

What Matters When Cotraining Robot Manipulation Policies on Everyday Human Videos?

Richard Li^{1,†}, Aditya Prakash², Andrew Wen¹, Saurabh Gupta², Yilun Du^{3,*}, Pulkit Agrawal^{1,*}

¹Massachusetts Institute of Technology

²University of Illinois Urbana-Champaign

³Harvard University

Abstract: Human video datasets used for cotraining robot manipulation policies largely consist of curated demonstrations where motions are orchestrated to resemble robot behavior and 3D hand poses are captured with specialized hardware. A more plentiful source of data is everyday Internet video, but it is an open question what factors enable transfer from such videos to robots. We investigate this using a new dataset of 532 human videos with 28 hours of high-quality triangulated hand labels and natural motions. We find that hand pose quality affects transfer, but even with accurate hands, the inherent motion gap hinders transfer unless the vision and policy networks specialize to each embodiment. Our cotraining recipe yields consistent improvements, with an absolute success rate gain of 29.7% in the low-robot-data regime across six manipulation tasks.

Keywords: Human-Robot Cotraining, Manipulation, Hand Pose Estimation

1 Introduction

A central challenge in building generalist robotic foundation models is data scarcity. Teleoperated demonstrations are expensive and slow to collect, while reinforcement learning remains difficult to scale due to challenges in reward design, exploration, and sim-to-real transfer. In contrast, Internet videos of humans performing everyday activities are abundant and offer a potentially scalable route to physical intelligence.

Prior work on learning from human video largely falls into two categories. The first uses Internet videos of everyday activities but relies on modular pipelines that decompose a policy into multiple components or stages [1, 2, 3]. The second achieves human-to-robot transfer by cotraining end-to-end policies, but relies on *aligned data*, where human demonstrations are carefully orchestrated to match robot motion, identical cameras are used, and specialized hardware provides accurate 3D hand pose [4, 5]. This setting differs fundamentally from Internet video, which exhibits unconstrained motion, heterogeneous cameras, task mismatch, and noisy hand pose estimates.

A rigorous empirical study of cotraining on Internet video is difficult because ground-truth 3D hand labels are unavailable, and cotraining pipelines can break down with poor labels. We therefore study cotraining in a proxy setting that preserves the key challenges of Internet video—egocentric views, natural motions, camera and task mismatch—while using extrinsic cameras to triangulate high-quality hand labels. Our goal is not to produce a method directly applicable to Internet video today, but rather to understand *what factors can enable* this regime, focusing on two key sources of difficulty: (1) 3D hand pose quality, and (2) the action gap induced by natural human motion and task mismatch. By identifying the bottlenecks, future work can concentrate efforts towards lifting these bottlenecks, e.g. by scaling up training data for hand pose estimators.

*Equal advising.

†Corresponding author. Contact rli14@mit.edu

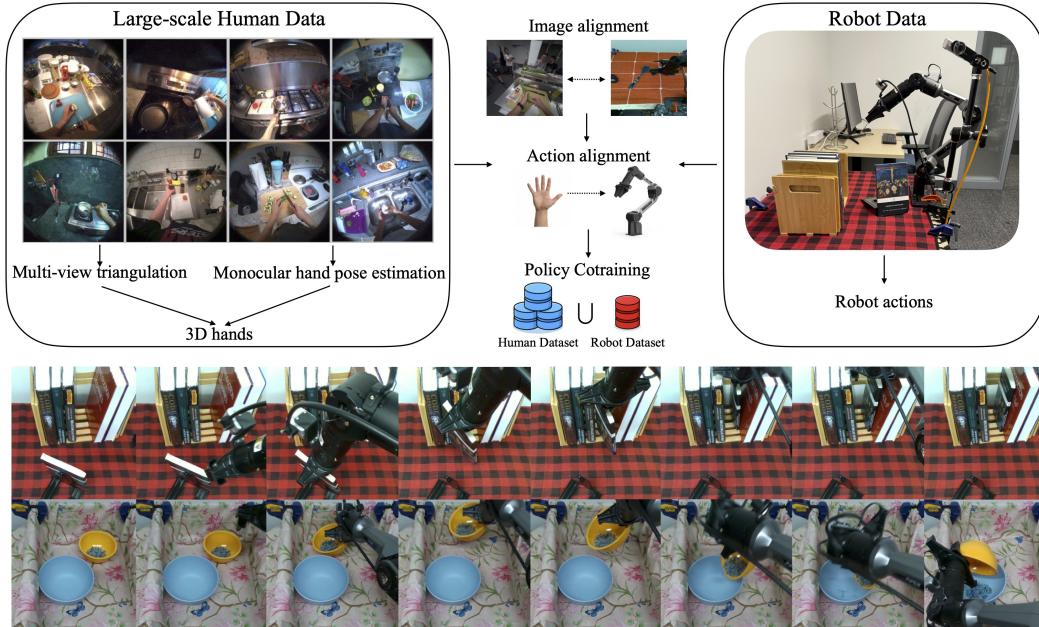


Figure 1: *Top*: System diagram showing data processing and policy cotraining steps. *Bottom*: Rollouts from cotrained policy manipulating unseen objects in unseen scenes. See interactive visualizations and videos: <https://richardrl.github.io/what-matters-cotraining-human-videos/>.

To enable controlled study, we construct a dataset of everyday human videos with high-quality 3D hand poses by triangulating EgoExo4D [6] with a multi-view pipeline. These labels upper bound what monocular hand pose estimation can achieve and can serve as training data for future monocular estimators. We isolate the role of hand quality by comparing our triangulated hands against a state-of-the-art monocular estimator [7].

Even with accurate 3D hand poses, successful transfer requires careful alignment across human and robot data. We identify *scale inconsistency in image space* as a previously overlooked issue when cotraining across datasets with different cameras. Simple image-space scale alignment significantly improves transfer, despite large differences in camera hardware. Additionally, we observe that standard cotraining recipes designed for lab data with smaller motion gaps yield little to no improvement over robot-only training given our natural motion human data. Through ablations, we show that the common design choices of using image bottlenecks and shared action encoders limit transfer. Thus, we propose a cotraining recipe that explicitly accounts for embodiment differences via token-level fusion, embodiment-specific action encoders and decoders, and upweighting robot data.

We evaluate our approach across six real-world manipulation tasks by training on 532 human videos and 3,000 robot demonstrations, and conduct 3,480 real-world rollouts. Our cotraining recipe significantly improves scaling behavior: human data provides the largest benefits in low-robot-data regimes (+29.7% absolute improvement) and continues to help as task-specific robot data scales.

Therefore, our main contributions are:

- We curate a large-scale dataset of everyday human videos with accurate 3D hands, and show our dataset has better robot transfer than a large-scale lab dataset [8].
- We show that higher-quality 3D hand labels yield greater transfer, with a large gap between monocular-estimated and triangulated hands on the same videos.
- We demonstrate the benefit of scale alignment in image space when cotraining across datasets with different cameras, a factor not currently accounted for in leading VLAs [9].

- We show that architectural and loss function decisions enabling embodiment-specific specialization are necessary for significant transfer from natural motion human data, whereas design choices effective for lab demonstrations do not transfer to natural motion videos.

Our overall contribution is to chart a path toward robot cotraining on Internet video by identifying and addressing the key challenges—hand quality and natural motion—that distinguish such data from lab demonstrations.

2 Related Work

Egocentric datasets with hand-object interaction (HOI): Large-scale egocentric video collections fall into two camps: datasets with accurate hand/head poses but limited motion and scene diversity, such as ARCTIC and EgoDex [10, 8], and natural-motion datasets with poor hand labels, such as HoloAssist [11], EpicKitchens [12], Ego4D [13], and EgoExo4D [6]. We build upon EgoExo4D by significantly processing and cleaning the hands to produce a new dataset with natural motions and clean hand labels.

Modular pipelines using HOI videos: Existing works exploit hand motion from Internet videos through modular pipelines that learn intermediate representations such as 3D poses [14, 15], affordances [1, 14, 16, 17, 15, 18, 19], or 2D tracks [3, 2, 20]. For example, [1] learns visual affordances from human Internet videos for post-grasp policies, and [2] generates keyframe-level interaction plans to condition robot policies. We hypothesize the lack of strong 3D hand pose estimators until recently led to this focus on modularity over end-to-end cotraining.

Co-training policies with hand and robot data: Another recent paradigm directly cotrains policies on both human and robot data, requiring 3D hand motion labels from lab datasets [21] or custom setups and data collection devices [8, 4, 22]. Egomimic and Egobridge [4, 5] study techniques for human-robot transfer but rely on small-scale lab data with orchestrated motions, and make assumptions that break as the motion gap grows (Sec. 6, [5]). Other works scale up cotraining but use data with limited scene diversity [23, 8] or noisy hand labels that yield minimal transfer [21, 11]. Our work differs in that we explicitly investigate what can enable cotraining on Internet videos.

3 TriHands Dataset

The main difficulty in studying the impact of hand label quality is the lack of a dataset of everyday human videos with clean hands. We build such a dataset by multiview triangulating such hands on 532 cooking videos from the EgoExo4D dataset [6].

The 2D keypoints used by the EgoExo4D team [24] are the primary bottleneck. We found that 3D model-based pose estimators [25] project to much more accurate 2D keypoints in poor lighting conditions and when the hand partially leaves the field-of-view, which are both common in egocentric and in-the-wild videos (Supp. A). Additionally, 2D hand keypoint estimators degrade significantly under self-occlusion, whereas 3D hand estimators always project to a complete set of 2D keypoints.

Given a strong 3D base model, an out-of-distribution dataset can be accurately annotated using a small number of partial (i.e. the visible subset of hand joints) 2D keypoint labels. We fine-tune the hand pose estimator with a reprojection loss, $\mathcal{L}_{\text{proj}} = \sum_{i \in \mathcal{I}} \|\mathbf{p}_i^{2\text{D}} - \pi_{\mathbf{K}}(\hat{\mathbf{p}}_i^{3\text{D}})\|_1$, where $\pi_{\mathbf{K}}$ is the projection operator with intrinsics \mathbf{K} , and \mathcal{I} is the set of joints with 2D annotations. Fine-tuning is performed with this loss on 4407 frames with partial 2D keypoint labels provided by EgoExo4D, which represents less than 0.144% of the final right-hand triangulated frames we can produce.

We triangulate 3D hands using the egocentric and $M \in \{4, 5\}$ exocentric views via DLT [26] with nonlinear refinement, picking the hand that maximizes camera count subject to reprojection error thresholds (details in Supp. B). After filtering and 0.4s linear interpolation, we obtain 3,042,406 right-hand frames at 30fps (over 28 hours of training data).

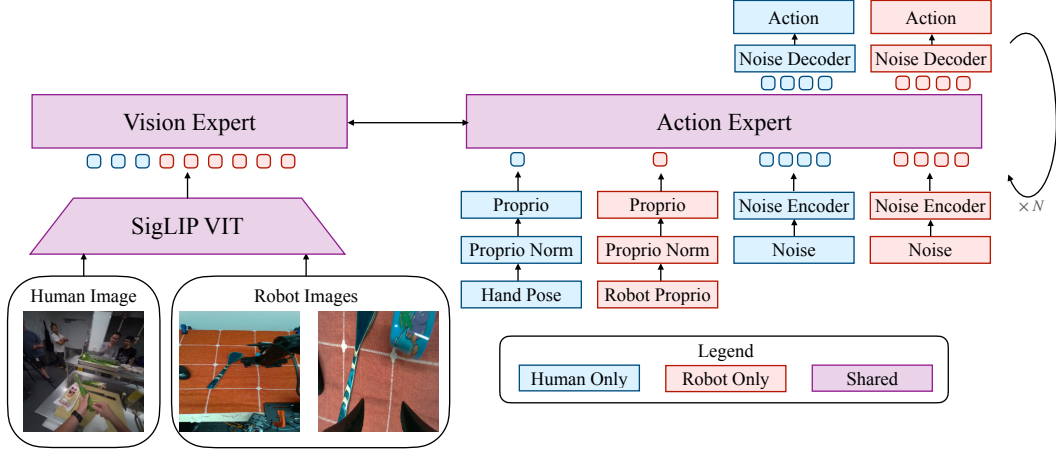


Figure 2: Input-output diagram for inference over human and robot data in our conditional flow matching architecture. Images shown with image-space alignment and resize. Color denotes weight sharing and N denotes inference steps. See Supp. I.

4 Framework

Our framework for cotraining policies on human videos and robot trajectories balances two competing goals: encouraging positive transfer by aligning observations and actions (Sec. 4.2, 4.3), while preventing harmful representation alignment by allowing the network to specialize to each embodiment (Sec. 4.4, 4.5). We adopt the spatial-algebra notation of [27], where a homogeneous transform ${}^bT^a = \begin{bmatrix} {}^bR^a & {}^b\mathbf{p}^a \\ \mathbf{0}^\top & 1 \end{bmatrix}$ represents frame a with respect to, and expressed in, frame b .

4.1 Data streams

We use images from an egocentric moving head-camera for the human data and we use a similarly-positioned egocentric camera for the robot data (Fig. 1). The cameras have significantly different sensor sizes and focal lengths, which makes transfer difficult (Sec. 4.2, Supp. D).

The human raw actions come from either triangulation (Sec. 3) or monocular hand pose estimation (Sec. 5) and must be transformed into a the robot action space to enable cotraining. The robot is a 6-DOF AgileX Piper arm with a parallel-jaw gripper, and its action space is the tool center point (TCP) pose and a discrete grasp command. Concretely, we define ${}^R\mathbf{a} = ({}^{\text{camera}}\mathbf{p}^{\text{TCP}}, {}^{\text{camera}}\mathbf{R}^{\text{TCP}}, g)$, where g is a discrete ternary variable representing *open*, *close*, and *no-op* grasp commands.

4.2 Image-space scale alignment

It is common for the cameras used in the robot and human datasets to be significantly different in terms of field of view and focal length. This motivates the need to align the image-space scale of objects in order to learn shared features. The pinhole equation $Z = \frac{f\Delta X}{\Delta u}$ shows that for a fixed 3D object extent ΔX , the pixel extent Δu is determined by the focal length f and depth Z , so cameras with different focal lengths produce different image-space scales for an object that is the same distance away from the cameras.

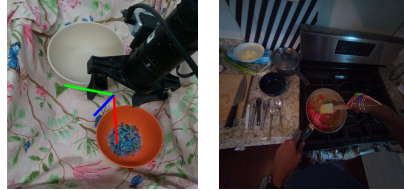
We undistort the human fisheye images to a pinhole projection [28, 29] and place the robot camera at z_{human} from the workspace, where z_{human} is the median wrist depth in the human dataset. However, since the robot camera FOV is much smaller, this distance causes the image to lose scene context. We therefore adjust the robot camera extrinsics: rotating it 90° clockwise so the wider FOV captures the vertical extent of the scene¹, and translating it to depth $z_{\text{robot}} = z_{\text{human}} \frac{f_s H_c}{f_c H_s}$, where H_c, H_s are

¹Hand-object interactions in the human dataset tend to occur towards the bottom of the image.

image heights and f_c, f_s the focal lengths (Supp. F). This approximately matches image-space object scale while retaining the 3D scene extent, and we show in Sec. 6 that it is crucial for transfer.

4.3 Action space alignment

For the network to learn shared representations across embodiments, the human and robot action marginals must share the same support. We map the human 3D hand pose to the robot action space by picking the “middle finger proximal” MANO joint frame with a rotation to match the robot TCP frame (Fig. 3), chosen because it best aligns middle contacts compared to the wrist frame. The remapped human action space in the current camera frame is ${}^H\mathbf{a} = (\text{camera } \mathbf{p}^{\text{middle}}, \text{camera } \mathbf{R}^{\text{middle middle}} \mathbf{R}^{\text{TCP}}, g_{\text{no-op}})$.



(a) Piper TCP. (b) MANO TCP.

Figure 3: Robot TCP frame vs. human TCP frame (not to scale).

Since imitation learning methods predict action chunks [30, 9], we cannot use the above equation directly because head camera motion introduces extreme multimodality.

We stabilize hand actions by transforming per-timestep camera-frame poses to the current head frame. The head poses for TriHands come from EgoExo4D [6], but could also be estimated from monocular video [31, 32, 7]. The human action chunk for observation \mathbf{o}_t is $\mathbf{a}_t, \dots, \mathbf{a}_{t+H} = {}^{c_t}\mathbf{T}^w {}^w\mathbf{T}^{c_{t+H}} {}^{c_{t+H}}\mathbf{a}^{t+H}$.

Despite alignment, a bias remains between the two action distributions: human hand motions are biased towards a camera region corresponding to hand chirality, and camera placement differences cause differences in action depths (Supp. G). We resolve this by mean-centering each dataset independently, filtering to the 1st–99th percentiles, and scaling to ± 1 . Rotations need no normalization since $R \in \text{SO}(3) \Rightarrow R_{ij} \in [-1, 1] \forall i, j \in \{1, 2, 3\}$.

4.4 Cross-embodiment architecture

For an ideal cross-embodiment policy, for human observation o_h and robot observation o_r of the same object in the same pose, we would like to learn an invariant function $f_\theta(o_h) = f_\theta(o_r) = z$ mapping to the embodiment-agnostic object state. If the encoder succeeds, a shared deterministic decoder would be forced to output $g_\phi(z) = {}^R\mathbf{a} = {}^H\mathbf{a}$. This mapping is guaranteed to fail given the large motion gap in our TriHands dataset, where tasks and motions are natural and unconstrained. Thus, embodiment-specific weights are necessary.

Our Transfusion-inspired architecture uses separate FFN experts for visual and action tokens [9, 33] (Fig. 2, Supp. I). It retains the full grid of ViT [34] image tokens and concatenates them with noisy action tokens via shared attention (*token-level fusion*). Embodiment-specific 3-layer MLPs serve as the action and proprioception encoders and decoders.

4.5 Cotraining algorithm and weighted loss

To prevent harmful representational alignment, we implicitly upweight the influence of the robot data. For each minibatch, we sample an equal number of samples independently from each embodiment. This scheme is equivalent to weighting the robot loss by $\frac{N_R + N_H}{N_R}$ and the human loss by $\frac{N_R + N_H}{N_H}$ relative to a uniform sampling strategy. Because the robot dataset is much smaller than the human dataset, this amplifies the influence of the robot data. See Supp. H.

5 Experimental Design

We aim to validate four hypotheses: **H1**: Cotraining on human data increases zero-shot generalization to new objects and backgrounds. **H2**: Human data transfers motion knowledge, instead of only

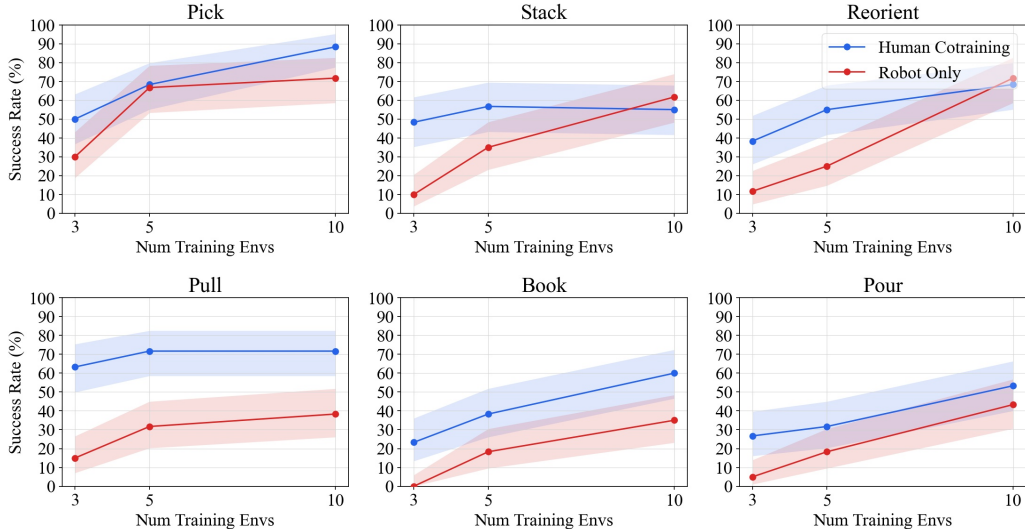


Figure 5: Per-task comparison between human cotraining with triangulated hands and robot-only training (95% Clopper-Pearson CI).

coarse visual features. **H3**: Higher hand quality leads to greater transfer. **H4**: Allowing the network to specialize to each embodiment leads to improved robot task performance.

Tasks and environments: To validate cotraining transfer, we would like to see consistent transfer across a range of tasks. We provide results on six tasks (Fig. 4), ranging from pick-and-place to more intricate tasks requiring complicated rotations and/or reaching the limit of the robot’s kinematic workspace. To test generalization (**H1**), we follow prior work [35] and focus on the controlled setting of within-category object generalization and background generalization. Each environment thus consists of a unique object and background. See Supp. K.

Data sources: Following data guidelines from prior work on robot scaling laws [35], we collect 50 demonstrations per environment. For each task, we have 10 training environments, resulting in a total of 500 demos per task. For approaches where we cotrain with human data in addition to the robot data, we always use RGB images sampled from all 532 videos from the TriHands dataset described above. The hand actions come from triangulation (TriHands) and monocular estimation [7] (**H3**). To further isolate the effect of hand quality, we add calibrated Gaussian noise (fit to the TriHands–HaWoR error distribution) to our triangulated hands at $0.5\times$ and $1.0\times$ the fitted standard deviation (Table 4).

Architecture ablations: To test **H4**, we ablate our token-level fusion and embodiment-specific encoders/decoders (Sec. 4.4) against two alternatives:

(1) an image-bottleneck (*CLS-token*) that pools ViT tokens to a single vector, as is common in prior work [4, 5, 30], and (2) shared action encoder/decoder weights across embodiments.

Experiments: We refer to training on both robot data and human data with our triangulated labels and recipe as *Human Cotraining* (HC), and training on robot data alone as *Robot Only* (RO). To test **H1**, we ablate the effect of human data at increasing robot data levels (Fig. 5). For every task, we

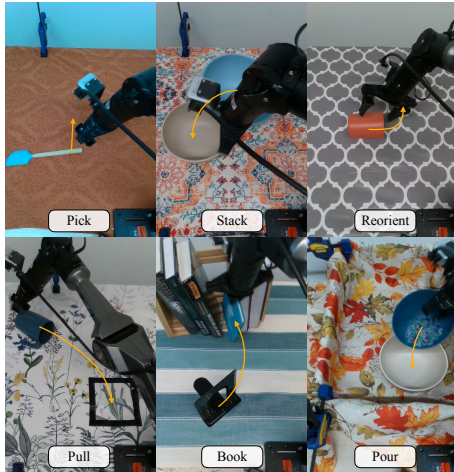


Figure 4: **Task visualizations**. Orange arrow shows representative object motion.

Table 1: Our cotraining recipe vs. others. Parentheses compare to Ours. All methods except Robot Only cotrained with TriHands, 3 envs of robot data. 95% Gaussian CI.

	Mean	Pick	Stack	Pull	Reorient	Book	Pour
Ours	41.5±12.1%	50.0%	48.0%	63.0%	38.0%	23.0%	26.7%
Ablations							
Robot only (vs. HC)	12.0±8.2%	30.0%	10.0%	15.0%	11.7%	0.00%	5.00%
CLS-token (vs. token fusion)	14.7±18.6%	0.00%	6.67%	61.7%	5.00%	6.67%	8.33%
Datasets							
EgoDex [8] (vs. TriHands)	19.4±13.4%	33.3%	45.0%	18.3%	8.33%	11.7%	0.00%
Methods							
HaWoR [7] (vs. TriHands)	24.7±16.9%	53.3%	10.0%	48.3%	15.0%	20.0%	1.70%
EgoBridge [5] (vs. BC)	16.6±10.7%	28.0%	10.0%	35.0%	20.0%	0.00%	6.67%
PiZero [9] (vs. sep. enc/dec)	17.5±6.0%	21.7%	21.7%	23.3%	15.0%	20.0%	3.30%

train on {3, 5, 10} environments (50 demos each), and evaluate in 4 test environments (15 rollouts each). To test **H2**, we analyze 90 rollouts each (across all 6 tasks) for HC and RO, for environments where the HC has significantly higher mean success rate than RC. We categorize failures as *global* (robot does not servo near the object) or *local* (robot fails task by a few cm). If HC often resolves RO’s local errors, that would indicate fine-grained motion transfer instead of coarse visual transfer.

Baselines: *CLS-token*: Vision tokens are attention-pooled to a single vector (image-bottleneck). *EgoDex* [8]: 145 hours of lab-collected egocentric data with limited scene diversity (vs. our 28 hours of natural scenes). *HaWoR* [7]: Human videos labelled with a monocular hand estimator instead of triangulation. *EgoBridge* [5]: Representational alignment loss based on action similarity; requires CLS-token architecture due to its optimal transport formulation. *PiZero* [9]: Shared action encoders and decoders across embodiments.

6 Results

Human cotraining significantly improves task performance over robot-only training.

From Table 2, we observe consistent gains across all robot data regimes using our recipe, with the largest improvements in the low-data setting: human cotraining improves absolute

success rates by 20%–48% at the 3-env level (Figure 5). The Pull task benefits particularly strongly from our human data and preprocessing, even across different architectures and methods (Table 1). In Figure 6, we provide a qualitative analysis of rollouts for the Human Cotraining model.

At higher robot data scales, the gap narrows—expected from prior work on cross-task cotraining [36], since the robot-only baseline trains on same-task, same-distribution demos. Despite these diminishing returns, improving zero-shot transfer in the low-data regime remains practically useful when paired with RL post-training [37]. Finally, TriHands outperforms EgoDex across all tasks despite 5× fewer hours of data, suggesting scene diversity matters more than scale (Table 1).

There is motion transfer on simple tasks, but motion transfer is inconclusive for our most complex task. We visually analyze the errors and find 100% of the Robot Only errors on 5/6 tasks are local errors. On these tasks, Human Cotraining outperforms Robot Only $82.8 \pm 11.4\%$ to $29.2 \pm 13.7\%$, indicating motion transfer. For the remaining task (Pour), Robot Only struggles to even servo to the object, so we cannot use this argument. We believe the overall recipe may be less effective on complex tasks due to the reliance on kinematic retargeting and the larger motion gap, and leave further investigation to future work.

Table 2: Success rates for HC and RO (95% Gaussian CI).

Method	3 envs	5 envs	10 envs
HC	41.7±12.1%	53.6±12.7%	66.1±10.5%
RO	12.0±8.2%	32.5±14.5%	53.6±13.4%

Hand quality matters, but the current state-of-the-art monocular hand estimator still shows transfer.

Monocular-estimated hands performance is worse when averaged across tasks compared to the triangulated hands (Table 1). However, the mean success rate for HaWoR is still higher than Robot Only (24.7% vs 11.9%) (Table 2). This indicates even noisy hand labels provide some benefit across certain tasks but greater improvements in hand pose estimation quality will unlock better transfer across other tasks. Given the metrics in Table 3, it is unsurprising the monocular transfer is worse — the hand joint translations in our 1-second prediction horizon can often be dominated by the noise in the depth values. Table 4 shows that transfer degrades monotonically with noise level, further confirming that higher quality hands lead to greater transfer.

Embodiment-specialization significantly increases human-to-robot transfer.

Prior works on human data with more robot-aligned motion use image-bottleneck architectures [4, 5, 30] and shared action decoders across embodiments [9]. We find that our token-fusion architecture outperforms the CLS-token bottleneck on all tasks except Pull (Table 1), likely because the large motion gap makes human and robot action chunks distinguishably different, and token-level fusion allows the model to specialize visual understanding per embodiment. Additionally, untying the action encoders and decoders leads to large improvements in 5 out of 6 tasks, for a similar reason.

Other ablation experiments Scale-aligning the human fisheye images with an extent-matched pinhole camera doubles performance over Robot Only (Table 5); a heavily-misaligned pinhole yields no transfer benefit. EgoBridge [5] improves over the CLS-token baseline on 4/6 tasks but lags behind our BC-only recipe (Table 1), likely because the larger action gap in our data makes its optimal transport loss based on action similarity unreliable.

7 Conclusion

We present a dataset of everyday human videos with accurate triangulated hand labels, and present a framework for successfully training on such videos with natural motions. Extensive experiments show improved zero-shot generalization to novel objects and backgrounds when cotraining with our human videos. Our hand quality experiments show that modern hand-pose estimators are starting to show transfer on natural Internet-type videos, and further improvements in hand pose estimation should unlock greater transfer. For future work, we seek to find new representational alignment and action retargeting techniques that work with the complexity of everyday videos. Overall, we believe our work analyzes difficulties and suggests solutions for Internet video cotraining.

Table 3: HaWoR hand pose error metrics (mm) using triangulated 3D hands as ground-truth.

MPJPE	PA-MPJPE	W-MPJPE	WA-MPJPE
185.67	15.72	161.85	77.86

Table 4: Effect of hand pose noise on transfer. 95% Gaussian CI.

Noise level	Mean	Reorient	Pick bowl
0.0	48.3 ± 19.6%	38.3%	58.3%
0.5	30.0 ± 13.1%	23.3%	36.7%
1.0	20.0 ± 6.5%	16.7%	23.3%

Table 5: Pinhole camera ablation (95% Clopper-Pearson CI). Supp. E.

Human Camera Model	Mean SR
Extent-Matched Pinhole (Ours)	51.2% [39.8, 62.6]
Medium Misalignment Pinhole	40.0% [29.2, 51.6]
Heavy Misalignment Pinhole	25.0% [16.0, 35.9]
Robot-Only Data	26.2% [17.0, 37.3]

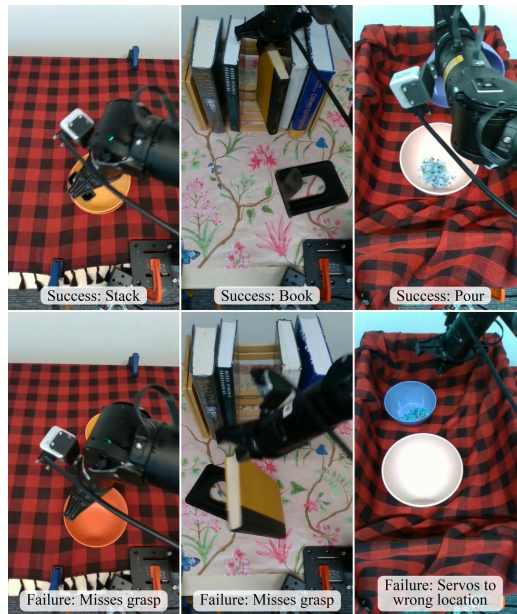


Figure 6: HC rollouts on unseen environments.

Acknowledgments

Richard Li was supported by the NSF Institute for Artificial Intelligence and Fundamental Interactions (Grant No. PHY-2019786) and the Felicis Scholars program. Aditya Prakash and Saurabh Gupta were supported by NSF Grant IIS-2007035. We thank John Marangola for his advice on the robot setup, and Antonia Bronars and Branden Romero for paper writing suggestions.

References

- [1] J. Shi, Z. Zhao, T. Wang, I. Pedroza, A. Luo, J. Wang, J. Ma, and D. Jayaraman. Zeromimic: Distilling robotic manipulation skills from web videos. In *IEEE International Conference on Robotics and Automation, ICRA*, 2025.
- [2] H. Bharadhwaj, A. Gupta, V. Kumar, and S. Tulsiani. Towards generalizable zero-shot manipulation via translating human interaction plans. In *IEEE International Conference on Robotics and Automation, ICRA*, 2024.
- [3] H. Bharadhwaj, R. Mottaghi, A. Gupta, and S. Tulsiani. Track2act: Predicting point tracks from internet videos enables generalizable robot manipulation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2024.
- [4] S. Kareer, D. Patel, R. Punamiya, P. Mathur, S. Cheng, C. Wang, J. Hoffman, and D. Xu. Egomimic: Scaling imitation learning via egocentric video. In *IEEE International Conference on Robotics and Automation, ICRA*, 2025.
- [5] R. Punamiya, D. Patel, P. Aphiwetsa, P. Kuppili, L. Y. Zhu, S. Kareer, J. Hoffman, and D. Xu. Ego-bridge: Domain adaptation for generalizable imitation from egocentric human data. In *Human to Robot: Workshop on Sensorizing, Modeling, and Learning from Humans*, 2025.
- [6] K. Grauman, A. Westbury, L. Torresani, K. Kitani, J. Malik, T. Afouras, K. Ashutosh, V. Baiyya, S. Bansal, B. Boote, et al. Ego-exo4d: Understanding skilled human activity from first-and third-person perspectives. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 19383–19400, 2024.
- [7] J. Zhang, J. Deng, C. Ma, and R. A. Potamias. Hawor: World-space hand motion reconstruction from egocentric videos. *arXiv preprint arXiv:2501.02973*, 2025.
- [8] R. Hoque, P. Huang, D. J. Yoon, M. Sivapurapu, and J. Zhang. Egodex: Learning dexterous manipulation from large-scale egocentric video. *arXiv preprint arXiv:2505.11709*, 2025.
- [9] K. Black, N. Brown, D. Driess, A. Esmail, M. Equi, C. Finn, N. Fusai, L. Groom, K. Hausman, B. Ichter, et al. $\pi 0$: A vision-language-action flow model for general robot control. corr, abs/2410.24164, 2024. doi: 10.48550. *arXiv preprint ARXIV.2410.24164*.
- [10] Z. Fan, O. Taheri, D. Tzionas, M. Kocabas, M. Kaufmann, M. J. Black, and O. Hilliges. Arctic: A dataset for dexterous bimanual hand-object manipulation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12943–12954, 2023.
- [11] X. Wang, T. Kwon, M. Rad, B. Pan, I. Chakraborty, S. Andrist, D. Bohus, A. Feniello, B. Tekin, F. V. Frujeri, N. Joshi, and M. Pollefeys. Holoassist: an egocentric human interaction dataset for interactive AI assistants in the real world. In *IEEE/CVF International Conference on Computer Vision, ICCV*, 2023.
- [12] D. Damen, H. Doughty, G. M. Farinella, S. Fidler, A. Furnari, E. Kazakos, D. Moltisanti, J. Munro, T. Perrett, W. Price, et al. Scaling egocentric vision: The epic-kitchens dataset. In *Proceedings of the European conference on computer vision (ECCV)*, pages 720–736, 2018.

- [13] K. Grauman, A. Westbury, E. Byrne, Z. Chavis, A. Furnari, R. Girdhar, J. Hamburger, H. Jiang, M. Liu, X. Liu, et al. Ego4d: Around the world in 3,000 hours of egocentric video. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18995–19012, 2022.
- [14] M. K. Srirama, S. Dasari, S. Bahl, and A. Gupta. HRP: human affordances for robotic pre-training. In *Robotics: Science and Systems (RSS)*, 2024.
- [15] A. Kannan, K. Shaw, S. Bahl, P. Mannam, and D. Pathak. DEFT: dexterous fine-tuning for hand policies. In *Conference on Robot Learning, (CoRL)*, 2023.
- [16] R. Mendonca, S. Bahl, and D. Pathak. Structured world models from human videos. In *Robotics: Science and Systems (RSS)*, 2023.
- [17] S. Bahl, R. Mendonca, L. Chen, U. Jain, and D. Pathak. Affordances from human videos as a versatile representation for robotics. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR*, 2023.
- [18] M. Goyal, S. Modi, R. Goyal, and S. Gupta. Human hands as probes for interactive object understanding. In *Computer Vision and Pattern Recognition (CVPR)*, 2022.
- [19] M. Chang, A. Prakash, and S. Gupta. Look ma, no hands! agent-environment factorization of egocentric videos. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.
- [20] C. Wen, X. Lin, J. I. R. So, K. Chen, Q. Dou, Y. Gao, and P. Abbeel. Any-point trajectory modeling for policy learning. In *Robotics: Science and Systems (RSS)*, 2024.
- [21] R. Yang, Q. Yu, Y. Wu, R. Yan, B. Li, A.-C. Cheng, X. Zou, Y. Fang, H. Yin, S. Liu, et al. Egovla: Learning vision-language-action models from egocentric human videos. *arXiv preprint arXiv:2507.12440*, 2025.
- [22] T. Tao, M. K. Srirama, J. J. Liu, K. Shaw, and D. Pathak. Dexwild: Dexterous human interactions for in-the-wild robot policies. *arxiv:2505.07813*, 2025.
- [23] R.-Z. Qiu, S. Yang, X. Cheng, C. Chawla, J. Li, T. He, G. Yan, D. J. Yoon, R. Hoque, L. Paulsen, et al. Humanoid policy~ human policy. *arXiv preprint arXiv:2503.13441*, 2025.
- [24] A. Sengupta, F. Jin, R. Zhang, and S. Cao. mm-pose: Real-time human skeletal posture estimation using mmwave radars and cnns. *IEEE sensors journal*, 20(17):10032–10044, 2020.
- [25] J. Romero, D. Tzionas, and M. J. Black. Embodied hands: Modeling and capturing hands and bodies together. *ACM Transactions on Graphics, (Proc. SIGGRAPH Asia)*, 36(6), Nov. 2017.
- [26] R. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, Cambridge, UK, 2nd edition, 2003.
- [27] R. Tedrake. *Robotic Manipulation*. 2024. URL <http://manipulation.mit.edu>.
- [28] J. Kannala and S. Brandt. A generic camera calibration method for fish-eye lenses. In *Proceedings of the 17th International Conference on Pattern Recognition, 2004. ICPR 2004.*, volume 1, pages 10–13. IEEE, 2004.
- [29] R. Szeliski. *Computer vision: algorithms and applications*. Springer Nature, 2022.
- [30] C. Chi, Z. Xu, S. Feng, E. Cousineau, Y. Du, B. Burchfiel, R. Tedrake, and S. Song. Diffusion policy: Visuomotor policy learning via action diffusion. *The International Journal of Robotics Research*, page 02783649241273668, 2023.
- [31] Z. Teed and J. Deng. Droid-slam: Deep visual slam for monocular, stereo, and rgb-d cameras. *Advances in neural information processing systems*, 34:16558–16569, 2021.

- [32] D. Maggio, H. Lim, and L. Carlone. Vggt-slam: Dense rgb slam optimized on the sl (4) manifold. *arXiv preprint arXiv:2505.12549*, 2025.
- [33] C. Zhou, L. Yu, A. Babu, K. Tirumala, M. Yasunaga, L. Shamis, J. Kahn, X. Ma, L. Zettlemoyer, and O. Levy. Transfusion: Predict the next token and diffuse images with one multi-modal model. *arXiv preprint arXiv:2408.11039*, 2024.
- [34] A. Dosovitskiy. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.
- [35] Y. Hu, F. Lin, P. Sheng, C. Wen, J. You, and Y. Gao. Data scaling laws in imitation learning for robotic manipulation. *arXiv preprint arXiv:2410.18647*, 2024.
- [36] J. Barreiros, A. Beaulieu, A. Bhat, R. Cory, E. Cousineau, H. Dai, C.-H. Fang, K. Hashimoto, M. Z. Irshad, M. Itkina, et al. A careful examination of large behavior models for multitask dexterous manipulation. *arXiv preprint arXiv:2507.05331*, 2025.
- [37] P. Intelligence, A. Amin, R. Aniceto, A. Balakrishna, K. Black, K. Conley, G. Connors, J. Darpinian, K. Dhabalia, J. DiCarlo, et al. $\pi_{0.6}^*$: A vla that learns from experience. *arXiv preprint arXiv:2511.14759*, 2025.
- [38] R. A. Potamias, J. Zhang, J. Deng, and S. Zafeiriou. Wilor: End-to-end 3d hand localization and reconstruction in-the-wild. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 12242–12254, 2025.
- [39] T. Cheng, D. Shan, A. Hassen, R. Higgins, and D. Fouhey. Towards a richer 2d understanding of hands at scale. *Advances in Neural Information Processing Systems*, 36:30453–30465, 2023.

A Hand label comparison

Ego-Exo4D provides 2D keypoint labels but does not release its fine-tuned MMPose model. Therefore, to demonstrate that our 3D-projected keypoints (**Ours**) are superior to their 2D keypoints (**Ego-Exo4D MMPose**), we conduct a human pairwise evaluation with 1700 trials and 17 subjects. We allow ties and exclude them from calculations. $P(\mathbf{Ours\ wins}) = 76.3\%$ (95% CI [74.0%, 78.6%]).

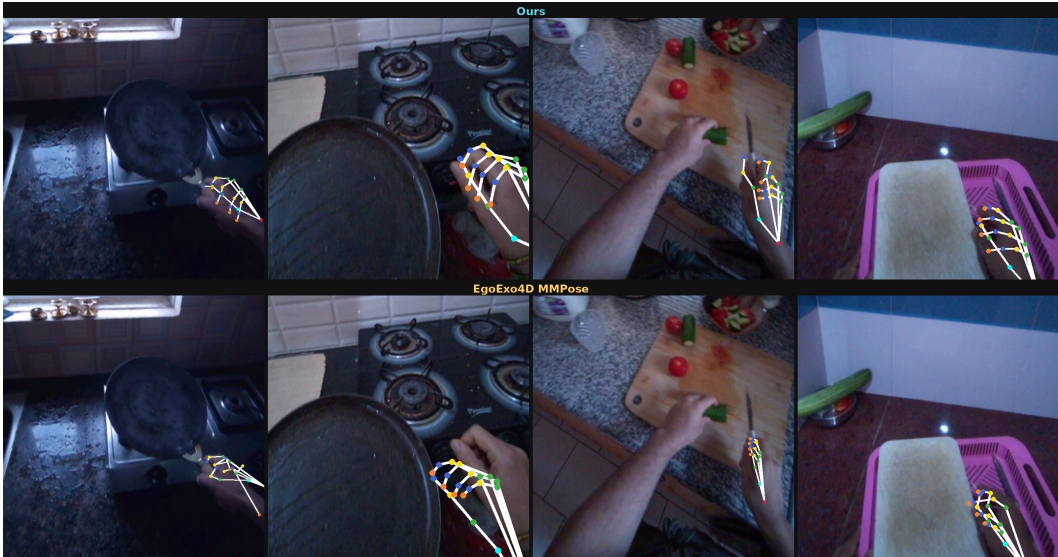


Figure 7: Top-row are ego-view projected keypoints from our 3D hands, and bottom row are corresponding projected keypoints from the default EgoExo4D 3D hand keypoints.

B TriHands triangulation pipeline

Each episode in EgoExo4D comes with one egocentric view from a head-mounted camera and $M \in \{4, 5\}$ exocentric views. For a given set of 2D hand-keypoints associated with a set of cameras, we use the direct-linear transformation (DLT) algorithm [26] as an efficient initial triangulation estimate then refine triangulated points with reprojection error above a threshold using nonlinear optimization.

To select the most accurate triangulation among all camera subsets, we incorporate several safeguards. Minimizing reprojection error alone is insufficient, as a set of cameras can exhibit low reprojection error yet yield poor 3D reconstructions due to small 2D keypoint errors. Instead, we maximize the number of cameras used while enforcing reprojection error thresholds. In addition, EgoExo4D videos often contain secondary individuals near the ego-camera wearer. To avoid their hands, we require the selected camera set to include valid hand detections for the egocentric view.

For a subset $S \subset C$ of the total set of cameras C , we set the reprojection threshold that must be satisfied for each camera c to be $0.01 \cdot \max(H_c, W_c)$, where H_c and W_c are the camera image dimensions in pixels. To simplify experiments, we only use right-hand human hand labels to train a single robot arm. After applying up to 0.4s linear interpolation between 3D joints of triangulated hands, we end up with 3,042,406 frames of reconstructed right hands at 30fps, which corresponds to over 28 hours of usable data for training.

Some additional factors we found to be important for our triangulation pipeline:

1. Most modern 3D hand pose estimators begin reconstruction from a cropped image of just the hand(s). We found the default Ultralytics YOLO v11 bounding box estimator included

Algorithm 1 Multiview Hand Triangulation Pipeline

Require: Multi-view video $\mathcal{V} = \{V_{\text{ego}}, V_{\text{exo}_1}, \dots, V_{\text{exo}_N}\}$, intrinsics $\{\mathbf{K}_c\}$, extrinsics $\{\mathbf{T}_c\}$ for each camera c

Ensure: 3D keypoints \mathbf{J}_{3D} , MANO parameters θ
where c denotes both a camera and its associated image, and

$$\rho_{\text{H}}(r) = \begin{cases} \frac{1}{2}r^2 & |r| \leq \delta \\ \delta|r| - \frac{1}{2}\delta^2 & |r| > \delta \end{cases} \text{ (Huber loss)}$$

// Stage 1: Hand Detection

- 1: **for all** frame f , camera c **do**
- 2: $\mathbf{B}_c^f \leftarrow \text{HANDDETECTOR}(V_c[f])$ ▷ bounding boxes
- 3: **end for**

// Stage 2: Single-View 3D Estimation

- 4: **for all** frame f , camera c , hand $h \in \{L, R\}$ **do**
- 5: $I_{\text{patch}} \leftarrow \text{CROP}(V_c[f], \mathbf{B}_c^f[h])$
- 6: $\mathbf{J}_{2D}, \theta_{\text{init}} \leftarrow \text{WiLoR}(I_{\text{patch}})$ ▷ 2D joints & MANO
- 7: **end for**

// Stage 3: Multiview Triangulation

- 8: **for all** frame f , hand h **do**
- 9: $\mathcal{C} \leftarrow \{c : \mathbf{B}_c^f[h] \text{ exists}\}$
- 10: **if** $|\mathcal{C}| < 2$ **then continue**
- 11: **end if**
- 12: **for all** $c \in \mathcal{C}$ **do**
- 13: $\mathbf{P}_c \leftarrow \mathbf{K}_c[\mathbf{I}|\mathbf{0}]\mathbf{T}_c^{-1}$
- 14: **end for**
- 15: **for all** subset $\mathcal{S} \subseteq \mathcal{C}$ with $c_{\text{ego}} \in \mathcal{S}, |\mathcal{S}| \geq 2$ **do**
- 16: **for all** joint $j = 1, \dots, 21$ **do**
- 17: $\mathbf{X}_j \leftarrow \text{DLT}(\{\mathbf{P}_c, \mathbf{J}_{2D}[c, j]\}_{c \in \mathcal{S}})$
- 18: $\mathbf{X}_j \leftarrow \arg \min_{\mathbf{X}} \sum_c \rho_{\text{H}}(\|\pi_c(\mathbf{X}) - \mathbf{J}_{2D}[c, j]\|)$
- 19: $\epsilon_j[c] \leftarrow \|\pi_c(\mathbf{X}_j) - \mathbf{J}_{2D}[c, j]\|$
- 20: **end for**
- 21: **end for**
- 22: $\mathcal{S}^* \leftarrow \arg \max_{|\mathcal{S}|} \text{s.t. } \geq 95\% \text{ joints: } \epsilon_j[c] < \tau_c$
- 23: $\mathbf{J}_{3D}[h, f] \leftarrow \mathbf{X}[\mathcal{S}^*]$
- 24: **end for**

// Stage 4: Temporal Interpolation

- 25: $\mathbf{J}_{3D} \leftarrow \text{SLERP}(\mathbf{J}_{3D}, g_{\text{max}} = 12)$ ▷ max gap: 12 frames

// Stage 5: Inverse Kinematics

- 26: **for all** frame f , hand h **do**
- 27: $\theta_0 \leftarrow \theta_{\text{init}}[c_{\text{ego}}, h, f]$ ▷ HaWoR init
- 28: $\mathbf{J}_{\text{tgt}} \leftarrow \mathbf{T}_{\text{cam}}^{-1} \mathbf{J}_{3D}[h, f]$
- 29: $\theta[h, f] \leftarrow \arg \min_{\theta} \|\mathcal{M}(\theta) - \mathbf{J}_{\text{tgt}}\|^2$
- 30: **end for**
- 31: **return** \mathbf{J}_{3D}, θ

with our 3D hand model WiLoR [38] to have an extremely high rate of false negatives (missing bounding box detections) as well as chirality errors. We addressed this by swapping the Ultralytics model for a version of Hands23 [39] finetuned on EpicKitchens.

2. Due to our swapping of the bounding box estimator, and due to different padding conventions for Ultralytics YOLO and Hands23, the cropped images going into WiLoR became out-of-distribution, and 3D hand reconstructions were of poor quality. We addressed this by training a small bounding box translation model that took the chirality, bounding box extents, and bounding box location of a Hands23 bounding box and outputted corresponding Ultralytics YOLO bounding box extents. We supervised this translation model by running Ultralytics YOLO and Hands23 on a subset of the videos and finding the intersection set of frames where both models detected hands.



Figure 8: TriHands triangulation visualizations (first five scenes).

3. Our triangulation pipeline produces triangulated 3D joints. To convert this into MANO parameters, we run GPU-batched inverse kinematics through the MANO model. Due to nonconvexity, this optimization often fails from random initialization - we initialize the optimization by using MANO parameter prediction from HaWoR [7], which has relatively accurately metric space parameters compared to WiLoR [38].

C Triangulation visualizations

We visualize 2D projections in the egocamera of our triangulations in Fig. 8, 9.



Figure 9: TriHands triangulation visualizations (remaining four scenes).

D Camera specifications

The human camera is an RGB fisheye camera with 110° HFOV and 110° VFOV, while the robot camera is an RGB RealSense D435i camera with 87° HFOV and 58° VFOV.

E Camera ablation details

We compare results cotraining on the pick spatula task, with the following intrinsics (h, w, f_x, f_y) per baseline: extent-matched (1280, 720, 690, 627), medium (1280, 720, 460, 460), heavy (720, 720, 211, 211). In addition, for the heavy misalignment model, we do not crop to fix the aspect ratio differences between human and robot camera.

F Image-space alignment extrinsic derivation

Consider a human camera with focal length f_c and vertical resolution H_c , and a robot camera with parameters f_s and H_s . Suppose the relevant portion of the human scene lies at depth z_{human} with respect to the camera frame (OpenCV camera frame convention). The top and bottom image rows,

v_T^c and v_B^c , correspond to 3D heights:

$$Y_T^c = \frac{z_{\text{human}} v_T^c}{f_c}, \quad Y_B^c = \frac{z_{\text{human}} v_B^c}{f_c},$$

so the vertical 3D span of the visible region is

$$L^c = Y_T^c - Y_B^c = \frac{z_{\text{human}} H_c}{f_c}.$$

We seek a robot-camera depth z_{robot} such that its captured region has the same vertical 3D span. Because its pixel span is $H_s = v_T^s - v_B^s$,

$$\frac{z_{\text{robot}} H_s}{f_s} = L^c.$$

Solving gives

$$z_{\text{robot}} = z_{\text{human}} \frac{H_c}{f_c} \frac{f_s}{H_s}.$$

This depth translation aligns the robot camera so that both cameras observe an equivalent 3D vertical extent.

G Action marginal distributions visualization

We visualize the distribution of human hand (MANO TCP) translations and robot TCP frame translations, before our mean-centering and unit-scaling normalization, in Fig. 10.

H Cotraining algorithm and loss function

The conditional flow-matching imitation learning loss for a single dataset \mathcal{D}_i , with observations o_t and action chunks A_t , is:

$$\mathcal{L}_{\mathcal{D}_i}(\theta) = \mathbb{E}_{\substack{(o_t, A_t) \sim \mathcal{D}_i \\ \tau \sim \text{Beta}(\alpha, \beta) \\ A_t^\tau \sim q(\cdot | A_t)}} \left[\|v_\theta(A_t^\tau, o_t) - u(A_t^\tau | A_t)\|_2^2 \right]$$

Uniformly sampling from the concatenated human and robot datasets is equivalent to sampling from a mixture-of-empirical-distributions:

$$\begin{aligned} D_R \cup D_H &= \frac{1}{n_R + n_H} \left(\sum_{i=1}^{n_R} \delta(x - x_i) + \sum_{j=1}^{n_H} \delta(x - x_j) \right) \\ &= \frac{n_R}{n_R + n_H} \left(\frac{1}{n_R} \sum_{i=1}^{n_R} \delta(x - x_i) \right) \\ &\quad + \frac{n_H}{n_R + n_H} \left(\frac{1}{n_H} \sum_{j=1}^{n_H} \delta(x - x_j) \right) \\ &= \frac{n_R}{n_R + n_H} D_R(x) + \frac{n_H}{n_R + n_H} D_H(x). \end{aligned}$$

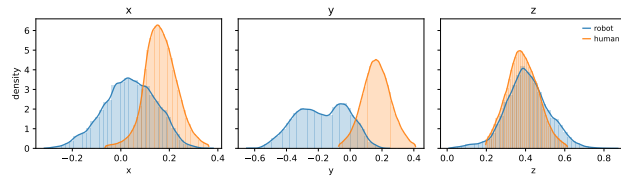
The overall loss would then be:

$$\mathcal{L}_{\mathcal{D}_H \cup \mathcal{D}_R}(\theta) = \frac{n_R}{n_R + n_H} \mathcal{L}_{D_R} + \frac{n_H}{n_R + n_H} \mathcal{L}_{D_H}$$

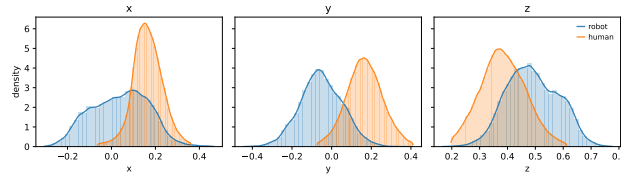
By independently sampling uniform batches from each dataset and summing the losses, we optimize instead:

$$\mathcal{L}_{\text{weighted}}(\theta) = \mathcal{L}_{D_R} + \mathcal{L}_{D_H}$$

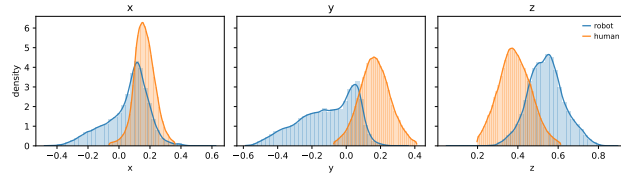
The full pseudo-code for our cotraining algorithm is in Alg. 2.



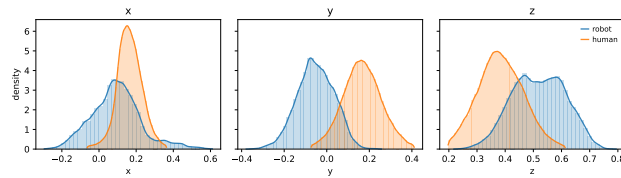
(a) Pick



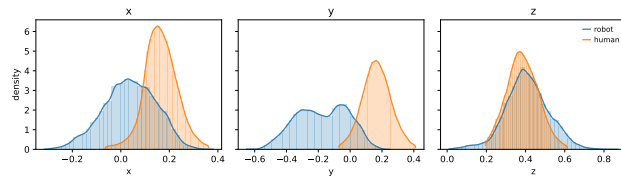
(b) Stack



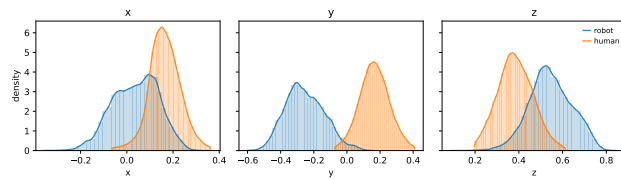
(c) Pull



(d) Reorient



(e) Book



(f) Pour

Figure 10: Comparison of robot and human action marginals in XYZ camera coordinates across all six tasks.

Algorithm 2 Cotraining conditional flow-matching policy

Require: Human dataset \mathcal{D}_H , robot dataset \mathcal{D}_R
Require: Flow-matching policy v_θ , target field $u(\cdot | A_t)$
Require: Action noising distribution $q(\cdot | A_t)$
Require: Timestep distribution $\text{Beta}(\alpha, \beta)$
Require: Batch sizes B_H, B_R
Require: Learning rates η_{vis} (SigLIP encoder) and η_{rest} (remaining modules)

- 1: Initialize v_θ : vision encoder initialized with SigLIP, remaining modules from scratch
- 2: Initialize Adam optimizer with parameter groups:
- 3: Adam($\{(\theta_{\text{vis}}, \eta_{\text{vis}}), (\theta_{\text{rest}}, \eta_{\text{rest}})\}$)
- 4: **while** stopping criterion not met **do**
- 5: **// Human batch**
- 6: Sample mini-batch $\{(o_t^{H,i}, A_t^{H,i})\}_{i=1}^{B_H} \sim \mathcal{D}_H$
- 7: Sample timesteps $\{\tau^{H,i}\}_{i=1}^{B_H} \sim \text{Beta}(\alpha, \beta)$
- 8: Sample noisy actions $\{A_t^{\tau, H,i}\}_{i=1}^{B_H} \sim q(\cdot | A_t^{H,i})$
- 9: Compute targets:
- 10: $u^{H,i} \leftarrow u(A_t^{\tau, H,i} | A_t^{H,i})$
- 11: $\mathcal{L}_H \leftarrow \frac{1}{B_H} \sum_{i=1}^{B_H} \|v_\theta(A_t^{\tau, H,i}, o_t^{H,i}) - u^{H,i}\|_2^2$
- 12:
- 13: **// Robot batch**
- 14: Sample mini-batch $\{(o_t^{R,i}, A_t^{R,i})\}_{i=1}^{B_R} \sim \mathcal{D}_R$
- 15: Sample timesteps $\{\tau^{R,i}\}_{i=1}^{B_R} \sim \text{Beta}(\alpha, \beta)$
- 16: Sample noisy actions $\{A_t^{\tau, R,i}\}_{i=1}^{B_R} \sim q(\cdot | A_t^{R,i})$
- 17: Compute targets:
- 18: $u^{R,i} \leftarrow u(A_t^{\tau, R,i} | A_t^{R,i})$
- 19: $\mathcal{L}_R \leftarrow \frac{1}{B_R} \sum_{i=1}^{B_R} \|v_\theta(A_t^{\tau, R,i}, o_t^{R,i}) - u^{R,i}\|_2^2$
- 20:
- 21: **// Cotraining update**
- 22: $\mathcal{L} \leftarrow \mathcal{L}_H + \mathcal{L}_R$
- 23: Adam.update($\nabla_\theta \mathcal{L}$)
- 24: **end while**
- 25: **return** θ

I Architecture details

Our architecture follows a Transfusion-inspired design that jointly models visual observations and continuous actions using a shared transformer backbone with modality-specific weights. The model operates on trajectories from two embodiments, human (H) and robot (R). RGB images are encoded using a shared SigLIP ViT encoder E_{img} , producing visual tokens $\mathbf{z}_{\text{img}}^d = E_{\text{img}}(\mathbf{x}^d) \in \mathbb{R}^{N \times D}$ for each dataset $d \in \{H, R\}$. Actions are corrupted by a flow-matching process at timestep t , yielding $\tilde{\mathbf{a}}_t^d$, and embedded using dataset-specific action encoders $\mathbf{z}_{\text{act}}^d = E_{\text{act}}^d(\tilde{\mathbf{a}}_t^d) \in \mathbb{R}^{T \times D}$. Image and action tokens are concatenated into a single sequence $\mathbf{z}^d = [\mathbf{z}_{\text{img}}^d; \mathbf{z}_{\text{act}}^d] \in \mathbb{R}^{(N+T) \times D}$, which forms the input to the Transformer.

Within each Transformer layer, self-attention is computed jointly over image and action tokens using modality-specific projections. For each modality $m \in \{\text{img}, \text{act}\}$, queries, keys, and values are obtained as $\mathbf{Q}_m^d = \mathbf{z}_m^d W_Q^m$, $\mathbf{K}_m^d = \mathbf{z}_m^d W_K^m$, and $\mathbf{V}_m^d = \mathbf{z}_m^d W_V^m$, where $W_{\{Q,K,V\}}^m$ depend only on modality and are shared across datasets. Concatenating these across modalities yields $\mathbf{Q}^d = [\mathbf{Q}_{\text{img}}^d; \mathbf{Q}_{\text{act}}^d]$, $\mathbf{K}^d = [\mathbf{K}_{\text{img}}^d; \mathbf{K}_{\text{act}}^d]$, and $\mathbf{V}^d = [\mathbf{V}_{\text{img}}^d; \mathbf{V}_{\text{act}}^d]$, and cross-modality self-attention is computed as $\mathbf{H}^d = \text{softmax}(\mathbf{Q}^d (\mathbf{K}^d)^\top / \sqrt{D}) \mathbf{V}^d$. This design allows image and action tokens to attend jointly. The final transformer output \mathbf{H}^d is passed to a dataset-specific flow-matching action decoder G_{act}^d , which outputs the denoised action at timestep $t - 1$.

For robot trajectories, we provide both egocentric and wrist camera images; for human trajectories, only egocentric images are available, and all token positions corresponding to the wrist stream are

Table 6: Model and training hyperparameters.

Component	Configuration
Optimization	
Visual Encoder LR	3.0×10^{-5}
MoE Parameters LR	3.0×10^{-4}
Adam Betas	(0.9, 0.95)
Self-attention	
Hidden Layers	6
Attention Heads	8
Key-Value Heads	1
Head Dim	256
SigLIP Vision Encoder	
Hidden Size	1152
Intermediate Size	4304
Hidden Layers	27
Attention Heads	16
Image Size	224
Patch Size	14
EgoBridge Loss	
Alignment Weight λ	1×10^{-4}
DTW γ	0.1
OT Loss Weight	1.0
Vision Expert	
Hidden Size	2048
Intermediate Size	4096
Proprioception Expert	
Hidden Size	1024
Intermediate Size	4096
Action Expert	
Hidden Size	1024
Intermediate Size	4096
Flow Matching Parameters	
Beta-distribution α	1.5
Beta-distribution β	1.0
Action/Proprio Encoder/Decoder MLPs	
Num Layers	3
Hidden Size	1024
Activation	LeakyReLU

masked out. Egocentric images are represented as 16×16 grids of image patches, while wrist images are spatially downsampled to 8×8 grids, with wrist positional embeddings obtained via bicubic interpolation. The complete model contains 1,362,151,162 parameters, of which 412,442,352 belong to the SigLIP visual encoder. Full architectural details are provided in Table 6, where the *Self-attention* refers to the modality-specific self-attention layers.

J Policy inference hyperparameters

The action chunks used in conditional flow matching (CFM) are sampled at 30hz for a 1 second chunk, for a horizon during training of $H^t = 30$. At inference, we execute open loop actions at half the training horizon, with $H^i = 15$. We also perform the same domain randomization at test time

as train time (color jitter and random cropping) as we find this avoids an error mode of CFM, where the policy becomes stuck in a stationary point.

K Task Details

K.1 Definitions

Pick: The task involves grasping a spatula by its handle and lifting it off the table.

Stack: The task entails lifting one of two bowls and placing it on top of the other to form a stable stack.

Pull: The task involves pulling a mug into a fixed 6 in \times 6 in square region marked on the table.

Reorient: The task entails reorienting a mug from a sideways pose into a stable upright configuration.

Book: The task involves picking up a book resting on a book holder and inserting it into an empty slot in a bookshelf.

Pour: The task entails lifting a small bowl filled with beads and pouring the beads into a larger bowl.

K.2 Success Metrics

Pick: A full success requires (1) lifting the spatula completely off the table, (2) securing the grasp on the handle, and (3) maintaining a stable hold after lifting.

Stack: A full success requires (1) grasping and lifting exactly one bowl and (2) placing it so that it rests stably on the other without either bowl tipping over.

Pull: A full success requires (1) the mug ending fully within the 6 in \times 6 in square and (2) remaining within the region after all motion stops.

Reorient: A full success requires (1) lifting the mug from its sideways orientation and (2) placing it right-side up such that it remains upright after release.

Book: A full success requires (1) lifting the book without disturbing the book holder and (2) inserting it fully into the target slot so that it is closed and aligned with neighboring books.

Pour: A full success requires (1) lifting the small bowl without spilling beads during pickup and (2) transferring more than 50% of the beads into the larger bowl.

K.3 Initial Condition Sampling Distributions

Pick: The spatula is initialized with a random pose (position and rotation each sampled uniformly at random) within the camera field of view (FOV).

Stack: Both bowls are initialized right-side up with positions sampled uniformly within the camera FOV.

Pull: The mug's position and rotation are sampled uniformly within the camera FOV; the square region remains fixed.

Reorient: The mug's pose is sampled uniformly within the camera FOV with a randomized rotation.

Book: The book's position and orientation are sampled randomly within the camera FOV.

Pour: The small and large bowls are initialized with positions sampled uniformly at random within the camera FOV.

Table 7: Training objects and tablecloths for each of the six tasks and first five training environments.

Task	Env 1	Env 2	Env 3	Env 4	Env 5
Pick	Spatula 1	Spatula 2	Spatula 4	Spatula 3	Spatula 5
	Tablecloth 1	Tablecloth 2	Tablecloth 3	Tablecloth 5	Tablecloth 4
Stack	Bowls (2,3)	Bowls (2,8)	Bowls (3,9)	Bowls (1,3)	Bowls (9,10)
	Tablecloth 1	Tablecloth 2	Tablecloth 3	Tablecloth 5	Tablecloth 4
Pull	Mug 2	Mug 4	Mug 5	Mug 3	Mug 1
	Tablecloth 1	Tablecloth 2	Tablecloth 3	Tablecloth 11	Tablecloth 4
Reorient	Mug 2	Mug 4	Mug 5	Mug 3	Mug 1
	Tablecloth 1	Tablecloth 2	Tablecloth 3	Tablecloth 5	Tablecloth 4
Book	Book 1	Book 4	Book 3	Book 5	Book 2
	Tablecloth 1	Tablecloth 2	Tablecloth 3	Tablecloth 11	Tablecloth 4
Pour	Bowls (4,8)	Bowls (3,8)	Bowls (1,9)	Bowls (5,9)	Bowls (2,10)
	Tablecloth 1	Tablecloth 2	Tablecloth 3	Tablecloth 11	Tablecloth 4

Table 8: Training objects and tablecloths for each of the six tasks and last five training environments.

Task	Env 6	Env 7	Env 8	Env 9	Env 10
Pick	Spatula 6	Spatula 7	Spatula 8	Spatula 9	Spatula 10
	Tablecloth 6	Tablecloth 7	Tablecloth 8	Tablecloth 9	Tablecloth 10
Stack	Bowls (5,6)	Bowls (4,5)	Bowls (6,7)	Bowls (4,7)	Bowls (8,9)
	Tablecloth 6	Tablecloth 7	Tablecloth 8	Tablecloth 9	Tablecloth 10
Pull	Mug 6	Mug 7	Mug 8	Mug 9	Mug 10
	Tablecloth 6	Tablecloth 7	Tablecloth 8	Tablecloth 9	Tablecloth 10
Reorient	Mug 6	Mug 7	Mug 8	Mug 9	Mug 10
	Tablecloth 6	Tablecloth 7	Tablecloth 8	Tablecloth 9	Tablecloth 10
Book	Book 6	Book 7	Book 8	Book 9	Book 10
	Tablecloth 6	Tablecloth 7	Tablecloth 8	Tablecloth 9	Tablecloth 10
Pour	Bowls (5,8)	Bowls (3,9)	Bowls (4,9)	Bowls (5,9)	Bowls (6,10)
	Tablecloth 6	Tablecloth 7	Tablecloth 8	Tablecloth 9	Tablecloth 10

L Environments

See Tables 7, 8, 9 for exact environment descriptions, to be paired with the following visualizations.

L.1 Environment Visualizations

Figures 11, 12, 13, 14 exhibit the training environments, and Figures 15, 16 exhibit the unseen testing environments, across all six tasks.

L.2 Object Visualizations

Figures 17, 18, 19, 20, 21, exhibit the training and testing objects for each of the six tasks. For each task, we detail the full set of objects actively involved in completing that task. This includes both objects the agent directly manipulates and objects that serve as targets or receivers of the manipulation. For example, in **Stack** this includes both bowls, in **Pour** it includes the small bead-filled bowl and the larger receiving bowl, and in **Book** it includes the book and the bookshelf slot.

M Results

We report policy performance under a unified evaluation setup across all six tasks and 15 trials each across four unseen test environments (Sec. K, L). For each training setting (3, 5, or 10 training environments), we select the best checkpoint. Tables 10, 11 summarize per-task success rates and overall trends in how increased training diversity affects generalization.

Table 9: Testing objects and tablecloths for each of the six tasks and four test environments.

Task	Env 1	Env 2	Env 3	Env 4
Pick	Spatula 11	Spatula 12	Spatula 11	Spatula 12
	Tablecloth 12	Tablecloth 12	Tablecloth 11	Tablecloth 11
Stack	Bowls (5,7)	Bowls (4,6)	Bowls (5,7)	Bowls (4,6)
	Tablecloth 12	Tablecloth 12	Tablecloth 11	Tablecloth 11
Pull	Mug 11	Mug 12	Mug 11	Mug 12
	Tablecloth 12	Tablecloth 12	Tablecloth 5	Tablecloth 5
Reorient	Mug 12	Mug 13	Mug 12	Mug 11
	Tablecloth 12	Tablecloth 12	Tablecloth 11	Tablecloth 11
Book	Book 11	Book 12	Book 11	Book 12
	Tablecloth 12	Tablecloth 12	Tablecloth 5	Tablecloth 5
Pour	Bowls (3,10)	Bowls (2,8)	Bowls (3,10)	Bowls (2,8)
	Tablecloth 12	Tablecloth 12	Tablecloth 5	Tablecloth 5

Table 10: **Human Cotraining success rate per task.** This table gives the quantitative results for the Human Cotraining method from Figure 5 broken down by task. Mean success rates are reported with 95% Clopper–Pearson confidence intervals.

	Mean [95% CI]	Env 1	Env 2	Env 3	Env 4
Pick					
3	50.0% [36.8, 63.2]	53.3%	53.3%	46.7%	46.7%
5	68.3% [55.9, 78.8]	66.7%	80.0%	53.3%	73.3%
10	88.3% [77.4, 95.2]	86.7%	100%	86.7%	80.0%
Stack					
3	48.3% [35.3, 61.5]	26.7%	46.7%	73.3%	46.7%
5	56.7% [43.2, 69.4]	66.7%	33.3%	80.0%	46.7%
10	55.0% [41.6, 67.9]	60.0%	46.7%	60.0%	53.3%
Pull					
3	63.3% [50.0, 75.2]	46.7%	53.3%	73.3%	80.0%
5	71.7% [58.6, 82.5]	66.7%	73.3%	80.0%	66.7%
10	71.7% [58.6, 82.5]	73.3%	53.3%	86.7%	73.3%
Reorient					
3	38.3% [26.2, 51.8]	40.0%	53.3%	26.7%	33.3%
5	55.0% [41.6, 67.9]	33.3%	66.7%	66.7%	53.3%
10	68.3% [55.9, 78.8]	66.7%	80.0%	73.3%	53.3%
Book					
3	23.3% [13.6, 36.6]	26.7%	33.3%	13.3%	20.0%
5	38.3% [26.2, 51.8]	46.7%	40.0%	40.0%	26.7%
10	60.0% [46.7, 72.4]	80.0%	40.0%	73.3%	46.7%
Pour					
3	26.7% [16.4, 40.3]	40.0%	33.3%	0.00%	33.3%
5	31.7% [20.3, 45.9]	46.7%	53.3%	13.3%	13.3%
10	53.3% [40.0, 66.3]	53.3%	46.7%	66.7%	46.7%

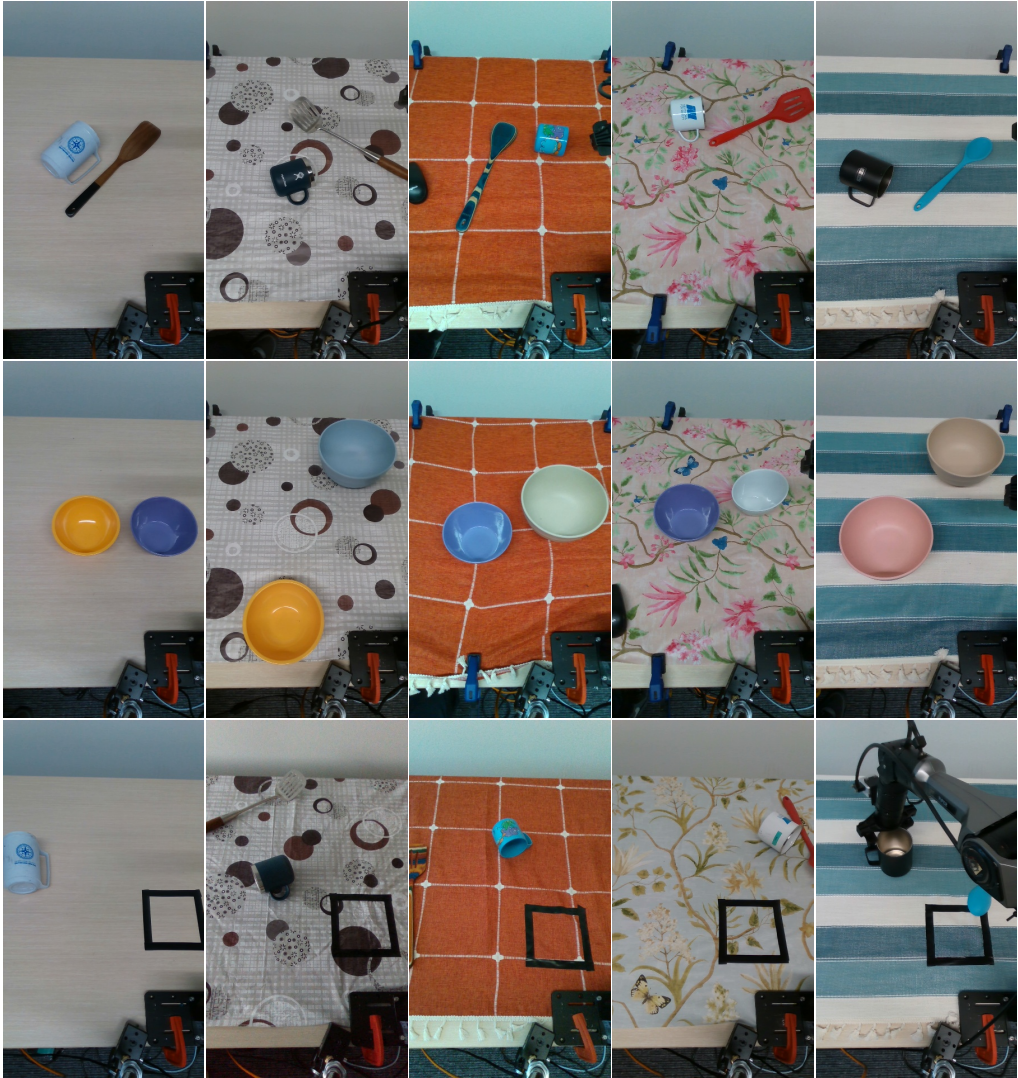


Figure 11: Training environments across first three tasks (**Pick**, **Stack**, **Pull** top to bottom) and first five scenes per task (columns).

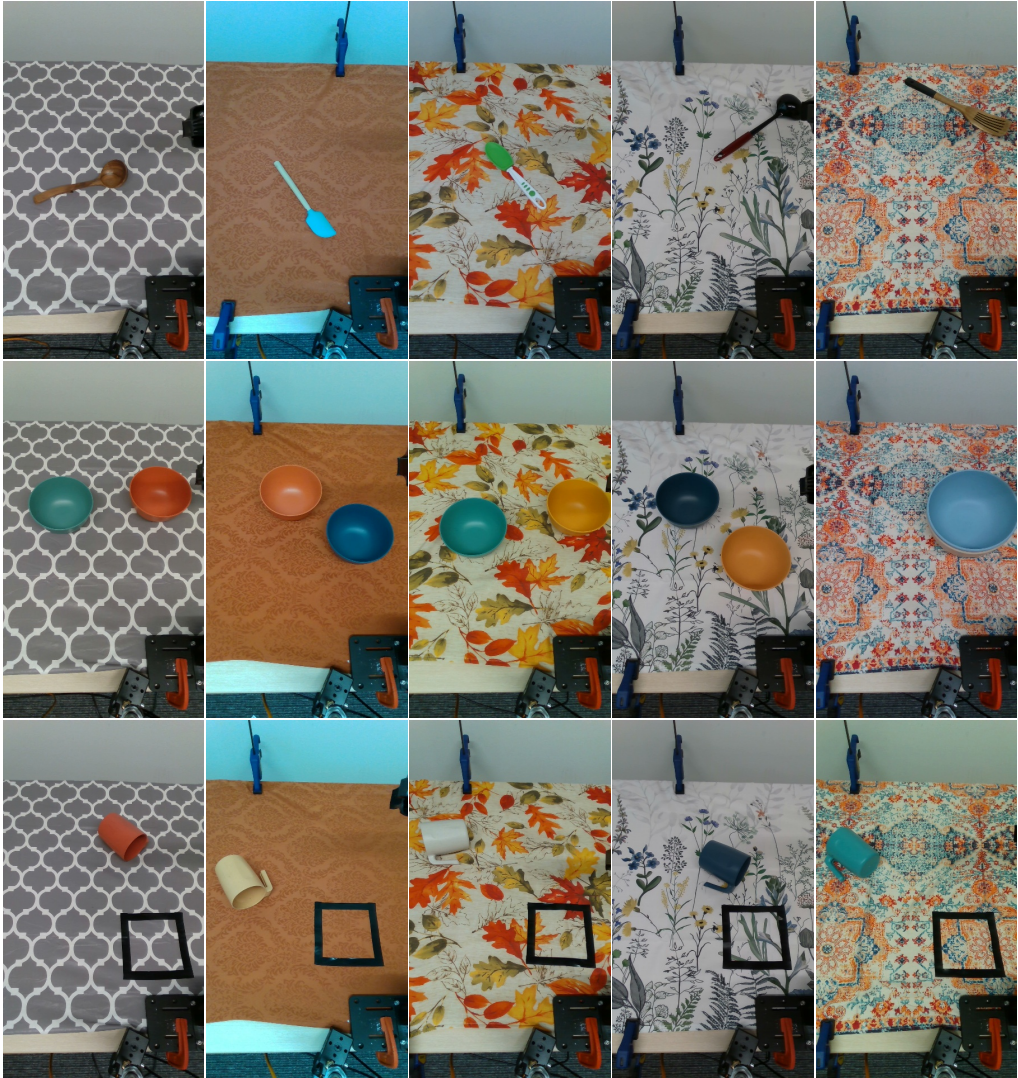


Figure 12: Training environments across first three tasks (**Pick**, **Stack**, **Pull** top to bottom) and last five scenes per task (columns).

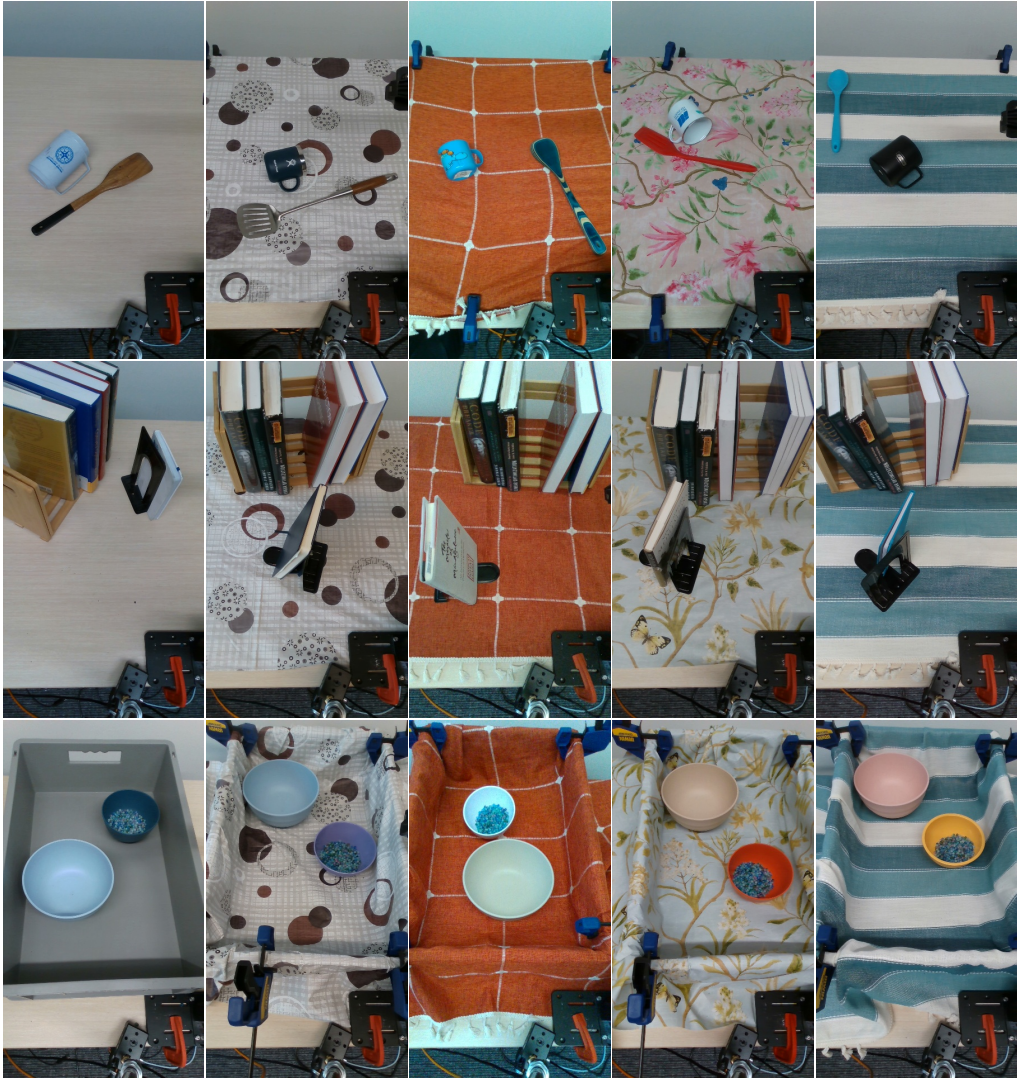


Figure 13: Training environments across last three tasks (**Reorient**, **Book**, **Pour** top to bottom) and first five scenes per task (columns).

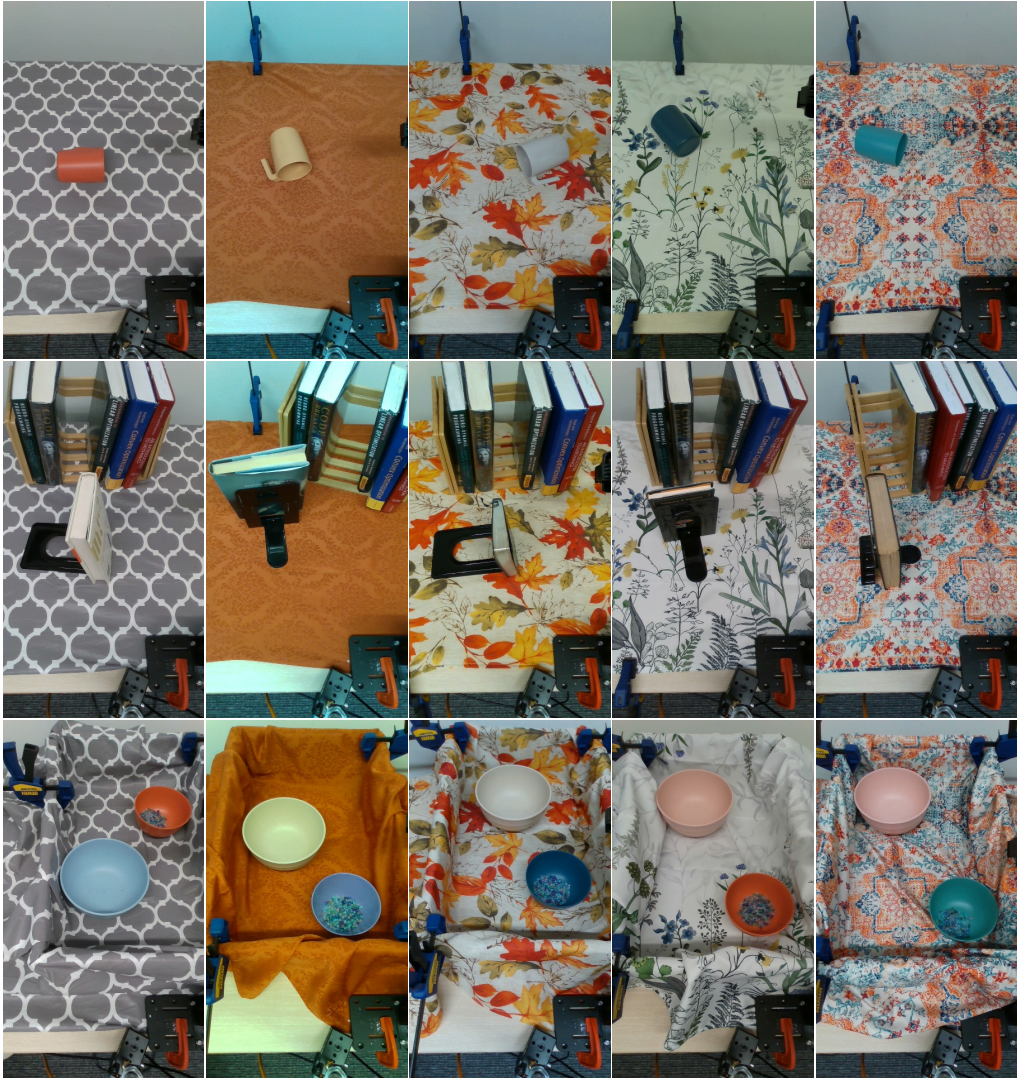


Figure 14: Training environments across last three tasks (**Reorient**, **Book**, **Pour** top to bottom) and last five scenes per task (columns).

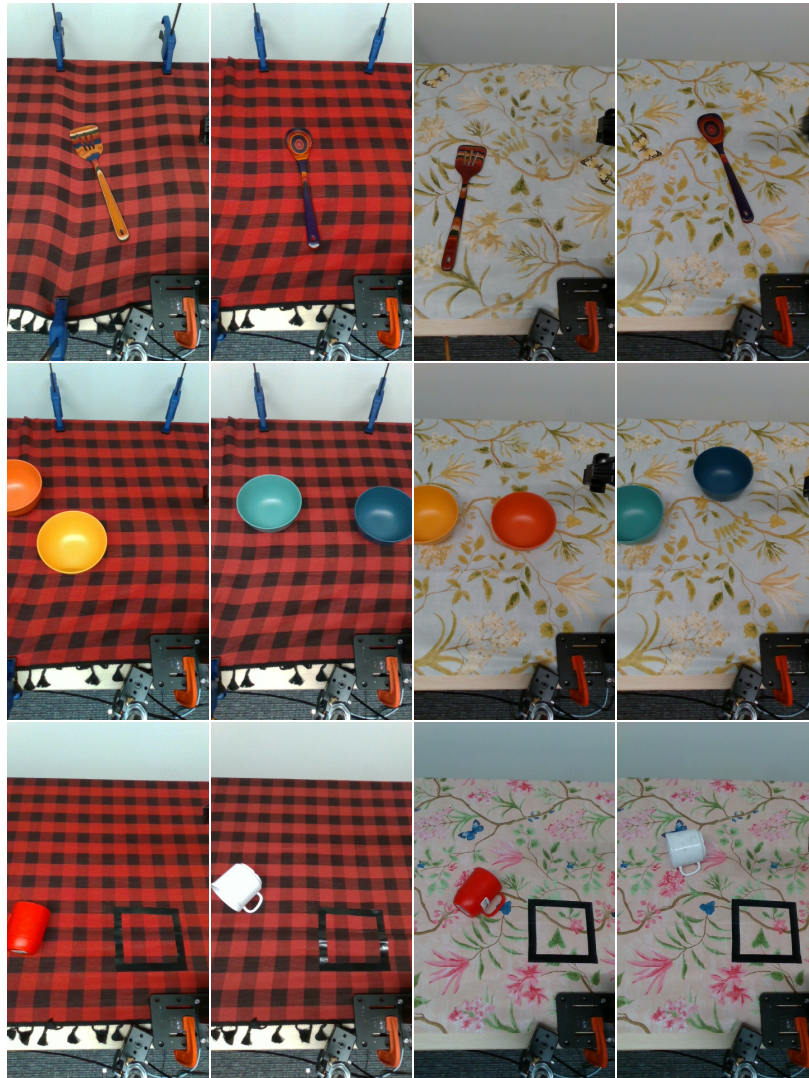


Figure 15: Test environments across first three tasks (**Pick**, **Stack**, **Pull** top to bottom) and four scenes per task (columns).



Figure 16: Test environments across last three tasks (**Reorient**, **Book**, **Pour** top to bottom) and four scenes per task (columns).

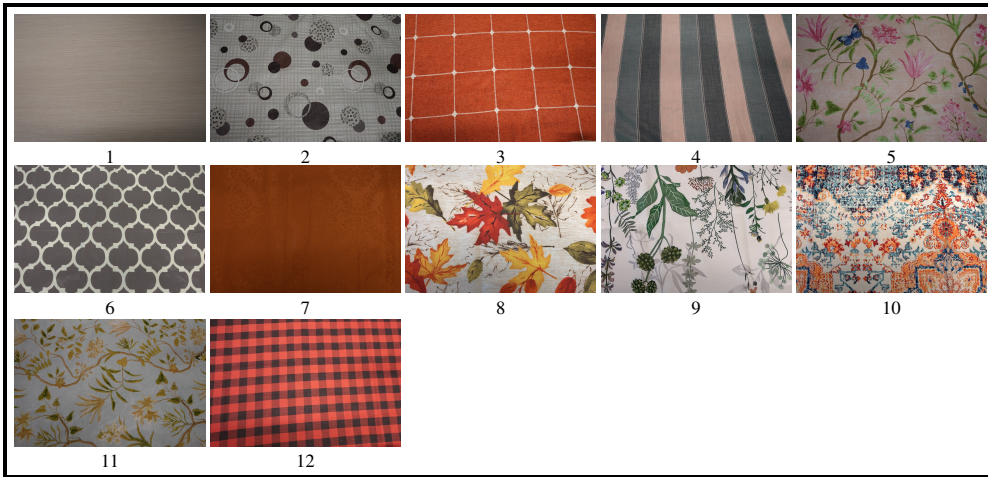


Figure 17: Tablecloths used during training/testing (labelled 1–12). 1-4, 6-10 were used in training and 12 was used in testing for all six tasks. **Pick, Stack, Reorient** used 5 in training and 11 in testing, while **Pull, Book, Pour** used 11 in training and 5 in testing.

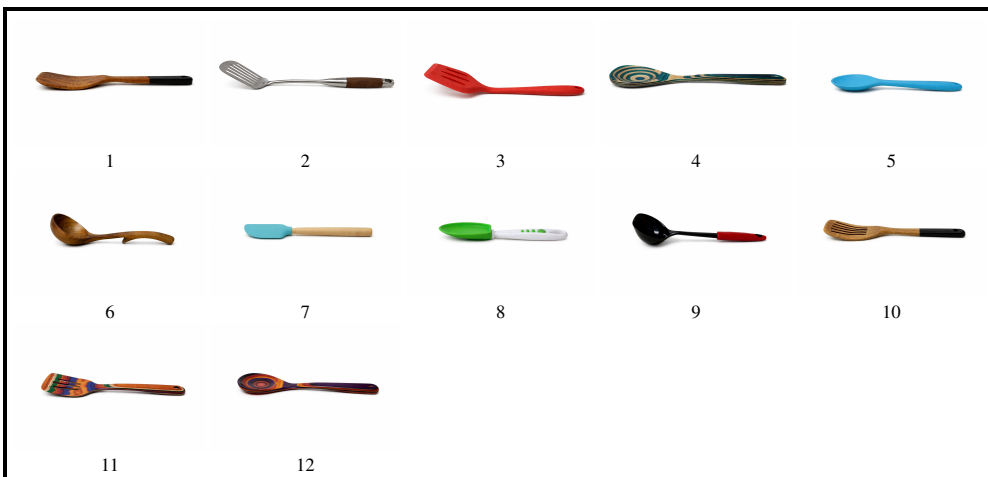


Figure 18: Spatulas used during training/testing (labelled 1-12). 1-10 were used in training and 11-12 were used in testing for **Pick**.



Figure 19: Mugs used during training/testing (labelled 1–13). 1-10 were used in training for both **Reorient** and **Pull**. 11-13 was used in testing for **Reorient**, and 11-12 was used in testing for **Pull**.



Figure 20: Bowls used during training/testing (labelled 1–10). Pairs $\{(2, 3), (2, 8), (9, 10), (3, 9), (1, 3), (5, 6), (4, 5), (6, 7), (4, 7), (8, 9)\}$ and pairs $\{(5, 7), (4, 6)\}$ were used in training and testing, respectively, for **Stack**. Pairs $\{(4, 8), (2, 10), (5, 9), (1, 9), (3, 8), (5, 8), (3, 9), (4, 9), (5, 10), (6, 10)\}$ and pairs $\{(2, 8) (3, 10)\}$ were used in training and testing, respectively, for **Pour**.

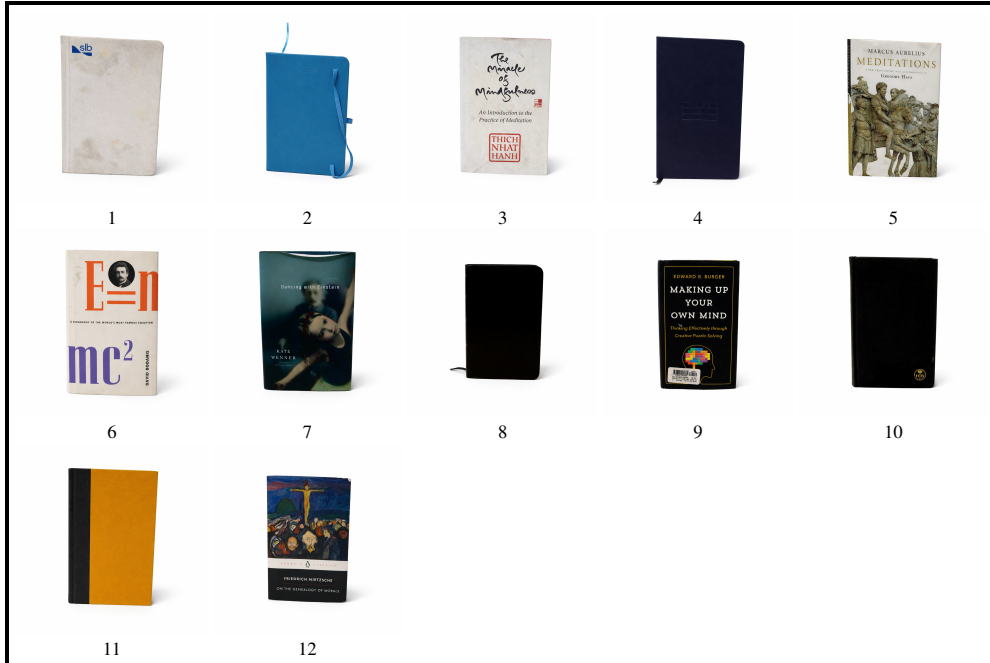


Figure 21: Books used during training/testing (labelled 1–12). 1-10 were used in training and 11-12 were used in testing for **book**.

Table 11: **Robot Only success rate per task.** This table gives the quantitative results for the Robot Only method from Figure 5 broken down by task. Mean success rates are reported with 95% Clopper–Pearson confidence intervals.

	Mean [95% CI]	Env 1	Env 2	Env 3	Env 4
Pick					
3	30.0% [19.0, 43.1]	33.3%	40.0%	26.7%	20.0%
5	66.7% [53.3, 78.3]	80.0%	73.3%	66.7%	46.7%
10	71.7% [58.6, 82.5]	66.7%	53.3%	80.0%	86.7%
Stack					
3	10.0% [4.1, 19.5]	6.67%	13.3%	13.3%	6.67%
5	35.0% [23.1, 48.4]	40.0%	20.0%	33.3%	46.7%
10	61.7% [48.2, 73.8]	46.7%	80.0%	46.7%	73.3%
Pull					
3	15.0% [7.3, 26.5]	0.00%	6.67%	13.3%	40.0%
5	31.7% [20.3, 45.9]	26.7%	20.0%	40.0%	40.0%
10	38.3% [26.2, 51.8]	26.7%	33.3%	26.7%	66.7%
Reorient					
3	11.7% [5.6, 21.8]	6.67%	6.67%	13.3%	20.0%
5	25.0% [15.1, 37.9]	20.0%	26.7%	20.0%	33.3%
10	71.7% [58.6, 82.5]	60.0%	60.0%	86.7%	80.0%
Book					
3	0.00% [0, 5.98]	0.00%	0.00%	0.00%	0.00%
5	18.3% [10.6, 29.8]	13.3%	20.0%	20.0%	20.0%
10	35.0% [23.1, 48.4]	53.3%	20.0%	46.7%	20.0%
Pour					
3	5.00% [1.7, 13.7]	0.00%	6.67%	6.67%	6.67%
5	18.3% [10.6, 29.8]	13.3%	20.0%	6.67%	33.3%
10	43.3% [30.6, 56.8]	13.3%	26.7%	66.7%	66.7%