arXiv:2610.11657v1 [cs.RO] 8 Oct 2026

YOCO: You Only Calibrate Once! Fast Mocap Calibration for Dexterous Teleoperation

Yu Zhang[1,2,3], **Yunqi Li**[1,3], **Yushi Du**[1], **Yi Ma**[1,2,3], **Yanchao Yang**[1,2,3]
[1]The University of Hong Kong, [2]Shenzhen Loop Area Institute, [3]TranscEngram

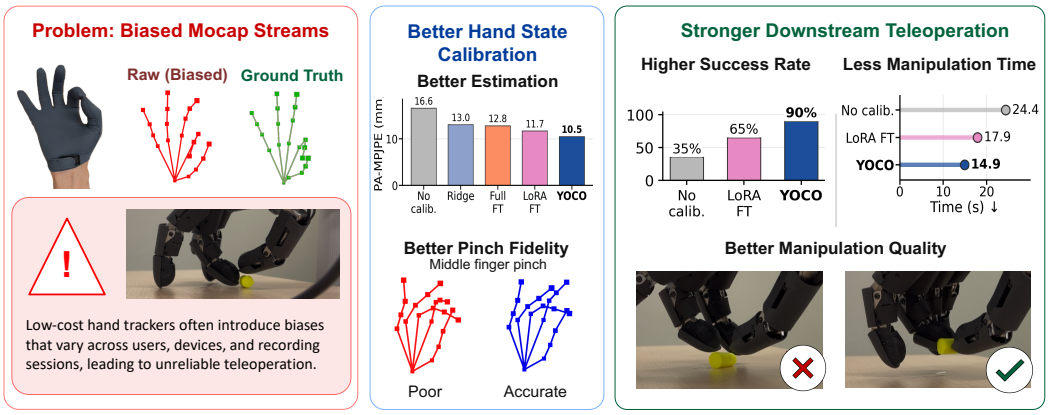

Figure 1: Overview of *YOCO*. Low-cost mocap streams suffer from user-, device-, and session-specific bias. Given a few paired calibration poses, *YOCO* predicts a calibration update that corrects future mocap poses and improves downstream dexterous teleoperation.

Abstract: Dexterous teleoperation requires reliable human-hand state estimations. However, common low-cost motion-capture gloves and markerless trackers often exhibit biases that vary across users, glove fit, and recording sessions, degrading retargeting and demonstration quality. We present *YOCO*, a fast few-shot, fine-tuning-free calibration framework that corrects biased hand-pose streams from a small set of paired raw and target poses. Instead of optimizing a separate model for every operator or session, *YOCO* conditions a calibration HyperNet on the paired examples and predicts LoRA-style updates for a frozen MANO hand-estimation module, turning per-user calibration into a lightweight feed-forward adaptation step while preserving the geometric prior of MANO and the efficiency of a compact estimator. We train *YOCO* with synthetic drift augmentations on InterHand2.6M and evaluate on augmented InterHand sequences, offline real glove data, and dexterous teleoperation tasks. Across these settings, *YOCO* improves calibration efficiency, hand-state estimation quality and teleoperation performance compared with uncalibrated input and standard calibration baselines.

Keywords: Hand Pose Estimation and Calibration, Teleoperation

1 Introduction

Teleoperation is a key tool for collecting dexterous hand demonstrations, but its effectiveness depends on the fidelity of the operator's hand-state estimates. High-end motion-capture systems can provide accurate kinematics, yet they are expensive and often difficult to deploy at scale. Lower-cost gloves and markerless trackers are more accessible, but their outputs are affected by user hand morphology gap, device fit, sensor drift, and session-dependent bias. Such errors are particularly

10th Conference on Robot Learning (CoRL 2026), Austin TX, USA.

detrimental in dexterous manipulation, where even small fingertip or joint offsets can alter contact patterns, pinch geometry, and retargeted robot motion.

Prior hand-mocap calibration typically operates within a particular sensing pipeline, learning glove-signal-to-pose mappings, correcting IMU alignment and drift, or personalizing sensor-to-pose models [1, 2, 3, 4]. We instead formulate few-shot calibration at the derived 3D hand-keypoint interface, after a tracker has produced its pose estimate. Given K paired raw and target poses from a user and session, *YOCO* conditions a HyperNet on those examples and generates LoRA updates for a frozen MANO-based estimator in one pass, without per-session gradient optimization. This couples a sensor-independent pose interface with amortized, session-conditioned weight generation: unlike a LoRA initialization shared across sessions, the generated updates depend on the current calibration set (Appendix B). The formulation requires no access to device-specific raw signals; our real-device evaluation remains limited to one tensile glove.

Our contributions are threefold. First, we formulate calibration of derived 3D hand keypoints from a few pose pairs, separating the correction from device-specific sensor-to-pose calibration. Second, we generate session-conditioned LoRA weights for a frozen MANO estimator in one pass; learned-prior and set-conditioned residual controls test whether the gains stem from pair-conditioned weight generation beyond a shared prior or set conditioning alone. Third, we evaluate accuracy and adaptation cost on augmented InterHand2.6M and paired real-glove/AVP data, then test the resulting hand poses in physical teleoperation.

2 Related Work

Mocap Devices: High-fidelity dexterous teleoperation depends on reliable capture of the operator hand, yet existing mocap interfaces exhibit a strong accuracy-accessibility trade-off. Glove- and exoskeleton-based systems provide stable kinematics but are costly, cumbersome, and sensitive to user-specific geometry and donning conditions [5, 1, 6, 2]. The resulting joint biases or fingertip drift propagate directly into the collected demonstrations. Markerless and RGB-based pipelines, including FreiHAND, MediaPipe Hands, and HaMeR, have improved accessibility [7, 8, 9, 10, 11, 12, 13], but still struggle with occlusion, hand-object interaction, cross-user morphology, and temporal stability. *YOCO* is complementary: rather than introducing a new tracker, it calibrates existing glove-based streams so that inexpensive devices become more consistent for downstream use.

Prior Hand-Mocap Calibration: Wang and Neff [1] learn a mapping from glove measurements to hand motion. Gosala et al. [4] use optical reference data to personalize a wearable sensor-to-pose model through side-tuning. FSGlove [2] jointly optimizes IMU alignment, hand shape, and kinematics, while the Transformer IMU Calibrator [3] estimates time-varying IMU drift and offsets. These approaches address calibration within particular sensing pipelines. *YOCO* instead takes their downstream 3D-keypoint representation as its input and learns to generate a session-specific correction of a frozen MANO estimator from a few paired poses. This interface makes the formulation independent of raw sensor modality; generalization beyond the evaluated tensile glove remains to be tested.

Dexterous Hand Retargeting and Teleoperation: Recent teleoperation systems have made dexterous data collection cheaper, more immersive, and more hardware-agnostic. Vision-based systems such as DexPilot and AnyTeleop reduce dependence on instrumented gloves [14, 15, 16], while VR-based platforms such as Open-TeleVision and OPEN TEACH improve operator immersion [17, 18]. In parallel, learning-based retargeting methods close the embodiment gap beyond purely kinematic alignment [19, 20, 21, 22], and new master-side or robot-side designs improve transferability and contact richness [23, 24, 25]. These efforts largely assume that the upstream hand observation is already accurate; *YOCO* targets that bottleneck and is compatible with the downstream stacks above.

HyperNetworks and Parameter-Efficient Adaptation: HyperNetworks generate the weights of one network with another [26], and task-conditioned variants support efficient cross-task adaptation

[27, 28, 29, 30, 31]. In parallel, parameter-efficient methods such as Compacter and LoRA specialize a frozen backbone with compact low-rank updates [32, 33], and hypernetworks have been shown to predict such updates from limited conditioning input [34]. *YOCO* brings these ideas to dexterous teleoperation: h_ψ predicts LoRA-style updates ΔW for the hand estimator from a handful of paired calibration poses, yielding fast and robust adaptation.

3 Method

3.1 Problem Formulation

We consider the problem of calibrating noisy mocap hand poses using a small set of paired calibration examples. A hand observation is represented as 21 3D joint keypoints $x \in \mathbb{R}^{21\times 3}$. Let

$$\mathcal{S} = \{(x_i^n, x_i^c)\}_{i=1}^K \tag{1}$$

denote the calibration set, where x_i^n is a noisy mocap hand-pose keypoint observation sampled from the noisy pose distribution $\mathcal{D}_{\text{noisy}}$, and x_i^c is its corresponding target 3D keypoint pose. The calibration set $\mathcal{S}$ is assumed to capture the approximately stationary bias of the current setting, such as a particular subject, glove fit, or recording session.

We formulate calibration as the composition of two networks with distinct roles.

Target network. A MANO estimator

$$g_\phi : \mathbb{R}^{21\times 3} \longrightarrow \mathbb{R}^{45}, \qquad \hat{\theta} = g_\phi(x), \tag{2}$$

maps 21 3D hand keypoints to MANO pose parameters $\hat{\theta}$. Its parameters ϕ are trained once on clean data and then held fixed; per-user and per-session calibration acts on g_ϕ only through additive LoRA updates rather than by re-optimizing ϕ.

Hypernetwork. A calibration HyperNet

$$h_\psi : \left(\mathbb{R}^{21\times 3} \times \mathbb{R}^{21\times 3}\right)^K \longrightarrow \mathcal{W}, \qquad \Delta W = h_\psi(\mathcal{S}), \tag{3}$$

takes the calibration set as input and outputs a set of low-rank LoRA matrices ΔW, where $\mathcal{W}$ denotes the space of LoRA adapters attached to g_ϕ. The parameters ψ are shared across all users and devices and constitute the only learned parameters of the calibration system; ϕ is fixed.

Calibrated estimator. Injecting ΔW into the target network yields a session-specific calibrated estimator

$$g_{\phi+\Delta W} : \mathbb{R}^{21\times 3} \longrightarrow \mathbb{R}^{45}, \tag{4}$$

which, composed with MANO forward kinematics, produces the calibrated 3D joints

$$\hat{x}_q^c \;=\; \Pi_{\text{MANO}}\big(g_{\phi+h_\psi(\mathcal{S})}(x_q^n),\, \beta\big) \;=:\; f_\psi(x_q^n;\, \mathcal{S}), \tag{5}$$

for any new query $x_q^n \sim \mathcal{D}_{\text{noisy}}$. The shape parameter β is handled consistently across methods in each evaluation setting, as described in Section 5.

Learning objective. Since ϕ is frozen, only the hypernetwork parameters ψ are trained, by minimizing the expected joint-position error of the resulting calibrated estimator:

$$\min_\psi \; \mathbb{E}_{\mathcal{S},\,(x_q^n, x_q^c)}\left[\, \mathcal{L}\big(f_\psi(x_q^n;\, \mathcal{S}),\, x_q^c\big) \,\right], \tag{6}$$

where $\mathcal{L}$ is an L_2 distance between hand joint positions. This objective encourages h_ψ to infer the underlying mocap bias from the paired calibration set and produce LoRA updates that correct unseen noisy poses drawn from the same distribution.

3.2 Model Overview

Figure 2 illustrates how the target network g_ϕ and the hypernetwork h_ψ defined in Section 3.1 are combined at deployment. Calibration consists of a one-time feed-forward conditioning pass

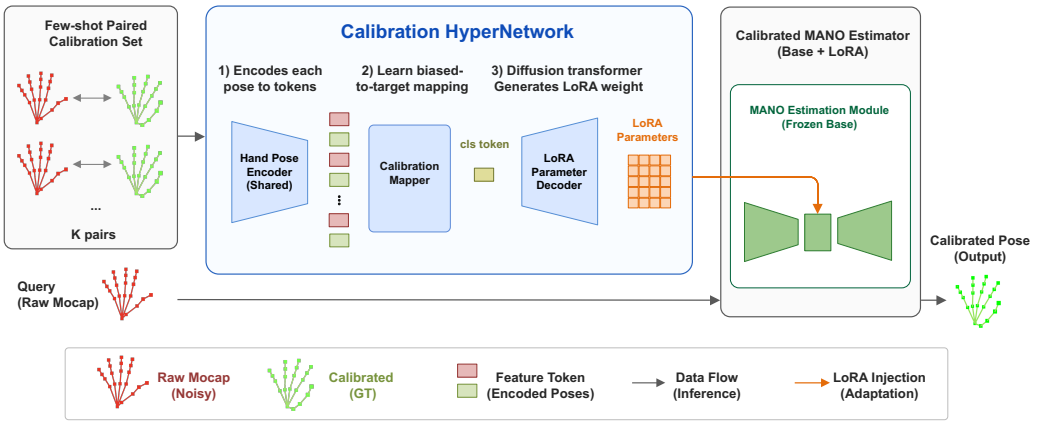

Figure 2: *YOCO* model architecture. Given a calibration set $\mathcal{S} = \{(x_i^n, x_i^c)\}_{i=1}^K$ of K paired noisy and target poses (left), the calibration HyperNet h_ψ (middle) encodes each pair, aggregates them through a transformer-based calibration mapper with one CLS token per LoRA matrix, and decodes the tokens into LoRA updates ΔW. These updates are injected once per session into the frozen MANO estimator g_ϕ (right) to form $g_{\phi+\Delta W}$, which maps each raw mocap query x_q^n at frame rate to calibrated MANO pose and 3D joints.

followed by a frame-rate query path, both fully gradient-free: (i) the HyperNet ingests the calibration set and outputs LoRA matrices, $\Delta W = h_\psi(\mathcal{S})$; (ii) these matrices are injected into the frozen MANO estimator to form the calibrated estimator $g_{\phi+\Delta W}$; (iii) each subsequent raw mocap pose x_q^n is mapped through $g_{\phi+\Delta W}$ to obtain $\hat{\theta}_q$, from which MANO reconstructs the calibrated joints $\hat{x}_q^c$. Step (i) is executed only once per session, while step (iii) runs at the mocap frame rate. The remainder of this section specifies the architectures of g_ϕ (Section 3.3) and h_ψ (Section 3.4).

3.3 Target Network: MANO Estimator Architecture

The MANO estimator g_ϕ consists of a hand-keypoint encoder and a MANO parameter decoder. The hand-keypoint encoder uses finger-wise CNN keypoint encoders followed by a transformer fusion layer. The MANO parameter decoder uses finger-wise MLPs and a unified MANO θ decoder. The finger-wise design prevents each finger estimate from being overly affected by unrelated fingers, while the fusion layer captures global hand-pose correlation. The decoder is deliberately lightweight so that it can run in real time and accept LoRA updates from the HyperNet.

3.4 Hypernetwork: Calibration HyperNet Architecture

The calibration HyperNet h_ψ realizes the mapping $\mathcal{S} \mapsto \Delta W$: its input is the K paired noisy/target poses in the calibration set, and its output is the LoRA low-rank matrices ΔW injected into the frozen MANO estimator g_ϕ. To produce useful adapters from only a handful of pairs, h_ψ must infer the session-specific drift pattern from a small amount of runtime calibration data.

The HyperNet consists of three submodules: (1) hand-keypoint encoder, (2) calibration mapping, and (3) LoRA parameter decoder. The hand-keypoint encoder reuses the encoder from the MANO estimator g_ϕ to extract pose features from each raw/calibrated pair. The calibration mapping is a transformer block with RoPE positional embedding [35] that models the mapping relationship across the paired examples. It includes one learnable CLS token for each LoRA matrix to be generated; after calibration-map aggregation, each CLS token conditions the generation of its corresponding LoRA matrix. The LoRA parameter decoder is a DiT-style decoder [36] that maps these LoRA-specific CLS tokens to the low-rank matrices in ΔW. We use the same LoRA adapter footprint for *YOCO* and the LoRA-Finetune baseline; details are provided in Appendix A.2.

4 Model Training

Training proceeds in two stages that mirror the role separation introduced in Section 3.1: first the target network g_ϕ is trained on clean data, then it is frozen and the hypernetwork h_ψ is trained to generate LoRA updates that correct simulated mocap bias.

4.1 Target Network Pretraining

We train the parameters ϕ of the MANO estimator g_ϕ on clean InterHand2.6M data, using the mean squared error between the predicted MANO pose parameter $\hat{\theta} = g_\phi(x)$ and the ground-truth MANO pose parameter. This pretraining establishes a hand-pose prior in MANO space that the HyperNet can later specialize through LoRA updates. After this stage, ϕ is frozen and never updated again.

4.2 Hypernetwork Training

With ϕ fixed, we train the hypernetwork parameters ψ so that h_ψ produces LoRA updates ΔW that compensate for realistic mocap bias. To expose h_ψ to a diverse family of bias patterns, we apply synthetic augmentations to InterHand2.6M sequences that simulate the drift behavior of glove-based mocap devices. The augmentations are designed to be strong and varied so that the augmented distribution covers, or at least overlaps with, realistic distribution shifts observed in real mocap data; the full strategy is provided in Appendix A.1.

Each training iteration uses one augmented sequence as a session. From this sequence we sample K paired calibration examples $\mathcal{S} = \{(x_i^n, x_i^c)\}_{i=1}^K$ for the HyperNet, and a disjoint set of query frames $\{(x_q^n, x_q^c)\}$ for supervision. The HyperNet generates session-specific LoRA matrices $\Delta W = h_\psi(\mathcal{S})$, which are injected into the frozen target network to form $g_{\phi+\Delta W}$. Each query pose is then passed through this calibrated estimator and MANO forward kinematics to obtain $\hat{x}_q^c = f_\psi(x_q^n; \mathcal{S})$, and ψ is updated by minimizing the mean squared error between $\hat{x}_q^c$ and the ground-truth calibrated joints x_q^c. Gradients flow only through ψ, and LoRA injection is differentiable in ΔW.

5 Experiment

Our experiments are designed to evaluate *YOCO* as a deployable calibration system rather than only as a pose-estimation model. We organize the evaluation around five research questions that together cover calibration accuracy, the source of *YOCO*'s advantage, sample efficiency in K, transfer to real low-cost glove data, and downstream impact on dexterous teleoperation: (1) Does *YOCO* reduce hand-pose error on biased mocap streams relative to uncalibrated input? (2) How does *YOCO* compare with optimization- and fine-tuning-based calibration baselines in accuracy and adaptation cost, and in particular, does the feed-forward HyperNet outperform gradient-based LoRA at matched adapter capacity? (3) How does performance scale with the number of paired calibration poses K? (4) Does *YOCO* transfer from synthetic drift to real low-cost glove data across users and glove fits? (5) Does improved hand-state calibration translate into measurable gains on downstream dexterous teleoperation tasks?

5.1 Experiment Setup

Training datasets. We pretrain the target network g_ϕ on the training subset of InterHand2.6M, which provides a large pool of clean MANO-annotated hand poses captured by a high-precision multi-view mocap system. We then train the hypernetwork h_ψ on a synthetic dataset constructed by applying the structured drift augmentations described in Appendix A.1 to InterHand2.6M, while keeping ϕ frozen. Each augmented sequence simulates one user/device/session with an internally consistent bias pattern, providing the diverse session-level distribution that the HyperNet must learn to summarize from only K paired examples.

Test datasets. We evaluate *YOCO* on three settings of increasing realism. On the synthetic side, we apply the same augmentation family to the InterHand2.6M validation set, instantiating 20 augmented sets with distinct drift patterns; all reported synthetic numbers are aggregated over these 20 sets to assess robustness across drift types rather than performance on a single bias. For real low-cost-glove evaluation, we collect a paired dataset using an Apple Vision Pro (AVP) as a high-end reference tracker and a low-cost tensile glove worn simultaneously, treating the AVP 3D keypoints as the calibration target. Split and sequence statistics are provided in Appendix C.3. For downstream evaluation, we deploy *YOCO* in a live teleoperation system on four physical manipulation tasks.

Evaluation setup. Unless otherwise specified, all methods use $K = 4$ paired calibration poses. For offline evaluations on InterHand2.6M and the real glove dataset, $\mathcal{S}$ is sampled from the first 200 frames of each validation sequence with a fixed random seed; the remaining frames form the evaluation split. The real-glove targets are synchronized AVP reference poses. For deployment, we record a fixed library of four 21-joint target poses with AVP once offline: index pinch, fully open hand, claw/half-close, and four straight fingers with the thumb near the palm (Appendix C.3). The targets use the mocap stream's joint order and coordinate convention. At the start of each teleoperation trial, the operator performs these prompts wearing only the low-cost glove; each raw observation is paired with its stored library target. No AVP is needed at runtime. All methods share the same shape-handling protocol and do not calibrate MANO shape β; details are in Appendices C.1 and C.2.

5.1.1 Teleoperation Protocol

We evaluate four physical tasks with a right robot hand: removing a fixed bottle's cap by twisting; lifting an earbud with a thumb–middle-finger pinch and holding it for 3 s; lifting a small glue bottle with the thumb, index, and middle fingers and holding it for 3 s; and unlocking a tissue-box lid by stabilizing it with four fingers and pulling with the thumb. These outcomes define task success. Two operators participate: one performs bottle-cap and earbud tasks, and the other performs glue-bottle and tissue-box tasks. The same operator tests all three calibration conditions for each assigned task. Operators can directly see the robot and thus can compensate for control error, but are blinded to the calibration method's identity. For each method–task pair, they receive 5 min of familiarization and then perform 10 evaluation trials; method order is randomized within each task. All conditions use the same trial budget and the same geometric-retargeting method [22]. A separate retargeting model is trained for each calibration condition using the same configuration. We report success counts and completion time, with the failure penalty for overall time specified in Table 3.

Evaluation metrics. We report a comprehensive set of pose, geometry, and deployment metrics. MPJPE and PA-MPJPE measure 3D joint accuracy before and after Procrustes alignment: MPJPE preserves residual offset, scale, and orientation bias, while PA-MPJPE isolates articulated pose accuracy. AUCJ is the normalized area under the joint PCK curve, computed over 101 uniformly spaced thresholds from 0 to 0.05 m. Tip-PA is the PA-aligned mean L2 error over the five MANO fingertips and is reported separately because fingertip geometry dominates dexterous-manipulation outcomes. Pinch error is the L2 distance between two fingertips on frames where the ground-truth pinch distance is below 10 mm; this isolates the regime that contact-rich tasks depend on. Calibration time is the test-time conditioning or adaptation cost incurred before applying the calibrated estimator: for *YOCO*, this is one forward pass of h_ψ; for fine-tuning baselines, it is gradient-based adaptation on the paired calibration set. For downstream evaluation, we report task success count over 10 trials and average completion time.

5.2 Experiment Results

RQ1: *YOCO* substantially reduces hand-pose error on biased mocap streams. On the augmented InterHand2.6M evaluation set, *YOCO* reduces PA-MPJPE from 16.55 mm (no calibration) to 10.45 mm, a 37% relative reduction, with corresponding gains across MPJPE, Tip-PA, AUCJ, and pinch error (Table 1). The improvement is largest on fingertip-sensitive metrics (Tip-PA drops from 37.15 to 23.29 mm), which matches the design intuition that calibration matters most where

Method	MPJPE↓ (mm)	PA-MPJPE↓ (mm)	Tip-PA↓ (mm)	AUCJ↑	Pinch↓ (mm)	Calib. s↓ (s)
No calibration	21.52	16.55	37.15	0.611	48.49	0.00
Ridge	17.62	13.00	28.28	0.676	45.35	**1.6**
Last-layer-Finetune	16.62	12.01	25.92	0.692	39.18	45.6
Full-Finetune	18.11	12.77	29.21	0.668	37.11	95.4
LoRA-Finetune	16.38	11.67	25.33	0.694	37.28	89.4
***YOCO* (ours)**	**15.64**	**10.45**	**23.29**	**0.702**	**34.90**	11.2

Table 1: Calibration results on the augmented InterHand2.6M evaluation setting. Calib. s reports the total adaptation time in seconds for whole validation sets on one H100.

small biases reshape contact geometry. These results confirm that the HyperNet successfully infers session-level drift from only $K = 4$ paired examples and that the predicted LoRA updates ΔW correct unseen queries from the same session.

RQ2: *YOCO* outperforms optimization- and fine-tuning-based baselines, and the advantage persists at matched LoRA capacity. We compare against five baselines spanning the natural spectrum of calibration strategies: (1) *No Calibration*, which uses raw mocap output; (2) *Ridge*, an optimization-based linear mapping fitted in MANO θ space from the paired examples; (3) *Last-layer-Finetune*, which gradient-tunes only the final MLP of the MANO θ decoder on the paired examples; (4) *Full-Finetune*, which gradient-tunes the entire g_ϕ; and (5) *LoRA-Finetune*, which injects the same LoRA adapter set as *YOCO* into g_ϕ and tunes only those adapters by gradient descent on the paired examples. LoRA-Finetune matches the adapter footprint, but its cold start does not control for *YOCO*'s learned cross-session prior. Appendix B adds a meta-learned LoRA initialization, a set-conditioned residual predictor, and a published side-tuning baseline to examine that distinction.

As shown in Table 1, *YOCO* achieves the best score on every precision metric while being substantially faster than every gradient-based baseline: 11.2 s of aggregate H100 conditioning for *YOCO* versus 45.6–95.4 s for the fine-tuning baselines. The accuracy advantage over LoRA-Finetune at matched adapter capacity (PA-MPJPE 10.45 vs. 11.67 mm, Tip-PA 23.29 vs. 25.33 mm) shows that the gains are not attributable to the LoRA adapters alone: feed-forward weight generation by h_ψ outperforms per-session gradient descent over the same parameter space. Ridge regression is the only baseline competitive on adaptation time, but its accuracy degrades severely. The deployment-latency gap is even larger on CPU, where *YOCO* requires 13.16 s versus 34.47–223.25 s for gradient fine-tuning baselines (Appendix C.4); this is the regime that matters for in-the-field teleoperation rigs without a dedicated GPU, which signifies the applicability of *YOCO*.

Method	MPJPE↓ (mm)	PA-MPJPE↓ (mm)	Tip-PA↓ (mm)	AUCJ↑	Pinch↓ (mm)	Calib.↓ (s)
No calibration	20.49	14.63	36.22	0.610	51.39	0.00
Ridge	24.71	19.21	45.50	0.573	57.54	**1.49**
Last-layer-Finetune	18.86	14.67	33.88	0.643	43.63	5.04
Full-Finetune	20.00	15.73	36.67	0.630	49.89	13.04
LoRA-Finetune	18.72	13.94	33.23	0.648	54.01	14.57
***YOCO* (ours)**	**17.75**	**13.14**	**31.73**	**0.657**	**42.79**	6.73

Table 2: Real-world mocap calibration results. Metrics are averaged over 3 seeds. Calib. s reports the total conditioning or adaptation time in seconds for whole datasets on one H100. Corresponding user-level std is provided in Appendix C.5.

RQ3: *YOCO* provides strong one-pass few-shot calibration. Figure 3 sweeps K from 1 to 16. Among methods that calibrate without further test-time optimization, *YOCO* achieves the lowest PA-MPJPE at $K \in \{1, 2, 4\}$: collecting four paired poses takes a few seconds, whereas collecting sixteen is closer to a minute. At $K = 1$, *YOCO* reaches 10.94 mm, below every gradient-based baseline at $K = 4$. At $K = 8$, gradient-based baselines close the gap, and at $K = 16$, Full-Finetune slightly surpasses the one-pass *YOCO*. The new YOCO-FT curve shows the effect of adding per-session fine-tuning after *YOCO* initialization; it improves accuracy further but gives up gradient-free adaptation. This pattern is consistent with an optimization or capacity limit at larger K, but does not isolate which factor dominates. The numerical multi-method sweep is reported in Appendix C.7.

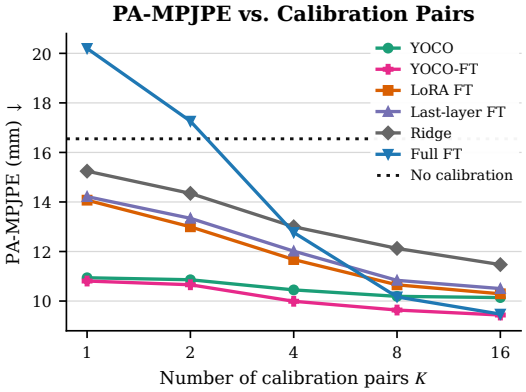

Figure 3: PA-MPJPE versus calibration-pair count, including *YOCO* followed by test-time fine-tuning (YOCO-FT). The dotted line marks no calibration.

RQ4: *YOCO* transfers to real low-cost glove data and generalizes across users. We evaluate on the paired AVP/tensile-glove dataset described in Appendix C.3, which exposes *YOCO* to the actual user- and fit-dependent distortions of a low-cost glove rather than synthetic drift. As shown in Table 2, *YOCO* attains the best score on every primary metric (MPJPE, PA-MPJPE, Tip-PA, AUCJ, pinch error) while requiring 6.73 s of conditioning versus 14.57 s for LoRA-Finetune and 13.04 s for Full-Finetune. The per-user standard deviations in Appendix C.5 indicate that this advantage is consistent across the four users in the dataset rather than driven by any single operator. Notably, Ridge regression degrades below the no-calibration baseline on real data, which suggests that the bias structure of a real glove is non-linear enough that linear correction in MANO θ space is actively harmful; this is the regime where a learned, structured calibrator earns its keep.

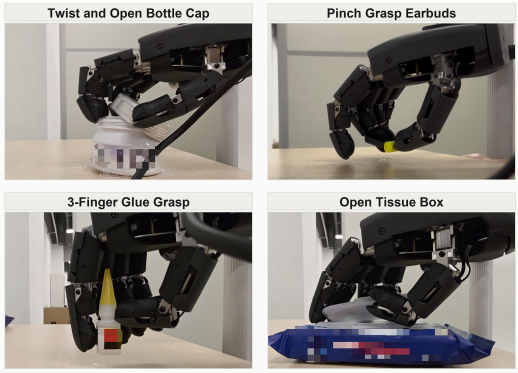

Figure 4: Goal states for the downstream dexterous manipulation tasks: twisting and opening a bottle cap, pinch-grasping earbuds, three-finger glue grasping, and tissue-box opening.

RQ5: The *YOCO* teleoperation pipeline improves task success. The tasks, operator assignment, visual feedback, familiarization, retargeting, and trial order are described in Section 5.1.1.

As shown in Table 3, *YOCO* achieves an overall success rate of 90%, compared with 65% for LoRA-Finetune and 35% for the uncalibrated baseline, and the lowest overall completion time across the four tasks. The largest per-task gains appear on contact-sensitive tasks: Earbud Pinch Grasping rises from 1/10 (LoRA-Finetune) to 8/10 (*YOCO*), and Tissue Box Opening from 8/10 to 10/10. The earbud task uses a middle-finger pinch, whereas the calibration poses include an index-finger pinch but no middle-finger pinch, suggesting transfer to an unobserved fingertip configuration. Because retargeting models are trained separately for each condition, the task results characterize the complete teleoperation pipelines rather than isolating calibration as the sole cause.

Method	Bottle Cap Opening		Earbud Pinch Grasping		Glue Three-Finger Grasping		Tissue Box Opening		Overall	
	Success↑	Time (s)↓	Success↑	Time (s)↓	Success↑	Time (s)↓	Success↑	Time (s)↓	Success Rate↑	Time (s)↓
No calibration	9/10	35.6	0/10	–	5/10	7.0	0/10	–	35%	24.4
LoRA-Finetune	9/10	34.1	1/10	6.0	**8/10**	**6.6**	8/10	**13.3**	65%	17.9
***YOCO* (ours)**	**10/10**	**27.3**	**8/10**	**4.2**	**8/10**	8.9	**10/10**	17.4	**90%**	**14.9**

Table 3: Downstream dexterous teleoperation results. We evaluate 4 tasks and count each task success over 10 trials and average completion time over successful trials. Failed trials are capped at one minute and excluded from task-specific average completion time, while each failed trial is penalized by assigning it the longest observed trial time in the overall average time. More stats are provided in Appendix C.8.

6 Conclusion

YOCO reframes mocap calibration as feed-forward weight generation: a hypernetwork h_ψ reads a handful of paired poses and emits LoRA updates ΔW that specialize a frozen MANO estimator g_ϕ, avoiding per-session optimization of the estimator. The downstream results show promising teleoperation gains for the evaluated calibration-and-retargeting pipelines.

Limitations *YOCO* focuses on pose calibration and leaves MANO shape estimation to a separate stage, which integrates cleanly with existing shape-fitting pipelines. The session-stationary bias assumption is consistent with sensor manufacturing tolerances of most commercial gloves, but high-frequency drift on less stable hardware would require re-calibration or a temporal extension of the HyperNet. The augmentation pipeline lets us scale training across diverse drift patterns at low cost, though covering high-frequency drift remains an open direction tied to sensor characterization.

Acknowledgments

This work is supported by the Early Career Scheme of the Research Grants Council (RGC) grant # 27207224, the HKU-100 Award, a donation from the Musketeers Foundation, and in part by the JC STEM Lab of Autonomous Intelligent Systems funded by The Hong Kong Jockey Club Charities Trust and computing resource provided by HKU-Shanghai Intelligent Computing Research Center. We thank TranscEngram for supporting data collection and providing hardware used in this project. This project is also supported and funded by Shenzhen Loop Area Institute under project FPF10120250005.

References

[1] Y. Wang and M. Neff. Data-driven glove calibration for hand motion capture. In *Proceedings of the 12th ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, pages 15–24, 2013. doi:10.1145/2485895.2485901.

[2] Y. Li, J. Zhang, W. Xu, T. Tang, and C. Lu. FSGlove: An inertial-based hand tracking system with shape-aware calibration. *arXiv preprint arXiv:2509.21242*, 2025. URL https://arxiv.org/abs/2509.21242.

[3] C. Zuo, J. Huang, X. Jiang, Y. Yao, X. Shi, R. Cao, X. Yi, F. Xu, S. Guo, and Y. Qin. Transformer IMU calibrator: Dynamic on-body IMU calibration for inertial motion capture. *ACM Transactions on Graphics*, 2025. URL https://arxiv.org/abs/2506.10580.

[4] N. Gosala, F. Wang, Z. Cui, H. Liang, O. Glauser, S. Wu, and O. Sorkine-Hornung. Self-calibrated multi-sensor wearable for hand tracking and modeling. *IEEE Transactions on Visualization and Computer Graphics*, 29(3):1769–1784, 2023. doi:10.1109/TVCG.2021.3131230.

[5] L. Dipietro, A. M. Sabatini, and P. Dario. A survey of glove-based systems and their applications. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 38(4):461–482, 2008. doi:10.1109/TSMCC.2008.923862.

[6] H. Zhang, S. Dalla Gasperina, S. N. Yousaf, T. Tsuboi, T. Narita, and A. D. Deshpande. Human-exoskeleton kinematic calibration to improve hand tracking for dexterous teleoperation. *arXiv preprint arXiv:2507.23592*, 2025. URL https://arxiv.org/abs/2507.23592.

[7] J. Romero, D. Tzionas, and M. J. Black. Embodied hands: Modeling and capturing hands and bodies together. *ACM Transactions on Graphics*, 36(6):245:1–245:17, 2017. doi:10.1145/3130800.3130883.

[8] C. Zimmermann and T. Brox. Learning to estimate 3d hand pose from single RGB images. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 4903–4911, 2017.

[9] C. Zimmermann, D. Ceylan, J. Yang, B. Russell, M. Argus, and T. Brox. FreiHAND: A dataset for markerless capture of hand pose and shape from single RGB images. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 813–822, 2019. doi:10.1109/ICCV.2019.00090.

[10] S. Hampali, M. Rad, M. Oberweger, and V. Lepetit. HOnnotate: A method for 3d annotation of hand and object poses. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3196–3206, 2020.

[11] F. Zhang, V. Bazarevsky, A. Vakunov, A. Tkachenka, G. Sung, C.-L. Chang, and M. Grundmann. MediaPipe hands: On-device real-time hand tracking. *arXiv preprint arXiv:2006.10214*, 2020. URL https://arxiv.org/abs/2006.10214.

[12] G. Moon, S.-I. Yu, H. Wen, T. Shiratori, and K. M. Lee. InterHand2.6M: A dataset and baseline for 3d interacting hand pose estimation from a single RGB image. In *Computer Vision – ECCV 2020*, pages 548–564. Springer, 2020. doi:10.1007/978-3-030-58565-5_33. URL https://arxiv.org/abs/2008.09309.

[13] G. Pavlakos, D. Shan, I. Radosavovic, A. Kanazawa, D. Fouhey, and J. Malik. Reconstructing hands in 3d with transformers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9826–9836, June 2024.

[14] A. Handa, K. Van Wyk, W. Yang, J. Liang, Y.-W. Chao, Q. Wan, S. Birchfield, N. Ratliff, and D. Fox. DexPilot: Vision-based teleoperation of dexterous robotic hand-arm system. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 9164–9170, 2020. doi:10.1109/ICRA40945.2020.9197124. URL https://arxiv.org/abs/1910.03135.

[15] Y. Qin, W. Yang, B. Huang, K. Van Wyk, H. Su, X. Wang, Y.-W. Chao, and D. Fox. AnyTeleop: A general vision-based dexterous robot arm-hand teleoperation system. In *Proceedings of Robotics: Science and Systems*, Daegu, Republic of Korea, 2023. doi:10.15607/RSS.2023.XIX.015. URL https://roboticsproceedings.org/rss19/p015.html.

[16] T. Lin, Y. Zhang, Q. Li, H. Qi, B. Yi, S. Levine, and J. Malik. Learning visuotactile skills with two multifingered hands, 2024. URL https://arxiv.org/abs/2404.16823.

[17] X. Cheng, J. Li, S. Yang, G. Yang, and X. Wang. Open-television: Teleoperation with immersive active visual feedback. In *Proceedings of The 8th Conference on Robot Learning*, volume 270 of *Proceedings of Machine Learning Research*, pages 2729–2749. PMLR, 2025. URL https://proceedings.mlr.press/v270/cheng25b.html.

[18] A. Iyer, Z. Peng, Y. Dai, I. Guzey, S. Haldar, S. Chintala, and L. Pinto. OPEN TEACH: A versatile teleoperation system for robotic manipulation. *arXiv preprint arXiv:2403.07870*, 2024. URL https://arxiv.org/abs/2403.07870.

[19] A. Sivakumar, K. Shaw, and D. Pathak. Robotic telekinesis: Learning a robotic hand imitator by watching humans on YouTube. In *Proceedings of Robotics: Science and Systems*, New York City, NY, USA, 2022. doi:10.15607/RSS.2022.XVIII.023. URL https://roboticsproceedings.org/rss18/p023.html.

[20] S. Park, S. Lee, M. Choi, J. Lee, J. Kim, J. Kim, and H. Joo. Learning to transfer human hand skills for robot manipulations. *arXiv preprint arXiv:2501.04169*, 2025. URL https://arxiv.org/abs/2501.04169.

[21] K. Li, P. Li, T. Liu, Y. Li, and S. Huang. ManipTrans: Efficient dexterous bimanual manipulation transfer via residual learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6991–7003, 2025. doi:10.1109/CVPR52734.2025.00656. URL https://openaccess.thecvf.com/content/CVPR2025/html/Li_ManipTrans_Efficient_Dexterous_Bimanual_Manipulation_Transfer_via_Residual_Learning_CVPR_2025_paper.html.

[22] Z.-H. Yin, C. Wang, L. Pineda, K. Bodduluri, T. Wu, P. Abbeel, and M. Mukadam. Geometric retargeting: A principled, ultrafast neural hand retargeting algorithm, 2025. URL https://arxiv.org/abs/2503.07541.

[23] Z. Wan, Z. Bi, Z. Zhou, H. Ren, Y. Zeng, Y. Li, L. Qi, X. Yang, M.-H. Yang, and H. Cheng. RAPID hand: A robust, affordable, perception-integrated, dexterous manipulation platform for generalist robot autonomy. *arXiv preprint arXiv:2506.07490*, 2025. URL https://arxiv.org/abs/2506.07490.

[24] H.-S. Fang, B. Romero, Y. Xie, A. Hu, B.-R. Huang, J. Alvarez, M. Kim, G. Margolis, K. Anbarasu, M. Tomizuka, E. Adelson, and P. Agrawal. DEXOP: A device for robotic transfer of dexterous human manipulation. *arXiv preprint arXiv:2509.04441*, 2025. URL https://arxiv.org/abs/2509.04441.

[25] K. Junge and J. Hughes. ADAPT-Teleop: Robotic hand with human matched embodiment enables dexterous teleoperated manipulation. *npj Robotics*, 3:31, 2025. doi:10.1038/s44182-025-00034-3.

[26] D. Ha, A. M. Dai, and Q. V. Le. HyperNetworks. In *International Conference on Learning Representations (ICLR)*, 2017. URL https://dblp.org/rec/conf/iclr/HaDL17.

[27] J. von Oswald, C. Henning, B. F. Grewe, and J. Sacramento. Continual learning with hypernetworks. In *International Conference on Learning Representations (ICLR)*, 2020. URL https://openreview.net/forum?id=SJgwNerKvB.

[28] R. Karimi Mahabadi, S. Ruder, M. Dehghani, and J. Henderson. Parameter-efficient multi-task fine-tuning for transformers via shared hypernetworks. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 565–576, 2021. URL https://aclanthology.org/2021.acl-long.47/.

[29] P. Zhou, W. Yao, Q. Luo, X. Zhou, and Y. Yang. Hyper-goalnet: Goal-conditioned manipulation policy learning with hypernetworks. *arXiv preprint arXiv:2512.00085*, 2025.

[30] H. Ren, L. Sun, X. Wang, P. Zhou, Z. Wu, S. Dong, D. Zou, Y. Zheng, and Y. Yang. HyPoGen: Optimization-biased hypernetworks for generalizable policy generation. In *International Conference on Learning Representations (ICLR)*, 2025. URL https://openreview.net/forum?id=CJWMXqAnAy.

[31] L. Sun, J. Wu, F. Chen, R. Liu, and Y. Yang. HyperTASR: Hypernetwork-driven task-aware scene representations for robust manipulation, 2025. URL https://arxiv.org/abs/2508.18802.

[32] R. Karimi Mahabadi, J. Henderson, and S. Ruder. Compacter: Efficient low-rank hypercomplex adapter layers. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 34, pages 1022–1035, 2021. URL https://proceedings.neurips.cc/paper/2021/hash/081be9fdff07f3bc808f935906ef70c0-Abstract.html.

[33] E. J. Hu, Y. Shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang, and W. Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations (ICLR)*, 2022. URL https://openreview.net/forum?id=nZeVKeeFYf9.

[34] N. Ruiz, Y. Li, V. Jampani, W. Wei, T. Hou, Y. Pritch, N. Wadhwa, M. Rubinstein, and K. Aberman. HyperDreamBooth: Hypernetworks for fast personalization of text-to-image models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6527–6536, 2024. URL https://openaccess.thecvf.com/content/CVPR2024/html/Ruiz_HyperDreamBooth_HyperNetworks_for_Fast_Personalization_of_Text-to-Image_Models_CVPR_2024_paper.html.

[35] J. Su, Y. Lu, S. Pan, A. Murtadha, B. Wen, and Y. Liu. Roformer: Enhanced transformer with rotary position embedding, 2023. URL https://arxiv.org/abs/2104.09864.

[36] W. Peebles and S. Xie. Scalable diffusion models with transformers, 2023. URL https://arxiv.org/abs/2212.09748.

[37] MANUS. MANUS Metagloves Pro, 2026. URL https://www.manus-meta.com/products/metagloves-pro.

[38] Rokoko. Rokoko Smartgloves, 2026. URL https://www.rokoko.com/products/smartgloves.

[39] StretchSense. StretchSense XR Gloves, 2026. URL https://stretchsense.com/.

[40] CyberGlove Systems LLC. CyberGlove II, 2026. URL https://www.cyberglovesystems.com/cyberglove-ii.

[41] CyberOrigin. CyberOrigin Data Glove and Motion Capture, 2025. URL https://cyberorigin.ai/trending/post?id=5&title=posts%2F942b8c447cbb6586.md.

A Method Details

A.1 Augmentation Strategy for HyperNet training

To train HyperLoRA under realistic calibration errors in glove-based mocap devices, we generate synthetic biases in the MANO pose space rather than directly perturbing 3D keypoints. Let $\theta_{d,t} \in R^{45}$ denote the MANO hand pose of frame t in sequence d, reshaped as $\theta_{d,t}^{f,j} \in R^3$ for finger f and joint level j. For each sequence, we sample a bias style ϕ_d once and keep it fixed across all frames. This reflects the observation that glove or hand-tracking miscalibration is usually session-consistent, rather than independent frame-wise noise.

The sampled bias combines several physically plausible error modes: per-finger zero offsets, per-joint residual offsets, global/finger/joint gain errors, adjacent-finger crosstalk, and proximal-to-distal coupling within each finger. Concretely, the biased pose is generated as

$$\tilde{\theta}_{d,t}^{f,j} = s_d^{f,j}\theta_{d,t}^{f,j} + a_{f,j} \sum_{g \in \mathcal{N}(f)} C_d^{f,g} s_d^{g,j} \theta_{d,t}^{g,j} + \sum_{k<j} U_d^{j,k} s_d^{f,k} \theta_{d,t}^{f,k} + o_d^{f,j}, \tag{7}$$

where $s_d^{f,j}$ is a clipped multiplicative gain, C_d models leakage between adjacent fingers, U_d models tendon-like proximal-to-distal coupling, $o_d^{f,j}$ is the additive offset, and $a_{f,j}$ attenuates perturbations from proximal to distal joints. These parameters are sampled from zero-mean Gaussian families with Bernoulli activation probabilities, so each sequence receives a different but internally consistent bias type.

We then apply MANO forward kinematics to both the calibrated and noisy poses:

$$x_{d,t}^c = \Pi_{\mathrm{MANO}}(\theta_{d,t}), \qquad x_{d,t}^n = \Pi_{\mathrm{MANO}}(\tilde{\theta}_{d,t}), \tag{8}$$

and store the resulting pair $(x_{d,t}^n, x_{d,t}^c)$ offline. During training, the calibration model receives the noisy joints $x_{d,t}^n$ as input and predicts the calibrated MANO pose. *YOCO* is conditioned by a small set of calibration examples

$$\mathcal{P}_d = \left\{ (x_{d,\tau_i}^n, x_{d,\tau_i}^c) \right\}_{i=1}^K, \tag{9}$$

which are encoded by the calibration mapping to generate sequence-specific LoRA parameters. This augmentation exposes the model to a diverse family of structured, kinematically valid calibration errors while preserving the temporal consistency expected from real glove or hand-tracking biases.

A.2 LoRA Adaptation Details

All layers in the MANO estimator are equipped with LoRA adapters. Unless otherwise specified, both *YOCO* and the LoRA-Finetune baseline use LoRA rank $r = 8$ and scaling factor $\alpha = 8$. *YOCO* generates the low-rank matrices for this full adapter set from the HyperNet, while the LoRA-Finetune baseline optimizes the same adapter set by gradient descent on the paired calibration examples. This keeps the adaptation capacity fixed across the feed-forward and optimization-based LoRA comparisons.

B Learned-Prior Controls and Additional Baselines

Cold-start LoRA-Finetune uses the same adapter footprint as *YOCO*, but lacks its learned prior across calibration sessions. To separate this effect from conditioning on the specific K pairs, *Meta-LoRA* learns a session-independent LoRA initialization using the same target estimator and adapter footprint; *Meta-LoRA+FT* adapts that initialization by gradient descent on the pairs. *Set-Cond. Residual* instead predicts MANO-pose residuals from the paired set without test-time gradients. *YOCO+LoRA-FT* fine-tunes the adapters generated by *YOCO*. We also adapt the published side-tuning calibration approach of Gosala et al. [4]; its original sensor inputs and supervision differ, so this is a related-method comparison rather than a matched-device calibration protocol.

	Real glove (AVP reference)		Augmented InterHand2.6M	
Method	PA-MPJPE↓	AUCJ↑	PA-MPJPE↓	AUCJ↑
LoRA-Finetune	13.94	0.648	11.67	0.694
Set-Cond. Residual	13.74	0.645	12.66	0.664
Side-Tuning [4]	13.48	0.641	15.35	0.618
Meta-LoRA	13.51	0.633	13.94	0.639
Meta-LoRA+FT	14.17	0.664	11.41	0.702
YOCO	**13.14**	0.657	10.45	0.702
YOCO+LoRA-FT	13.28	**0.665**	**9.99**	**0.707**

Table 4: Learned-prior controls and additional calibration baselines at $K = 4$. PA-MPJPE is in millimeters; AUCJ is the normalized joint-PCK area.

Table 4 shows that, compared with the session-independent Meta-LoRA initialization, *YOCO* reduces PA-MPJPE from 13.51 to 13.14 mm on the real glove data and from 13.94 to 10.45 mm on augmented InterHand2.6M. The set-conditioned residual control and Side-Tuning provide additional checks on the adaptation mechanism and prior work. Further LoRA fine-tuning improves some scores, at the cost of test-time optimization.

C Experiment Details

C.1 MANO Beta Handling

For the evaluation on InterHand2.6M dataset, we use the ground truth shape parameter for all methods. For the evaluation on real-world dataset, we use optimization-based shape estimation method to estimate the shape parameter for all methods. For the evaluation on downstream dexterous manipulation tasks, we use a MANO shape parameter β pre-computed when collecting the AVP mocap poses for the preset-pose library, and hold this same β fixed for all methods.

C.2 Target Pose Selection

For all offline evaluation, we report the average performance over 3 random seeds for the selection of paired calibration poses. In addition, the calibration poses are selected only from the first 200 frames of the validation sequence, and evaluation is performed on the remaining frames to avoid data leakage and evaluate the generalization ability of *YOCO* and the baselines to uncalibrated poses.

C.3 Real-World Data Protocol

Our real-world glove/mocap dataset contains four users. For each frame, the noisy observation x^n is the low-cost glove-derived hand keypoint pose, while the calibrated target x^c is the synchronized 3D hand keypoint pose from Apple Vision Pro (AVP). AVP is used here as a high-end reference source for offline evaluation data, not as a device required during deployment-time calibration. Each user contributes one sequence for evaluation. Each sequence is approximately 1500 frames long at 30 FPS.

Index pinch. Thumb tip and index fingertip touch; remaining fingers stay naturally open.

Fully open. Palm open with all five fingers extended.

Claw / half close. Fingers are partially flexed in a mid-closure hand shape.

Four fingers straight, thumb near palm. Index, middle, ring, and little fingers are straight while the thumb is adducted close to the palm.

Figure 5: Preset pose library used to obtain deployable calibration targets. The four prompts are index pinch, fully open, claw / half close, and four fingers straight with the thumb near the palm.

C.4 CPU Deployment Calibration Time

The Calibration time in Table 1 and 2 are computed using H100 GPU. However, in practical teleoperation deployments, calibration may need to run on a CPU-only host. Table 5 reports the average CPU adaptation time required before inference. We use the HyperNet forwarding time for *YOCO* and the optimization/solving time for gradient-based or regression baselines.

Method	Timing Source	CPU calibration time (s)
YOCO (ours)	`HyperNet Forward`	13.16
Ridge	`Closed Form Solver`	0.37
Last-layer-Finetune	`Gradient Descent`	34.47
LoRA-Finetune	`Gradient Descent`	59.58
Full-Finetune	`Gradient Descent`	223.25

Table 5: Average CPU calibration time required before real deployment.

C.5 Real-World Per-User Variability

Table 6 reports the per-user standard deviation of the real-world mocap calibration results. This table corresponds to Table 2 to show how much each method varies across users.

Method	MPJPE↓ (mm)	PA-MPJPE↓ (mm)	Tip-PA↓ (mm)	AUCJ↑	Pinch↓ (mm)	Calib. s↓ (s)
No calibration	1.14	0.76	3.18	0.021	7.01	0.00
Ridge	4.32	2.95	9.12	0.053	12.53	0.68
Last-layer-Finetune	0.87	1.01	2.47	0.017	6.23	0.55
Full-Finetune	1.80	1.52	4.79	0.030	7.65	0.75
LoRA-Finetune	1.22	1.06	3.73	0.023	9.50	0.80
***YOCO* (ours)**	1.34	1.07	3.84	0.029	6.44	3.30

Table 6: Per-user standard deviation of the real-world mocap calibration results.

C.6 Mocap Device Selection

Several glove-based hand motion capture systems are currently available, including professional EMF-tracking gloves such as MANUS Metagloves Pro, IMU/EMF-based products such as Rokoko

Smartgloves, stretch-sensor gloves such as StretchSense gloves, and bend-sensor data gloves such as CyberGlove II [37, 38, 39, 40]. In our real-world experiment, we use OpenCyberGlove, a low-cost tensile-sensor-based glove [41]. This choice is motivated by both practicality and evaluation purpose: compared with high-end commercial mocap gloves, OpenCyberGlove provides an affordable platform for real data collection, while its tensile/strain sensor readings are naturally affected by user-specific hand geometry, glove fit, hand size, and hand shape. Therefore, it provides a representative and challenging test case for our calibration setting, where the goal is precisely to compensate for person- and fit-dependent distortions and recover accurate hand motion from limited paired calibration data.

C.7 Additional K-Sweep Results

Method	K	MPJPE↓ (mm)	PA-MPJPE↓ (mm)	Tip-PA↓ (mm)	AUCJ↑	Pinch L2↓ (mm)
YOCO (ours)	1	16.07	10.94	24.40	0.694	38.04
	2	16.10	10.86	24.30	0.694	36.66
	4	15.64	10.45	23.29	0.702	34.90
	8	15.21	10.19	22.26	0.709	33.87
	16	15.10	10.14	21.94	0.712	33.70
LoRA FT	1	18.96	14.07	31.33	0.651	43.44
	2	17.80	13.00	28.54	0.671	42.01
	4	16.38	11.67	25.33	0.694	37.28
	8	15.44	10.65	22.92	0.709	33.42
	16	15.12	10.29	22.06	0.714	32.89
Last-layer FT	1	19.10	14.22	31.64	0.649	43.54
	2	18.04	13.34	29.20	0.669	42.57
	4	16.62	12.01	25.92	0.692	39.18
	8	15.54	10.84	23.16	0.709	35.72
	16	15.18	10.50	22.26	0.715	35.10
Ridge	1	20.18	15.24	33.87	0.633	47.44
	2	19.10	14.35	31.66	0.654	48.25
	4	17.62	13.00	28.28	0.676	45.35
	8	16.76	12.13	26.44	0.688	46.35
	16	16.07	11.47	24.87	0.698	45.03
Full FT	1	25.62	20.20	48.08	0.570	48.71
	2	22.79	17.26	40.76	0.607	45.11
	4	18.11	12.77	29.21	0.668	37.11
	8	15.39	10.17	22.39	0.706	32.60
	16	14.55	9.47	20.39	0.722	30.88

Table 7: Detailed K-sweep results on the augmented InterHand2.6M evaluation setting. All rows use the overall split aggregated over 3 seeds and 20 augmented units.

C.8 Additional Teleoperation Statistics

Method	Bottle Cap Opening Avg. Time (s)↓	Earbud Pinch Grasping Avg. Time (s)↓	3-Finger Glue Grasping Avg. Time (s)↓	Tissue Box Opening Avg. Time (s)↓
No calibration	37.9	7.0	10.5	42.0
LoRA-Finetune	37.5	6.9	**8.1**	19.0
***YOCO* (ours)**	**27.3**	**4.8**	9.9	**17.4**
Longest trial	59.0	7.0	14.0	42.0

Table 8: Detailed downstream teleoperation completion-time statistics. Values are average completion time in seconds over 10 trials; lower is better. The longest-trial row reports the longest observed trial time for each task.

C.9 Device and Compute Platform

All training runs and timing measurements are performed on a single-GPU server. Table 9 summarizes the hardware and operating system used for model training and timing experiments.

Component	Value
CPU	2× Intel Xeon Platinum 8468V
RAM	2.0 TiB
GPU	1× NVIDIA H100 80 GB HBM3

Table 9: Device and compute platform used for training and timing experiments.

C.10 Training Hyperparameters

Table 10 reports the main optimization, HyperNet, LoRA, and paired-pose preference-extractor hyperparameters used in our training runs.

Setting	Value
Seed	42
Optimizer	Adam
Total steps	10,000
Batch size	1024
Peak learning Rate	`3e-4`
Final Learning Rate	`1e-6`
Warmup Steps	500
Gradient clipping	1.0
LoRA rank / alpha	Rank 8; alpha 8.0
LoRA dropout	0.1
Calibration Mapper Transformer Depth	6
Calibration Mapper Transformer Number of Heads	8
Decoder Flow Steps	8
Decoder DIT Number of Layers	4
Decoder DIT Number of Heads	4
Hidden Dimension	256

Table 10: Training hyperparameters used by *YOCO*.