Safety Starts in the Dataset: Auditing Temporal Action-Label Misalignment Before Training

Socrates Osorio*
Electrical Engineering and Computer Sciences
University of California, Berkeley
Berkeley, CA, USA
Email: socratesj.osorio@berkeley.edu

Joy Zheyun Yang*
University of Oxford
Oxford, United Kingdom
Email: joy@robots.ox.ac.uk

***Abstract*—Most robot safety machinery acts on the policy: collision checks, shields, runtime monitors, red-teaming. But a learned policy can be unsafe because the data it was trained on was silently corrupt, before any architecture or runtime monitor exists to catch it. We study one such defect that is invisible to every standard data dashboard: *temporal action-label misalignment*, where logging, controller queues, dataset preprocessing, or asynchronous teleoperation shift an action stream by a few control steps. Each observation is still realistic and each action is still realistic. The bug is which timestep an action supervises. Across a 21-case RoboMimic shift grid, states and observations are unchanged after trimming and task-mean action marginals move by at most 0.118 standardized 1D Wasserstein distance, yet a 0.2 s ($+4$-frame) shift collapses behavior-cloning success from $591/800$ to $51/800$ on Lift and from $634/800$ to $209/800$ on Can. A policy fit on these labels has been actively optimized to act for the wrong instant. We argue that auditing the temporal integrity of a logged corpus is a *safety-by-design* primitive: it reduces risk at the source rather than patching it at deployment. We present ActSync, a CPU-scale audit-and-repair primitive whose default is to *abstain when uncertain*. It searches integer shifts, scores each by held-out state-transition consistency, repairs labels only on high-confidence homogeneous global estimates, and uses only logged states and actions, no rewards, rollouts, or success labels. ActSync recovers the exact shift in $21/21$ cases, restores 93.1% (Lift) and 81.2% (Can) of the lost success in every paired seed (sign-test $p = 0.0078$), and abstains on all $180/180$ marginal-preserving negative controls at its conservative threshold. The abstention gate is itself a practice-time monitor applied at the data layer: when the corpus cannot be safely repaired, the audit refuses to write and raises a warning rather than shipping a silently biased prior.**

I. Safety Has a Data-Layer Origin

The dominant picture of robot safety acts on the *policy*: constraint enforcement (joint limits, collision avoidance), runtime monitoring and out-of-distribution detection, shielding and safe fallback, and red-teaming of the deployed system. These are necessary. But they all presuppose that the artifact being protected was trained on data that means what it says. A policy can be unsafe not because its monitor failed or its architecture is brittle, but because the supervision it learned from was attached to the wrong moment in time, and nothing in the training pipeline ever checked.

This paper isolates one such data-layer defect that the community is not currently auditing for: **temporal action-label misalignment**. Behavior cloning, and offline learning generally, assume each logged action is paired with the observation that caused it. In practice that assumption is fragile: camera and proprioception logging, controller queues, dataset preprocessing, and teleoperation replay can each shift an action stream by a few control steps [1]. The result is insidious precisely because it is plausible. Each observation is still realistic, and each action is still realistic. The defect is which state transition an action supervises.

a) Why standard data auditing misses it: The data-quality tools used in robotics today, demonstration count, success rate, action-distribution histograms, and mutual-information curation [2, 6], score states and actions *separately*. A temporal shift moves none of those statistics enough to trigger a flag. Across our 21-case Lift/Can/Square shift grid, states and observations are unchanged after trimming, and task-mean action marginals move by at most 0.118 clean-standardized 1D Wasserstein distance, a magnitude no dataset dashboard will surface. Yet the policy fit on those labels has been actively optimized to act for the wrong timestep. We could find no published preflight audit of the temporal cleanliness of any large logged robot corpus (RoboMimic, BridgeData V2, Open X-Embodiment, DROID [15, 20, 16, 11]). By default, the defect is untested.

b) Position: audit the data before you train: Reducing risk at the source means cleaning the corpus before any policy, architecture, or runtime monitor is built on top of it. We make this concrete with a controlled measurement of how much a small shift biases the learned prior, a CPU-scale audit primitive that defaults to abstention, and a deployment rule that treats the audit's refusal-to-write as a first-class outcome. The design-time intervention and a practice-time monitor are, in this case, the same mechanism: the audit's confidence gate is a monitor applied at the data layer.

c) Scope honesty: This is a position-plus-bounded-evidence paper. We inject synthetic integer shifts into public low-dimensional RoboMimic demonstrations and measure behavior cloning. We do not run image policies, foundation-model policies, or real-robot logs. The scope is deliberate: low-dimensional, single-task, well-instrumented BC is the *most favorable* measurement setting, which makes it a clean lower bound on the magnitude of the bias. Harder settings are unlikely to be *more* robust to the same defect, and §V

names the experiments that would test transfer. Two further bounds deserve emphasis. First, our clean BC references are themselves modest (73.9%/79.2%), so the measured collapse quantifies the risk in this regime rather than isolating the shift's effect on a saturated policy. Second, real latency is rarely a constant offset: under jittered asynchronous delays our probes stay within one frame in $108/108$ cases but are exact in only $42/108$ (Appendix), so for drifting latency ActSync's value narrows from repair toward detect-and-abstain, which the gate is designed to report honestly.

II. A Controlled Measurement of the Risk

We use RoboMimic proficient-human low-dimensional datasets for Lift, Can, and Square [15], with applied-action control at 20 Hz [18], so a 4-frame shift is 0.2 s, a plausible logging or controller-latency scale. For a label shift k, the corrupted dataset replaces a_t by a_{t+k} and trims invalid endpoints. The corresponding repair shift is $r = -k$.

We train BC under four matched conditions, clean-trim, corrupted, ActSync-repaired, and oracle-repaired, all trimmed to matched valid timesteps so comparisons cannot benefit from different sample counts. The primary stress is $+4$ on Lift and Can, eight matched training seeds per task, 100 rollouts per condition and seed (800 per cell), horizon 400. All headline rollouts use the same local evaluation stack.

a) The defect is severe under the easiest conditions: Table I reports the result. A 0.2 s shift collapses BC success from 73.9% to 6.4% on Lift and from 79.2% to 26.1% on Can. This is the magnitude of risk a downstream learner inherits if the audit step is skipped, in a setting designed to make the BC stage easy. A wrong-direction repair is worse than no repair (§IV), so the audit cannot be cavalier about *how* it writes.

TABLE I
Risk from a 0.2 s ($+4$-frame) action-label shift, and ActSync recovery. Eight matched BC seeds per task, 100 rollouts per condition and seed. **Rec.** is the fraction of the clean-vs-corrupt gap ActSync recovers.

Task	Clean	Corrupt	ActSync	Oracle	Rec.
Lift	591/800	51/800	554/800	565/800	0.931
Can	634/800	209/800	554/800	552/800	0.812

b) Not a starvation effect: Doubling corrupt-policy training from 20 to 40 epochs for the first three Lift seeds reached $0/150$ success, versus $19/150$ for baseline corrupt checkpoints. More compute on the wrong labels does not climb out. The failure is in the data, not the budget.

III. ActSync: A Safety-by-Design Primitive

ActSync is a preflight audit-and-repair primitive. The contribution is the *pipeline*, search, held-out scoring, confidence gating, deterministic repair, abstention, and downstream behavioral validation, not a claim that any one detector is uniquely powerful. Simple alignment scores are also strong at detection, and we say so explicitly.

a) Method: Let a trajectory contain observations o_t, low-dimensional state features $s_t = f(o_t)$, and logged actions a_t. ActSync considers candidate repair shifts $r \in [-K, K]$. For each r it pairs a_{t+r} with $\Delta s_t = s_{t+1} - s_t$ whenever both indices are valid. The default estimator fits a ridge model [7] from shifted actions to state deltas on a demonstration-level train split and scores held-out Huber error [8]:

$$\text{score}(r) = \tfrac{1}{|\mathcal{V}_r|} \sum_{(a,\Delta s)\in\mathcal{V}_r} \rho_\delta(\Delta s - g_r(a)).$$

$\hat{r} = \arg\min_r \text{score}(r)$ with ties broken toward smaller $|r|$, and confidence is the relative gap between the best and second-best finite scores. A moving-state filter keeps higher-motion timesteps. All runs use a $70/30$ demo-level split, standardized inputs, ridge $\alpha = 0.01$, Huber $\delta = 1.0$, and a ≥ 60th-percentile motion filter. Estimator choices are tuned only on logged-data audits, never on rollouts or success labels. A leave-one-task-out check is exact for all three held-out choices over the loss, motion-filter, and ridge-candidate grids. For an accepted estimate, repair is deterministic: keep timesteps where $t + \hat{r}$ is valid and replace each label with $a_{t+\hat{r}}$. The same trimming applies to clean and oracle references.

b) Why it is cheap enough to be a default: For low-dimensional manipulation, the short-horizon action predicts the next change in end-effector state, and a wrong offset weakens that relation. ActSync exploits this directly, so it runs in seconds on a CPU, orders of magnitude cheaper than a single training epoch. A safety check that is essentially free has no excuse not to be run before every dataset release.

c) On simple baselines: Correlation is exact on $20/21$ task-offset cases and within one frame on all 21, and an in-sample linear lag score is exact on $21/21$ (Table II). What no bare detector provides is the rest of the pipeline: held-out scoring, the confidence gate, the abstention rule, the session-heterogeneity check, the wrong-repair and clean false-positive controls, and the matched-seed validation. The safety-by-design framing is detector-agnostic. Any detector can be slotted into the same gated workflow.

A. Detection is reliable and cheap

Across 21 Lift/Can/Square task-offset cases, eef-only ridge with moving filtering recovers the exact repair shift in $21/21$ cases. The three unshifted full-dataset cases are zero in $3/3$, so the audit does not hallucinate a repair on clean logs. A uniform-random baseline is exact only 0.077 of the time. Runtime is 1.031 s per full-dataset case and 0.051 s per 10-demo case.

B. Repair restores most of the risk, in both directions

The pooled result of §II is consistent across matched seeds. In every $+4$ seed pair ActSync beats the corrupted policy, with a two-sided sign-test $p = 0.0078$ on each task: Lift is $8/8$ with mean delta $+0.629$ (95% seed CI $[+0.522, +0.736]$, min $+0.350$) and Can is $8/8$ with mean delta $+0.431$ (95% CI $[+0.236, +0.626]$, min $+0.030$). A -4 directionality gate (Table III) shows the same pattern in $8/8$ matched seeds for both tasks ($p = 0.0078$ each). The defect, and the repair, are not sign-specific.

TABLE II
OFFSET-ESTIMATION AND AUDIT ABLATIONS OVER THE 21 TASK-OFFSET CASES.

Setting	Cases	Exact	$\leq$1 frame
Correlation	21	0.952	1.000
In-sample linear lag score	21	1.000	1.000
Per-demo correlation vote	21	0.857	1.000
Ridge eef+gripper	21	0.762	1.000
Ridge eef-only	21	1.000	1.000
Random offset	21	0.077	0.231
10 demos, eef-only	21	0.714	1.000
50 demos, eef-only	21	0.952	1.000

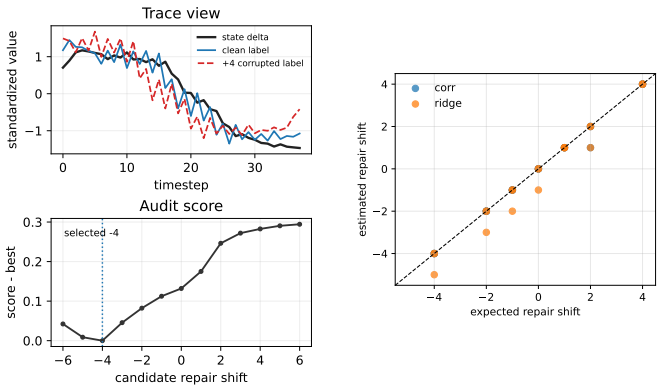

Fig. 1. Left: a Lift $+4$ trace, where the corrupted labels remain plausible but are paired with the wrong state-change instant. Right: estimated vs. expected repair shifts across task-offset cases.

IV. SAFETY BY ABSTENTION

A design-time data fix is only safe if it refuses to act when it cannot diagnose the corpus. A silently mis-repaired dataset is the worst outcome: an undetected biased prior that survives every downstream stage and ships. We built ActSync so that, in doubt, it does not write. This is the bridge to safety-by-practice: the same confidence gate that decides whether to repair is a runtime monitor over the dataset.

a) Wrong-direction repair is worse than no repair: Against oracle alignment on Lift/Can/Square $+4$ datasets, the expected repair ranks first in $3/3$ checks. No repair leaves mean MSE 0.0695, while opposite-sign repair leaves 0.1521. A Lift $+4$ wrong-sign repair policy reaches $0/50$ success, below the corrupted policy at $4/50$. The audit must be able to say "do not write," and the confidence gate is that mechanism.

b) Negative-control abstention: We built two marginal-preserving negative controls on 100-demo clean subsamples: within-trajectory circular action shifts by ≥ 8 frames, and time-reversed actions. These preserve each trajectory's action set but break the short-horizon timing relation. A permissive 0.05 threshold would keep $53/180$ such cases, but the conservative $\tau \geq 0.10$ threshold **abstains on** $180/180$. The audit does not manufacture plausible repairs from data corrupted in a way it cannot diagnose.

c) Clean and heterogeneous-corpus abstention: On full clean Lift/Can/Square datasets the selected repair is zero in $3/3$.

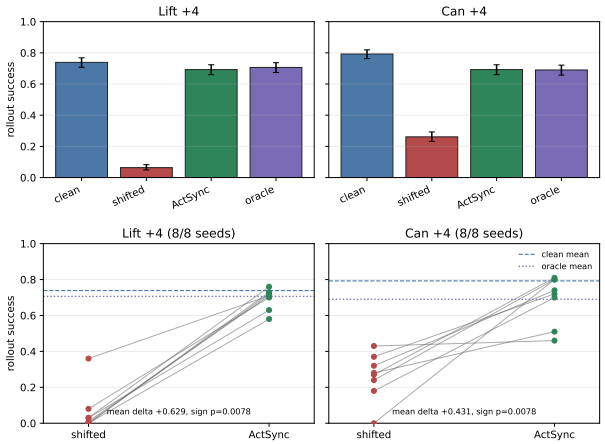

Fig. 2. Top: pooled BC recovery. Bottom: matched-seed ActSync-vs-corrupt deltas. ActSync is positive in every seed pair on both tasks.

TABLE III
EIGHT-SEED -4 DIRECTIONALITY GATES, 100 ROLLOUTS PER CONDITION/SEED.

Task	Clean	Corrupt	ActSync	Oracle
Lift	439/800	51/800	554/800	542/800
Can	589/800	4/800	571/800	556/800

On a 1200-case 100-demo clean stress test the exact-zero rate is 81.2% with no repair larger than one frame, and at $\tau \geq 0.10$ it keeps $332/1200$ subsamples, all exact zero. On a 1803-case $10/25/50$-demo test, $\tau \geq 0.05$ keeps $1099/1800$ and removes all repairs larger than one frame. A 54-case pseudo-session audit mixing shifted and clean sessions, the realistic case where shards from different operators, robots, or logging revisions are concatenated, is globally exact in $18/18$ homogeneous cases, and all 36 mixed cases produce conflicting session-level estimates. The rule is to abstain on session disagreement.

d) The deployment rule: Repair only full-dataset or sufficiently large, high-confidence, exact, homogeneous global estimates. Inspect manually for ≥ 100-demo nonzero repairs at $\tau \geq 0.10$. Warn but do not rewrite low-data one-frame estimates. Abstain on low confidence, intermediate global estimates, session disagreement, image-only logs without low-dim state, or weak action-state coupling. A failed audit is still a useful safety outcome: it stops an expensive training run before its results become un-interpretable, and it produces a dataset-card warning instead of a silently shipped prior.

V. OPEN QUESTIONS

This is a bounded study. The following would confirm or falsify its transfer, and each is a natural design-time-safety contribution. (i) **Real-corpus probing**: are BridgeData V2, Open X-Embodiment, and DROID temporally clean by ActSync's gate, treating the abstain set as "unknown" rather than "clean"? Measured latency statistics from real logging, teleoperation, and controller stacks would also ground the stress scale we chose synthetically. (ii) **Chunked actions**: for action-

chunked policies such as ACT [22], the safest use is to audit and repair the primitive stream before chunking. A Lift/Can $+4$ diagnostic confirms pre-chunk repair makes all 26,873 evaluated 8-step state-aligned chunks identical to oracle, while corrupted chunks have nonzero error (Appendix). Whether label-equivalence translates to rollout recovery for VLA-scale chunked policies is open. (iii) **Localized and per-channel estimation**: the current estimator assumes one global shift. Mixed-offset corpora, per-session drift, and per-channel delays (e.g., arm and gripper streams logged through different queues) need a localization method, and are arguably the deployment-critical case. (iv) **Adversarial robustness**: our negative controls are constructive. Can the gate be fooled by action streams that preserve confidence but break causality? (v) **Stronger policies**: Square BC, Lift/Can BC-RNN, GPU-strong MLP, and Diffusion Policy [4] clean gates failed at our compute budget, so architecture-independent recovery is not claimed.

VI. Related Work

a) Data quality in imitation learning: Behavior cloning is the simplest offline IL baseline [17], with interactive variants motivated by compounding error [19], and surveys emphasize that demonstration quality and covariate shift shape downstream performance [1]. Recent work formalizes imitation-data quality through distribution shift and curation objectives [2, 6]. These tools assume each (s_t, a_t) is correctly paired. We audit that precondition.

b) Delay and time-alignment: Classical time-delay estimation [12, 13] and delayed-MDP work [10, 21, 3] make the decision process delay-aware. Temporal alignment for video and video-based imitation aligns demonstrations or embeddings across executions [5, 9]. Our setting differs: the environment is unchanged but stored labels are shifted before training, so the operational question is whether a stored corpus should be repaired *before* learning, independent of the learner.

c) Policy classes and datasets: Diffusion Policy [4] and action-chunked transformers [22] improve the action-distribution model but do not address whether labels were attached to the right timestep, and they consume the same corpora [14, 15, 20, 16, 11] our audit targets.

VII. Conclusion

The safest place to catch a class of robot failures is in the dataset, before any policy exists to be monitored. Temporal action-label misalignment is an under-tested, statistics-invisible defect that silently optimizes a policy to act for the wrong instant. We measured its severity, gave a CPU-scale audit-and-repair primitive that defaults to abstention, and framed dataset temporal-integrity auditing as a safety-by-design primitive whose abstention gate is itself a practice-time monitor at the data layer.

Appendix A
Additional Evidence

a) Per-seed primary results: Table IV gives the per-seed successes underlying Table I (ΔC is ActSync minus corrupt, ΔO ActSync minus oracle). ActSync beats corrupt in every seed pair.

TABLE IV
Primary $+4$ per-seed successes out of 100 rollouts.

Task	Seed	Clean	Corrupt	ActSync	Oracle	ΔC	ΔO
Can	0	78	18	70	75	+52	-5
Can	1	72	24	80	80	+56	+0
Can	2	73	0	80	77	+80	+3
Can	3	83	32	74	67	+42	+7
Can	4	84	37	72	61	+35	+11
Can	5	86	27	81	81	+54	+0
Can	6	71	43	46	51	+3	-5
Can	7	87	28	51	60	+23	-9
Lift	0	78	8	70	73	+62	-3
Lift	1	79	3	76	81	+73	-5
Lift	2	72	36	71	61	+35	+10
Lift	3	80	0	72	74	+72	-2
Lift	4	80	0	73	72	+73	+1
Lift	5	77	3	63	63	+60	+0
Lift	6	61	1	71	71	+70	+0
Lift	7	64	0	58	70	+58	-12

b) Matched-seed contrasts: Table V summarizes the paired-seed gains for the primary $+4$ result and the -4 directionality gates. Every contrast is positive in $8/8$ seeds with two-sided sign-test $p = 0.0078$.

TABLE V
Matched-seed contrasts (ActSync minus corrupt unless noted).

Task	Shift	Pos.	Mean	95% CI
Lift	+4	8/8	0.629	[0.522,0.736]
Can	+4	8/8	0.431	[0.236,0.626]
Lift	-4	8/8	0.629	[0.579,0.678]
Can	-4	8/8	0.709	[0.602,0.815]

c) Clean-reference gates (scope limits): Policy-recovery comparisons are expanded only after the clean-reference policy is meaningful. If a recipe cannot solve the clean-trim dataset, corrupt-vs-repair is uninterpretable. Table VI lists the failed gates that bound the current claim. Optional Transport/Tool Hang $+4$ offset smokes still recovered the expected repair shift, so these are policy-interpretability stops, not detector failures.

TABLE VI
Clean-reference gates that stopped unsupported expansions.

Task / recipe	Clean-reference condition	Success
Square CPU-medium BC	clean-trim +4	3/20
Square BC-RNN	clean-trim +4	0/20
Square GPU-strong MLP	clean-trim +4	0/100
Square Diffusion Policy	clean-trim +4	0/30
Lift/Can Diffusion Policy	clean-trim +4	0/10
Transport GPU-strong MLP	clean-trim +4	0/30
Tool Hang GPU-strong MLP	clean-trim	0/120
Lift/Can BC-RNN	clean-trim +4	0/20

d) Confidence calibration: Table VII summarizes the conservative triage rule. Thresholds are set from saved logged-data audits, never from rollouts. High-confidence exact global estimates can drive deterministic repair, while low-data nonzero estimates trigger inspection.

TABLE VII
CONFIDENCE TRIAGE FROM SAVED CPU AUDITS. SHIFTED SUBSAMPLES: 84 CASES. CLEAN 100-DEMO: 1200 UNSHIFTED SUBSAMPLES.

Audit set / threshold	Coverage	Exact	Within 1
Shifted, $\tau \geq 0.00$	1.000	0.667	0.964
Shifted, $\tau \geq 0.10$	0.393	0.818	0.970
Clean 100-demo, $\tau \geq 0.00$	1.000	0.812	1.000
Clean 100-demo, $\tau \geq 0.05$	0.480	0.972	1.000
Clean 100-demo, $\tau \geq 0.10$	0.277	1.000	1.000

e) Robustness diagnostics: Adding Gaussian action-log noise up to $0.20\times$ each task's action standard deviation preserves exact recovery on all $30/30$ Lift/Can/Square $+4$ noise cases. An interpolated fractional-delay probe is within one frame in $72/72$ cases (median absolute error 0.5 frame, nearest-integer recovery $49/72$). A harder jittered asynchronous-delay probe with sinusoidal or slow-random variation around ± 4-frame means is within one frame in $108/108$ cases but exact in $42/108$, reinforcing triage over blind exact repair under nonconstant latency. Varying the ridge split over 50 seeds keeps $1050/1050$ full-dataset estimates within one frame (exactness 0.941). A ridge-sensitivity sweep is exact in $189/189$ loss/regularization cases, and moving-filter percentiles 50–70 are exact in $63/63$ cases.

f) Action-chunking applicability: For chunked policies, repair the primitive stream before chunk construction. Across horizons $4, 8, 16, 32$, ActSync chunks match oracle chunks exactly (max error 0.0) while corrupted chunks carry nonzero error (e.g., horizon-8 mean chunk MSE 0.082 across $26{,}873$ chunks). A small low-dimensional 8-step chunk predictor trained on ActSync labels matches the clean-oracle oracle-label MSE on both Lift (0.0407) and Can (0.0421), versus corrupted (0.0883 / 0.0661). This is a label-equivalence diagnostic, not a rollout claim.

g) Reproducibility: All aggregate artifacts are generated from saved JSON/CSV results in our project repository. We plan to publicly release the audit, corruption, repair, plotting, and aggregation scripts, together with saved rollout logs and trained-policy metadata sufficient to reproduce every reported table, in a future code release.

REFERENCES

[1] Brenna D. Argall, Sonia Chernova, Manuela Veloso, and Brett Browning. A survey of robot learning from demonstration. *Robotics and Autonomous Systems*, 57(5): 469–483, 2009. doi: 10.1016/j.robot.2008.10.024.

[2] Suneel Belkhale, Yuchen Cui, and Dorsa Sadigh. Data quality in imitation learning. In *Advances in Neural Information Processing Systems*, volume 36, 2023.

[3] Baiming Chen, Mengdi Xu, Liang Li, and Ding Zhao. Delay-aware model-based reinforcement learning for continuous control. *Neurocomputing*, 450:119–128, 2021. doi: 10.1016/j.neucom.2021.04.015.

[4] Cheng Chi et al. Diffusion policy: Visuomotor policy learning via action diffusion. *The International Journal of Robotics Research*, 44(10–11):1684–1704, 2025. doi: 10.1177/02783649241273668.

[5] Debidatta Dwibedi, Yusuf Aytar, Jonathan Tompson, Pierre Sermanet, and Andrew Zisserman. Temporal cycle-consistency learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1801–1810, 2019. doi: 10.1109/CVPR.2019.00190.

[6] Joey Hejna et al. Robot data curation with mutual information estimators. In *Proceedings of Robotics: Science and Systems*, Los Angeles, CA, USA, June 2025. doi: 10.15607/RSS.2025.XXI.023.

[7] Arthur E. Hoerl and Robert W. Kennard. Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics*, 12(1):55–67, 1970. doi: 10.1080/00401706.1970.10488634.

[8] Peter J. Huber. Robust estimation of a location parameter. *The Annals of Mathematical Statistics*, 35(1):73–101, 1964. doi: 10.1214/aoms/1177703732.

[9] William Huey, Huaxiaoyue Wang, Anne Wu, Yoav Artzi, and Sanjiban Choudhury. Imitation learning from a single temporally misaligned video. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pages 26283–26310. PMLR, 2025.

[10] Konstantinos V. Katsikopoulos and Sascha E. Engelbrecht. Markov decision processes with delays and asynchronous cost collection. *IEEE Transactions on Automatic Control*, 48(4):568–574, 2003. doi: 10.1109/TAC.2003.809799.

[11] Alexander Khazatsky et al. DROID: A large-scale in-the-wild robot manipulation dataset. In *Proceedings of Robotics: Science and Systems*, Delft, Netherlands, July 2024. doi: 10.15607/RSS.2024.XX.120.

[12] Charles H. Knapp and G. Clifford Carter. The generalized correlation method for estimation of time delay. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 24(4):320–327, 1976. doi: 10.1109/TASSP.1976.1162830.

[13] Lennart Ljung. *System Identification: Theory for the User*. Prentice Hall, Upper Saddle River, NJ, 2 edition, 1999.

[14] Ajay Mandlekar et al. ROBOTURK: A crowdsourcing platform for robotic skill learning through imitation. In *Proceedings of The 2nd Conference on Robot Learning*, volume 87 of *Proceedings of Machine Learning Research*, pages 879–893. PMLR, 2018.

[15] Ajay Mandlekar et al. What matters in learning from offline human demonstrations for robot manipulation. In *Proceedings of the 5th Conference on Robot Learning*, volume 164 of *Proceedings of Machine Learning Research*, pages 1678–1690. PMLR, 2022.

[16] Open X-Embodiment Collaboration. Open x-embodiment: Robotic learning datasets and RT-X models. In *IEEE International Conference on Robotics and Automation*, pages 6892–6903, May 2024. doi: 10.1109/ICRA57147.2024.10611477.

[17] Dean A. Pomerleau. ALVINN: An autonomous land vehicle in a neural network. In *Advances in Neural Information Processing Systems*, volume 1, 1988.

[18] robomimic Core Team. robomimic documentation: Environments. https://robomimic.github.io/docs/modules/environments.html, 2023. Accessed 2026-05-29.

[19] Stephane Ross, Geoffrey Gordon, and Drew 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*, volume 15 of *Proceedings of Machine Learning Research*, pages 627–635. PMLR, 2011.

[20] Homer Rich Walke et al. Bridgedata v2: A dataset for robot learning at scale. In *Proceedings of The 7th Conference on Robot Learning*, volume 229 of *Proceedings of Machine Learning Research*, pages 1723–1736. PMLR, 2023.

[21] Thomas J. Walsh, Ali Nouri, Lihong Li, and Michael L. Littman. Learning and planning in environments with delayed feedback. *Autonomous Agents and Multi-Agent Systems*, 18(1):83–105, 2009. doi: 10.1007/s10458-008-9056-7.

[22] Tony Z. Zhao, Vikash Kumar, Sergey Levine, and Chelsea Finn. Learning fine-grained bimanual manipulation with low-cost hardware. In *Robotics: Science and Systems*, 2023. doi: 10.15607/RSS.2023.XIX.016.