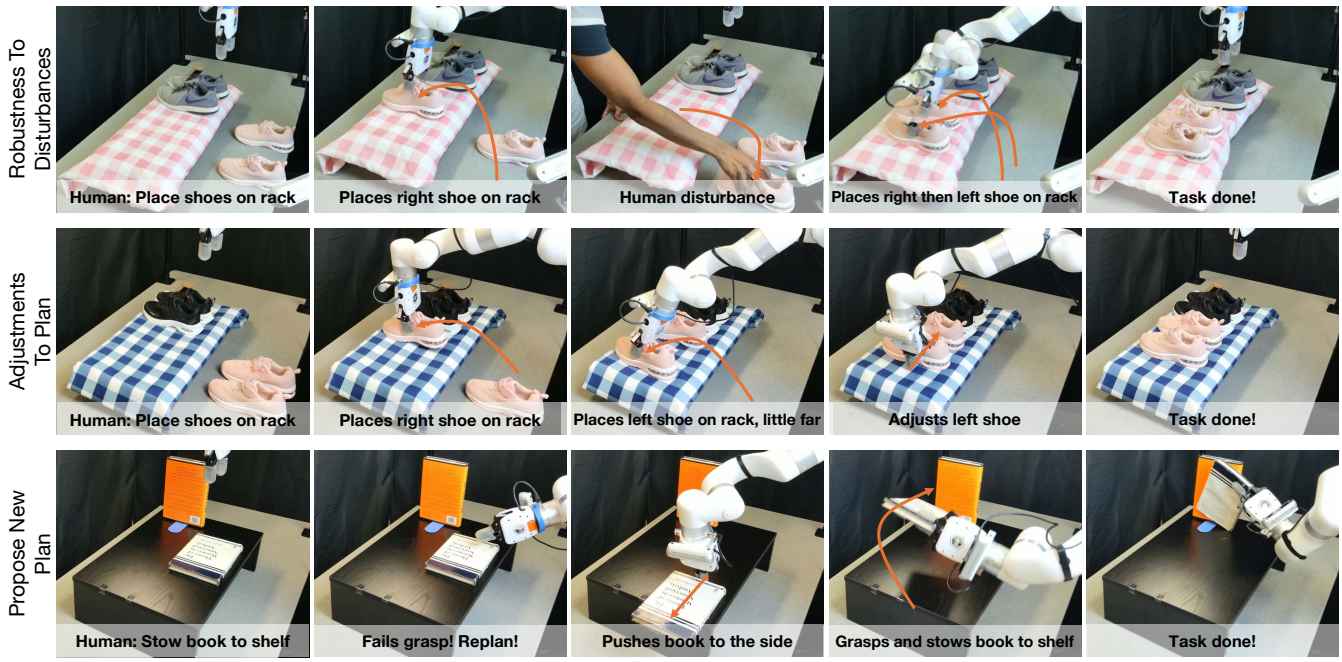

Fig. 4: **Scenarios demonstrating capabilities of our framework.** The framework is robust to disturbances and can adapt in response to unexpected events. Additionally, it can propose new plans when the original ones become infeasible.

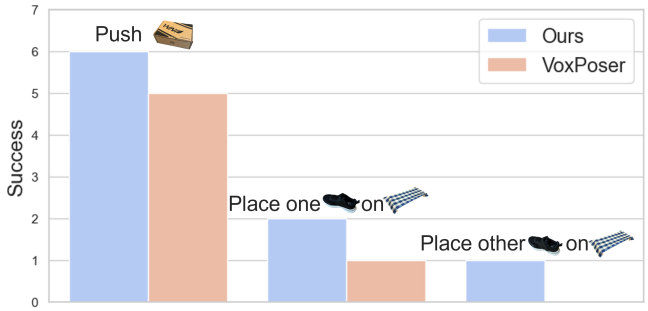

Fig. 5: **Multi-Step Task Chaining Comparison with VoxPoser.** Our framework outperforms VoxPoser at every step of the task sequence.

as pushing the shoe box either too far or not far enough, failed grasping attempts, collisions with the environment during object manipulation, or improper placement of the shoes—resulting in both shoes being stacked on top of each other and subsequently falling.

E. Robustness, Adjusting Plans, and Re-Planning

Unlike previous works that rely on open-loop plans, our approach leverages closed-loop plans, enabling adjustments during execution. This feature gives rise to several capabilities, as demonstrated in Figure 4.

In the first scenario, a human interrupts the robot while it is in the process of placing shoes on the ground. The framework demonstrates resilience by recovering from the interruption. The robot re-grasps the shoe and successfully completes the task by placing both shoes on the rack.

In the second scenario, when the robot attempts to place the left shoe, it detects that the shoe is not positioned close enough to the right shoe. To address this, the VLM predicts a corrective action, suggesting that the robot push the left shoe closer to the right shoe to form a proper pair.

In the third scenario, the robot is tasked with stowing a book on a shelf. However, the initial grasp attempt fails because the book is too large to be handled effectively. In response, the VLM predicts an alternative strategy to complete the task, adjusting the approach to ensure successful placement.

V. Conclusion and Limitations

In this work, we introduced Iterative Keypoint Reward (IKER), a framework that leverages VLMs to generate visually grounded reward functions for robotic manipulation in open-world environments. By using keypoints from RGB-D observations, our approach enables precise SE(3) control and integrates priors from VLMs without relying on rigid instructions. IKER bridges simulation and real-world execution through a real-to-sim-to-real loop, training policies in simulation and deploying them in physical environments. Experiments across diverse tasks demonstrate the framework's ability to handle complex, long-horizon challenges with adaptive strategies and error recovery. This work represents a step toward more intelligent and flexible robots capable of operating effectively in dynamic, real-world settings.

Despite these advancements, our approach has certain limitations. We use a simplified version of real-to-sim transfer, which may not fully capture the complexities of real-world environments. A more comprehensive system identification might be necessary for more precise manipulation. Additionally, while our framework reconstructs multiple objects in the environment, our current implementation does not account for tasks involving complicated multi-object interactions, limiting our evaluation primarily to single-object manipulation at each stage. Lastly, even with GPU-accelerated

simulation, the current simulators require approximately 5 minutes of training per task, which can be time-consuming for some applications.

VI. Acknowledgements

We thank Aditya Prakash, Arjun Gupta, Binghao Huang, Hanxiao Jiang, Kaifeng Zhang, Unnat Jain, and members of UIUC Vision and Robotics Labs for fruitful discussions. This work does not relate to the positions of Shubham Garg and Hooshang Nayyeri at Amazon. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of the sponsors.

References

[1] OpenAI, "Gpt-4 technical report," *arXiv*, 2023.

[2] A. Zeng, A. Wong, S. Welker, K. Choromanski, F. Tombari, A. Purohit, M. Ryoo, V. Sindhwani, J. Lee, V. Vanhoucke *et al.*, "Socratic models: Composing zero-shot multimodal reasoning with language," *arXiv preprint arXiv:2204.00598*, 2022.

[3] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, G. Krueger, and I. Sutskever, "Learning transferable visual models from natural language supervision," 2021. [Online]. Available: https://arxiv.org/abs/2103.00020

[4] C. Jia, Y. Yang, Y. Xia, Y.-T. Chen, Z. Parekh, H. Pham, Q. V. Le, Y. Sung, Z. Li, and T. Duerig, "Scaling up visual and vision-language representation learning with noisy text supervision," 2021. [Online]. Available: https://arxiv.org/abs/2102.05918

[5] J. Li, D. Li, C. Xiong, and S. Hoi, "Blip: Bootstrapping language-image pre-training for unified vision-language understanding and generation," in *International conference on machine learning*. PMLR, 2022, pp. 12 888–12 900.

[6] J. Li, D. Li, S. Savarese, and S. Hoi, "Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models," 2023. [Online]. Available: https://arxiv.org/abs/2301.12597

[7] J.-B. Alayrac, J. Donahue, P. Luc, A. Miech, I. Barr, Y. Hasson, K. Lenc, A. Mensch, K. Millican, M. Reynolds, R. Ring, E. Rutherford, S. Cabi, T. Han, Z. Gong, S. Samangooei, M. Monteiro, J. Menick, S. Borgeaud, A. Brock, A. Nematzadeh, S. Sharifzadeh, M. Binkowski, R. Barreira, O. Vinyals, A. Zisserman, and K. Simonyan, "Flamingo: a visual language model for few-shot learning," 2022. [Online]. Available: https://arxiv.org/abs/2204.14198

[8] J. Yu, Z. Wang, V. Vasudevan, L. Yeung, M. Seyedhosseini, and Y. Wu, "Coca: Contrastive captioners are image-text foundation models," 2022. [Online]. Available: https://arxiv.org/abs/2205.01917

[9] S. Nasiriany, F. Xia, W. Yu, T. Xiao, J. Liang, I. Dasgupta, A. Xie, D. Driess, A. Wahid, Z. Xu *et al.*, "Pivot: Iterative visual prompting elicits actionable knowledge for vlms," *arXiv preprint arXiv:2402.07872*, 2024.

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

[11] F. Liu, K. Fang, P. Abbeel, and S. Levine, "Moka: Open-vocabulary robotic manipulation through mark-based visual prompting," *arXiv preprint arXiv:2403.03174*, 2024.

[12] D. Venuto, S. N. Islam, M. Klissarov, D. Precup, S. Yang, and A. Anand, "Code as reward: Empowering reinforcement learning with vlms," *arXiv preprint arXiv:2402.04764*, 2024.

[13] W. Yuan, J. Duan, V. Blukis, W. Pumacay, R. Krishna, A. Murali, A. Mousavian, and D. Fox, "Robopoint: A vision-language model for spatial affordance prediction for robotics," *arXiv preprint arXiv:2406.10721*, 2024.

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

[15] M. Ahn, A. Brohan, N. Brown, Y. Chebotar, O. Cortes, B. David, C. Finn, C. Fu, K. Gopalakrishnan, K. Hausman *et al.*, "Do as i can, not as i say: Grounding language in robotic affordances," *arXiv preprint arXiv:2204.01691*, 2022.

[16] J. Liang, W. Huang, F. Xia, P. Xu, K. Hausman, B. Ichter, P. Florence, and A. Zeng, "Code as policies: Language model programs for embodied control," in *arXiv preprint arXiv:2209.07753*, 2022.

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

[18] A. Brohan, N. Brown, J. Carbajal, Y. Chebotar, J. Dabis, C. Finn, K. Gopalakrishnan, K. Hausman, A. Herzog, J. Hsu *et al.*, "Rt-1: Robotics transformer for real-world control at scale," *arXiv preprint arXiv:2212.06817*, 2022.

[19] A. Brohan, N. Brown, J. Carbajal, Y. Chebotar, X. Chen, K. Choromanski, T. Ding, D. Driess, A. Dubey, C. Finn *et al.*, "Rt-2: Vision-language-action models transfer web knowledge to robotic control," *arXiv preprint arXiv:2307.15818*, 2023.

[20] O. M. Team, D. Ghosh, H. Walke, K. Pertsch, K. Black, O. Mees, S. Dasari, J. Hejna, T. Kreiman, C. Xu *et al.*, "Octo: An open-source generalist robot policy," *arXiv preprint arXiv:2405.12213*, 2024.

[21] S. Huang, Z. Jiang, H. Dong, Y. Qiao, P. Gao, and H. Li, "Instruct2act: Mapping multi-modality instructions to robotic actions with large language model," *arXiv preprint arXiv:2305.11176*, 2023.

[22] M. Xu, P. Huang, W. Yu, S. Liu, X. Zhang, Y. Niu, T. Zhang, F. Xia, J. Tan, and D. Zhao, "Creative robot tool use with large language models," *arXiv preprint arXiv:2310.13065*, 2023.

[23] H. Zhou, M. Ding, W. Peng, M. Tomizuka, L. Shao, and C. Gan, "Generalizable long-horizon manipulations with large language models," *arXiv preprint arXiv:2310.02264*, 2023.

[24] N. Di Palo and E. Johns, "Keypoint action tokens enable in-context imitation learning in robotics," *arXiv preprint arXiv:2403.19578*, 2024.

[25] F. Zeng, W. Gan, Y. Wang, N. Liu, and P. S. Yu, "Large language models for robotics: A survey," *arXiv preprint arXiv:2311.07226*, 2023.

[26] K. Rana, J. Haviland, S. Garg, J. Abou-Chakra, I. Reid, and N. Suenderhauf, "Sayplan: Grounding large language models using 3d scene graphs for scalable task planning," *arXiv preprint arXiv:2307.06135*, 2023.

[27] S. Belkhale, T. Ding, T. Xiao, P. Sermanet, Q. Vuong, J. Tompson, Y. Chebotar, D. Dwibedi, and D. Sadigh, "Rt-h: Action hierarchies using language," in *https://arxiv.org/abs/2403.01823*, 2024.

[28] M. Dalal, T. Chiruvolu, D. Chaplot, and R. Salakhutdinov, "Plan-seq-learn: Language model guided rl for solving long horizon robotics tasks," *arXiv preprint arXiv:2405.01534*, 2024.

[29] W. Zhao, J. Chen, Z. Meng, D. Mao, R. Song, and W. Zhang, "Vlmpc: Vision-language model predictive control for robotic manipulation," in *Robotics: Science and Systems*, 2024.

[30] W. Yu, N. Gileadi, C. Fu, S. Kirmani, K.-H. Lee, M. G. Arenas, H.-T. L. Chiang, T. Erez, L. Hasenclever, J. Humplik *et al.*, "Language to rewards for robotic skill synthesis," *arXiv preprint arXiv:2306.08647*, 2023.

[31] Y. J. Ma, W. Liang, G. Wang, D.-A. Huang, O. Bastani, D. Jayaraman, Y. Zhu, L. Fan, and A. Anandkumar, "Eureka: Human-level reward design via coding large language models," *arXiv preprint arXiv: Arxiv-2310.12931*, 2023.

[32] Y. J. Ma, W. Liang, H.-J. Wang, S. Wang, Y. Zhu, L. Fan, O. Bastani, and D. Jayaraman, "Dreureka: Language model guided sim-to-real transfer," 2024. [Online]. Available: https://arxiv.org/abs/2406.01967

[33] D. Kappler, F. Meier, J. Issac, J. Mainprice, C. G. Cifuentes, M. Wüthrich, V. Berenz, S. Schaal, N. Ratliff, and J. Bohg, "Real-time perception meets reactive motion generation," *IEEE Robotics and Automation Letters*, vol. 3, no. 3, pp. 1864–1871, 2018.

[34] B. Wen, W. Lian, K. Bekris, and S. Schaal, "Catgrasp: Learning category-level task-relevant grasping in clutter from simulation," in *2022 International Conference on Robotics and Automation (ICRA)*. IEEE, 2022, pp. 6401–6408.

[35] ——, "You only demonstrate once: Category-level manipulation from single visual demonstration," *arXiv preprint arXiv:2201.12716*, 2022.

[36] X. Huang, I. Walker, and S. Birchfield, "Occlusion-aware reconstruction and manipulation of 3d articulated objects," in *2012 IEEE international conference on robotics and automation*. IEEE, 2012, pp. 1365–1371.

[37] X. Li, H. Wang, L. Yi, L. J. Guibas, A. L. Abbott, and S. Song, "Category-level articulated object pose estimation," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 3706–3715.

[38] X. Wang, B. Zhou, Y. Shi, X. Chen, Q. Zhao, and K. Xu, "Shape2motion: Joint analysis of motion parts and attributes from 3d

shapes," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 8876–8884.

[39] J. Mu, W. Qiu, A. Kortylewski, A. Yuille, N. Vasconcelos, and X. Wang, "A-sdf: Learning disentangled signed distance functions for articulated shape representation," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 13 001–13 011.

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

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

[42] M. Shridhar, L. Manuelli, and D. Fox, "Cliport: What and where pathways for robotic manipulation," *arXiv preprint arXiv: Arxiv-2109.12098*, 2021.

[43] Y. Jiang, C. Wang, R. Zhang, J. Wu, and L. Fei-Fei, "Transic: Sim-to-real policy transfer by learning from online correction," 2024. [Online]. Available: https://arxiv.org/abs/2405.10315

[44] J. Gu, D. S. Chaplot, H. Su, and J. Malik, "Multi-skill mobile manipulation for object rearrangement," *arXiv preprint arXiv: Arxiv-2209.02778*, 2022.

[45] S. Yenamandra, A. Ramachandran, K. Yadav, A. Wang, M. Khanna, T. Gervet, T.-Y. Yang, V. Jain, A. W. Clegg, J. Turner, Z. Kira, M. Savva, A. Chang, D. S. Chaplot, D. Batra, R. Mottaghi, Y. Bisk, and C. Paxton, "Homerobot: Open-vocabulary mobile manipulation," *arXiv preprint arXiv: Arxiv-2306.11565*, 2023.

[46] B. Huang, Y. Chen, T. Wang, Y. Qin, Y. Yang, N. Atanasov, and X. Wang, "Dynamic handover: Throw and catch with bimanual hands," *arXiv preprint arXiv:2309.05655*, 2023.

[47] Y. Chen, C. Wang, L. Fei-Fei, and C. K. Liu, "Sequential dexterity: Chaining dexterous policies for long-horizon manipulation," *arXiv preprint arXiv:2309.00987*, 2023.

[48] Y. Qin, B. Huang, Z.-H. Yin, H. Su, and X. Wang, "Dexpoint: Generalizable point cloud reinforcement learning for sim-to-real dexterous manipulation," 2022. [Online]. Available: https://arxiv.org/abs/2211.09423

[49] A. Kumar, Z. Fu, D. Pathak, and J. Malik, "RMA: rapid motor adaptation for legged robots," in *Robotics: Science and Systems XVII, Virtual Event, July 12-16, 2021*, D. A. Shell, M. Toussaint, and M. A. Hsieh, Eds., 2021. [Online]. Available: https://doi.org/10.15607/RSS.2021.XVII.011

[50] T. He, C. Zhang, W. Xiao, G. He, C. Liu, and G. Shi, "Agile but safe: Learning collision-free high-speed legged locomotion," 2024. [Online]. Available: https://arxiv.org/abs/2401.17583

[51] K. J. Åström and P. Eykhoff, "System identification—a survey," *Automatica*, vol. 7, no. 2, pp. 123–162, 1971.

[52] J. Tan, T. Zhang, E. Coumans, A. Iscen, Y. Bai, D. Hafner, S. Bohez, and V. Vanhoucke, "Sim-to-real: Learning agile locomotion for quadruped robots," *arXiv preprint arXiv: Arxiv-1804.10332*, 2018.

[53] P. Chang and T. Padir, "Sim2real2sim: Bridging the gap between simulation and real-world in flexible object manipulation," *arXiv preprint arXiv: Arxiv-2002.02538*, 2020.

[54] V. Lim, H. Huang, L. Y. Chen, J. Wang, J. Ichnowski, D. Seita, M. Laskey, and K. Goldberg, "Planar robot casting with real2sim2real self-supervised learning," *arXiv preprint arXiv: Arxiv-2111.04814*, 2021.

[55] K. Bousmalis, A. Irpan, P. Wohlhart, Y. Bai, M. Kelcey, M. Kalakrishnan, L. Downs, J. Ibarz, P. Pastor, K. Konolige *et al.*, "Using simulation and domain adaptation to improve efficiency of deep robotic grasping," in *2018 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2018, pp. 4243–4250.

[56] K. Arndt, M. Hazara, A. Ghadirzadeh, and V. Kyrki, "Meta reinforcement learning for sim-to-real domain adaptation," 2019. [Online]. Available: https://arxiv.org/abs/1909.12906

[57] K. Rao, C. Harris, A. Irpan, S. Levine, J. Ibarz, and M. Khansari, "Rl-cyclegan: Reinforcement learning aware simulation-to-real," 2020. [Online]. Available: https://arxiv.org/abs/2006.09001

[58] OpenAI, I. Akkaya, M. Andrychowicz, M. Chociej, M. Litwin, B. McGrew, A. Petron, A. Paino, M. Plappert, G. Powell, R. Ribas, J. Schneider, N. Tezak, J. Tworek, P. Welinder, L. Weng, Q. Yuan, W. Zaremba, and L. Zhang, "Solving rubik's cube with a robot hand," *arXiv preprint arXiv: Arxiv-1910.07113*, 2019.

[59] J. Tobin, R. Fong, A. Ray, J. Schneider, W. Zaremba, and P. Abbeel, "Domain randomization for transferring deep neural networks from simulation to the real world," in *2017 IEEE/RSJ international conference on intelligent robots and systems (IROS)*. IEEE, 2017, pp. 23–30.

[60] R. Antonova, F. Ramos, R. Possas, and D. Fox, "Bayessimig: Scalable parameter inference for adaptive domain randomization with isaacgym," *arXiv preprint arXiv:2107.04527*, 2021.

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

[62] B. Wen, W. Yang, J. Kautz, and S. Birchfield, "Foundationpose: Unified 6d pose estimation and tracking of novel objects," *arXiv preprint arXiv:2312.08344*, 2023.

[63] H.-S. Fang, C. Wang, H. Fang, M. Gou, J. Liu, H. Yan, W. Liu, Y. Xie, and C. Lu, "Anygrasp: Robust and efficient grasp perception in spatial and temporal domains," *IEEE Transactions on Robotics*, 2023.

[64] S. Gu, E. Holly, T. Lillicrap, and S. Levine, "Deep reinforcement learning for robotic manipulation with asynchronous off-policy updates," in *2017 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2017, pp. 3389–3396.

[65] I. Popov, N. Heess, T. Lillicrap, R. Hafner, G. Barth-Maron, M. Vecerik, T. Lampe, Y. Tassa, T. Erez, and M. Riedmiller, "Data-efficient deep reinforcement learning for dexterous manipulation," *arXiv preprint arXiv:1704.03073*, 2017.

[66] M. Vecerik, T. Hester, J. Scholz, F. Wang, O. Pietquin, B. Piot, N. Heess, T. Rothörl, T. Lampe, and M. Riedmiller, "Leveraging demonstrations for deep reinforcement learning on robotics problems with sparse rewards," *arXiv preprint arXiv:1707.08817*, 2017.

[67] A. Rajeswaran, V. Kumar, A. Gupta, G. Vezzani, J. Schulman, E. Todorov, and S. Levine, "Learning complex dexterous manipulation with deep reinforcement learning and demonstrations," *arXiv preprint arXiv:1709.10087*, 2017.

[68] H. Qi, A. Kumar, R. Calandra, Y. Ma, and J. Malik, "In-hand object rotation via rapid motor adaptation," in *Conference on Robot Learning*. PMLR, 2023, pp. 1722–1732.

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