Sequential Asymmetric Imitation for Learning Coupled Robot Policies

Yincong Chen[1,*] **Ranpeng Qiu**[1,3,*] **Zihao Li**[1,2] **Yanan Zhou**[1,4]

Guoqiang Ren[1] **Weiming Zhi**[1,4,†]

[1] Zeno AI [2] Zhejiang University
[3] Zhejiang University of Technology [4] The University of Sydney
[*] Equal contribution
[†] Corresponding Author

Abstract: Collaborative mobile manipulation requires robots to coordinate with a partially observed partner while physically interacting through shared objects. This is difficult because failures often arise not from poor local skills, but from mistimed waiting, yielding, pulling, releasing, or repositioning. We study this problem with two bimanual mobile manipulators coupled through rigid and deformable objects. We propose *Sequential Asymmetric Imitation* (SAI), a single-teleoperator curriculum for learning coupled multi-robot behaviors without synchronized dual-operator demonstrations or explicit inter-robot communication. SAI trains Robot A from unilateral demonstrations with a compliant human partner, trains Robot B against the deployed Robot A policy, and then refines Robot A using sparse interventions near coordination failures. This staged process exposes the policies to increasingly realistic partner behaviors, including delay, phase mismatch, insufficient yielding, and interaction conflict. Across real-world dual-robot manipulation tasks, SAI improves task success, phase synchronization, and partner-contingent yielding over independent imitation and curriculum-ablation baselines. These results suggest that physically coupled collaboration can be learned through the structure of the imitation curriculum, rather than through synchronized multi-operator demonstrations or explicit coordination mechanisms.

1 Introduction

Dual-robot manipulation enables robots to handle large, heavy, and deformable objects that are difficult for a single robot to control. However, success in this setting requires more than two locally competent policies. When robots pull, lift, align, transport, or spread a shared object, each robot's motion changes the load, contact state, and feasible motion of the other. Small timing errors can therefore produce large interaction forces, object deformation, loss of grasp, or task failure. Recent work has made substantial progress in robot manipulation from demonstrations, including mobile manipulation, dual-arm manipulation, and visuomotor policy learning with expressive architectures such as action chunking transformers and diffusion policies [7, 3, 2]. Yet physically coupled dual-robot tasks introduce a distinct coordination problem. The robots must align task phases, adapt to partner delays, yield under interaction conflict, and resume coordinated motion after perturbations. Independent policies often fail because each robot follows its own nominal motion phase, even when the partner is delayed, stalled, or out of position.

Existing approaches face practical bottlenecks in this setting. Model-based planning and centralized control can coordinate multiple robots when object geometry, contacts, and interaction constraints are well specified [23, 21, 22], but these assumptions are hard to maintain with deformable objects, intermittent contacts, friction, and long-horizon mobile manipulation. Multi-agent reinforcement learning provides a general coordination framework [16], but real-robot deployment remains difficult

Figure 1: **Local skills are not enough for dual-robot coordination.** Independent policies can fail under phase mismatch, insufficient yielding, and physical interaction conflict. SAI induces coupled behavior.

due to sparse rewards [17], reward shaping, and simulation-to-real transfer. Imitation learning is more direct, but standard dual-robot imitation typically requires synchronized joint demonstrations, where both robots are controlled as a coordinated pair. Collecting such data requires multiple teleoperation channels, calibrated interfaces, and operators who can maintain precise temporal alignment [14].

This data-collection requirement motivates our central question: *Can coupled dual-robot policies be learned from single-teleoperator demonstrations, without synchronized dual-operator data, reinforcement learning, or explicit inter-robot communication?*

We propose *Sequential Asymmetric Imitation* (SAI), a staged imitation curriculum for learning physically coupled dual-robot behavior. SAI decomposes coordination learning into three asymmetric stages. First, Robot A learns basic task execution from unilateral demonstrations with a compliant human or passive partner. Second, Robot A is deployed as a frozen policy while a single operator teleoperates Robot B, allowing Robot B to learn against the timing, motion profile, and errors of its future robot partner. Third, both policies are deployed together, and sparse interventions near coordination failures refine Robot A with behaviors such as waiting, yielding, slowing, and recovery.

Our key hypothesis is that inter-robot coordination can be induced by structuring the partner distribution seen during imitation learning. SAI shifts training from compliant support, to a deployed learned partner, to closed-loop robot-robot execution with targeted corrections. This exposes the policies to realistic coordination errors, including partner delay, phase mismatch, insufficient yielding, and interaction conflict, enabling coordination through local observations and shared physical interaction without explicit communication or centralized action prediction.

Concretely, our technical contributions are:

1. **Sequential Asymmetric Imitation.** We introduce SAI, a single-teleoperator curriculum for learning physically coupled dual-robot policies without synchronized dual-operator demonstrations, reinforcement learning, or explicit inter-robot communication.
2. **Partner-distribution curriculum.** We show that shifting the training partner from compliant support, to a frozen learned policy, to closed-loop robot-robot execution induces phase alignment and partner-contingent yielding.
3. **Real-world dual-robot validation.** We evaluate SAI on contact-rich real-world tasks involving rigid and deformable shared objects, showing improvements in task success, phase synchronization, and yielding behavior over independent imitation and curriculum-ablation baselines.

2 Related Work

Dual-Robot and Multi-Arm Manipulation: Classical methods coordinate multiple manipulators by modeling object geometry, contacts, and kinematic constraints, but these assumptions weaken with deformable materials, intermittent contacts, and unmodeled interaction dynamics [11, 4]. Learning-based methods reduce manual modeling through visuomotor imitation [1, 18], including diffusion policies, keypoint representations [9], and bimanual foundation models [12]. Most, however, rely on synchronized joint trajectories that supervise both agents as one coordinated behavior [19, 20]. This

increases data-collection cost, entangles agent roles, and can make policies sensitive to partner timing deviations [10]. SAI instead induces dual-robot coordination through staged partner-distribution shifts using single-teleoperator, asynchronous demonstrations.

Teleoperation-Based Imitation Learning: Teleoperation-based imitation methods such as ACT [7] and Diffusion Policy [3] achieve strong visuomotor manipulation performance, but collaborative dual-robot tasks require demonstrations that capture temporal coupling between agents. Existing multi-arm systems often use stationary bimanual setups or specialized capture and teleoperation interfaces [2, 13] to obtain aligned trajectories, increasing hardware and operator requirements. Recent work also studies efficient data collection and dataset scaling for robot manipulation [24, 25], but synchronized collaborative data on mobile manipulators, studied in this work, remain expensive to collect. SAI assumes joint demonstrations are unavailable and bootstraps coordination through a sequential, asymmetric *single*-teleoperator curriculum.

Interactive Imitation and Intervention Learning: Robot learning from demonstration has studied how skills can be acquired from human examples through general imitation-learning formulations [26], dynamical-system-based policies [27], and synchronized movement primitives for bimanual manipulation [28]. These approaches typically assume that the relevant coordination structure is present in the demonstrations or specified through primitive-level synchronization. Interactive imitation methods, including DAgger and intervention-based variants, reduce covariate shift by collecting corrections under the learner-induced state distribution [5, 15]. SAI uses this principle only in its final stage, but targets coordination-specific failures such as partner delay, phase mismatch, and near-failure contact configurations [6]. Targeted loss masking and nominal replay preserve existing skills while adding partner-contingent yielding and recovery.

3 Problem Formulation

Dual-Robot Imitation Learning Setup: We formulate cooperative dual-robot manipulation as a decentralized partially observable control problem with global state $s_t \in \mathcal{S}$. At each timestep t, each robot $i \in A, B$ receives a local visuo-proprioceptive observation $o_i^t = O_i(s_t)$, and acts according to its own policy:

$$a_A^t \sim \pi_A(\cdot \mid o_A^t), \qquad a_B^t \sim \pi_B(\cdot \mid o_B^t). \tag{1}$$

The policies do not exchange messages, partner states, future actions, or latent embeddings. Coordination can only emerge through local observations of the shared workspace and the physical effects of manipulating the same object. Given demonstration data, the objective is to learn decentralized policies π_A and π_B by supervised imitation such that their joint rollout $\tau = (o_A^t, o_B^t, a_A^t, a_B^t)_{t=1}^T$ satisfies the task-success criteria with high probability.

Collaboration Challenges: Local skill does not imply cooperative skill. Even if each robot can grasp, lift, pull, or transfer objects in isolation, joint deployment can fail because both policies interact through the shared object and workspace. We focus on three recurring bottlenecks in physically coupled dual-robot manipulation:

- **Temporal phase coupling:** the robots must align key task transitions, such as grasping, motion onset, lifting, lowering, and release, despite variable execution timing.
- **Partner-contingent yielding:** each robot must slow, pause, or re-time its motion when the partner is delayed, stalled, obstructed, or out of phase.
- **Interaction conflict mitigation:** the joint behavior must avoid excessive internal forces, object deformation, opposing motions, and workspace collisions.

We call this ability *implicit policy coupling*: partner-responsive coordination that arises from local observations and shared physical interaction, without explicit inter-robot communication. A coupled policy pair should maintain phase-aligned progress, yield to partner deviations, and avoid interaction-induced failures during closed-loop execution.

Single-Teleoperator Constraints: Standard dual-robot imitation learning assumes synchronized joint demonstrations, where both robots are controlled and recorded as one coordinated trajectory. This

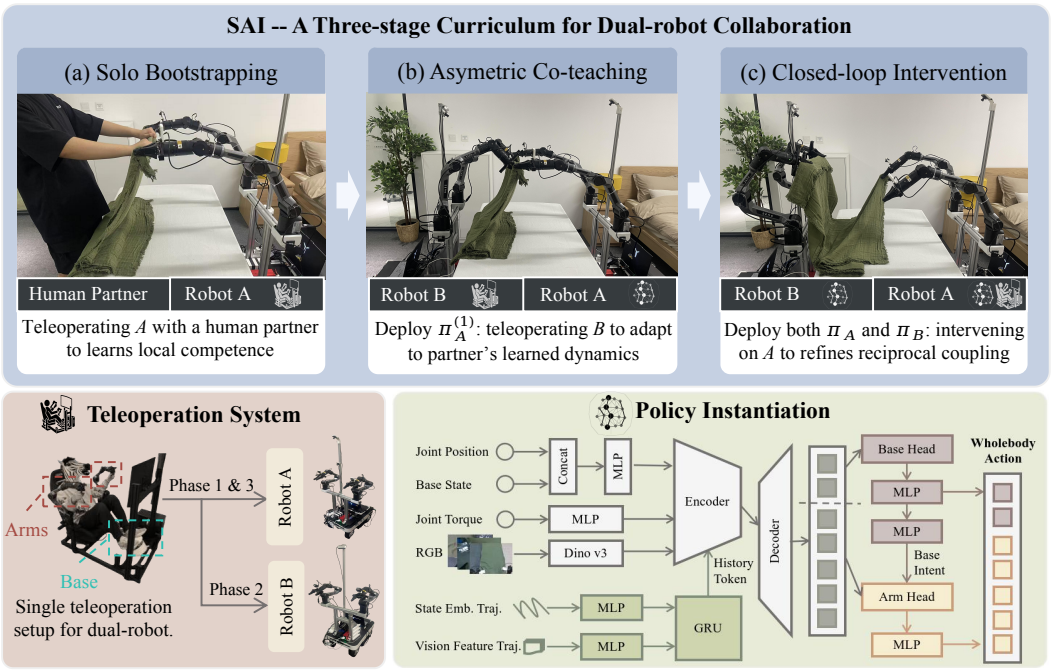


Figure 2: **System overview of SAI.** A single whole-body teleoperator [8] collects a staged curriculum for policy coupling; bottom panels show the teleoperation interface and policy instantiation.

can require multiple teleoperation channels, calibrated interfaces, and precise temporal alignment between operators. We instead impose a single-teleoperator constraint: only one robot can be directly teleoperated at a time through one hardware interface. Under this constraint, supervision is decomposed into three asymmetric datasets: $\mathcal{D}_A^{\text{solo}}$teleoperate Robot A with a compliant human or passive partner; $\mathcal{D}_B^A$teleoperate Robot B while the learned Robot-A policy is deployed; $\mathcal{D}_A^{\text{int}}$ intervene on Robot A during closed-loop dual-robot deployment. The superscript in $\mathcal{D}_B^A$ indicates that Robot B demonstrations are collected under the partner distribution induced by the learned Robot-A policy. The central question is whether these sequential, asymmetric datasets are sufficient to learn coupled decentralized policies that approach the cooperative behavior normally obtained from synchronized joint demonstrations.

4 Sequential Asymmetric Imitation

Sequential Asymmetric Imitation (SAI) learns physically coupled dual-robot manipulation without requiring synchronized dual-operator demonstrations. The key idea is to build coordination gradually: first train one robot to perform the task with external support, then train the second robot against the deployed first robot, and finally correct the remaining closed-loop coordination failures through sparse human interventions.

SAI follows the data-collection sequence

$$\mathcal{D}_A^{\text{solo}} \rightarrow \mathcal{D}_{B|A}^{\text{demo}} \rightarrow \mathcal{D}_A^{\text{int}}. \tag{2}$$

Here, $\mathcal{D}_A^{\text{solo}}$ contains unilateral demonstrations for Robot A, $\mathcal{D}_{B|A}^{\text{demo}}$ contains Robot-B demonstrations collected while Robot A executes its learned policy, and $\mathcal{D}_A^{\text{int}}$ contains corrective Robot-A interventions collected during joint deployment. Each robot acts from its own local history, including recent images, proprioception, and previous actions. The policies do not receive privileged partner state, explicit communication, or centralized coordination signals.

4.1 Stage 1: Bootstrapping Robot A

SAI begins with unilateral demonstrations for Robot A. In these demonstrations, Robot A performs the task while the other side of the shared object is supported by a compliant human or passive partner. This gives Robot A basic task competence, such as grasping, pulling, aligning, and exposing task-relevant regions, without requiring two robots or two synchronized operators.

Because the Stage-1 partner is human but the deployment partner is Robot B, a policy trained directly on these demonstrations may overfit to human-specific appearance and occlusion patterns. We therefore apply partner-masking augmentation to frames in which the supporting partner is visible: the partner region is replaced with randomized appearance perturbations, such as color jitter, blur, noise, or texture changes, with details in Appendix B. This encourages Robot A to focus on task-relevant object state rather than the visual identity of the supporting partner. Robot A is trained by behavioral cloning on the augmented demonstrations, producing the initial policy $\pi_A^{(1)}$. This policy can perform the task with external support, but it has not yet learned to coordinate with an autonomous robot partner.

4.2 Stage 2: Training Robot B Against Deployed Robot A

The second stage freezes $\pi_A^{(1)}$ and deploys it on Robot A. A single operator then teleoperates Robot B while interacting with the deployed Robot-A policy. This produces $\mathcal{D}_{B|A}^{\text{demo}}$, where Robot B observes the partner behavior it will face at test time.

During data collection, Robot A is executed with bounded deployment variation, such as speed scaling, action noise, timing offsets, or perturbed initial object states. These variations expose Robot B to realistic partner deviations and encourage waiting, yielding, recovery, and resumption behaviors. Robot B is then trained by behavioral cloning on the teleoperated actions, producing $\pi_B^{(2)}$. This stage is asymmetric: Robot B is not trained against an ideal partner or a manually synchronized human partner, but against the actual learned behavior of Robot A. As a result, Robot B learns to compensate for Robot A's timing, motion profile, and residual errors.

4.3 Stage 3: DAgger-Style Intervention Fine-Tuning

After Stage 2, Robot B has learned to coordinate with Robot A, but Robot A has still only been trained with a human-supported partner. The largest remaining distribution mismatch is therefore on Robot A: it must adapt from a compliant human partner to the autonomous behavior of Robot B. In practice, the remaining failures often occur at coordination boundaries, where Robot A pulls too early, fails to yield, continues while Robot B is delayed, or recovers poorly after phase mismatch.

To correct these failures, we deploy $\pi_A^{(1)}$ and $\pi_B^{(2)}$ together and collect sparse human interventions on Robot A. The intervention process is DAgger-style: the learned policies induce the closed-loop states, and the operator provides corrective labels only near states where coordination begins to fail. Each intervention records the current Robot-A history and the corrective action supplied by the operator. The resulting intervention dataset $\mathcal{D}_A^{\text{int}}$ is aggregated with the original Robot-A demonstrations to fine-tune Robot A:

$$\pi_A^{(3)} = \text{BC}\left(\mathcal{D}_A^{\text{solo}} \cup \mathcal{D}_A^{\text{int}}\right). \tag{3}$$

In practice, intervention samples are upweighted relative to nominal replay so that Robot A preserves its original task skill while adapting near coordination-critical states. When only part of the action requires correction, we optionally apply an action-dimension mask so that the loss supervises only the corrected components; for example, an operator may correct the base motion while leaving the arm command unchanged. If no mask is used, the full corrective action is used as the supervision target. This update is A-centric because Robot B has already been trained against deployed Robot-A behavior, whereas Robot A has not yet been trained against an autonomous Robot-B partner. Stage 3 recalibrates Robot A to the coupled deployment distribution, teaching it to slow down, wait, yield, or

recover when the joint system deviates from the nominal phase. Safety overrides may be applied to Robot B during data collection, but they are not used as training labels.

4.4 Policy Instantiation

SAI is a training curriculum and can be instantiated with different decentralized visuomotor policies. In our experiments, each robot uses an ACT-style action-chunking policy with temporal history encoding and a cascaded whole-body action head. The policy receives local RGB observations from the shared top-view camera and egocentric wrist cameras, together with the robot's proprioceptive state and previous actions. Visual features are extracted with a frozen DINOv3-80M backbone, projected into image tokens, and fused with state features.

To provide temporal context for coordination decisions, we maintain a history window of $H = 30$ steps and select 12 frames using logarithmic temporal sampling. Multi-view visual features and full-body state features are fused and encoded by a single-layer GRU, whose final hidden state forms the history token. The ACT decoder then predicts a whole-body action chunk with a cascaded head: a base-action MLP first predicts $a_{\text{base}} \in \mathbb{R}^3$, corresponding to (v_x, v_y, ω), and an arm-action MLP then predicts $a_{\text{arm}} \in \mathbb{R}^{14}$ conditioned on both the decoder query and the predicted base action. This architecture improves local execution reliability by encouraging base-arm consistency and preserving temporal context. However, the partner-responsive behavior comes from the SAI curriculum itself: each robot learns from progressively more realistic partner behavior, rather than from explicit communication, privileged partner state, or a centralized coordination loss.

5 Experimental Results

Our evaluation addresses four questions: **(Q1) Task performance:** Does SAI improve collaborative success across contact-rich real-world dual-robot manipulation tasks? **(Q2) Partner delays:** Does SAI induce yielding when the partner is unexpectedly delayed? **(Q3) Policy components:** Which lightweight architectural components improve execution reliability, and do they explain the cooperative gains? **(Q4) Backbone compatibility:** Does the SAI data curriculum remain effective with different imitation-learning backbones?

Across comparisons, we keep the observation space, action space, and policy architecture fixed whenever possible, so changes in collaborative behavior can be attributed primarily to the training curriculum. Unless otherwise stated, policies use the ACT-style whole-body architecture described in Appendix A: frozen DINOv3 visual features, a temporal history encoder over RGB, proprioception, and torque feedback, and a cascaded action head that predicts mobile-base commands before arm-gripper commands. Each robot acts from local observations only, with no explicit inter-robot communication, privileged partner state, or centralized coordination loss. We report three percentage metrics. *Success* measures task completion. *Phase Sync.* measures whether both robots complete predefined task checkpoints within the task-specific timing tolerance and without a disruptive interaction. These checkpoints are fixed before evaluation and are listed in Appendix C. *Yield/Wait* measures whether the robot slows, pauses, or resumes appropriately when its partner is delayed or out of phase.

5.1 Q1: Does SAI improve collaboration across real-world tasks? Yes!

We evaluate SAI on four real-world dual-robot manipulation tasks: **Bed-throw Spreading** and **Tablecloth Spreading**, which require deformable-object alignment; **Laundry Collection**, which requires asynchronous coordination in a shared workspace; and **Painting Transport**, which requires contact-rich rigid-object transport. As shown in Fig. 3, these tasks cover phase mismatch, partner timing variation, shared-workspace interference, and unmodeled tracking latency.

We compare three training pipelines under the same policy architecture. *Independent Imitation* trains each robot from separately collected unilateral demon-

Table 1: **Per-task evaluation.** SAI improves task success and process-level coupling metrics across deformable spreading, shared-workspace collection, and rigid-object transport.

Task	Method	Success ↑	Phase Sync. ↑	Yield/Wait ↑
Bed-throw	Independent Imit.	23.3%	30.8%	26.7%
	Partner-Cond. Imit.	36.7%	60.8%	35.0%
	SAI	**53.3%**	**62.5%**	**68.3%**
Tablecloth	Independent Imit.	43.3%	54.4%	46.7%
	Partner-Cond. Imit.	60.0%	67.8%	61.7%
	SAI	**66.7%**	**68.9%**	**70.0%**

Figure 3: **Real-world task suite.** Representative rollouts across deformable spreading, shared-workspace collection, and rigid-object transport.

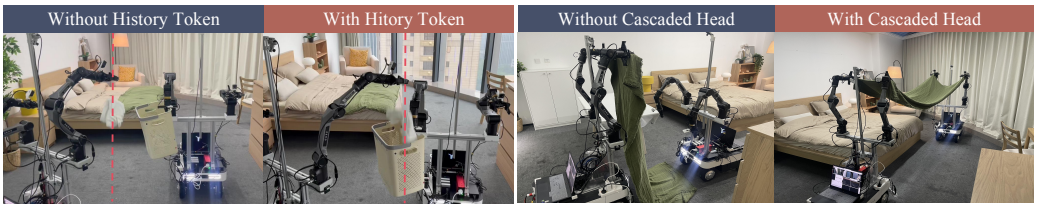
strations. *Partner-Conditioned Imitation* trains Robot B against the deployed Robot-A policy but omits the final intervention stage. *SAI* uses the full three-stage curriculum, including closed-loop Robot-A interventions.

Table 1 shows that SAI improves task success and process-level coordination across all four tasks. Independent Imitation often learns local manipulation primitives but fails during joint execution because the robots follow incompatible nominal timings. Partner-Conditioned Imitation improves phase alignment by exposing Robot B to the deployed Robot-A policy; for example, Phase Sync. in Bed-throw Spreading increases from 30.8% to 60.8%. However, Robot A remains trained only under the human-supported Stage-1 distribution, limiting its ability to adapt when Robot B is delayed or out of phase. Full SAI further improves Success and Yield/Wait behavior, indicating that sparse closed-loop interventions teach Robot A partner-responsive correction behaviors without explicit communication or reinforcement learning.

5.2 Q2: Does SAI induce yielding under partner delay? Yes!

To isolate partner-contingent yielding, we introduce a controlled delay during cooperative deployment in Tablecloth Spreading. Robot B is manually paused for a short interval, and we evaluate whether Robot A continues its nominal pulling motion or adapts to the delayed partner. This targets a common local coordination failure: if Robot A keeps pulling while Robot B remains stationary, the cloth becomes misaligned and the interaction can destabilize. Figure 4 shows representative rollout snapshots. The top row shows Independent Imitation, where Robot A continues its nominal motion despite Robot B being paused. This creates asymmetric pulling and leads to cloth misalignment. The bottom row shows SAI, where Robot A slows down and waits after observing that Robot B has not advanced, then resumes once the partner recovers. These snapshots illustrate the behavior captured by the aggregate Yield/Wait metric in Table 1: SAI increases partner-contingent slowing and waiting rather than simply replaying a nominal action sequence.

Figure 4: **Partner-contingent yielding under delay.** When Robot B is paused in Tablecloth Spreading, Independent Imitation continues pulling, increasing cloth tension and causing Robot B to lose its grasp. SAI instead slows, waits, and resumes after the partner recovers.



(a) Early release (b) Early lowering

Figure 5: **Policy ablations on execution failures.** The history token reduces premature release, while the cascaded base-arm head reduces premature lowering by improving whole-body action consistency.

5.3 Q3: Do history and cascading components in the policy improve reliability? Yes!

SAI primarily changes the training distribution, but reliable real-robot deployment also depends on the policy's ability to execute whole-body actions consistently. We therefore ablate two components of the policy instantiation: the temporal history token and the cascaded base-arm action head. These ablations use the same sensorimotor interface and training pipeline, changing only the tested architectural component.

Table 2 and Fig. 5 show that temporal history reduces premature release in Laundry Collection from 64.5% to 16.1%, because the policy can distinguish intermediate transport states from terminal release states. The cascaded base-arm head reduces premature lowering from 51.6% to 9.7%, because arm commands are conditioned on the predicted base motion rather than produced by an independent flat head. These components improve local execution reliability, but they do not explain the cooperative gains by themselves: the baseline pipelines in Table 1 use the same architecture and still fail more often under partner delay and phase mismatch. Thus, the architecture stabilizes primitive execution, while SAI induces partner-responsive coordination through the data curriculum.

Table 2: Failure-specific policy ablations.

Failure mode	Variant	Rate ↓
Premature release	w/o Hist.	64.5%
	w/ Hist.	**16.1%**
Premature lowering	w/o Casc.	51.6%
	w/ Casc.	**9.7%**

5.4 Q4: Is SAI agnostic to specific imitation-learning backbones? Yes!

SAI operates at the data-curriculum level and is not tied to a particular action decoder. To test backbone compatibility, we instantiate the same curriculum with two imitation-learning backbones, Action Chunking Transformer (ACT) and Diffusion Policy, while keeping the sensorimotor interface and task protocol fixed. We evaluate this comparison on Painting Transport, a contact-rich task that requires synchronized grasping, lifting, transport, and placement. As shown in Table 3, SAI improves over Independent Imitation for both backbones. With ACT, SAI improves task success from 33.3% to 56.7%, Phase Sync. from 42.7% to 74.7%, and Yield/Wait

Table 3: **Backbone compatibility on Painting Transport.** SAI improves over Independent Imitation with both ACT and Diffusion Policy.

Action Head	Training Pipeline	Success ↑	Phase Sync. ↑	Yield/Wait ↑
ACT	Independent Imit.	33.3%	42.7%	34.4%
	SAI	**56.7%**	**74.7%**	**68.9%**
Diffusion	Independent Imit.	23.5%	35.2%	30.7%
	SAI	**52.9%**	**62.2%**	**76.9%**

from 34.4% to 68.9%. With Diffusion Policy, SAI also improves all three metrics, including a large gain in Yield/Wait from 30.7% to 76.9%. These results support the interpretation that SAI is a data-curriculum contribution rather than an artifact of a particular action decoder.

6 Conclusions and Limitations

We introduced *Sequential Asymmetric Imitation* (SAI), a single-teleoperator curriculum for learning decentralized dual-robot manipulation without synchronized dual-operator demonstrations. SAI trains Robot A from solo demonstrations, trains Robot B against the deployed Robot-A policy, and uses sparse interventions to correct coordination failures. Across deformable spreading, shared-workspace collection, and rigid-object transport, SAI improves task success, phase synchronization, and partner-contingent yielding over independent imitation. These results suggest that coordinated multi-robot behavior can emerge from structured imitation data, without reinforcement learning, explicit inter-robot communication, or dense synchronized demonstrations.

Limitations: SAI assumes that each robot can learn useful standalone primitives, that partner progress is partially observable from local sensory inputs, and that failures can be corrected through meaningful single-agent interventions. Tasks requiring simultaneous correction of both robots may not fit the current A-centric design. The main limitation is failure-mode coverage: our Stage-3 intervention set is only about 30% of the baseline dataset size. Although this improves *Yield/Wait*, recovery has not saturated. More targeted interventions around rare delays and near-failure contact states may improve robustness, while excessive correction may bias the policy toward unnecessary waiting.

References

[1] H. Kim, Y. Ohmura, and Y. Kuniyoshi, "Goal-conditioned dual-action imitation learning for dexterous dual-arm robot manipulation," *IEEE Transactions on Robotics*, vol. 40, pp. 2287–2305, 2024.

[2] Z. Fu, T. Z. Zhao, and C. Finn, "Mobile ALOHA: Learning bimanual mobile manipulation using low-cost whole-body teleoperation," in *8th Annual Conference on Robot Learning (CoRL)*, 2024.

[3] 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*, vol. 44, no. 10-11, pp. 1684–1704, 2025.

[4] E. Hannus, T. N. Le, D. Blanco-Mulero, and V. Kyrki, "Dynamic Manipulation of Deformable Objects using Imitation Learning with Adaptation to Hardware Constraints," in *Proceedings of the 2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 12655–12662, 2024.

[5] S. Ross, G. Gordon, and D. Bagnell, "A reduction of imitation learning and structured prediction to no-regret online learning," in *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics (AISTATS)*, pp. 627–635, 2011.

[6] J. Spencer, S. Choudhury, M. Barnes, M. Schmittle, M. Chiang, P. Ramadge, and S. Srinivasa, "Learning from Interventions: Human-robot interaction as both explicit and implicit feedback," in *Proceedings of Robotics: Science and Systems (RSS) XVI*, 2020.

[7] T. Z. Zhao, V. Kumar, S. Levine, and C. Finn, "Learning Fine-Grained Bimanual Manipulation with Low-Cost Hardware," in *Proceedings of Robotics: Science and Systems (RSS) XIX*, 2023.

[8] Z. Li, Y. Zhou, R. Qiu, H. Wu, G. Ren, and W. Zhi, "TriPilot-FF: Coordinated Whole-Body Teleoperation with Force Feedback," *arXiv preprint arXiv:2602.09888*, 2026.

[9] J. Gao, X. Jin, F. Krebs, N. Jaquier, and T. Asfour, "Bi-KVIL: Keypoints-based Visual Imitation Learning of Bimanual Manipulation Tasks," in *Proceedings of the 2024 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 16850–16857, 2024.

[10] T. Lin, Z.-H. Yin, H. Qi, P. Abbeel, and J. Malik, "Twisting Lids Off with Two Hands," in *Proceedings of the 8th Annual Conference on Robot Learning (CoRL)*, 2024.

[11] B. Aksoy and J. T. Wen, "Collaborative manipulation of deformable objects with predictive obstacle avoidance," in *Proceedings of the 2024 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 16766–16772, 2024.

[12] S. Liu, L. Wu, B. Li, H. Tan, H. Chen, Z. Wang, K. Xu, H. Su, and J. Zhu, "RDT-1B: a Diffusion Foundation Model for Bimanual Manipulation," in *Proceedings of the Thirteenth International Conference on Learning Representations (ICLR)*, 2025.

[13] C. Chi, Z. Xu, C. Pan, E. Cousineau, B. Burchfiel, S. Feng, R. Tedrake, and S. Song, "Universal Manipulation Interface: In-The-Wild Robot Teaching Without In-The-Wild Robots," in *Proceedings of Robotics: Science and Systems (RSS) XX*, 2024.

[14] W. Zhi, T. Zhang, and M. Johnson-Roberson, "Instructing Robots by Sketching: Learning from Demonstration via Probabilistic Diagrammatic Teaching," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 15047–15053, 2024.

[15] R. Hoque, A. Mandlekar, C. Garrett, K. Goldberg, and D. Fox, "IntervenGen: Interventional data generation for robust and data-efficient robot imitation learning," in *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 2840–2846, 2024.

[16] R. Lowe, Y. I. Wu, A. Tamar, J. Harb, P. Abbeel, and I. Mordatch, "Multi-agent actor-critic for mixed cooperative-competitive environments," in *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 30, 2017.

[17] H. Ravichandar, A. S. Polydoros, S. Chernova, and A. Billard, "Recent advances in robot learning from demonstration," *Annual Review of Control, Robotics, and Autonomous Systems*, vol. 3, no. 1, pp. 297–330, 2020.

[18] W. Zhi, T. Lai, L. Ott, and F. Ramos, "Diffeomorphic Transforms for Generalised Imitation Learning," in *Proceedings of The 4th Annual Learning for Dynamics and Control Conference (L4DC)*, pp. 508–519, 2022.

[19] K. Shaw, Y. Li, J. Yang, M. K. Srirama, R. Liu, H. Xiong, R. Mendonca, and D. Pathak, "Bimanual Dexterity for Complex Tasks," in *Proceedings of the 8th Annual Conference on Robot Learning (CoRL)*, 2024.

[20] J.-J. Jiang, X.-M. Wu, Y.-X. He, L.-A. Zeng, Y.-L. Wei, D. Zhang, and W.-S. Zheng, "Rethinking Bimanual Robotic Manipulation: Learning with decoupled interaction framework," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 12427–12437, 2025.

[21] W. Liu, M. Ren, K. Song, M. Y. Wang, and Z. Xiong, "A Planning Framework for Complex Flipping Manipulation of Multiple Mobile Manipulators," *IEEE Robotics and Automation Letters*, vol. 10, no. 5, pp. 5162–5169, 2025.

[22] T. Lai, W. Zhi, T. Hermans, and F. Ramos, "Parallelised diffeomorphic sampling-based motion planning," in *Conference on Robot Learning (CoRL)*, pp. 81–90, 2022.

[23] F. De Vincenti and S. Coros, "Centralized Model Predictive Control for Collaborative Loco-Manipulation," in *Proceedings of Robotics: Science and Systems (RSS) XIX*, 2023.

[24] J. Gao, A. Xie, T. Xiao, C. Finn, and D. Sadigh, "Efficient Data Collection for Robotic Manipulation via Compositional Generalization," in *Proceedings of Robotics: Science and Systems (RSS)*, 2024.

[25] V. Saxena, M. Bronars, N. R. Arachchige, K. Wang, W. C. Shin, S. Nasiriany, A. Mandlekar, and D. Xu, "What Matters in Learning from Large-Scale Datasets for Robot Manipulation," in *Proceedings of the Thirteenth International Conference on Learning Representations (ICLR)*, 2025.

[26] T. Osa, J. Pajarinen, G. Neumann, J. A. Bagnell, P. Abbeel, and J. Peters, "An algorithmic perspective on imitation learning," *Foundations and Trends in Robotics*, vol. 7, no. 1-2, pp. 1–179, 2018.

[27] H. Ravichandar, I. Salehi, and A. P. Dani, "Learning Partially Contracting Dynamical Systems from Demonstrations," in *Proceedings of the Conference on Robot Learning (CoRL)*, pp. 369–378, 2017.

[28] P. Thota, H. Ravichandar, and A. P. Dani, "Learning and Synchronization of Movement Primitives for Bimanual Manipulation Tasks," in *Proceedings of the IEEE 55th Conference on Decision and Control (CDC)*, pp. 945–950, 2016.

[29] O. Siméoni, H. V. Vo, M. Seitzer, F. Baldassarre, M. Oquab, C. Jose, V. Khalidov, M. Szafraniec, S. Yi, M. Ramamonjisoa, F. Massa, D. Haziza, L. Wehrstedt, J. Wang, T. Darcet, T. Moutakanni, L. Sentana, C. Roberts, A. Vedaldi, J. Tolan, J. Brandt, C. Couprie, J. Mairal, H. Jégou, P. Labatut, and P. Bojanowski, "DINOv3," *arXiv preprint arXiv:2508.10104*, 2025.

[30] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer, "High-Resolution Image Synthesis with Latent Diffusion Models," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

A Implementation Details and Reproducibility

This section summarizes the policy architecture, observation and action spaces, and training hyperparameters used in our real-robot experiments. Unless otherwise stated, all policies use an ACT-style action chunking backbone with temporal history encoding and a cascaded whole-body action head.

A.1 Network Architecture Details

Visual Encoder. We use a frozen DINOv3 (ViT-Base) backbone [29] to extract spatial features from RGB observations, including the shared top camera and egocentric wrist cameras. The feature maps are projected with a 1×1 convolution and flattened into image tokens before entering the transformer encoder. Two-dimensional sinusoidal positional embeddings are added to preserve spatial information.

History encoder. To provide temporal context for long-horizon decisions, we maintain a history window of $H = 30$ steps. For efficiency, we select 12 historical frames with logarithmic temporal sampling, using denser samples near the current timestep and sparser samples further in the past. For each sampled timestep, multi-view visual features are pooled and projected, while the full-body state is encoded by an MLP into a 64-dimensional latent. The visual and state latents are concatenated, passed through a fusion MLP, and encoded by a single-layer GRU with hidden dimension 512. Padding masks are used when historical frames are unavailable. The final GRU hidden state is used as the history token z^t_{hist}.

Cascaded whole-body action head. The ACT decoder outputs action-query embeddings. Instead of predicting all action dimensions with a single flat head, we use a cascaded structure. A 2-layer MLP first predicts the mobile-base action $a_{\text{base}} \in \mathbb{R}^3$, corresponding to (v_x, v_y, ω). The predicted base intent is then concatenated with the decoder query and passed to a second 2-layer MLP to predict the arm-gripper action $a_{\text{arm}} \in \mathbb{R}^{14}$. This factorization conditions arm and gripper commands on the imminent base motion and improves whole-body action consistency.

A.2 State and Action Space Definitions

Observation space. At each timestep, the observation consists of RGB images, robot proprioception, and torque feedback. RGB images are captured from the shared top-view camera and egocentric wrist cameras. The proprioceptive state is 17-dimensional, containing the 3D base state and 14D arm-gripper state. Torque feedback consists of measured arm-side torque signals, which provide implicit physical interaction cues during contact-rich manipulation.

Action space.

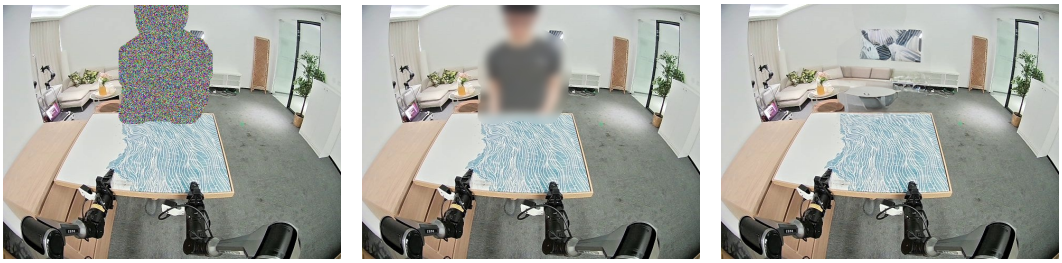
The policy predicts an action chunk of horizon K. Each action step has dimension $d_a = 17$, consisting of a 3D base command (v_x, v_y, ω) and a 14D arm-gripper command. During execution, overlapping chunks are combined with exponential temporal ensembling using coefficient $\alpha = 0.01$.

Table 4: Hyperparameters for policy instantiation.

Hyperparameter	Value
Architecture	
Visual backbone	Frozen DINOv3
History window	30 steps
Log. sampling steps	12 steps
State encoder dim.	64
GRU hidden dim.	512
ACT model dim.	512
ACT enc./dec. layers	4 / 1
Attention heads	8
Obs. and Action	
Proprio. state dim.	17 $(3 + 14)$
Physical feedback	Arm torque
Action horizon	100
Action dim.	17 $(3 + 14)$
Training	
Temp. ensemble coeff.	0.01
Optimizer	AdamW
Learning rate	1×10^{-5}
Weight decay	1×10^{-4}
Visual backbone	Frozen

A.3 Training Details and Hyperparameters

All policies are trained with AdamW. During Stage 3, interventions may correct only part of the action space, such as stopping the base while maintaining arm posture. We therefore use binary action-loss masks $M^t \in 0, 1^{K \times d_a}$ for intervention samples, so gradients are applied only to explicitly corrected action dimensions. This prevents sparse corrective



(a) RGB randomization (b) Gaussian blur (c) Stable Diffusion inpainting

Figure 6: **Partner-region randomization.** Training-time replacements applied inside the detected human-partner region: RGB randomization, Gaussian blur, and Stable Diffusion Inpainting.

labels from overwriting nominal behavior in unaffected dimensions.

Key hyperparameters are summarized in Table 4.

B Partner Masking and Randomization

Offline partner-region detection. During Phase 1, Robot A interacts with a compliant human partner. Since the deployment partner is Robot B, directly training on raw Phase-1 images may cause the policy to exploit human-specific appearance cues, such as clothing, skin color, body texture, or human body shape. We therefore apply an offline partner-region masking pipeline to reduce this morphology-induced visual shortcut.

For each raw observation $o_A^t \in \mathbb{R}^{H\times W\times 3}$, we use an off-the-shelf instance segmentation model to obtain a binary human-region mask $\hat{m}^t \in \{0,1\}^{H\times W}$. To cover boundary artifacts, thin uncovered regions, and nearby shadows, we dilate the mask with a circular structuring element:

$$m^t = \hat{m}^t \oplus \mathcal{K}_\rho, \tag{4}$$

where $\mathcal{K}_\rho$ is a circular dilation kernel and ρ is the dilation radius. This preprocessing is applied only during offline data loading and does not introduce additional policy parameters.

Partner-region randomization. Given the dilated mask m^t, we construct an augmented observation:

$$\bar{o}_A^t = (1 - m^t) \odot o_A^t + m^t \odot r(o_A^t), \tag{5}$$

where $r \sim \mathcal{R}$ is a stochastic replacement operator applied only inside the masked partner region. The goal is not to remove all interaction evidence, but to make human-specific appearance cues unreliable during policy learning. This encourages the policy to rely less on the visual identity of the human partner and more on task-relevant scene cues, such as object geometry, deformation, contact progress, and workspace configuration. During offline training-time data loading, $\mathcal{R}$ randomly samples one of the following partner-region transformations:

- **RGB Randomization:** pixels inside the mask are replaced with independently sampled RGB values, disrupting local color and texture statistics.
- **Gaussian Blur:** a strong local blur is applied inside the mask to remove high-frequency appearance details while preserving only a coarse foreground region.
- **Stable Diffusion Inpainting:** the masked region is repaired using a mask-guided latent diffusion inpainting model [30]. Unlike classical interpolation-based inpainting, this operator generates locally plausible content conditioned on the surrounding image context.

By varying the appearance of the partner region across these transformations, the policy is regularized against partner-appearance shortcuts. All transformations are applied offline during data loading and are not used during policy inference. Figure 6 visualizes the three partner-region transformations. These examples show how the detected human-partner region is altered during training while preserving the surrounding workspace context.

Table 7: Task-level event definitions for Phase Sync. evaluation.

Task	Phase Sync. checkpoints
Bed-throw Spreading	(1) Robot A hands the bed throw to Robot B, and Robot B establishes a stable grasp; (2) both robots retreat from the handoff area while keeping the object controlled; (3) the robots jointly move toward the bed region with matched timing and without excessive dragging; (4) both robots place the bed throw at the target area in a coordinated manner.
Tablecloth Spreading	(1) both robots establish stable grasps before spreading; (2) the robots spread the tablecloth with compatible timing so that the cloth remains approximately aligned and tensioned; (3) both robots complete final placement without premature release or large disturbance to the cloth.
Laundry Collection	(1) the robots enter the shared workspace and approach the laundry item without blocking each other; (2) the collection and delivery motion is completed while the robots maintain a compatible spatial order in the shared area.
Painting Transport	(1) both robots establish stable grasps on the painting; (2) they lift the painting together without excessive tilt; (3) they enter the transport path while maintaining compatible relative positions; (4) they transport the painting with stable coordination along the path; (5) they lower and place the painting at the target location without premature release.

C Evaluation Protocol and Rollout Statistics

General protocol. All policies are evaluated on real-robot rollouts. The percentages reported in the main tables are computed from integer counts and rounded to one decimal place. *Success* is a rollout-level metric. *Phase Sync.* and *Yield/Wait* are event-level descriptive metrics computed over predefined checkpoints or partner-contingent opportunities within the same rollout set. Therefore, their denominators can be larger than the number of rollouts, but the events should not be interpreted as independent rollout trials. Table 5 summarizes the rollout count used for each main experiment.

Table 5: Rollout statistics for the main experiments.

Experiment	Evaluation scope	Conditions	Rollouts / condition
Q1	Per-task comparison	4 tasks $\times$ 3 methods	30
Q3	Policy ablations	Failure-specific variants	31
Q4	Backbone compatibility	Diffusion on Painting	34

Q1: per-task evaluation. For the main per-task comparison, each task-method pair is evaluated over 30 rollouts. Success is computed at the rollout level with denominator 30. Phase Sync. is computed over task-specific synchronization checkpoints, and Yield/Wait is computed over task-specific partner-contingent opportunities. The number of checkpoints and opportunities is fixed per task and kept identical across methods. Table 6 lists the corresponding event-level denominators for each task.

Table 6: Event-level denominators for Q1.

Task	Rollouts	Phase ckpts.	Phase denom.	Yield denom.
Bed-throw	30	4	120	60
Tablecloth	30	3	90	60
Laundry	30	2	60	60
Painting	30	5	150	90

Phase Sync. For each task, we divide the cooperative process into ordered event stages. These stages define the checkpoints used to compute *Phase Sync.*. The Phase Sync. checkpoints for each task are listed in Table 7. For a task with K checkpoints per rollout and N evaluated rollouts, the denominator is $N \times K$. The numerator is the number of checkpoints for which both robots reach and complete the corresponding stage within the predefined timing tolerance and without causing a task-disrupting interaction. Thus, *Phase Sync.* is computed as the percentage of synchronized checkpoints over all evaluated checkpoints. If a rollout fails before reaching a later stage, all unreached checkpoints are counted as failed. Stage definitions and timing thresholds are fixed per task and kept identical across all evaluated methods. For example, Painting Transport has five checkpoints per rollout; with 30 rollouts, the Phase Sync. denominator is $30 \times 5 = 150$.

Yield/Wait. Yield/Wait measures whether a robot responds appropriately to a partner-contingent opportunity. A response is counted as correct when the robot slows down, waits, yields, adjusts pose, or resumes only after the partner becomes ready.

Q3: policy-component ablations. The policy ablation study evaluates failure-specific rates over 31 targeted rollout segments. These segments are initialized near the task phases where the corresponding

failures are most likely to occur, rather than from the beginning of the full task. Premature release is evaluated in the high-risk delivery/release phase of Laundry Collection and measures whether the robot releases the garment before reaching the valid receiving or drop-off state. Premature lowering is evaluated in the high-risk transition phase of spreading or transport tasks and measures whether the arm enters a lowering or release phase before the mobile base has completed the corresponding transport phase. The history-token ablation is evaluated on premature release, while the cascaded-head ablation is evaluated on premature lowering.

Q4: backbone compatibility. For backbone compatibility, we evaluate the same SAI data curriculum with different imitation-learning backbones while keeping the task protocol, observation space, and action space fixed. The comparison is conducted on Painting Transport. The ACT results use the same 30-rollout Painting Transport protocol as Q1. The Diffusion Policy comparison is evaluated over 34 rollouts per condition. Success is computed at the rollout level, while Phase Sync. and Yield/Wait use the same Painting Transport event definitions described above.