# $\chi_0$: Resource-Aware Robust Manipulation via Taming Distributional Inconsistencies

Checheng Yu, Chonghao Sima, Gangcheng Jiang, Hai Zhang, Haoguang Mai, Hongyang Li, Huijie Wang, Jin Chen, Kaiyang Wu, Li Chen, Lirui Zhao, Modi Shi, Ping Luo, Qingwen Bu, Shijia Peng, Tianyu Li, Yibo Yuan

[1]**Kinetix AI** [2]**The University of Hong Kong**

Code: https://github.com/OpenDriveLab/kai0

Blog: https://mmlab.hk/research/kai0

***Abstract*—High-reliability long-horizon manipulation has traditionally relied on large-scale data and compute. For deformable garments, however, we identify that the primary bottleneck to robustness is not resource scale alone, but the distributional shift among the human demonstration distribution, the policy's inductive bias, and the test-time execution distribution. These inconsistencies cause compounding errors in multi-stage tasks such as flattening, folding, handover, sorting, and hanging. To mitigate them, we present $\chi_0$, a resource-aware framework for robust dual-arm garment manipulation. Our approach builds on three technical pillars: (i) Model Arithmetic, a weight-space merging strategy that efficiently soaks up diverse distributions of demonstrations, varying from object appearance to state variations; (ii) Stage Advantage, a stage-aware advantage estimator that provides stable, dense progress signals and avoids the numerical instability of value-difference methods; and (iii) Train-Deploy Alignment, which bridges the deployment gap via spatio-temporal augmentation, heuristic DAgger corrections, and temporal chunk-wise smoothing. With roughly 20 hours of demonstrations per task and 8 A100 GPUs, $\chi_0$ improves success rate over the open-source $\pi_{0.5}$ baseline by nearly 250%. We offer this as a field report on combining end-to-end vision-language-action policies with modular alignment mechanisms for scalable non-rigid manipulation.**

## I. Introduction

Production-ready robustness is the grand challenge of modern robotics. While autonomous vehicles show operational viability in complex urban environments, matching this reliability for manipulation in unstructured, human-centered settings remains open, and the challenge is especially salient for non-rigid objects. Garments do not admit a stable 6-DoF pose representation; they wrinkle, self-occlude, slip under contact, and contain many visually similar but semantically distinct states. These properties make laundry-like manipulation a representative testbed for deformable-object perception, representation, planning, and control.

Such robustness requires a policy to operate within a vast search space, and the prevailing industrial paradigm has pivoted toward scaling large robotic foundation models [3, 2, 1]. While architectural evolution and resource scaling are essential, we argue that the decisive factor in robust policy execution is not scale alone. Our key insight, inherited from the RSS main-track version of this work, is that the hidden bottleneck is *the inconsistency among the distributions governing the three pillars of robot learning: data collection, model training, and policy deployment*. These inconsistencies are not always evident from a single success-rate number; they appear in execution smoothness, throughput, and retry cost [1].

We encapsulate the robot learning pipeline into three distributions. $P_{\text{train}}$ denotes the distribution of human expert demonstrations used to train an imitation policy. $Q_{\text{model}}$ denotes the inductive bias learned by the policy, i.e., a state-to-plausible-actions mapper. $P_{\text{test}}$ denotes the distribution of executed trajectories during deployment, which differs from direct policy outputs due to inference latency, actuator limits, contact dynamics, and recovery behavior. For deformable garments, this mismatch is amplified: demonstrations sparsely cover a high-dimensional state manifold, small contact errors induce large cloth-state changes, and a visually plausible action can be wrong for the current stage.

Real-world deployment reveals three systematic inconsistencies, summarized in Fig. 1. First, due to the extreme dimensionality of the task, $P_{\text{train}}$ is inherently sparse relative to the full solution manifold, resulting in a $Q_{\text{model}}$ that is biased toward the limited training distribution. Second, the latency between model inference ($Q_{\text{model}}$) and control-level execution ($P_{\text{test}}$) introduces temporal mismatch, rendering theoretically optimal plans suboptimal during inference [4, 11]. Third, despite frequent failures during inference, the policy often lacks failure-recovery ability; minor perturbations in $P_{\text{test}}$ can trigger catastrophic divergence even when the nominal task states are present in $P_{\text{train}}$ [10].

We propose $\chi_0$, a resource-efficient framework that resolves these misalignments within the constraints of physical robotics. It retains an end-to-end VLA policy as the action generator, but adds modular mechanisms for data coverage, task-stage representation, and deployment smoothing. This hybrid design raises a key question: how should end-to-end and modular paradigms be combined for non-rigid object manipulation? Training with roughly 20 hours of demonstrations on 8×A100 GPUs, $\chi_0$ outperforms the open-source $\pi_{0.5}$ baseline by nearly **250%** in success rate on long-horizon garment tasks.

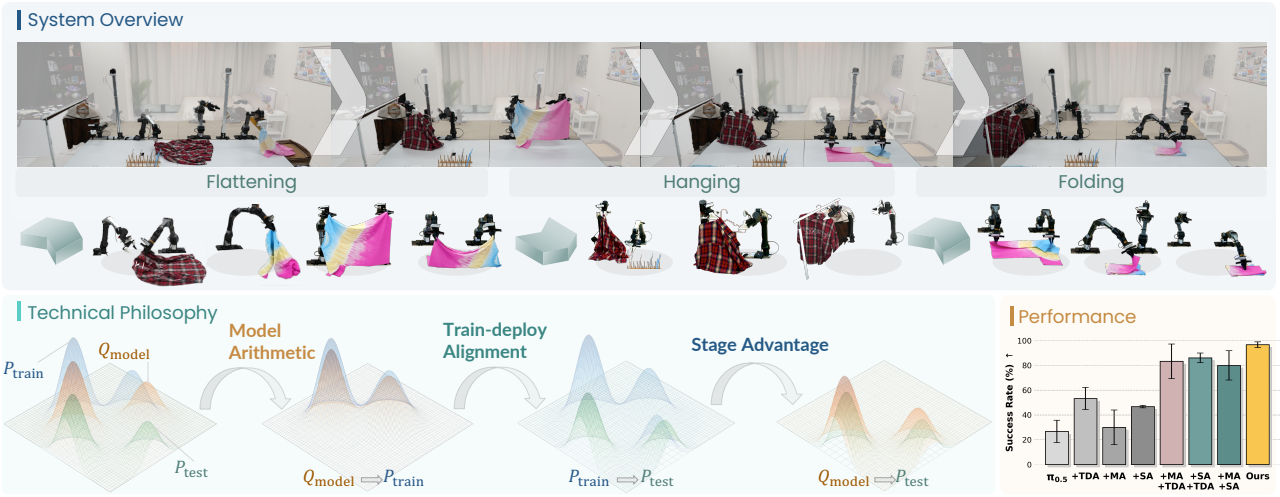



Fig. 1: **System overview.** $\chi_0$ studies long-horizon deformable garment manipulation with two dual-arm robots. Distributional inconsistencies are inherent to robot learning ($P_{\text{train}}$: expert demonstrations; $Q_{\text{model}}$: policy inductive bias; $P_{\text{test}}$: deployment trajectories). $\chi_0$ addresses the pairwise mismatches through Model Arithmetic, Stage Advantage, and Train-Deploy Alignment, enabling robust flattening, folding, sorting, handover, and hanging.

## II. Problem Setup: Distributional Alignment

To formalize the above view, we consider a finite-horizon Markov Decision Process with state space $\mathcal{S}$, action space $\mathcal{A}$, and horizon $H$. A trajectory $\tau = (s_0, a_0, s_1, a_1, \ldots, s_H)$ evolves under real-world dynamics $s_{t+1} \sim T(\cdot \mid s_t, a_t, \xi)$ with initial-state distribution $\mu(s_0)$. The trajectory distribution induced by a stochastic policy $\pi(a \mid s; \phi)$ is

$$P_\pi(\tau) = \mu(s_0) \prod_{t=0}^{H-1} \pi(a_t \mid s_t; \phi) T(s_{t+1} \mid s_t, a_t, \xi). \quad (1)$$

Let $P_{\text{real}}$ denote the distribution over successful real-world trajectories for a task. The training distribution $P_{\text{train}}$ is induced by human expert demonstrations on real robots, and $Q_{\text{model}} \triangleq \pi(a \mid s; \hat{\phi})$ is learned by maximizing imitation likelihood over the demonstration set $\mathcal{D} = \{\tau^{(i)}\}_{i=1}^{N}$ [10, 8]. Finally, real-robot execution produces $P_{\text{test}}$ by composing $Q_{\text{model}}$ with an inference and control operator $I(\tilde{a}_t \mid a_t, s_t)$, where $\tilde{a}_t$ denotes the actually executed action. This matters for deformables: the intended joint-space chunk and the executed motion can diverge exactly when cloth contact is most sensitive.

We categorize the resulting gaps as **coverage deficiency**, **temporal mismatch**, and **failure cascade**. Coverage deficiency: $P_{\text{train}}$ undersamples the high-dimensional manifold $P_{\text{real}}$, biasing $Q_{\text{model}}$ toward limited modes. Temporal mismatch: long-horizon tasks induce visually similar but semantically distinct states across stages, and inference-control latency misaligns planned and realized chunks. Failure cascade: the lack of recovery behaviors in $P_{\text{train}}$ leaves the policy unable to self-correct from perturbations in $P_{\text{test}}$. $\chi_0$ addresses these three inconsistencies with targeted alignment strategies.

## III. Methodology

### A. Pipeline of $\chi_0$

$\chi_0$ integrates three complementary technical pillars across the robot learning cycle. Model Arithmetic expands policy coverage by merging models trained on complementary data subsets in weight space. Stage Advantage addresses temporal mismatch at the policy-learning phase through stage-aware progress supervision. Train-Deploy Alignment closes the loop between deployment and training through inference optimization and complementary data augmentation. Together, these modules convert a generic VLA backbone into a deployable policy for long-horizon deformable garment manipulation.

### B. Model Arithmetic for Coverage Deficiency

Limited demonstrations cause coverage deficiency in $P_{\text{train}}$, biasing learned policies toward narrow manipulation patterns. Simply scaling demonstrations until $P_{\text{train}}$ approximates $P_{\text{real}}$ is prohibitively expensive for garments, where each collection cycle demands extensive human operation and real-robot occupancy.

We propose **Model Arithmetic (MA)**, a weight-space merging strategy that combines policies trained on complementary data subsets via validation-based optimization. Unlike Mixture-of-Experts (explicit routers, complex training) or output ensembling, MA directly merges parameters into a unified policy with no inference-time routing cost. Given data subsets $\{D_1, D_2, \ldots, D_n\}$ sampled from $P_{\text{train}}$, we train policies $\{\theta_1, \theta_2, \ldots, \theta_n\}$ independently and synthesize their weights by

$$\theta_{\text{merged}} = \sum_{i=1}^{n} \alpha_i \theta_i, \quad \alpha_i \geq 0, \quad \sum_i \alpha_i = 1. \quad (2)$$

The mixing coefficients minimize a held-out validation loss, and the critical design choice is validation-set selection. We

construct an out-of-distribution validation set from recovery trajectories collected via DAgger [10, 7]; these recovery behaviors are naturally absent from the original nominal demonstrations. Based on this split, we ablate average weighting, inverse-loss weighting, gradient-descent weighting, and greedy search [12, 6]. Validation-guided weight-space synthesis combines diverse unimodal policies into a single multi-modal policy, mitigating the bias of $Q_{\text{model}}$ induced by sparse garment demonstrations.

### C. Stage Advantage for Long-Horizon Progress

While MA mitigates $Q_{\text{model}}$ bias, the policy still struggles with long-horizon execution in $P_{\text{test}}$. Visually similar states across stages cause misapplied behaviors and compounding errors: a partially folded shirt can resemble an earlier flattening state, yet the correct action depends on whether the robot is recovering, folding, handing over, or hanging.

Prior work uses advantage as a progress signal with advantage-weighted regression [9, 1]. One common formulation obtains advantage implicitly as $A(s,a) = V(s') - V(s)$, taking the difference between independently predicted progress values. This amplifies frame-wise noise into high-variance training signals, and estimating global progress without stage awareness can make $V(s)$ multi-valued.

To obtain a stable and accurate signal, we treat advantage as a **direct modeling target**: $A(s,a) = f_\theta(s,s')$, where $f_\theta$ predicts relative progress from $s$ to $s'$. We construct training pairs by randomly sampling a time span $\Delta$ and setting $s' = s_{t+\Delta}$, avoiding overfitting to a fixed temporal discretization. To resolve multi-valued ambiguity, **Stage Advantage** conditions this estimator on a semantic stage label $g$:

$$A_{\text{stage}}(s,a,g) = f_\theta(s,s' \mid g). \tag{3}$$

We use manually annotated stage labels represented as a normalized scalar $g \in \{0, \frac{1}{S}, \ldots, \frac{S-1}{S}\}$ for $S$ stages. Following prior advantage-weighted policy learning [5, 1], the continuous prediction is thresholded into a binary optimality indicator $I = \mathbb{1}[A_{\text{stage}} > \epsilon]$, which upweights high-quality data while reducing temporal confusion.

### D. Train-Deploy Alignment for Physical Execution

Even with improved coverage and stage-aware progress, deployment introduces new inconsistencies between $Q_{\text{model}}$ and $P_{\text{test}}$. Inference-control latency causes misplaced execution and drift, especially for action-chunking policies: the gap between inference and chunk execution breaks temporal continuity across chunks, causing abrupt transitions and degraded cloth-contact stability. We therefore adopt **temporal chunk-wise smoothing**. Let $\mathbf{a}^{old}$ denote the current action buffer and $\mathbf{a}^{new}$ the newly predicted action chunk. We drop stale prefix commands according to a threshold $d_{\max}$, blend the overlap between the residual buffer and the new chunk, and append the remaining suffix of $\mathbf{a}^{new}$. This suppresses abrupt target changes at chunk switches and can be combined with real-time chunking methods [4, 11].

TABLE I: **Real-robot garment tasks.** All start from arbitrary initial states; Tasks B and C add conditional long-horizon structure.

| Task | Goal and main challenge |
|---|---|
| A | Flatten and fold a T-shirt within a time limit; tests recovery and basic folding. |
| B | Retrieve a T-shirt or collared shirt, then fold or hand over by type; tests perception-conditioned branching. |
| C | Hang a collared shirt on a rod under occlusion; tests contact-rich, longest-horizon manipulation. |

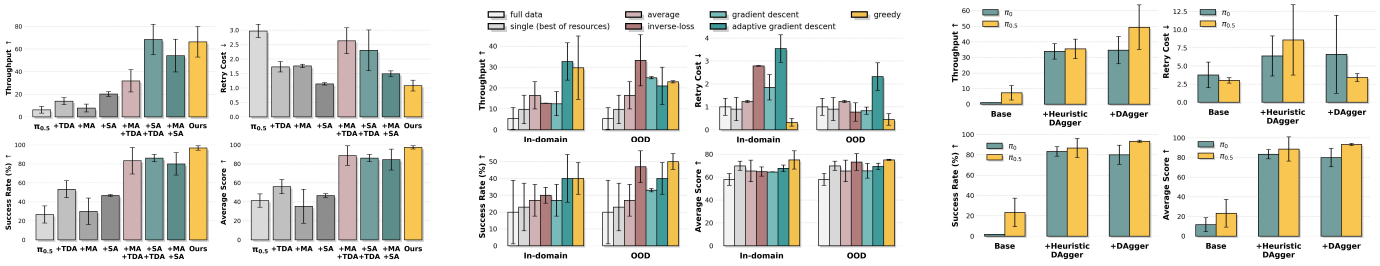
To expand $P_{\text{train}}$ toward deployment-critical regions, we further close the loop between deployment and training. Standard on-policy DAgger expands $P_{\text{train}}$ toward failure-adjacent regions but is time-consuming because it waits for natural failures. We propose a **Heuristic DAgger** variant that directly initializes the system in manually designed recoverable failure states, such as misaligned grasps or partial drops, and records corrective demonstrations. To diversify data at zero robot time, we also apply **spatio-temporal augmentation**: horizontal flipping with left/right arm swapping, and partial frame-skipping to synthesize speed variations. These data-side and execution-side mechanisms jointly reduce the gap between demonstrations and physical deployment.

## IV. Experimental Setup and Results

Our evaluation targets collaborative long-horizon garment manipulation: flattening from arbitrary states, folding, handover, sorting, and hanging. These contact-rich tasks require state recovery and thus magnify the distributional shifts above. The system uses two dual-arm robots with multi-view RGB observations (wrist and fixed main-view cameras). We curate roughly 20 hours of demonstrations per task, spanning diverse garment states, colors, materials, configurations, and lighting. We employ full-parameter fine-tuning on 8×A100 GPUs via a flow-matching objective [3].

**Metrics and baselines.** We report success rate (SR), throughput (TP), retry cost, and a rule-based average score. SR is successful trials over total attempts; TP is successful completions per evaluation time; retry cost counts retries; average score gives partial credit to task milestones, normalized to 100. We use $\pi_{0.5}$ as the primary base policy and include $\pi_0$ where applicable, since they are the only open-source VLA policies that achieved viable performance on our tasks after fine-tuning. Other reported policies did not produce tractable behavior in this setting after the same 20-hour data budget.

**System-level effect.** Fig. 2(a) reports the performance breakdown of each module on Task A. By selecting the optimal configuration for MA, SA, and TDA, performance scales monotonically as components are added. SA is the dominant factor for throughput, whereas TDA drives success rate but can incur higher retry cost. This trade-off aligns with our distributional view: TDA encourages persistent recovery rather than silent failure, which naturally improves task completion at the expense of additional corrective actions. The complete system outperforms the open-source $\pi_{0.5}$ baseline by nearly




(a) System efficacy. (b) Model Arithmetic. (c) Recovery data.

Fig. 2: **Representative experimental results.** (a) Adding MA, SA, and TDA monotonically improves the base policy, with the complete $\chi_0$ system achieving the best trade-off. (b) MA variants outperform single-best and full-data candidates, especially when OOD recovery data is used for validation. (c) Heuristic and standard DAgger improve success and throughput by injecting recovery-critical states into $P_{\text{train}}$.

250% in success rate and sustains a 24-hour real-robot stress test in our deployment.

**Model Arithmetic.** Fig. 2(b) benchmarks MA variants against non-MA baselines. All MA variants outperform both the single-best subset-trained candidate and the full-data candidate. This suggests fine-tuned VLAs contain complementary local modes that are easier to consolidate via weight-space synthesis than monolithic retraining. OOD validation data also provides a more robust criterion than in-domain validation, with lower standard error and higher performance. DAgger recovery trajectories make a useful validation split, exposing whether $Q_{\text{model}}$ covers deployment-critical modes missing from nominal $P_{\text{train}}$.

**Stage Advantage.** SA provides more stable progress supervision than value-difference training: the direct two-timestep signal avoids subtracting two noisy value predictions, and stage conditioning removes the multi-valued ambiguity when similar cloth appearances map to different subgoals. In Task B, with conditional retrieval and handover/folding branches, this stability reduces idling and retries. These results indicate that a lightweight semantic-stage representation supplies much of the planning structure needed for long-horizon deformable manipulation without an explicit cloth dynamics model.

**Train-Deploy Alignment.** Fig. 2(c) evaluates recovery-data injection across models and tasks. Heuristic DAgger significantly improves failure recovery, achieving higher SR and TP than the baseline; its higher retry cost reflects active self-correction, not inefficiency. Unlike standard DAgger, heuristic initialization reaches useful recovery states without waiting for natural failures. On the execution side, temporal chunk-wise smoothing outperforms synchronous/asynchronous inference [11] and temporal ensembling [13] in most settings, and can be combined with RTC for further gains. Overall, robust non-rigid manipulation depends not only on policy quality, but also on aligning the policy with physical contact, latency, and recoverable failure.

## V. Conclusion

$\chi_0$ studies deformable garment manipulation as a distributional alignment problem across demonstrations, learned policies, and real-robot deployment. The results suggest that strong VLA backbones become substantially more reliable when paired with modular mechanisms for coverage, stage-aware progress, and execution alignment. This offers one practical path toward scalable non-rigid object manipulation under realistic data and compute budgets.

## References


[1] Ali Amin, Raichelle Aniceto, Ashwin Balakrishna, Kevin Black, Ken Conley, Grace Connors, James Darpinian, Karan Dhabalia, Jared DiCarlo, et al. $\pi^*_{0.6}$: A VLA that learns from experience. *arXiv.org*, 2511.14759.

[2] Kevin Black, Noah Brown, James Darpinian, Karan Dhabalia, Danny Driess, Adnan Esmail, Michael Robert Equi, Chelsea Finn, Niccolo Fusai, Manuel Y Galliker, et al. $\pi$0.5: A vision-language-action model with open-world generalization. In *CoRL*, 2025.

[3] Kevin Black, Noah Brown, Danny Driess, Adnan Esber, Michael Equi, Chelsea Finn, et al. $\pi_0$: A vision-language-action flow model for general robot control. In *RSS*, 2025.

[4] Kevin Black, Manuel Y Galliker, and Sergey Levine. Real-time action chunking for robot control. In *NeurIPS*, 2025.

[5] Li Chen, Chonghao Sima, Kashyap Chitta, Antonio Loquercio, Ping Luo, Yi Ma, and Hongyang Li. Intelligent robot manipulation requires self-directed learning. *Authorea Preprints*, 2026.

[6] Gabriel Ilharco, Marco Tulio Ribeiro, Mitchell Wortsman, Suchin Gururangan, Ludwig Schmidt, Hannaneh Hajishirzi, and Ali Farhadi. Editing models with task arithmetic. In *ICLR*, 2023.

[7] Michael Kelly, Chelsea Sidrane, Katherine Driggs-Campbell, and Mykel J Kochenderfer. HG-DAgger: Interactive imitation learning with human experts. In *ICRA*, 2019.

[8] Takayuki Osa, Joni Pajarinen, Gerhard Neumann, J Andrew Bagnell, Pieter Abbeel, and Jan Peters. An algorithmic perspective on imitation learning. *Foundations and Trends® in Robotics*, 2018.

[9] Xue Bin Peng, Aviral Kumar, Grace Zhang, and Sergey Levine. Advantage-weighted regression: Simple and scalable off-policy reinforcement learning. *ICLR*, 2021.

[10] Stephane Ross, Geoffrey Gordon, and Drew Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In *AISTATS*, 2011.

[11] Jiaming Tang, Yufei Sun, Yilong Zhao, Shang Yang, Yujun Lin, Zhuoyang Zhang, James Hou, Yao Lu, Zhijian Liu, and Song Han. Vlash: Real-time vlas via future-state-aware asynchronous inference. *arXiv preprint arXiv:2512.01031*, 2025.

[12] Mitchell Wortsman, Gabriel Ilharco, Samir Ya Gadre, Rebecca Roelofs, Raphael Gontijo-Lopes, Ari S Morcos, Hongseok Namkoong, Ali Farhadi, Yair Carmon, Simon Kornblith, et al. Model soups: Averaging weights of multiple fine-tuned models improves accuracy without increasing inference time. In *ICML*, 2022.

[13] Tony Z Zhao, Vikash Kumar, Sergey Levine, and Chelsea Finn. Learning fine-grained bimanual manipulation with low-cost hardware. In *RSS*, 2023.