

Pre-training Visual Dexterity in Simulation

Sarthak Kamat^{1*} Adam Rashid^{2*} Satvik Sharma¹ Aseem Doriwala³
Chelsea Finn¹ Phillip Isola² C. Karen Liu¹

spd.bot

Abstract: Large-scale pre-training has made robot policy fine-tuning increasingly data-efficient, but this progress has largely been driven by datasets and embodiments built around simple parallel-jaw grippers. Dexterous, multi-fingered hands remain comparatively data-starved because real teleoperation is costly to scale, while human hand video is off-embodiment and requires lossy pose estimation and retargeting. We introduce *Simulation Pre-training for Dexterity* (SPD), a pre-training framework for dexterous manipulation that uses data entirely collected in simulation. In SPD, humans manipulate virtual objects inside a VR headset, enabling on-embodiment trajectories and robot-free collection. With the help of five operators, we collect 75 hours of multi-task dexterous manipulation over one week, and use it to pre-train a causal transformer on a sequence modeling objective. We study the benefits of simulation pre-training on real-world tasks by fine-tuning on 1–2 hours of physical demonstrations on a 56-DoF bimanual dexterous setup. We find that our approach outperforms training behavior cloning policies from scratch, showing that simulation teleoperation is a viable pre-training source for real-world dexterous manipulation. We perform ablation studies, measuring the benefits of history conditioning and short action chunks for reactive control.

1 Introduction

Dexterous manipulation is a challenging problem in robotics because it demands precise control during periods of high contact and partial observability, with low tolerance for latency. Behavior cloning has been shown to be an effective technique for training bimanual dexterous policies [1], but performance is limited by the quantity and coverage of task demonstrations. To address this, recent work pre-trains policies on large multi-task datasets, before fine-tuning on a smaller set of demonstrations of the target task [2, 3, 4, 5]. This amortizes the cost of learning broadly useful visuomotor representations and makes policy learning data-efficient.

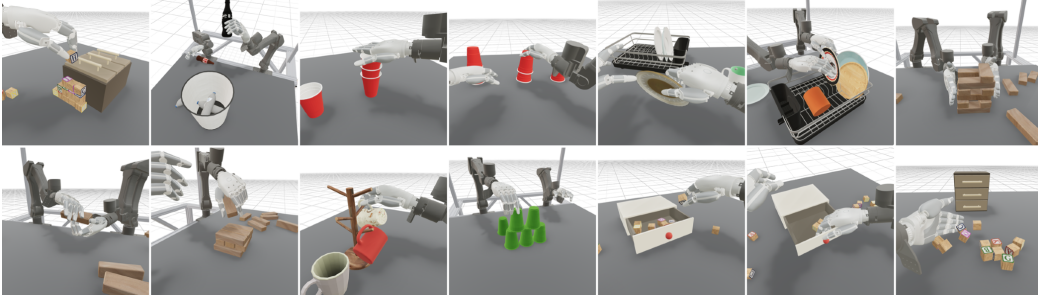
Teleoperation data is the ideal source for pre-training as it has no train-test divergence, but is challenging for multi-fingered hands because their cost and fragility limit data throughput. Handheld interfaces, such as UMI, enable scalable data collection without a robot [6, 4, 7], but are also difficult to extend to hands, as the device must avoid an embodiment gap while giving the operator control over every degree of freedom. Human hand video is abundant and cheap, but extracting robot-usable supervision from it is difficult because contact-rich hand-object interaction involves occlusion, resulting in noisy hand pose reconstruction. Even when human hands can be recorded accurately, such as by wearing motion-capture gloves, there remain differences in contact points and finger actuation, resulting in demonstrations that cannot be faithfully executed on the robot.

In this work, we study whether data collected in physics simulators can enable scalable pre-training for dexterous hands. Our framework, *Simulation Pre-training for Dexterity* (SPD), consists of two teleoperation systems: `spd-vr`, virtual reality software used to collect pre-training data, and `spd-teleop`, a real-world teleoperation system used to collect post-training data. The two systems are aligned, sharing top and wrist camera placements, an identical pair of dexterous hands and 6-DoF arms, and a similar collection of objects. Since our VR data collection has no dependence

*Equal contribution. Correspondence to: sartk@cs.stanford.edu, abrashid@mit.edu.

¹Stanford University ²MIT ³Scale AI

Pre-training in Simulation
75 hours of diverse manipulation



Real World Dexterity
1 hour of fine-tuning

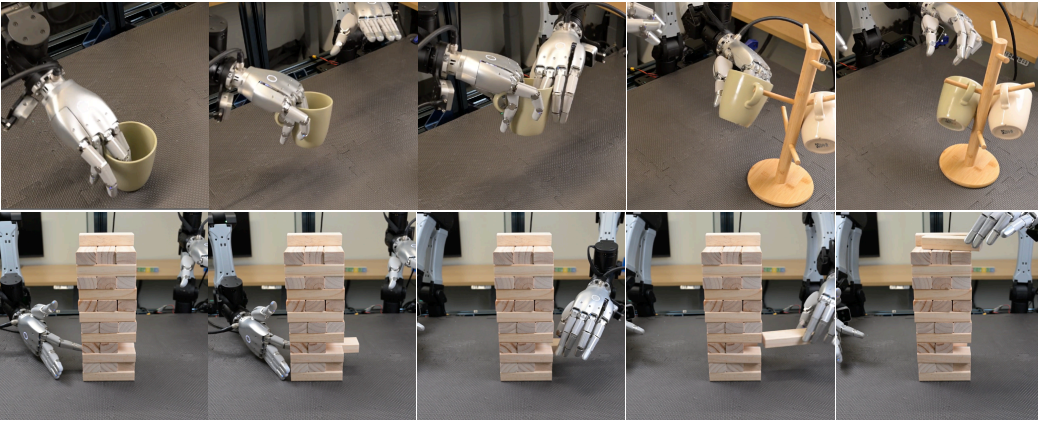


Figure 1: **Simulation pre-training for dexterity.** Top: 75 hours of teleoperated simulation data spanning six scenes and hundreds of object instances, covering contact-rich low-level dexterous behavior. Bottom: the pre-trained policy adapts to the real-world dexterous tasks shown, with 1 hour of real demonstrations.

on physical hardware, it enables fast resets of virtual objects, parallel operation, and decentralized collection.

Using `spd-vr`, five operators collect 75 hours of long-horizon, multi-task demonstrations over one week, spanning six scenes and hundreds of object instances. We pre-train a diffusion transformer policy on this dataset and fine-tune it on each downstream task with 1–2 hours of robot data. Across five real-world tasks — plate racking, mug hanging, playing Jenga, cup stacking, and tossing bottles into a bin — pre-training on simulation data outperforms training from scratch. Our ablations further show the value of conditioning on visuomotor history. With context, the policy can re-plan its actions frequently, staying reactive without losing temporal coherence, and this configuration benefits the most from pre-training.

Beyond our experimental findings, we release our pre-training dataset (`spd-75h`), VR teleoperation software (`spd-vr`), and six curated scenes with tuned contact parameters to support future work on simulation pre-training for dexterity.

2 Related Work

Pre-training for robot policies. Multi-task imitation learning has become a dominant paradigm for robot policy pre-training, with large robot datasets [8, 9, 10, 11] and ALOHA-style teleoperation datasets [12] enabling policies to learn from broad task and scene diversity. These datasets have supported increasingly general manipulation policies, including multi-task imitation

and transformer-based policies [13, 14, 15, 16, 17, 18, 19]. Recent vision-language-action models extend this paradigm by adapting pre-trained vision-language backbones for action prediction [20, 21, 2, 22, 23]. Beyond zero-shot generalization, these works have shown that large-scale pre-trained policies can serve as useful priors for downstream fine-tuning across new tasks, embodiments, modalities, and action spaces [17, 21, 24, 18, 23]. However, most of this progress has focused on manipulation with parallel-jaw grippers, whereas *SPD* studies whether simulation pre-training can be extended to contact-rich manipulation with dexterous hands.

Learning from humans. Because real-world robot teleoperation is expensive to scale, prior work has explored off-robot data sources that collect useful manipulation supervision without continuously operating a physical robot. One line of work uses handheld or portable interfaces to preserve action labels while reducing robot dependence [25, 26, 27, 28, 29]. Another line of work pre-trains from human video or internet-scale visual data, either through learned representations [30, 31, 32, 33, 34, 35] or through more policy-oriented video pre-training methods [36, 37, 38, 39, 40]. These approaches are attractive because they turn abundant human activity into scalable supervision, but human videos generally lack robot action labels and require action inference, pose reconstruction, retargeting, or latent-action discovery before they can supervise robot policies. Handheld and wearable systems address some of these issues by producing action-labeled data, but they often require specialized hardware, exoskeletons, or visual post-processing, and bulky interfaces can limit the dexterity of the human demonstrator. In contrast, *SPD* uses simulation teleoperation to collect action-labeled data directly on the target dexterous-hand embodiment, avoiding post-hoc human-to-robot retargeting while retaining the scalability benefits of off-robot data collection.

Simulation for dexterous manipulation. Simulation has been widely used for dexterous manipulation because it enables large-scale training for high-dimensional, contact-rich hands [41, 42, 43, 44, 45]. Subsequent sim-to-real systems have demonstrated increasingly capable dexterous skills using reinforcement learning with domain randomization from state-based and visual inputs [46, 47, 48]. Dexterous simulation benchmarks and training suites have expanded the scope of dexterous learning beyond individual hand-designed tasks, providing diverse objects, articulated environments, and multi-hand embodiments for training and evaluation [49, 50, 51]. In parallel-jaw and simple end-effector settings, recent sim-and-real co-training work shows that mixing real robot demonstrations with synthetic simulation data can improve vision-based manipulation policies, especially when the simulated data preserves task structure through digital cousins or controlled simulated demonstrations [52]. For dexterous robotic hands, simulation has often been used either with VR teleoperation systems for demonstration collection [53, 54, 55], or for RL-based sim-to-real transfer of task-specific skills [41, 42, 46, 56, 48]. In contrast, *SPD* uses simulation teleoperation as a broad pre-training data source: we collect diverse, action-labeled demonstrations directly on the target dexterous-hand embodiment and train a visuomotor policy intended to be fine-tuned across downstream real-world tasks, rather than optimized for a single simulated skill.

3 Method

We study contact-rich visuomotor control with a pair of dexterous, multi-fingered robot hands. Our goal is to pre-train a policy on large-scale simulation data so that it can be adapted to downstream real-world dexterous manipulation tasks using only a small number of task-specific demonstrations. We describe our methodology for VR teleoperation (Section 3.1), real-world teleoperation (Section 3.2), and policy learning (Section 3.3).

3.1 VR Data Collection

During VR data collection, the operator controls the target bimanual dexterous robot directly in the MuJoCo physics simulator [57]. The simulation steps on a computer at 480 Hz, capturing hand poses from the cameras on the connected headset at 60 Hz. The detected wrist pose and fingertip positions are used to drive the simulated robot arms and hands through inverse kinematics. All objects are

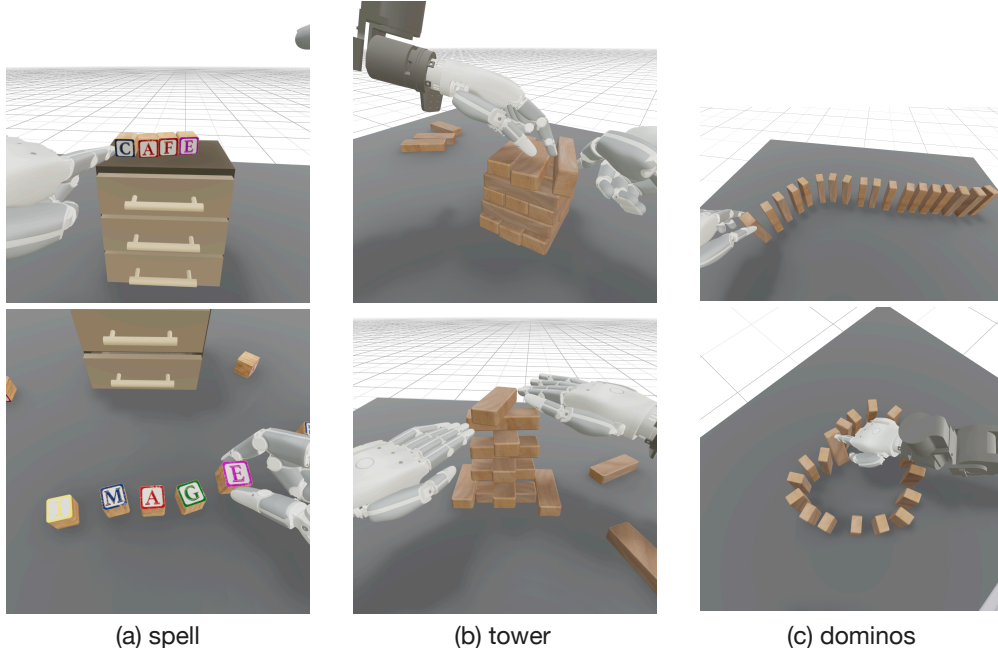


Figure 2: **Diverse operator strategies on open-ended simulation tasks.** Each pair of frames shows two distinct strategies the same operator pool produced on a single long-horizon task: (a) spelling a target word from a pile of letter blocks, (b) stacking blocks into a Jenga tower, and (c) arranging dominos into a chain.

virtual, and hand-object contacts are physically simulated. The robot arms are made translucent to minimize occlusion during collection.

We collect data across six scenes: spelling blocks, dishes, mugs, bottles, cups, and Jenga bricks. On episode reset, we randomize a new task prompt, along with asset selection, physical properties, and initial positions of all objects. With the help of five operators, we collect approximately 2,000 episodes (75 hours) over one week. Tasks are long-horizon and open-ended, specifying the desired outcome without constraining strategy or subtask ordering, which yields diverse, contact-rich behavior across object instances, poses, and strategies (Figure 2).

After data collection, we filter out extended periods of non-contact, and render our trajectories in parallel using an adapted version of the Madrona renderer [58]. In order to increase visual randomization, we compute visual augmentations as GPU transforms applied after data-loading. During rendering, we save segmentation masks alongside the wrist-camera frames and use them to randomly tint object colors and swap background and table textures. We also perform a symmetry augmentation: swapping the two arms and reflecting the corresponding images, proprioception, and actions.

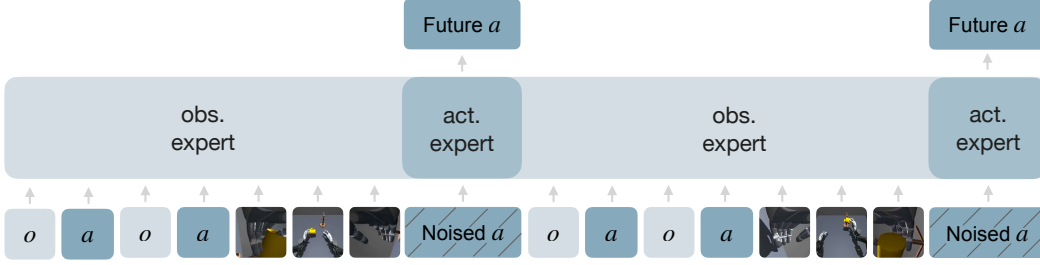
3.2 Real-World Data Collection

Our hardware consists of two upgraded YAM Pro arms, each equipped with a 22-DoF Sharpa Wave dexterous hand. To reduce overheating due to the weight of the hands, we replace the 10:1 gear-ratio J3 and J4 motors on the YAM arms with 40:1. The setup includes three RealSense D405 cameras, a top camera mounted between the arms, and wrist cameras mounted on the ulnar side of each hand.

Real-world teleoperation mirrors our VR teleoperation for re-targeting and control, except we replace the headset’s built-in hand tracking with Manus gloves for finger tracking and an attached Quest controller for wrist tracking. This produces smoother, more precise control, which we find necessary to control the robot through occlusions and at a distance.

Training

prefix parallel with causal sliding-window transformer



Inference

asynchronous with rolling KV cache

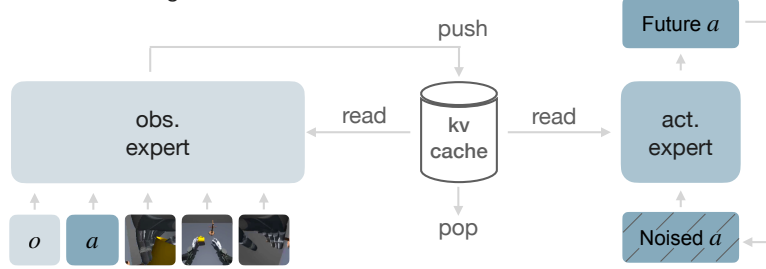


Figure 3: **Policy Architecture.** Our policy is a diffusion transformer that consumes a sequence of proprioception (o), action (a), multi-view images, and noised action chunks as interleaved tokens. In training, the model denoises all action chunks simultaneously with a causal mask, amortizing its sequence length. We additionally use sliding window attention in training, with a matched rolling KV cache for efficient inference.

3.3 Policy Architecture and Training

Our pre-training data spans diverse behaviors and low-level dexterous primitives, so the policy must capture multimodal action distributions. Following prior work on robot pre-training, we adopt a diffusion transformer that denoises action chunks conditioned on visual and proprioceptive observations [2, 5]. Our data lacks dense language annotations and broad scene coverage, so we forgo language conditioning. Instead, we condition on the visuomotor history, similar to [19].

The model consumes an interleaved sequence of proprioception, action, and visual tokens together with noised future action chunks, and supervises the denoised chunks with a flow-matching velocity-prediction objective [59]. Each training sequence spans 256 timesteps recorded at 30 Hz; to keep training efficient over this long context, the model denoises all chunks in parallel under a causal mask, amortizing the cost of processing the full sequence.

Visual observations from each camera are encoded into patches by a frozen, pre-trained vision transformer and pooled into a compact set of tokens via cross-attention from learnable queries. To make this pooling context-dependent, the pooled visual tokens cross-attend to the original ViT patches every two transformer blocks, as in [60], and we subsample image inputs every 8 timesteps to remove redundancy between adjacent frames. Non-visual modalities use modality-specific linear input and output projections over a shared transformer trunk, and, as in [2], the action-denoising expert maintains its own set of weights.

All attention is causal, with tokens assigned temporal positions from their timestamps via rotary embeddings. Noised action chunks additionally receive absolute positional embeddings encoding both the flow-matching timestep and the position within the chunk. To support a fixed-length KV cache at deployment, every layer uses sliding-window attention with a 32-timestep window. We train all models with Muon at a fixed learning rate of 10^{-3} and maintain an exponential moving average of the weights for inference.

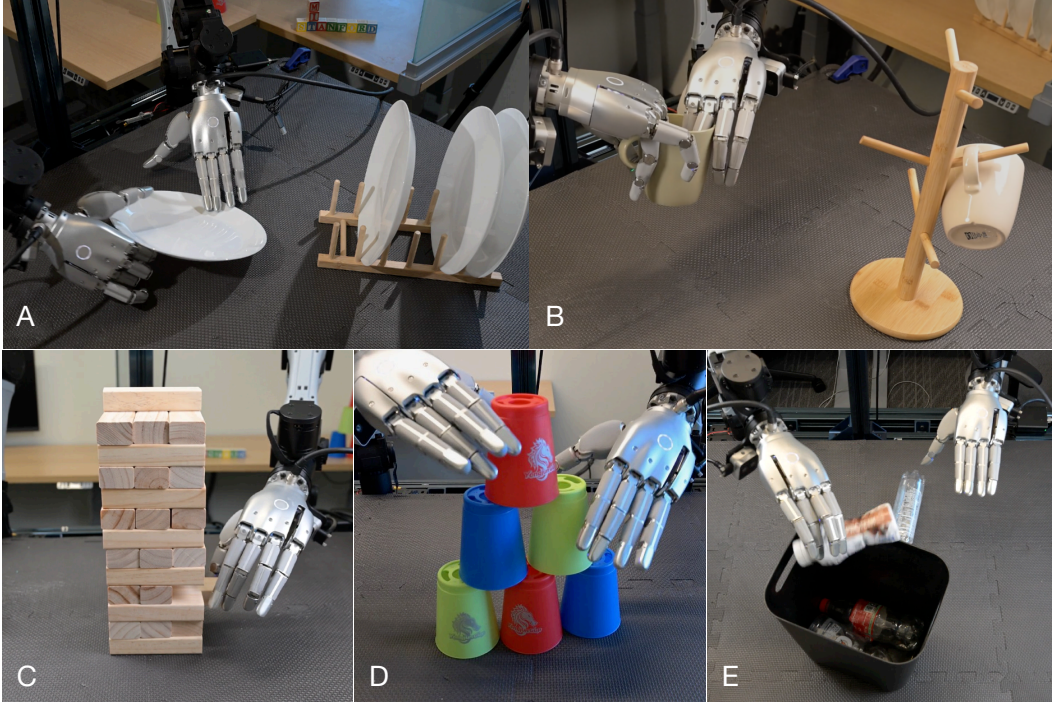


Figure 4: **Autonomous rollouts of five tasks.** (A) plate racking: lifting plates and racking them in a dish rack; (B) mug hanging: hanging a mug on a mug tree after a bimanual handover; (C) jenga playing: pushing a block out of the tower with one hand, pulling it free with the other, and placing it on top; (D) cup stacking: unstacking nested cups and building a pyramid; (E) bottles in bin: tossing bottles into a bin.

4 Experimental Results

We study the benefits of simulation pre-training by fully fine-tuning our policy on 1–2 hours of real-world demonstrations collected on our physical robot per task. We evaluate on five tasks consisting of objects that are similar, but not identical to those seen in pre-training: (A) plate racking, (B) mug hanging, (C) jenga playing, (D) cup stacking, and (E) bottles in bin, shown in Figure 4. We include videos of each task on our [project page](#).

4.1 Benefits of Simulation Pre-training

We compare our method against a baseline with the same architecture trained on real-world demonstrations of each task from scratch; results are in Figure 5. We perform 20 trials per checkpoint, and plot the fraction of trials that reach a certain amount of task progress.

On all five tasks, SPD reaches nearly every stage more often than the from-scratch BC baseline, and has higher average task progress. We also plot training loss curves for each task, which prior work shows to be positively correlated with downstream task performance for behavior cloning policies, unlike validation loss [5]. The loss curves match our real-world observations, with SPD checkpoints starting and converging to lower flow-matching loss values than from-scratch BC.

4.2 Ablations on History Conditioning and Chunk Size

We study the roles of history conditioning and action chunking by sweeping the sliding window size $w \in \{1, 32\}$ and the action chunk size $c \in \{8, 32\}$. Each of the four variants is trained both from the pre-trained checkpoint and from scratch, sharing the architecture, fine-tuning data, and training

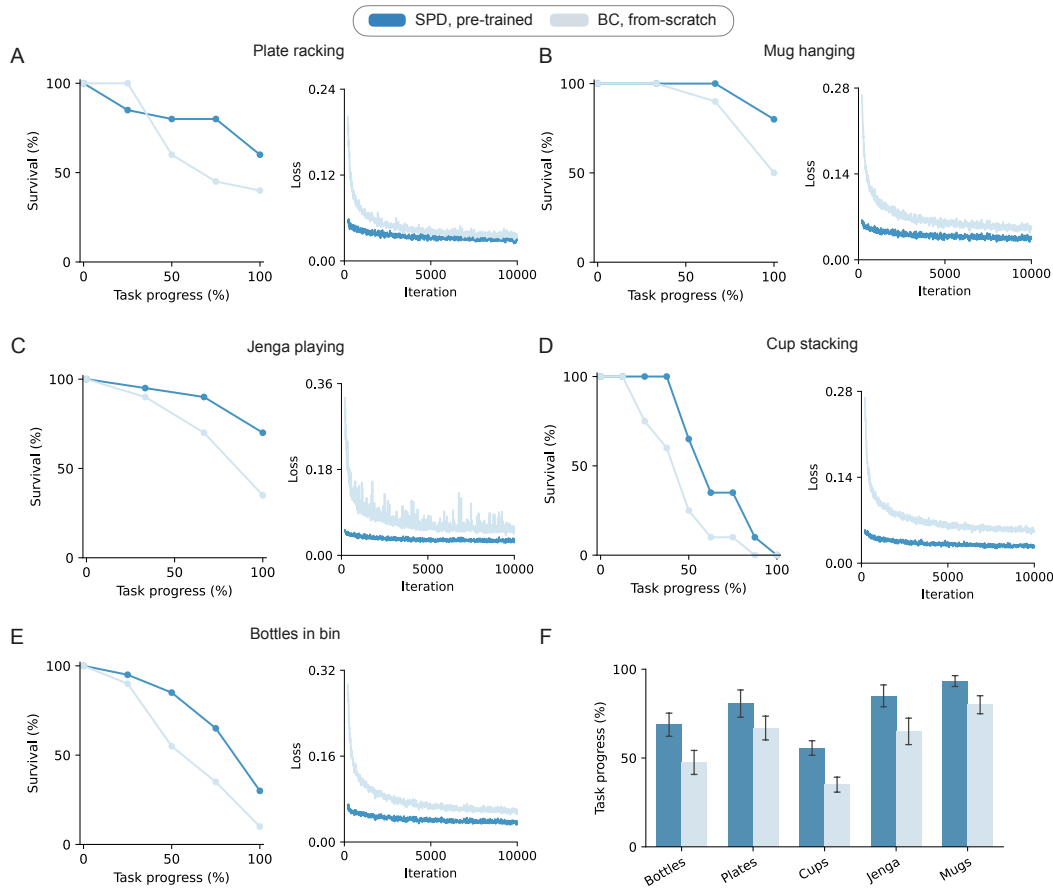


Figure 5: **Simulation pre-training improves real-world dexterity.** We evaluate five bimanual dexterous tasks on the real robot, comparing SPD pre-trained checkpoints to those trained from scratch (BC). In (A-E), we report the fraction of trials that reach each successive manipulation stage and plot training loss curves. In (F), we plot average task progress; error bars show standard error.

time of Section 3.3, and is evaluated on all five tasks under the protocol of Section A.4. Figure 6 plots per-task progress for each variant, with the full numbers in Table 1.

Prior works, such as π_0 [2], train with a single frame of context and a one-second action chunk, corresponding to $w = 1$, $c = 32$. In this single-frame setting, we observe that reducing the chunk size to $c = 8$ collapses performance, making the policy visibly shaky and less temporally coherent. Adding history removes this trade-off: with a 32-step window, the 8-step chunk becomes the strongest variant in both training regimes, drawing temporal consistency from its context and reactivity from its shorter chunk. This configuration also benefits the most from pre-training: its average progress improves by 18 points over its from-scratch counterpart, compared to 3 points or less for the other variants.

5 Conclusion

We present *Simulation Pre-training for Dexterity* (SPD), a simulation pre-training framework for dexterous manipulation. SPD collects scalable, action-labeled, on-embodiment demonstrations in simulation without requiring physical robot hardware during pre-training. Pre-training a causal diffusion-transformer policy on this data improves real-world fine-tuning on five bimanual dexterous tasks compared to training from scratch. Our ablations show that history conditioning enables short, reactive action chunks and yields the largest gains from pre-training.

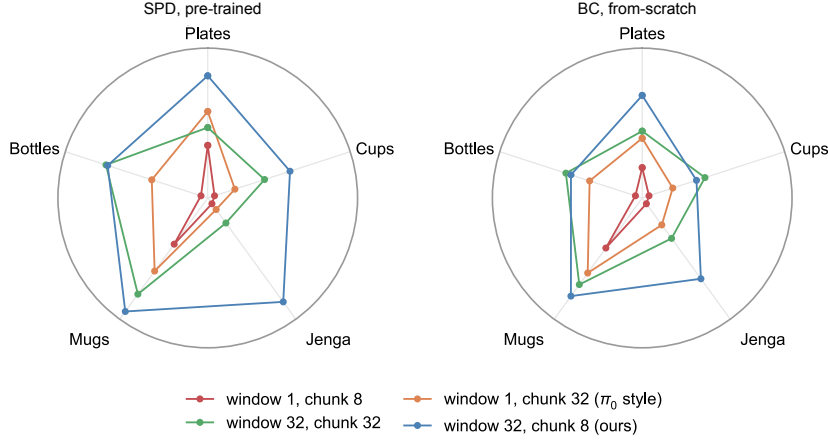


Figure 6: **Ablations on history conditioning and chunk size.** Visualization of the average task progress in Table 1, for policies pre-trained with SPD (left) and trained from scratch (right).

Table 1: **Average task progress for all variants.** Progress (%) for every sliding-window (w) and action-chunk (c) combination, with the best variant per task bolded within each training regime. $w = 32$, $c = 8$ is the chosen configuration.

	w	c	plates	mugs	jenga	cups	bottles
SPD, pre-trained	1	8	31.9	35.0	0.0	0.0	0.0
		32	55.6	58.3	5.0	15.0	36.2
	32	8	80.6	93.3	85.0	55.6	68.8
		32	44.4	78.3	16.7	36.9	70.0
BC, from-scratch	1	8	16.2	38.3	0.0	0.0	0.0
		32	36.9	60.0	18.3	17.5	33.8
	32	8	66.9	80.0	65.0	35.0	47.5
		32	41.9	70.0	30.0	41.2	51.2

Limitations. While SPD improves real-world performance, it still depends on simulation scenes whose physics are tuned well enough for operators to produce realistic behaviors. If object masses, friction, or contact responses differ too much from real-world expectations, the collected demonstrations may encode strategies that transfer poorly. Our current pre-training data is also limited in scene and object diversity, and the real-world evaluation uses objects that are similar to those seen in simulation. Evaluating on more out-of-distribution objects, scenes, and task variations would better indicate how broadly the pre-trained policy generalizes.

Future work. Simulation teleoperation is one of several routes to scaling dexterous pre-training, and a natural next step is to study how it complements the sources discussed in Section 1 — real-world teleoperation and egocentric human video — within a mixed pre-training corpus. A second direction is scale: pre-training on more scenes, objects, and hours would allow characterizing the forms of generalization that emerge from simulation data itself. Finally, teleoperation is not the only way to generate experience in simulation. Reinforcement learning scales with compute rather than operator time, and our pre-trained policy offers a natural initialization for it.

Acknowledgments

We thank Scale AI for collecting the spd-75h dataset, Arthur Allshire and Ritvik Singh for sharing the ABC teleoperation infrastructure, which was adapted for SPD, and Sharpa for their support. Sarthak Kamat and Adam Rashid are supported by the National Science Foundation (NSF) Graduate Research Fellowship Program. Satvik Sharma is supported by Meta and the NSF under

Grant Numbers 2342246 and 2327974. This work was supported under project ID 43 as part of the Swiss AI Initiative, through a grant from the ETH Domain and computational resources provided by the Swiss National Supercomputing Centre (CSCS) under the Alps infrastructure. This work was also supported by a Packard Fellowship to P.I., ONR MURI grant N00014-22-1-2740, NSF Award 2153854, and Stanford HAI.

References

- [1] T. Z. Zhao, J. Tompson, D. Driess, P. Florence, S. K. S. Ghasemipour, C. Finn, and A. Wahid. Aloha unleashed: A simple recipe for robot dexterity. In P. Agrawal, O. Kroemer, and W. Burgard, editors, *Proceedings of The 8th Conference on Robot Learning*, volume 270 of *Proceedings of Machine Learning Research*, pages 1910–1924. PMLR, 06–09 Nov 2025. URL <https://proceedings.mlr.press/v270/zhao25b.html>.
- [2] K. Black, N. Brown, D. Driess, A. Esmail, M. Equi, C. Finn, N. Fusai, L. Groom, K. Hausman, B. Ichter, S. Jakubczak, T. Jones, L. Ke, S. Levine, A. Li-Bell, M. Mothukuri, S. Nair, K. Pertsch, L. X. Shi, J. Tanner, Q. Vuong, A. Walling, H. Wang, and U. Zhilinsky. π_0 : A vision-language-action flow model for general robot control, 2026. URL <https://arxiv.org/abs/2410.24164>.
- [3] T. L. Team, J. Barreiros, A. Beaulieu, A. Bhat, R. Cory, E. Cousineau, H. Dai, C.-H. Fang, K. Hashimoto, M. Z. Irshad, M. Itkina, N. Kuppuswamy, K.-H. Lee, K. Liu, D. McConachie, I. McMahon, H. Nishimura, C. Phillips-Grafflin, C. Richter, P. Shah, K. Srinivasan, B. Wulfe, C. Xu, M. Zhang, A. Alspach, M. Angeles, K. Arora, V. C. Guizilini, A. Castro, D. Chen, T.-S. Chu, S. Creasey, S. Curtis, R. Denitto, E. Dixon, E. Dusel, M. Ferreira, A. Goncalves, G. Gould, D. Guoy, S. Gupta, X. Han, K. Hatch, B. Hathaway, A. Henry, H. Hochshtein, P. Horgan, S. Iwase, D. Jackson, S. Karamcheti, S. Keh, J. Masterjohn, J. Mercat, P. Miller, P. Mitiguy, T. Nguyen, J. Nimmer, Y. Noguchi, R. Ong, A. Onol, O. Pfannenstiehl, R. Poyner, L. P. M. Rocha, G. Richardson, C. Rodriguez, D. Seale, M. Sherman, M. Smith-Jones, D. Tago, P. Tokmakov, M. Tran, B. V. Hoorick, I. Vasiljevic, S. Zakharov, M. Zolotas, R. Ambrus, K. Fetzer-Borelli, B. Burchfiel, H. Kress-Gazit, S. Feng, S. Ford, and R. Tedrake. A careful examination of large behavior models for multitask dexterous manipulation. 2025. URL <https://arxiv.org/abs/2507.05331>.
- [4] G. A. Team. Gen-1: Scaling embodied foundation models to mastery. *Generalist AI Blog*, 2026. <https://generalistai.com/blog/apr-02-2026-GEN-1>.
- [5] A. Allshire, H. G. Singh, R. Singh, A. Rashid, H. Choi, D. McAllister, J. Yu, Y. Chen, H. Huang, P. Abbeel, X. Chen, R. Duan, P. Isola, J. Malik, F. Shentu, G. Shi, P. Wu, and A. Kanazawa. Scalable behavior cloning with open data, training, and evaluation. *arXiv preprint*, 2026. URL <https://abc.bot/>.
- [6] C. Chi, Z. Xu, C. Pan, E. Cousineau, B. Burchfiel, S. Feng, R. Tedrake, and S. Song. Universal manipulation interface: In-the-wild robot teaching without in-the-wild robots, 2024. URL <https://arxiv.org/abs/2402.10329>.
- [7] S. Robotics. Act-2 preview: Generalizing reliability. *Sunday Robotics Blog*, jul 2026. URL <https://www.sunday.ai/blog/act-2-preview>.
- [8] S. Dasari, F. Ebert, S. Tian, S. Nair, B. Bucher, K. Schmeckpeper, S. Singh, S. Levine, and C. Finn. Robonet: Large-scale multi-robot learning, 2020. URL <https://arxiv.org/abs/1910.11215>.
- [9] H. Walke, K. Black, A. Lee, M. J. Kim, M. Du, C. Zheng, T. Zhao, P. Hansen-Estruch, Q. Vuong, A. He, V. Myers, K. Fang, C. Finn, and S. Levine. Bridgedata v2: A dataset for robot learning at scale, 2024. URL <https://arxiv.org/abs/2308.12952>.

- [10] E. Collaboration, A. O'Neill, A. Rehman, A. Gupta, A. Maddukuri, A. Gupta, A. Padalkar, A. Lee, A. Pooley, A. Gupta, A. Mandlekar, A. Jain, A. Tung, A. Bewley, A. Herzog, A. Irpan, A. Khazatsky, A. Rai, A. Gupta, A. Wang, A. Kolobov, A. Singh, A. Garg, A. Kembhavi, A. Xie, A. Brohan, A. Raffin, A. Sharma, A. Yavary, A. Jain, A. Balakrishna, A. Wahid, B. Burgess-Limerick, B. Kim, B. Schölkopf, B. Wulfe, B. Ichter, C. Lu, C. Xu, C. Le, C. Finn, C. Wang, C. Xu, C. Chi, C. Huang, C. Chan, C. Agia, C. Pan, C. Fu, C. Devin, D. Xu, D. Morton, D. Driess, D. Chen, D. Pathak, D. Shah, D. Büchler, D. Jayaraman, D. Kalashnikov, D. Sadigh, E. Johns, E. Foster, F. Liu, F. Ceola, F. Xia, F. Zhao, F. V. Frueh, F. Stulp, G. Zhou, G. S. Sukhatme, G. Salhotra, G. Yan, G. Feng, G. Schiavi, G. Berseth, G. Kahn, G. Yang, G. Wang, H. Su, H.-S. Fang, H. Shi, H. Bao, H. B. Amor, H. I. Christensen, H. Furuta, H. Bharadhwaj, H. Walke, H. Fang, H. Ha, I. Mordatch, I. Radosavovic, I. Leal, J. Liang, J. Abou-Chakra, J. Kim, J. Drake, J. Peters, J. Schneider, J. Hsu, J. Vakil, J. Bohg, J. Bingham, J. Wu, J. Gao, J. Hu, J. Wu, J. Wu, J. Sun, J. Luo, J. Gu, J. Tan, J. Oh, J. Wu, J. Lu, J. Yang, J. Malik, J. Silvério, J. Hejna, J. Booher, J. Thompson, J. Yang, J. Salvador, J. J. Lim, J. Han, K. Wang, K. Rao, K. Pertsch, K. Hausman, K. Go, K. Gopalakrishnan, K. Goldberg, K. Byrne, K. Oslund, K. Kawaharazuka, K. Black, K. Lin, K. Zhang, K. Ehsani, K. Leckala, K. Ellis, K. Rana, K. Srinivasan, K. Fang, K. P. Singh, K.-H. Zeng, K. Hatch, K. Hsu, L. Itti, L. Y. Chen, L. Pinto, L. Fei-Fei, L. Tan, L. J. Fan, L. Ott, L. Lee, L. Weihs, M. Chen, M. Lepert, M. Memmel, M. Tomizuka, M. Itkina, M. G. Castro, M. Spero, M. Du, M. Ahn, M. C. Yip, M. Zhang, M. Ding, M. Heo, M. K. Srirama, M. Sharma, M. J. Kim, M. Z. Irshad, N. Kanazawa, N. Hansen, N. Heess, N. J. Joshi, N. Suenderhauf, N. Liu, N. D. Palo, N. M. M. Shafiullah, O. Mees, O. Kroemer, O. Bastani, P. R. Sanketi, P. T. Miller, P. Yin, P. Wohlhart, P. Xu, P. D. Fagan, P. Mitrano, P. Sermanet, P. Abbeel, P. Sundaresan, Q. Chen, Q. Vuong, R. Rafailov, R. Tian, R. Doshi, R. Martín-Martín, R. Baijal, R. Scalise, R. Hendrix, R. Lin, R. Qian, R. Zhang, R. Mendonca, R. Shah, R. Hoque, R. Julian, S. Bustamante, S. Kirmani, S. Levine, S. Lin, S. Moore, S. Bahl, S. Dass, S. Sonawani, S. Tulsiani, S. Song, S. Xu, S. Haldar, S. Karamcheti, S. Adebola, S. Guist, S. Nasiriany, S. Schaal, S. Welker, S. Tian, S. Ramamoorthy, S. Dasari, S. Belkhale, S. Park, S. Nair, S. Mirchandani, T. Osa, T. Gupta, T. Harada, T. Matsushima, T. Xiao, T. Kollar, T. Yu, T. Ding, T. Davchev, T. Z. Zhao, T. Armstrong, T. Darrell, T. Chung, V. Jain, V. Kumar, V. Vanhoucke, V. Guizilini, W. Zhan, W. Zhou, W. Burgard, X. Chen, X. Chen, X. Wang, X. Zhu, X. Geng, X. Liu, X. Liangwei, X. Li, Y. Pang, Y. Lu, Y. J. Ma, Y. Kim, Y. Chebotar, Y. Zhou, Y. Zhu, Y. Wu, Y. Xu, Y. Wang, Y. Bisk, Y. Dou, Y. Cho, Y. Lee, Y. Cui, Y. Cao, Y.-H. Wu, Y. Tang, Y. Zhu, Y. Zhang, Y. Jiang, Y. Li, Y. Li, Y. Iwasawa, Y. Matsuo, Z. Ma, Z. Xu, Z. J. Cui, Z. Zhang, Z. Fu, and Z. Lin. Open x-embodiment: Robotic learning datasets and rt-x models, 2025. URL <https://arxiv.org/abs/2310.08864>.
- [11] A. Khazatsky, K. Pertsch, S. Nair, A. Balakrishna, S. Dasari, S. Karamcheti, S. Nasiriany, M. K. Srirama, L. Y. Chen, K. Ellis, P. D. Fagan, J. Hejna, M. Itkina, M. Lepert, Y. J. Ma, P. T. Miller, J. Wu, S. Belkhale, S. Dass, H. Ha, A. Jain, A. Lee, Y. Lee, M. Memmel, S. Park, I. Radosavovic, K. Wang, A. Zhan, K. Black, C. Chi, K. B. Hatch, S. Lin, J. Lu, J. Mercat, A. Rehman, P. R. Sanketi, A. Sharma, C. Simpson, Q. Vuong, H. R. Walke, B. Wulfe, T. Xiao, J. H. Yang, A. Yavary, T. Z. Zhao, C. Agia, R. Baijal, M. G. Castro, D. Chen, Q. Chen, T. Chung, J. Drake, E. P. Foster, J. Gao, V. Guizilini, D. A. Herrera, M. Heo, K. Hsu, J. Hu, M. Z. Irshad, D. Jackson, C. Le, Y. Li, K. Lin, R. Lin, Z. Ma, A. Maddukuri, S. Mirchandani, D. Morton, T. Nguyen, A. O'Neill, R. Scalise, D. Seale, V. Son, S. Tian, E. Tran, A. E. Wang, Y. Wu, A. Xie, J. Yang, P. Yin, Y. Zhang, O. Bastani, G. Berseth, J. Bohg, K. Goldberg, A. Gupta, A. Gupta, D. Jayaraman, J. J. Lim, J. Malik, R. Martín-Martín, S. Ramamoorthy, D. Sadigh, S. Song, J. Wu, M. C. Yip, Y. Zhu, T. Kollar, S. Levine, and C. Finn. Droid: A large-scale in-the-wild robot manipulation dataset, 2025. URL <https://arxiv.org/abs/2403.12945>.
- [12] T. Z. Zhao, V. Kumar, S. Levine, and C. Finn. Learning fine-grained bimanual manipulation with low-cost hardware, 2023. URL <https://arxiv.org/abs/2304.13705>.

- [13] D. Kalashnikov, J. Varley, Y. Chebotar, B. Swanson, R. Jonschkowski, C. Finn, S. Levine, and K. Hausman. Mt-opt: Continuous multi-task robotic reinforcement learning at scale, 2021. URL <https://arxiv.org/abs/2104.08212>.
- [14] E. Jang, A. Irpan, M. Khansari, D. Kappler, F. Ebert, C. Lynch, S. Levine, and C. Finn. Bc-z: Zero-shot task generalization with robotic imitation learning, 2022. URL <https://arxiv.org/abs/2202.02005>.
- [15] M. Shridhar, L. Manuelli, and D. Fox. Perceiver-actor: A multi-task transformer for robotic manipulation, 2022. URL <https://arxiv.org/abs/2209.05451>.
- [16] A. Brohan, N. Brown, J. Carbajal, Y. Chebotar, J. Dabis, C. Finn, K. Gopalakrishnan, K. Hausman, A. Herzog, J. Hsu, J. Ibarz, B. Ichter, A. Irpan, T. Jackson, S. Jesmonth, N. J. Joshi, R. Julian, D. Kalashnikov, Y. Kuang, I. Leal, K.-H. Lee, S. Levine, Y. Lu, U. Malla, D. Manjunath, I. Mordatch, O. Nachum, C. Parada, J. Peralta, E. Perez, K. Pertsch, J. Quiambao, K. Rao, M. Ryoo, G. Salazar, P. Sanketi, K. Sayed, J. Singh, S. Sontakke, A. Stone, C. Tan, H. Tran, V. Vanhoucke, S. Vega, Q. Vuong, F. Xia, T. Xiao, P. Xu, S. Xu, T. Yu, and B. Zitkovich. Rt-1: Robotics transformer for real-world control at scale, 2023. URL <https://arxiv.org/abs/2212.06817>.
- [17] O. M. Team, D. Ghosh, H. Walke, K. Pertsch, K. Black, O. Mees, S. Dasari, J. Hejna, T. Kreiman, C. Xu, J. Luo, Y. L. Tan, L. Y. Chen, P. Sanketi, Q. Vuong, T. Xiao, D. Sadigh, C. Finn, and S. Levine. Octo: An open-source generalist robot policy, 2024. URL <https://arxiv.org/abs/2405.12213>.
- [18] S. Liu, L. Wu, B. Li, H. Tan, H. Chen, Z. Wang, K. Xu, H. Su, and J. Zhu. Rdt-1b: a diffusion foundation model for bimanual manipulation, 2025. URL <https://arxiv.org/abs/2410.07864>.
- [19] I. Radosavovic, B. Shi, L. Fu, K. Goldberg, T. Darrell, and J. Malik. Robot learning with sensorimotor pre-training. In J. Tan, M. Toussaint, and K. Darvish, editors, *Proceedings of The 7th Conference on Robot Learning*, volume 229 of *Proceedings of Machine Learning Research*, pages 683–693. PMLR, 06–09 Nov 2023. URL <https://proceedings.mlr.press/v229/radosavovic23a.html>.
- [20] A. Brohan, N. Brown, J. Carbajal, Y. Chebotar, X. Chen, K. Choromanski, T. Ding, D. Driess, A. Dubey, C. Finn, P. Florence, C. Fu, M. G. Arenas, K. Gopalakrishnan, K. Han, K. Hausman, A. Herzog, J. Hsu, B. Ichter, A. Irpan, N. Joshi, R. Julian, D. Kalashnikov, Y. Kuang, I. Leal, L. Lee, T.-W. E. Lee, S. Levine, Y. Lu, H. Michalewski, I. Mordatch, K. Pertsch, K. Rao, K. Reymann, M. Ryoo, G. Salazar, P. Sanketi, P. Sermanet, J. Singh, A. Singh, R. Soricut, H. Tran, V. Vanhoucke, Q. Vuong, A. Wahid, S. Welker, P. Wohlhart, J. Wu, F. Xia, T. Xiao, P. Xu, S. Xu, T. Yu, and B. Zitkovich. Rt-2: Vision-language-action models transfer web knowledge to robotic control, 2023. URL <https://arxiv.org/abs/2307.15818>.
- [21] M. J. Kim, K. Pertsch, S. Karamcheti, T. Xiao, A. Balakrishna, S. Nair, R. Rafailov, E. Foster, G. Lam, P. Sanketi, Q. Vuong, T. Kollar, B. Burchfiel, R. Tedrake, D. Sadigh, S. Levine, P. Liang, and C. Finn. Openvla: An open-source vision-language-action model, 2024. URL <https://arxiv.org/abs/2406.09246>.
- [22] J. Lee, J. Duan, H. Fang, Y. Deng, S. Liu, B. Li, B. Fang, J. Zhang, Y. R. Wang, S. Lee, W. Han, W. Pumacay, A. Wu, R. Hendrix, K. Farley, E. VanderBilt, A. Farhadi, D. Fox, and R. Krishna. Molmoact: Action reasoning models that can reason in space, 2025. URL <https://arxiv.org/abs/2508.07917>.
- [23] K. Bousmalis, G. Vezzani, D. Rao, C. Devin, A. X. Lee, M. Bauza, T. Davchev, Y. Zhou, A. Gupta, A. Raju, A. Laurens, C. Fantacci, V. Dalibard, M. Zambelli, M. Martins, R. Pevceviciute, M. Blokzijl, M. Denil, N. Batchelor, T. Lampe, E. Parisotto, K. Žohna, S. Reed, S. G. Colmenarejo, J. Scholz, A. Abdolmaleki, O. Groth, J.-B. Regli, O. Sushkov, T. Rothörl, J. E.

- Chen, Y. Aytar, D. Barker, J. Ortiz, M. Riedmiller, J. T. Springenberg, R. Hadsell, F. Nori, and N. Heess. Robocat: A self-improving generalist agent for robotic manipulation, 2023. URL <https://arxiv.org/abs/2306.11706>.
- [24] M. J. Kim, C. Finn, and P. Liang. Fine-tuning vision-language-action models: Optimizing speed and success, 2025. URL <https://arxiv.org/abs/2502.19645>.
- [25] M. Xu, H. Zhang, Y. Hou, Z. Xu, L. Fan, M. Veloso, and S. Song. Dexumi: Using human hand as the universal manipulation interface for dexterous manipulation, 2025. URL <https://arxiv.org/abs/2505.21864>.
- [26] T. Cheng, K. Chen, L. Chen, L. Zhang, Y. Zhang, Y. Ling, M. Hamad, Z. Bing, F. Wu, K. Sharma, and A. Knoll. Tacumi: A multi-modal universal manipulation interface for contact-rich tasks, 2026. URL <https://arxiv.org/abs/2601.14550>.
- [27] Z. Wang. Umi-3d: Extending universal manipulation interface from vision-limited to 3d spatial perception, 2026. URL <https://arxiv.org/abs/2604.14089>.
- [28] J. Koh, H. Jung, N. Kim, W. Ko, and C. Nam. Dex-mouse: A low-cost portable and universal interface with force feedback for data collection of dexterous robotic hands, 2026. URL <https://arxiv.org/abs/2604.15013>.
- [29] L. Wu, C. Yu, J. Ren, L. Chen, Y. Jiang, R. Huang, G. Gu, and H. Li. Freetacman: Robot-free visuo-tactile data collection system for contact-rich manipulation, 2026. URL <https://arxiv.org/abs/2506.01941>.
- [30] S. Nair, A. Rajeswaran, V. Kumar, C. Finn, and A. Gupta. R3m: A universal visual representation for robot manipulation, 2022. URL <https://arxiv.org/abs/2203.12601>.
- [31] T. Xiao, I. Radosavovic, T. Darrell, and J. Malik. Masked visual pre-training for motor control, 2022. URL <https://arxiv.org/abs/2203.06173>.
- [32] I. Radosavovic, T. Xiao, S. James, P. Abbeel, J. Malik, and T. Darrell. Real-world robot learning with masked visual pre-training, 2022. URL <https://arxiv.org/abs/2210.03109>.
- [33] S. Karamcheti, S. Nair, A. S. Chen, T. Kollar, C. Finn, D. Sadigh, and P. Liang. Language-driven representation learning for robotics, 2023. URL <https://arxiv.org/abs/2302.12766>.
- [34] Y. J. Ma, W. Liang, V. Som, V. Kumar, A. Zhang, O. Bastani, and D. Jayaraman. Liv: Language-image representations and rewards for robotic control, 2023. URL <https://arxiv.org/abs/2306.00958>.
- [35] S. A. Sontakke, J. Zhang, S. M. R. Arnold, K. Pertsch, E. Bıyık, D. Sadigh, C. Finn, and L. Itti. Roboclip: One demonstration is enough to learn robot policies, 2023. URL <https://arxiv.org/abs/2310.07899>.
- [36] H. Wu, Y. Jing, C. Cheang, G. Chen, J. Xu, X. Li, M. Liu, H. Li, and T. Kong. Unleashing large-scale video generative pre-training for visual robot manipulation, 2023. URL <https://arxiv.org/abs/2312.13139>.
- [37] S. Ye, J. Jang, B. Jeon, S. Joo, J. Yang, B. Peng, A. Mandlekar, R. Tan, Y.-W. Chao, B. Y. Lin, L. Liden, K. Lee, J. Gao, L. Zettlemoyer, D. Fox, and M. Seo. Latent action pretraining from videos, 2025. URL <https://arxiv.org/abs/2410.11758>.
- [38] R. Zheng, D. Niu, Y. Xie, J. Wang, M. Xu, Y. Jiang, F. Castañeda, F. Hu, Y. L. Tan, L. Fu, T. Darrell, F. Huang, Y. Zhu, D. Xu, and L. Fan. Egoscale: Scaling dexterous manipulation with diverse egocentric human data, 2026. URL <https://arxiv.org/abs/2602.16710>.

- [39] Q. Li, Y. Deng, Y. Liang, L. Luo, L. Zhou, C. Yao, L. Zeng, Z. Feng, H. Liang, S. Xu, Y. Zhang, X. Chen, H. Chen, L. Sun, D. Chen, J. Yang, and B. Guo. Scalable vision-language-action model pretraining for robotic manipulation with real-life human activity videos, 2025. URL <https://arxiv.org/abs/2510.21571>.
- [40] J. Zeng, Q. Bu, B. Wang, W. Xia, L. Chen, H. Dong, H. Song, D. Wang, D. Hu, P. Luo, H. Cui, B. Zhao, X. Li, Y. Qiao, and H. Li. Learning manipulation by predicting interaction, 2024. URL <https://arxiv.org/abs/2406.00439>.
- [41] OpenAI, M. Andrychowicz, B. Baker, M. Chociej, R. Jozefowicz, B. McGrew, J. Pachocki, A. Petron, M. Plappert, G. Powell, A. Ray, J. Schneider, S. Sidor, J. Tobin, P. Welinder, L. Weng, and W. Zaremba. Learning dexterous in-hand manipulation, 2019. URL <https://arxiv.org/abs/1808.00177>.
- [42] 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, 2019. URL <https://arxiv.org/abs/1910.07113>.
- [43] 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, 2018. URL <https://arxiv.org/abs/1709.10087>.
- [44] Y. Qin, Y.-H. Wu, S. Liu, H. Jiang, R. Yang, Y. Fu, and X. Wang. Dexmv: Imitation learning for dexterous manipulation from human videos, 2022. URL <https://arxiv.org/abs/2108.05877>.
- [45] A. Gupta, V. Kumar, C. Lynch, S. Levine, and K. Hausman. Relay policy learning: Solving long-horizon tasks via imitation and reinforcement learning, 2019. URL <https://arxiv.org/abs/1910.11956>.
- [46] A. Handa, A. Allshire, V. Makoviychuk, A. Petrenko, R. Singh, J. Liu, D. Makoviichuk, K. V. Wyk, A. Zhurkevich, B. Sundaralingam, Y. Narang, J.-F. Lafleche, D. Fox, and G. State. Dextreme: Transfer of agile in-hand manipulation from simulation to reality, 2024. URL <https://arxiv.org/abs/2210.13702>.
- [47] H. Qi, B. Yi, S. Suresh, M. Lambeta, Y. Ma, R. Calandra, and J. Malik. General in-hand object rotation with vision and touch. In J. Tan, M. Toussaint, and K. Darvish, editors, *Proceedings of The 7th Conference on Robot Learning*, volume 229 of *Proceedings of Machine Learning Research*, pages 2549–2564. PMLR, 06–09 Nov 2023. URL <https://proceedings.mlr.press/v229/qi23a.html>.
- [48] R. Singh, A. Allshire, A. Handa, N. Ratliff, and K. V. Wyk. Dextrah-rgb: Visuomotor policies to grasp anything with dexterous hands, 2025. URL <https://arxiv.org/abs/2412.01791>.
- [49] C. Bao, H. Xu, Y. Qin, and X. Wang. Dexart: Benchmarking generalizable dexterous manipulation with articulated objects, 2023. URL <https://arxiv.org/abs/2305.05706>.
- [50] Y. Chen, T. Wu, S. Wang, X. Feng, J. Jiang, S. M. McAleer, Y. Geng, H. Dong, Z. Lu, S.-C. Zhu, and Y. Yang. Towards human-level bimanual dexterous manipulation with reinforcement learning, 2022. URL <https://arxiv.org/abs/2206.08686>.
- [51] A. Petrenko, A. Allshire, G. State, A. Handa, and V. Makoviychuk. Dexpbt: Scaling up dexterous manipulation for hand-arm systems with population based training, 2023. URL <https://arxiv.org/abs/2305.12127>.
- [52] A. Maddukuri, Z. Jiang, L. Y. Chen, S. Nasiriany, Y. Xie, Y. Fang, W. Huang, Z. Wang, Z. Xu, N. Chernyadev, S. Reed, K. Goldberg, A. Mandlekar, L. Fan, and Y. Zhu. Sim-and-real co-training: A simple recipe for vision-based robotic manipulation, 2025. URL <https://arxiv.org/abs/2503.24361>.

- [53] Y. Park, J. S. Bhatia, L. L. Ankile, and P. Agrawal. Dexhub and dart: Towards internet scale robot data collection. *ArXiv*, abs/2411.02214, 2024. URL <https://api.semanticscholar.org/CorpusID:273821640>.
- [54] Y. Ravan, A. Rashid, A. Yu, K. McClennen, G. Huh, K. Yang, Z. Yang, Q. Yu, X. Wang, P. Isola, and G. Yang. Lucid-xr: An extended-reality data engine for robotic manipulation, 2026. URL <https://arxiv.org/abs/2605.00244>.
- [55] X. Jiang, Q. Yuan, E. U. Dincer, H. Zhou, G. Li, X. Li, J. Haag, N. Schreiber, K. Li, G. Neumann, and R. Lioutikov. Iris: An immersive robot interaction system, 2025. URL <https://arxiv.org/abs/2502.03297>.
- [56] T. Chen, M. Tippur, S. Wu, V. Kumar, E. Adelson, and P. Agrawal. Visual dexterity: In-hand reorientation of novel and complex object shapes. *Science Robotics*, 8(84), Nov. 2023. ISSN 2470-9476. doi:10.1126/scirobotics.adc9244. URL <http://dx.doi.org/10.1126/scirobotics.adc9244>.
- [57] E. Todorov, T. Erez, and Y. Tassa. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 5026–5033, 2012. doi:10.1109/IROS.2012.6386109.
- [58] L. G. Rosenzweig, B. Shacklett, W. Xia, and K. Fatahalian. High-throughput batch rendering for embodied ai. In *SIGGRAPH Asia 2024 Conference Papers*, SA ’24, New York, NY, USA, 2024. Association for Computing Machinery. ISBN 9798400711312. doi:10.1145/3680528.3687629. URL <https://doi.org/10.1145/3680528.3687629>.
- [59] Y. Lipman, R. T. Q. Chen, H. Ben-Hamu, M. Nickel, and M. Le. Flow matching for generative modeling, 2023. URL <https://arxiv.org/abs/2210.02747>.
- [60] A. Jaegle, F. Gimeno, A. Brock, O. Vinyals, A. Zisserman, and J. Carreira. Perceiver: General perception with iterative attention. In M. Meila and T. Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 4651–4664. PMLR, 18–24 Jul 2021. URL <https://proceedings.mlr.press/v139/jaegle21a.html>.
- [61] K. Zakka. Mink: Python inverse kinematics based on MuJoCo, 2024. URL <https://github.com/kevinzakka/mink>.
- [62] O. Siméoni, H. V. Vo, M. Seitzer, F. Baldassarre, M. Oquab, C. Jose, et al. DINOv3. *arXiv preprint arXiv:2508.10104*, 2025.

A Appendix

A.1 spd-vr: Simulation Teleoperation System

spd-vr streams a MuJoCo [57] simulation to a Meta Quest 3 headset and maps the operator’s tracked hands onto the simulated robot. All computation runs on a host workstation; the headset runs only a WebXR client that renders the scene and reports hand poses. At load time the client receives every scene body as a mesh, streamed in binary at 60 Hz, and sends back hand poses over a USB tether. The simulation steps at 480 Hz with the `implicitfast` integrator, elliptic friction cones, and one no-slip iteration; control, streaming, and recording run at 60 Hz.

Collection is organized by a task registry: each task defines a natural-language prompt, a target duration, and a reset function that randomizes asset selection, object placement, and physical properties; every sampled value is logged so episodes can be reconstructed exactly. The operator controls recording with a three-button foot pedal (checkpoint, pause, revert/skip); checkpoints are rejected while a hand is in contact with an object, so reverting always restores a contact-free state where the operator can undo a mistake. After collection, we cut spans with more than ten seconds of no hand-object contact and render all clips in parallel with a MuJoCo Warp adaptation of the Madrona batch renderer [58] at 224×168 , preserving instance segmentation masks for the visual augmentations described in Section 3.1.

A.2 spd-75h Dataset

spd-75h comprises 1,930 teleoperated simulation episodes (~ 75 hours) collected across six scenes on the target bimanual dexterous embodiment. Table 2 lists the per-task episode and duration breakdown.

Table 2: spd-75h **dataset statistics**. Per-task episode counts and durations, grouped by scene. Spelling variants are merged into a single SPELLING task, and tasks with fewer than ten episodes are omitted. Durations are reported in minutes at 30 Hz.

Scene	Task	Episodes	Minutes
Jenga	Hollow tower	92	567
	Tower	87	473
	Dominos	107	471
	Criss-cross	103	386
	Handover (L→R)	96	172
	Handover (R→L)	72	109
Spelling Blocks	Spelling	168	587
	Sort and unload	25	144
	Pyramid	32	136
	Sort vowels/consonants	50	109
Mugs	Hang mug	406	491
Dishes	Rack dishes	129	285
	Plate dishes	79	109
Cups	Pyramid	44	109
	Stack two threes	46	67
	Unstack	30	47
Bottles	Toss in bin	350	253
Total		1,916	4,516

A.3 spd-teleop: Real-World Teleoperation System

spd-teleop mirrors the retargeting from spd-vr, and follows the system design from abc [5] for hardware processes. The stack is a set of single-purpose processes — one per arm, hand, camera, and logical component — communicating over ZeroMQ publish-subscribe on local IPC sockets, with a lightweight wire format of a JSON header plus raw array bytes. Each process runs a fixed-rate loop: the leader streams commands at 60 Hz, arm and hand followers servo at 120 Hz, and cameras publish at 30 fps.

Wrists. A WebXR client on the Quest headset streams controller poses at 60 Hz. A pedal press anchors the operator’s wrist pose and the robot’s end-effector pose when tracking engages, after which motion is applied as relative deltas with orientation changes re-expressed in the end-effector frame. Wrist targets are solved using mink-based differential IK [61] (four QP iterations per tick with posture, joint-limit, and velocity tasks), run in a dedicated process, with exponential smoothing and target interpolation for jumps larger than 8 cm.

Fingers. Manus gloves provide 25 tracked keypoints per hand. For each digit, the fingertip position in a palm-centric frame is mapped to a target in the robot hand’s palm frame by a per-operator affine map, fit by least squares from a short calibration routine in which the operator holds prescribed poses. The five fingertip targets then drive mocap-based IK in a fixed-base simulation of the hand, whose resulting 22 joint angles are the hand command.

Robot control. Arm commands are joint positions tracked by per-joint PD gains with a gravity-compensation feedforward torque computed by MuJoCo inverse dynamics on a model that includes the hand as end-effector payload; motors are driven over CAN at 1 Mbit/s with internal servo threads at 250 Hz. Hands are position-controlled through the vendor SDK over Ethernet. Episodes are recorded as per-stream HDF5 with parallel timestamp arrays: camera frames as JPEG at native rate, arm observations at 120 Hz, and commands at 60 Hz, all resampled offline onto the 30 Hz training grid.

A.4 Real-World Evaluation

We evaluate each checkpoint with 20 trials per task from randomized initial object placements. Each trial is scored against the per-task rubric in Table 3, and reported task progress is the achieved score normalized by the task’s maximum.

Table 3: **Evaluation task rubrics.** Max score is the maximum achievable score for the task; reported progress is the achieved score normalized by this maximum.

Task	Setting	Max Score	Scoring Rubric
bottles in bin	4 bottles, 1 bin	4	+1 for each bottle tossed into the bin. 60-second timeout.
plate racking	2 plates, 1 dish rack	4	+1 for lifting each plate; +1 for racking it
cup stacking	6 cups	8	+1 for each cup placed correctly; +1 for each subsequent destack move
jenga playing	1 tower	3	+1 for pushing a middle block out; +1 for pulling it from the other side without collapsing the tower; +1 for placing it on top
mug hanging	1 mug, 1 mug tree	3	+1 for lifting the mug; +1 for the handover; +1 for hanging it on the hook

A.5 Model Architecture

Our policy is a 222M-parameter diffusion transformer that consumes an interleaved multimodal token sequence and denoises future action chunks. Each training sequence spans 256 timesteps at 30 Hz. At every timestep the sequence carries one proprioception token and one previous-action token (both 56-D, normalized); every eighth timestep it additionally carries four pooled tokens per camera and an 8-step noised action chunk.

Vision pathway. Images from the three cameras are encoded by a frozen DINOv3 ViT-B/16 [62] into patch features. Per camera and frame, four learnable queries pool the patch bank into four tokens via cross-attention. To make the pooling context-dependent, the pooled tokens re-attend to the raw patch bank through camera-specific cross-attention blocks interleaved every two trunk blocks, as in [60].

Trunk. The trunk is an 8-block transformer with hidden size 768, 12 attention heads, and MLP expansion factor 4, with modality-specific linear input and output projections. All attention is causal; tokens receive temporal positions from their timestamps via rotary embeddings, and every layer uses sliding-window attention over a 32-timestep window. Following π_0 [2], the action-denoising expert maintains its own 58M-parameter set of weights, while observation tokens share the base trunk weights.

Flow head. Noised action chunks are constructed as $x_t = (1 - t)x_0 + tx_1$ with $x_0 \sim \mathcal{N}(0, I)$ and $t \sim \mathcal{U}[0, 1]$, and the model is supervised to predict the velocity $v = x_1 - x_0$ [59]. Each chunk token receives two additive embeddings: a Gaussian Fourier embedding of the flow time t and a sinusoidal embedding of the position within the chunk, each mapped through a two-layer MLP. During training, all chunks in the sequence are denoised in parallel under the causal mask with independent per-chunk t , which amortizes the cost of processing the 256-timestep context over 32 chunk predictions. To reduce distribution shift from conditioning on history, proprioception and action inputs are perturbed with i.i.d. Gaussian noise ($\sigma = 0.03$) during training.

Inference. At deployment the transformer runs as an incremental engine over a rolling KV cache matched to the 32-timestep training window. Each control tick appends the current observation tokens to the cache; on chunk boundaries the engine integrates the flow ODE with 10 Euler steps and emits the next 8 actions.

A.6 Training Hyperparameters

Table 4 lists the hyperparameters used for simulation pre-training. Table 5 lists only the settings that differ during real-world fine-tuning; all other hyperparameters are inherited from pre-training.

Table 4: **Pre-training hyperparameters.**

Hyperparameter	Value
Batch size	64
Learning rate	1×10^{-3}
Learning rate schedule	constant
Weight decay	0.1
Optimizer	Muon (matrices), AdamW (rest)
Parameters	222M
Parameters (vision encoder)	86M
Parameters (action expert)	58M
EMA half-life	20 steps
Training steps	170k
Sample rate	30 Hz
Action chunk steps	8
Image subsample steps	8
Flow-matching noise schedule	uniform
Observation noise	0.03
Action noise	0.03
Vision backbone	DINOv3 ViT-B/16
Vision queries	4

Table 5: **Real-world fine-tuning hyperparameters.** Only settings that differ from pre-training (Table 4) are listed; all other hyperparameters are inherited. Dataset size is reported per task.

	bottles	plates	cups	jenga	mugs
Training steps	6k	6k	10k	6k	6k
Dataset size (minutes)	72	70	121	48	44
Dataset size (episodes)	270	161	217	193	238