DISPER: Disentangled Perception for Generalizable Post-Training

Zhuo Cao[1], Yingxi Lu[1], Xuanyi Xie[1]

***Abstract*— We present our progress on generalizable post-training, especially for geometry. The key idea is to disentangle different perception modules in both architecture and training. Our current results show great potential on the architecture and the proper post-training pattern is still under exploration. Our project is open-sourced on Github.**

I. INTRODUCTION

Policies trained in simulation can acquire strong geometry-level generalization by being exposed to diverse object shapes, layouts, and environmental variations. Such simulation-trained policies often exhibit robust geometric reasoning, yet their deployment in the real world remains challenging due to the persistent sim-to-real dynamics gap. Differences in physical properties, sensing noise, and unmodeled dynamics can substantially degrade real-world performance, even when geometric generalization is well preserved in simulation [1].

A common strategy to mitigate this gap is *real-world post-training*, where a pretrained policy is further adapted using real-world data. Prior work frequently adopts reinforcement learning (RL) algorithms for this purpose, leveraging online interaction to improve robustness and task success. However, naïvely fine-tuning the entire policy during post-training—regardless of the specific optimization method—can lead to overfitting to a limited set of real-world experiences, thereby eroding the geometry-level generalization learned during simulation pretraining [2].

Motivated by this tension, we focus on real-world post-training as the problem setting, while adopting imitation-based methods such as DAgger as a lightweight and stable alternative to full-scale real-world RL. Our central hypothesis is that the loss of geometric generalization during post-training is not primarily caused by the choice of optimization algorithm, but rather by representational entanglement within the policy. Specifically, when perception and control are tightly coupled, post-training updates intended to correct dynamics mismatch can inadvertently alter geometric reasoning.

To address this issue, we propose to disentangle the policy's perception and behavior components, enabling selective optimization during post-training. Our key intuition is that geometric perception should remain stable—preserving the ability to generalize across shapes and configurations—while only behavior-relevant components are adapted to real-world dynamics. By maintaining this disentangled structure, we aim to improve real-world post-training outcomes without sacrificing geometry-level generalization.

[1]Institute for Interdisciplinary Information Sciences, Tsinghua University

II. RELATED WORKS

A. Real-World Post-Training

Recent progress in robotic manipulation has shown that post-training policies directly on real hardware can substantially improve task success rates and robustness. Many existing approaches adopt reinforcement learning as the primary post-training mechanism. For example, Luo et al. propose HiL-Serl [2], which applies a Reinforcement Learning with Prior Data (RLPD) pipeline to adapt policies in the real world, achieving near-perfect performance on a range of challenging manipulation tasks. Similarly, RL-100 introduced by Lei et al. [3] presents a staged post-training pipeline—imitation learning $\rightarrow$ offline RL $\rightarrow$ online RL—executed entirely on real robotic systems, resulting in high reliability across diverse tasks. In a complementary direction, Josifovski et al. propose Safe Continual Domain Adaptation (SCDA) [4], which integrates safe RL and continual learning to adapt simulation-trained policies while mitigating catastrophic forgetting.

While these methods demonstrate impressive real-world performance, they commonly report reduced generalization under geometric variation. In particular, policies adapted to specific object instances or configurations often exhibit noticeable performance degradation when evaluated on novel geometries, indicating limited geometry-level generalizability after post-training.

Our work is motivated by the hypothesis that this limitation arises from an implicit entanglement between **task behavior** and **geometry perception** within the policy network. During real-world post-training on a specific object geometry, both components are jointly optimized. While updating task behavior is necessary to correct for dynamics mismatch and improve robustness, simultaneously adapting geometry perception can induce overfitting to the observed geometries, thereby degrading generalization to unseen shapes. To address this challenge, we aim to disentangle geometry perception from task behavior, enabling effective real-world post-training while preserving geometry-level generalizability.

B. Perception Disentanglement in Visuomotor Policy

A growing body of work in visuomotor learning explores disentangled representations to factorize distinct sources of variation in visual observations, with the goal of improving robustness and generalization. SEAR, proposed by Gmelin et al. [5], learns separate representations for the *agent* and the *environment* using segmentation masks and auxiliary supervision. Similarly, DEAR introduced by Pore et al. [6] disentangles agent-centric and world-centric features without reconstructing the full observation, leading to improved sample efficiency in visually complex environments.

In the context of object-centric manipulation, DOCIR by Emukpere et al. [7] proposes distinct latent streams for the robot, the object of interest, and surrounding obstacles, and demonstrates strong zero-shot generalization to unseen objects and real-world scenes. Collectively, these approaches aim to isolate semantically meaningful latent factors that facilitate policy learning.

However, in most existing methods, disentangled perceptual representations are still co-adapted with the policy during training or real-world post-training. In contrast, our approach explicitly constrains geometry perception during post-training. By freezing the geometry encoder, we preserve stable geometric representations learned in simulation, while allowing task-relevant perceptual and behavioral components to adapt. This design targets real-world post-training specifically, enabling adaptation to real dynamics without sacrificing geometry-level generalizability.

III. METHOD

As our reflection, one key to the reason why previous post-training methods usually lead to loss of generalizability, especially on geometry, is that the policy components related to geometry perception and behavior planning are coupled together. Therefore, during real-world post-training, this coupling easily causes the co-optimization of both the planning part and geometry perception, thus over-fitting to the scenes the policy is fine-tuned on.

We propose a natural method to try avoiding such issues – disentangling geometry perception and planning modules during both pre-training and post-training. The geometry module should be optionally congealable for further disentanglement.

A. Data Collection

Currently we focus on precise grasp of tiny pegs whose diameter is within 0.5∼1.0cm, which requires very high precision since little displacement could lead to failure.

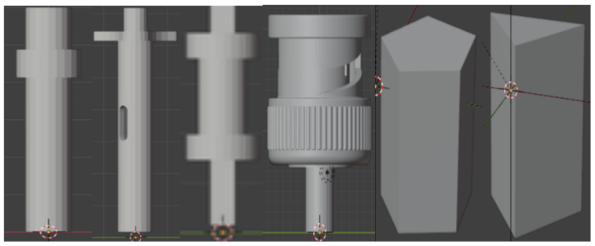


Fig. 1: **Asset Examples** The leftmost four pegs are from established datasets and the last two are generated.

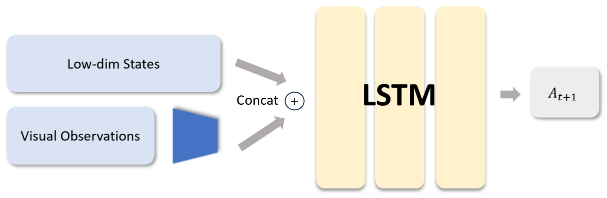
As Figure 1 shows, we have a wide range of peg assets from: *a.* Established datasets including AutoMate [8], ManipulationNet [9], etc. These assets possess high diversity in geometry. *b.* Controlled generation using Blender [10] scripts. We have generated different polygons with a variety of scales.

Since we are doing imitation learning under sim2real setting, we have to collect trajectories in simulator. In our work, we basically leverage MimicGen [11] in NVIDIA Isaac Lab [12]. It inputs a little number of demonstrations from teleoperation in simulator and then automatically synthesizes huge datasets. However, we find that the teleoperated trajectories in simulation have very low quality especially for precise grasp since human operator has to adjust multiple times with the lack of perception, which leads to bad success rate of both synthesis process and learned policy.

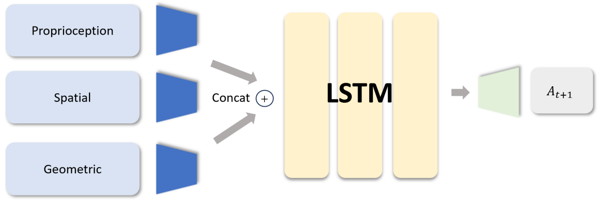
To solve this issue, we use manual motion planning to collect those demonstrations, that is, tracking manually calibrated key frames linearly with default inverse kinematics solver. The procedure is not too heavy since we only need 10∼15 demonstrations for each asset. It turns out significantly helpful to improve the quality of collected data.

B. Pre-Training

We train our base policy using BC-RNN algorithm upon RoboMimic [13] framework. Figure 2a illustrates its default architecture, which inputs all low-dimensional states and encoded visual perceptions concatenated into a single vector, and go through a mutli-layer LSTM [14] with historic inputs to get the final output action.



(a) Original Architecture



(b) New Architecture

Fig. 2: **Policy Architectures** Compared with the original architecture, the key modification is to create separated encoding groups for different meaning types of information.

To implement our idea, we have to modify the architecture with the motivation that the encoding process should be grouped by information instead of modalities and separated with each other. As Figure 2b demonstrates, our new architecture takes 3 encoding groups:

- **Proprioception** Robot states including joint angels, end-effector pose and gripper value could be directly sent to the policy.
- **Spatial** The relative spatial information is critical for the policy to navigation to the object target. For state-based policy, it could be simply the 6D delta pose to

the peg; For vision-based version, it should be certain latent vector extracted from input images.

- **Geometric** The geometric information helps the policy to decide the proper pose to stably grasp a specific peg. It should be ignored for training specialists. When it comes to generalist, the state-based version just requires a one-hot vector to indicate which asset is being used and for vision-based policy, again, it should be certain latent vector extracted from input images.

With such an architecture, all direct inputs should be low-dimensional vectors. Therefore, we need to pre-encode input images into certain latent vectors, which will be discussed in Section III-C.

C. Vision Encoding

As mentioned in Section III-B, we need vision encoders that can disentangle spatial and geometric information in images. We train such encoders via contrastive learning, specifically, based on SimCLR [15].

The core design of SimCLR [15] is positive and negative sampling. Briefly speaking, the learning process aims to maximize the representation agreement of positive samples and minimize that of negative samples. Formally, it uses InfoNCE loss described as:

$$\mathscr{L} = -\frac{\sum_i \exp(\mathrm{sim}(\mathbf{p}_{i,1}, \mathbf{p}_{i,2})/\tau)}{\sum_i \exp(\mathrm{sim}(\mathbf{p}_{i,1}, \mathbf{p}_{i,2})/\tau) + \sum_j \exp(\mathrm{sim}(\mathbf{n}_{j,1}, \mathbf{n}_{j,2})/\tau)} \tag{1}$$

where $\mathbf{p}, \mathbf{n}$ are positive and negative samples respectively, sim is the agreement metric (usually the cosine similarity) and τ is the temperature hyper-parameter. The original sample strategy is to use two images randomly transformed from the same image as a positive pair and images transformed from different images as negative samples.

In our case, clearly, we need to train 2 different encoders with 2 sampling strategies accordingly.

For geometric encoder, we can simply sample images from the trajectories of the same asset as positive samples and different assets as negative samples.

For spatial encoder, it turns out not as simple as geometry since spatial label, i.e. delta pose, is continuous and cannot be sampled trivially. Essentially, we want to sample images according to the distance relation of images and this is actually another form of agreement measurement. Based on Equation (2), we add weights to assign different attentions for samples considering their spatial relations, which results in our Weighted InfoNCE loss:

$$\mathscr{L} = -\frac{\sum_i \exp(||\mathbf{d}_{i,1} - \mathbf{d}_{i,2}||_2/\sigma) \cdot \exp(\mathrm{sim}(\mathbf{p}_{i,1}, \mathbf{p}_{i,2})/\tau)}{\sum_i \exp(\mathrm{sim}(\mathbf{p}_{i,1}, \mathbf{p}_{i,2})/\tau)} \tag{2}$$

where $\mathbf{d}$ is the delta pose label and σ is the temperature hyper-parameter for weights. We no long distinguish positive and negative samples explicitly and instead, softly adjust the update attentions for different sample pairs through re-weighting.

D. Post-Training

With our imitation learning setting, it is natural to adopt DAgger [16] as the post-training technique. In simulation, we can simply do DAgger [16] with state-based specialists. In real-world deployment, HG-DAgger [17] is preferred by replacing specialists with human teleoperator since there is no high-quality specialist policy any more.

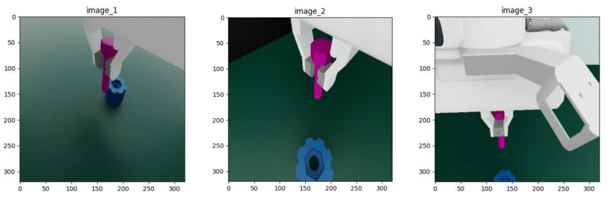
Shown in Figure 3, we introduce the idea of module freezing during post-training. The geometric encoder group is optionally frozen to potentially avoid the loss of geometry generalizability after post-training.

IV. EXPERIMENTS

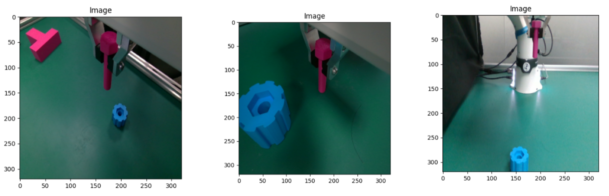
We have done verification of our method in state-space and is working on the vision-based pipeline.

A. Environment Setup

Our embodiment is a 7-DoF Franka Research 3 robot arm with 2 wrist cameras (Intel RealSense D405) and 1 side camera (Intel RealSense D435i). Figure 4 shows the camera views of both the real-world setup and our calibrated ones in simulation.



(a) Camera Views in Simulation



(b) Camera Views in Real-World

Fig. 4: **Camera Views of Our Setup**

For training and testing assets, we have collected datasets for 8 different pegs from the sources introduced in Section III-A. The domain ranges we cover during both data collection and testing are listed in Table I. Besides the spatial and geometric variation of pegs, we also apply randomization for initial robot pose and scene lights.

(cm)	**Coordinate**	**Scale**
x	[15.0, 35.0]	[0.5, 2.0]
y	[-10.0, 10.0]	[0.5, 2.0]
z	N/A	[6.0, 10.0]

TABLE I: **Domain Ranges of Assets**

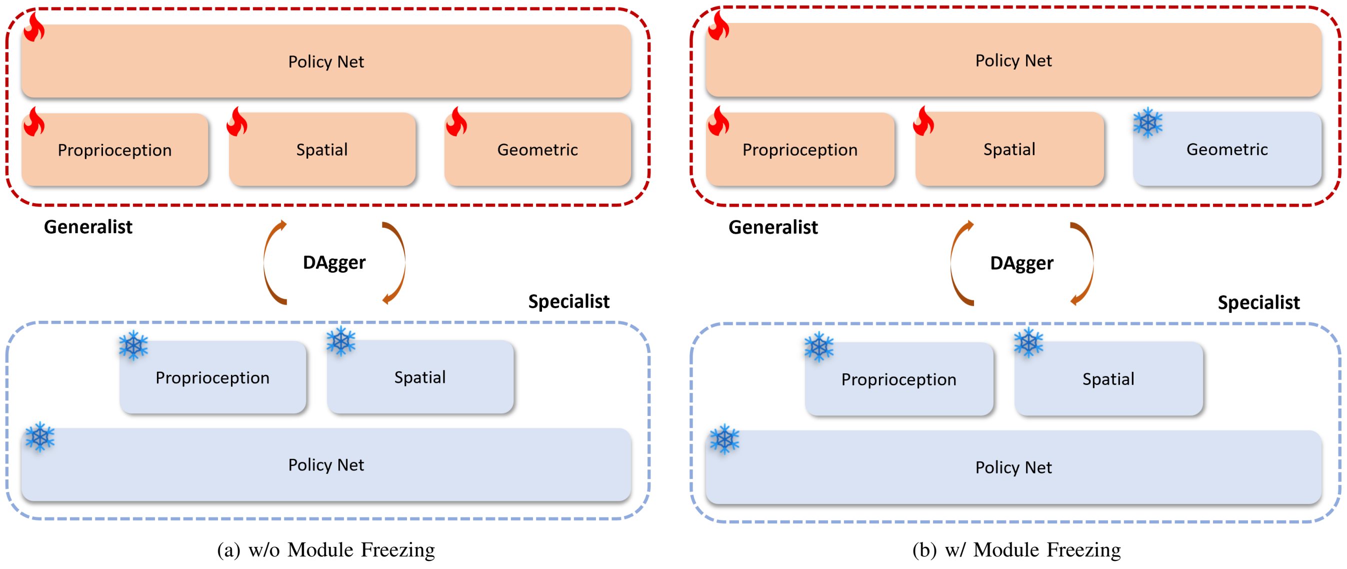


Fig. 3: **Post-Training Scheme** We do post-training in two patterns considering the module freezing idea. Here we optionally freeze the geometric encoder group.

B. *Architecture Comparison*

Asset ID		1	2	3
Original Arch.	**Specialist**	88%	56%	62%
	Generalist	97%	96%	91%
New Arch.	**Specialist**	**100%**	**100%**	**100%**
	Generalist	**99%**	**99%**	**99%**

TABLE II: **Success Rate Comparison of Original and New Architecture** All specialists are trained with 3k trajectories and generalists are trained with 3k×3 trajectories.

Shown in Table II, as a simple validation on 3 assets, our new architecture consistently outperforms the original one for both specialists and generalists. This indicates that the design to re-arrange encoders into groups via information types already works as an improvement for base policy.

C. *Post-Training Performance*

For real-world post-training, usually one can fine-tune policy on only few cases due to effort limit. In order to verify our method properly, we run DAgger [16] in simulation also on a single asset, with the corresponding specialist as the teacher policy.

Table III and Figure 5 reveal the fact that after post-training with DAgger [16] there will be over-fitting to different degrees without surprise.

However, freezing the geometric part does not seem to help overcome this issue further. This is probably because there is still some entanglement inside the following policy net. We plan to test more novel architectures that potentially provide further decoupling, for instance, the one demonstrated in Figure 6.

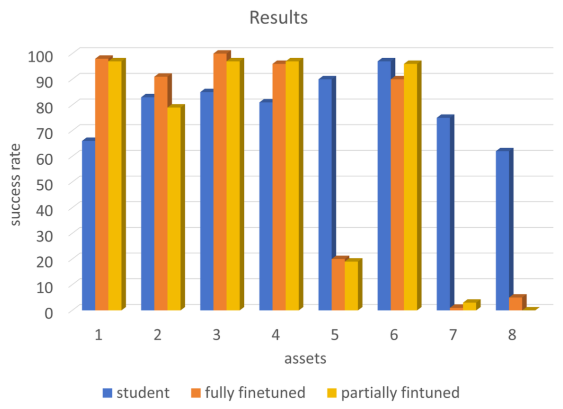


Fig. 5: **Post-Training Results** Student policy is Generalist II and fine-tuned on asset 1.

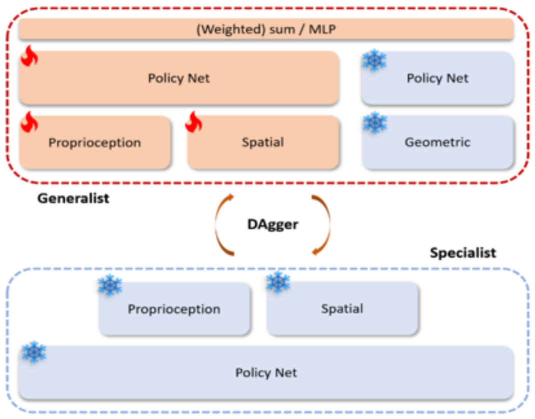
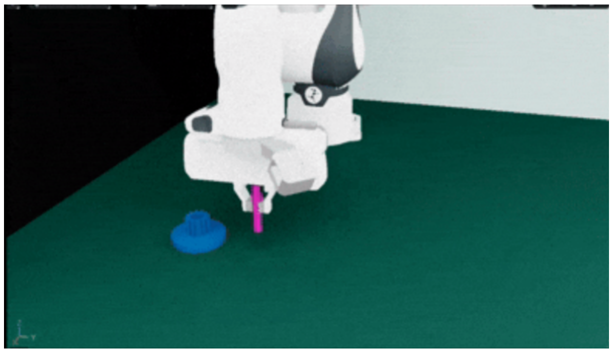


Fig. 6: **Architecture Example** Hopefully this design would further disentangle perceptions compared with the current one.

	1	2	3	4	5	6	7	8
Specialist	**100%**	84%	100%	100%	100%	72%	93%	89%
Generalist I (5000epoch)	**99%**	99%	99%	99%	100%	100%	99%	90%
Generalist II (1200epoch)	**66%**	83%	85%	81%	90%	97%	75%	62%
Fine-tuned Generalist II (full)	**98%**	91%	100%	96%	20%	99%	1%	5%
Fine-tuned Generalist II (geometry frozen)	**97%**	79%	97%	97%	19%	96%	3%	0%

TABLE III: **Performance Comparison** We choose to fine-tune Generalist II which is trained with less epochs since Generalist I with more epochs already achieves high success rates on most assets. To align with the real-world post-training, we only do DAgger [16] on asset 1.


(a) Roll out in Simulation

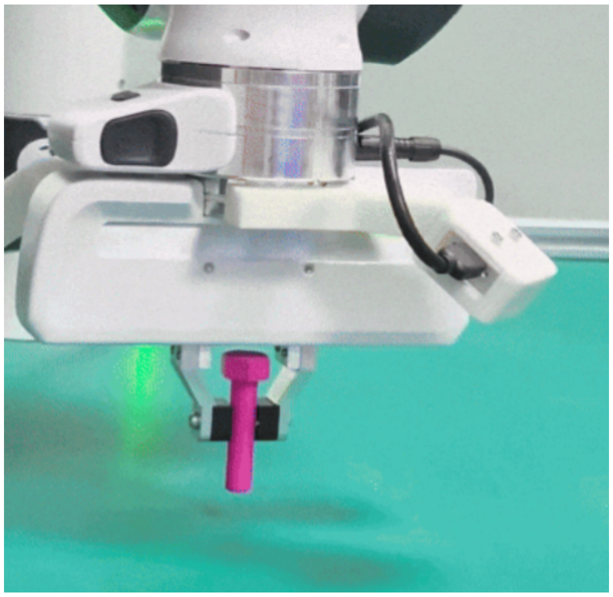

(b) Roll out in Real-World

Fig. 7: **Roll-out Setup in Both Simulation and Real-World**

D. *Vision-Based Policy and Real-World Progress*

We have trained vision-based policy which can achieve ∼45% success rate currently. Real-world HG-DAgger [17] is still on-going since it costs much human labor to collect data and converge. Figure 7 shows our setup in both simulation and real-world.

V. CONCLUSIONS

In this work, we investigated generalizable post-training for robotic manipulation with an emphasis on preserving geometry-level generalization. Through systematic architectural and post-training analysis, we derive several key lessons and insights.

First, our results show that *structuring policy representations by information type*—explicitly separating proprioceptive, spatial, and geometric inputs—consistently improves base policy performance for both specialist and generalist settings. This suggests that architectural inductive bias alone can significantly enhance learning efficiency and stability, even before considering real-world adaptation.

Second, while disentangling perception modules and freezing the geometry encoder during post-training are intuitively appealing, our experiments indicate that these mechanisms alone provide limited gains in preserving geometry-level generalization. In particular, overfitting under post-training persists even when the geometric encoder is frozen, implying that the primary source of coupling between task behavior and geometry perception does not reside solely in the perceptual encoders.

These findings lead to a new insight: *the entanglement responsible for geometry overfitting is likely embedded within the downstream policy network itself.* As a result, effective preservation of geometric generalization requires deeper architectural decoupling beyond perception-level disentanglement. Based on this insight, an important direction for future work is to redesign policy architectures that explicitly decouple task behavior learning from geometry-dependent reasoning at the decision-making level. Such architectural separation may enable real-world post-training to correct dynamics mismatch while fundamentally preventing geometry-specific overfitting, paving the way toward more robust and generalizable sim-to-real reinforcement learning systems.

REFERENCES

[1] L. Da, J. Turnau, T. P. Kutralingam, A. Velasquez, P. Shakarian, and H. Wei, "A survey of sim-to-real methods in rl: Progress, prospects and challenges with foundation models," 2025.

[2] J. Luo, C. Xu, J. Wu, and S. Levine, "Precise and dexterous robotic manipulation via human-in-the-loop reinforcement learning," 2025.

[3] K. Lei, H. Li, D. Yu, Z. Wei, L. Guo, Z. Jiang, Z. Wang, S. Liang, and H. Xu, "Rl-100: Performant robotic manipulation with real-world reinforcement learning," 2025.

[4] J. Josifovski, S. Gu, M. Malmir, H. Huang, S. Auddy, N. Navarro-Guerrero, C. Spanos, and A. Knoll, "Safe continual domain adaptation after sim2real transfer of reinforcement learning policies in robotics," 2025.

[5] K. Gmelin, S. Bahl, R. Mendonca, and D. Pathak, "Efficient RL via disentangled environment and agent representations," in *Proceedings of the 40th International Conference on Machine Learning* (A. Krause,

E. Brunskill, K. Cho, B. Engelhardt, S. Sabato, and J. Scarlett, eds.), vol. 202 of *Proceedings of Machine Learning Research*, pp. 11525–11545, PMLR, 23–29 Jul 2023.

[6] A. Pore, R. Muradore, and D. Dall'Alba, "Dear: Disentangled environment and agent representations for reinforcement learning without reconstruction," 2024.

[7] D. Emukpere, R. Deffayet, B. Wu, R. Brégier, M. Niemaz, J.-L. Meunier, D. Proux, J.-M. Renders, and S. Kim, "Disentangled object-centric image representation for robotic manipulation," 2025.

[8] B. Tang, I. Akinola, J. Xu, B. Wen, A. Handa, K. V. Wyk, D. Fox, G. S. Sukhatme, F. Ramos, and Y. Narang, "Automate: Specialist and generalist assembly policies over diverse geometries," 2024.

[9] Y. Chen, K. Kimble, E. H. Adelson, T. Asfour, P. Chanrungmaneekul, D. Kragic, X. Li, Y. Li, A. Prather, N. Pollard, M. A. Roa-Garzon, R. Seney, S. Wang, K. Zhang, Y. Zhu, and K. Hang, "Manipulationnet: Benchmarking real-world robot manipulation at scale through physical skill challenges and embodied multimodal reasoning," 2025.

[10] B. D. Team, "Blender 5.0," 2025.

[11] A. Mandlekar, S. Nasiriany, B. Wen, I. Akinola, Y. Narang, L. Fan, Y. Zhu, and D. Fox, "Mimicgen: A data generation system for scalable robot learning using human demonstrations," 2023.

[12] M. Mittal, P. Roth, J. Tigue, A. Richard, O. Zhang, P. Du, A. Serrano-Muñoz, X. Yao, R. Zurbrügg, N. Rudin, L. Wawrzyniak, M. Rakhsha, A. Denzler, E. Heiden, A. Borovicka, O. Ahmed, I. Akinola, A. Anwar, M. T. Carlson, J. Y. Feng, A. Garg, R. Gasoto, L. Gulich, Y. Guo, M. Gussert, A. Hansen, M. Kulkarni, C. Li, W. Liu, V. Makoviychuk, G. Malczyk, H. Mazhar, M. Moghani, A. Murali, M. Noseworthy, A. Poddubny, N. Ratliff, W. Rehberg, C. Schwarke, R. Singh, J. L. Smith, B. Tang, R. Thaker, M. Trepte, K. V. Wyk, F. Yu, A. Millane, V. Ramasamy, R. Steiner, S. Subramanian, C. Volk, C. Chen, N. Jawale, A. V. Kuruttukulam, M. A. Lin, A. Mandlekar, K. Patzwaldt, J. Welsh, H. Zhao, F. Anes, J.-F. Lafleche, N. Moënne-Loccoz, S. Park, R. Stepinski, D. V. Gelder, C. Amevor, J. Carius, J. Chang, A. H. Chen, P. de Heras Ciechomski, G. Daviet, M. Mohajerani, J. von Muralt, V. Reutskyy, M. Sauter, S. Schirm, E. L. Shi, P. Terdiman, K. Vilella, T. Widmer, G. Yeoman, T. Chen, S. Grizan, C. Li, L. Li, C. Smith, R. Wiltz, K. Alexis, Y. Chang, D. Chu, L. J. Fan, F. Farshidian, A. Handa, S. Huang, M. Hutter, Y. Narang, S. Pouya, S. Sheng, Y. Zhu, M. Macklin, A. Moravanszky, P. Reist, Y. Guo, D. Hoeller, and G. State, "Isaac lab: A gpu-accelerated simulation framework for multi-modal robot learning," *arXiv preprint arXiv:2511.04831*, 2025.

[13] A. Mandlekar, D. Xu, J. Wong, S. Nasiriany, C. Wang, R. Kulkarni, L. Fei-Fei, S. Savarese, Y. Zhu, and R. Martín-Martín, "What matters in learning from offline human demonstrations for robot manipulation," 2021.

[14] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, p. 1735–1780, Nov. 1997.

[15] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, "A simple framework for contrastive learning of visual representations," 2020.

[16] S. Ross, G. J. Gordon, and J. A. Bagnell, "A reduction of imitation learning and structured prediction to no-regret online learning," 2011.

[17] M. Kelly, C. Sidrane, K. Driggs-Campbell, and M. J. Kochenderfer, "Hg-dagger: Interactive imitation learning with human experts," 2019.