ActionABI: Turning Undocumented Action Tensors into Verified Converters or Honest Refusals

Krishi Attri
krishiattriwork@gmail.com

Abstract

Robot learning datasets and deployments routinely carry action tensors whose meaning—which channel drives which degree of freedom, its sign, scale, whether targets are absolute or delta, in which frame, with what pipeline lag, and whether the gripper is inverted—is undocumented, untrusted, or missing. Existing tools either check *declared* metadata against a specification or define new schemas going forward; neither retrofits legacy or adversarial data carrying no metadata at all. We present ActionABI, a C++20 evidence-first CLI that scores a finite, declared action-interface grammar against timestamped trajectories, retains a calibrated observational-equivalence set instead of breaking ties, and emits a converter only when every required field is supported by evidence—abstaining rather than forcing a unique answer. On a 100-case synthetic ambiguity gate, ActionABI eliminates all 25 false unique certifications a forced-argmin baseline makes, at full coverage. Against ground-truth latent contracts from real ManiSkill GPU simulation (90 traces, 45 contracts), structural fields (permutation, sign) recover to 0.80/0.84 overall after a lag-observable fix, while scale is confounded by the physical controller, not the grammar. A 35-dataset Hugging Face Hub audit scores recovered conventions against documented ones over 139 field-labels with 0 contradictions and 0 false uniques, finding that "OXE" is not one action convention and that ≥ 5 gripper-polarity conventions coexist. Three measurement-driven defects were found, root-caused, fixed, and regression-tested; ActionABI's C++ core runs load-bearing in a sibling adaptation benchmark at 5.7×10^{-14} parity and a $\sim$13$\times$ CPU-regime speedup, while its CUDA path honestly fails a preregistered 5$\times$ gate. We make no "first" claims and no hardware claims; every number traces to a committed report.

1 Introduction

A policy's action tensor is only meaningful given an *action-interface contract*: which numeric channel controls which physical degree of freedom, its sign and scale, whether it commands a delta or an absolute pose, in what frame, with what pipeline lag, and whether its gripper channel is inverted. When documented and correct, this contract is invisible. When it is undocumented, stale, or silently wrong—a mislabeled dataset, a legacy log with no surviving metadata, a third-party checkpoint with an unverifiable `README`—a wrong guess does not corrupt a file; it moves a real or simulated robot the wrong way. This asymmetry—a wrong network-field guess corrupts data, a wrong action-semantics guess moves a robot—is the central fact separating action-ABI recovery from most "infer an undocumented format from evidence" problems, and it forces a design commitment: the system must be able to say *"I don't know"* and refuse to certify a unique semantics rather than guess.

The premise is not invented. Our Hub audit (§5.3) finds at least five mutually-incompatible gripper-polarity conventions in the wild—most starkly, ALOHA [21] documents 0=closed/1=open while Open X-Embodiment [17] documents 1=close/$-$1=open/0=no-change for the same numeric channel. The same value carries the opposite physical meaning across two live dataset families. Two adjacent lines of work do not close the gap. *Specification-checking tools* (ExecSpec) detect drift from *declared* metadata assumed knowable and checkable; they do not infer semantics when metadata is missing. *Forward-looking schemas* (ORTF, the LeRobot action-representation docs) specify what a *new* episode must declare; they do not retrofit the large legacy population that predates or ignores such a schema. Neither targets what we call **forensic action-ABI recovery**: from passive trajectory evidence alone, with no trusted metadata, recover as much action semantics as the evidence supports, report calibrated equivalence sets where it does not, and refuse an unsafe converter rather than force an answer.

Contributions. (1) A forensic-recovery framing distinct from both specification-checking and schema-standardization, substantiated by an explicit literature check (§2). (2) A calibrated-equivalence, abstention-first scoring design (§3) that measurably eliminates false unique certifications relative to a forced-argmin baseline (0 vs. 25 on 100 cases; §5.1). (3) A ground-truth-labeled supervised accuracy measurement from real ManiSkill GPU dynamics (§5.2): structural fields recover, scale is a controller confound, reported with pre-/post-fix numbers held apart. (4) A 35-dataset Hub audit (§5.3) over 139 field-labels—0 contradictions, 0 false uniques—quantifying real convention fragmentation. (5) Three measurement-driven defects found, fixed, and regression-tested (§4), disclosed with before/after numbers. (6) A

load-bearing systems artifact: ActionABI's C++ core runs as a live pybind11 backend inside a sibling belief loop at 5.7×10^{-14} parity (§5.4) while its CUDA path fails a preregistered gate (§5.5). We make **no "first" claims**; every novelty statement is "we are not aware of prior work that…"

2 Related Work

Interface specification and metadata-mismatch checking (closest collision). ExecSpec [18] (arXiv:2606.03724, an unreviewed single-author cs.CR preprint) formalizes an executable policy specification (model, action representation, unnormalizer, controller convention) and shows metadata mismatches collapse task success (28/28→2/28, 26/26→0/26). It detects drift *analytically from declared metadata*, assuming the correct metadata is knowable pre-rollout. ActionABI's distinct case is when metadata is *missing, untrusted, or absent* and the only evidence is the trajectory itself; we cite it first because it is our nearest neighbor and flag it as what it is—not a peer-reviewed result to lean on.

Dataset standardization and forward-declared schemas. ORTF v0.2 [9] is an episode/manifest schema a validator checks; it says what a well-formed episode *must* declare, not what an underspecified legacy one *means*. It is a commercial, non-peer-reviewed vendor resource, cited descriptively and never "beaten." The **LeRobot action-representation docs** [12] document the two canonical action spaces (joint vs. end-effector) and delta-vs-absolute, but do not name permutation, sign, or scale as axes and leave gripper per-dataset—the exact taxonomy our grammar overlaps.

Action-space design and shared representations. Demystifying Action Space Design [7] (arXiv:2602.23408, 500+ trained models) is a training-time design-choice study, not deployment-time recovery of an already-fixed convention. **Implicit Kinematic Policies** [8] (ICRA 2022) learns a policy jointly expressive across Cartesian/joint spaces and compensates small encoder offsets as a side effect—a policy-learning robustness trick, not a provenance-preserving audit that reports identified/ambiguous/unsupported fields and refuses conversion.

Self-modeling and protocol reverse engineering—the epistemic lineage. Robot self-modeling (Science Robotics 2022/2023 [2, 5]; Reconfigurable Robot Identification, arXiv:2403.10496 [10]) infers a robot's own *morphology* from self-observation, not what an external action tensor means. General-CS **protocol reverse engineering** (Discoverer [4] lineage; BinPRE, CCS 2024 [14]; automatic state-machine inference, arXiv:2412.02540 [19]) shares ActionABI's exact epistemic shape—infer an undocumented format from traces with field-level confidence—but targets network/binary formats under a different failure cost. That difference is precisely why ActionABI has a converter-refusal gate PRE tools generally lack. We do not claim novelty of equivalence-class reasoning itself; **novelty rests on the combination** (trajectory-only evidence + physical action semantics + calibrated abstention + converter refusal), not on any single ingredient.

3 Method: evidence-first forensic recovery

3.1 Grammar and separable held-out scoring

The declared grammar is finite: a **permutation** π of channels; per-channel **sign** $\sigma \in \{+1, -1\}$; per-channel positive **scale** on the grid $\{0.5, 0.75, 1.0, 1.25, 1.5, 2.0\}$; a **target** in {delta, absolute, velocity}; a base/tool **frame**; a nonnegative integer **lag**; and **gripper** inversion. A contract θ *decodes* a raw action a channel-wise, $\text{decode}(a, \theta)_i = \sigma_i \, s_i \, a_{\pi(i)}$. Canonical JSONL input begins with one metadata record (source filename, SHA-256, extraction date, state columns, units) followed by samples carrying `episode_id`, `t_ns`, `state`, `action`.

The C++ scorer (`src/score_cpu.cpp`) splits episodes into a train fraction and an *episode-disjoint* held-out remainder (zero row leakage). For contract θ and output channel c it forms the residual $r_c = \text{decode}(a, \theta)_c - o_c(\theta)$ between the decoded command and the target-and-lag-dependent observable o_c, and scores it with a robust Huber loss [11]

$$\ell(r) = \begin{cases} \frac{1}{2} r^2, & |r| \le \delta, \\ \delta \left(|r| - \frac{1}{2}\delta \right), & |r| > \delta, \end{cases}$$

accumulating a per-channel held-out mean $L_c(\theta) = \text{mean}\, \ell(r_c)$; the held-out loss is $\frac{1}{D} \sum_c L_c(\theta)$. Because this held-out loss is separable—the mean of per-dimension losses—the optimal channel assignment for a fixed (target, lag) with its best per-cell (σ, s) is a *linear-assignment problem* over the C++ `per_dimension_loss` matrix, and the permutation is resolved by *brute-force* over the small finite permutation group. This evaluates the entire declared finite grammar with authentic C++ residuals without enumerating its billions of members: for the labeled bridge (§5.2) a trace-independent basis of 432 per-cell contracts ($6 \times 12 \times 2 \times 3$) feeds a brute force over all 720 permutations.

3.2 The single-step-delayed observable

Lag is a subtle observable and the source of a real defect (§4). The physical response of a lagged contract is a *single-step delta arriving `lag` steps late*, $\text{executed}_t = \text{decode}(\text{raw}_{t-\text{lag}})$. With offset = lag+1, the correct delta observable is the one-step transition at the delayed index,

$$o^{\text{delta}} = \text{state}[t{+}\text{lag}{+}1] - \text{state}[t{+}\text{lag}],$$

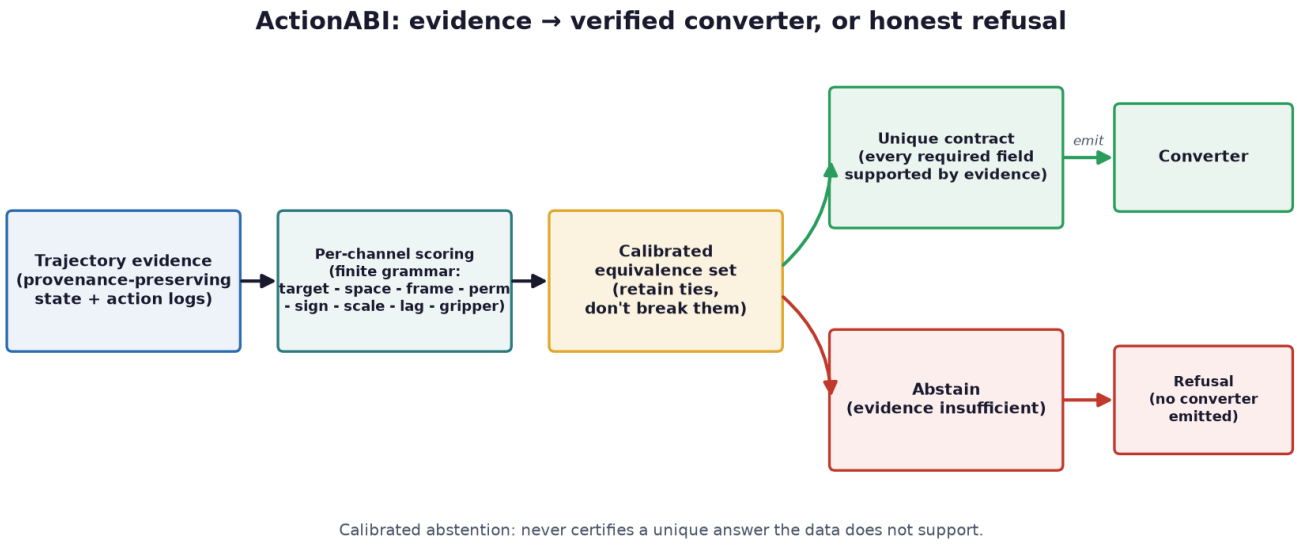

Figure 1: Evidence in, a verified converter or an honest refusal out. The calibrated equivalence set is the heart of the tool: when two contracts fit the data equally well, ActionABI keeps both and abstains rather than certifying a lie.

the velocity observable divides by the elapsed time, and the absolute observable is state[t+lag+1]. This reduces to state[t+1] − state[t] at lag 0 and matches ActionABI's Python reference scorer for every lag.

3.3 Calibrated equivalence and abstention

Rather than certifying the single minimum-loss hypothesis (forced argmin), ActionABI performs a *paired bootstrap* [6] over held-out residual rows: any hypothesis whose per-row loss-gap 95% CI to the argmin includes zero joins the argmin's **equivalence set**. A field is certified unique only when its equivalence set is a singleton; otherwise ActionABI reports the set and abstains. Fields with no discriminating evidence at all (e.g. gripper inversion under a pose-only observable) are **unsupported**, not guessed. The bootstrap alone calibrates *sampling* noise and assumes the best hypothesis's residuals are zero-mean; on real dynamics this fails (§4, defect 2), so it is augmented with a **fail-closed, bias-robust guard** that estimates a systematic-bias bound from the argmin's held-out residual structure and abstains under detected misspecification rather than certify a biased answer (Fig. 2).

3.4 Converter refusal

A converter is emitted only when every required field is supported and, where applicable, uniquely identified or safely defaultable; otherwise ActionABI refuses rather than emit a converter with invented metadata—the safety-relevant asymmetry motivating the whole design. Documentation, when present, is used only as a *label to score recovered conventions against*, never as an inference input.

4 Measurement-Driven Hardening: Three Fixed Defects

ActionABI's supervised and audit evidence is new precisely because it is the first ground-truth-labeled and largest documentation-scored measurement the tool has faced, and that measurement surfaced three real code defects—not measurement artifacts—that synthetic and unlabeled evaluation could not have found. Each was root-caused, fixed, and guarded by a regression test shown to fail on the old code and pass on the fix.

Defect 1—lag observable spanned lag+1 steps. *Measurement:* on the labeled bridge, all fields degraded as lag grew (permutation 0.92→0.58→0.49 for lag 0/1/2) and lag itself was barely recovered (0.25–0.43). *Root cause:* `src/score_cpu.cpp` (and `cuda/score_cuda.cu`) used offset = lag+1 with observable state[t+lag+1] − state[t], so for lag>0 it summed lag+1 consecutive deltas, giving the true lagged contract a spurious residual so it lost to wrong-lag hypotheses; it also disagreed with the Python reference scorer, which uses consecutive deltas. *Fix:* score against the one-step transition at the delayed index (§3.2), applied identically in C++ and CUDA. *Regression test:* a new CTest builds a trace under the single-step-delayed model at lag ∈ {1, 2} and asserts the true contract is now near-lossless and strictly beats every wrong-lag hypothesis; it fails 7/8 assertions on the old span and passes 8/8 on the fix. C++/Python residual parity now holds for every lag at max gap 2.8×10^{-16}. *Before/after:* Table 3—overall permutation 0.63→0.80, lag 0.39→0.66, lag=2 permutation 0.49→0.89.

Defect 2—calibration not robust to systematic response bias. *Measurement:* pre-fix the calibrated set still produced 4 false uniques (all lag>0) with truth coverage 0.02, despite abstaining on 96% of traces. *Root*

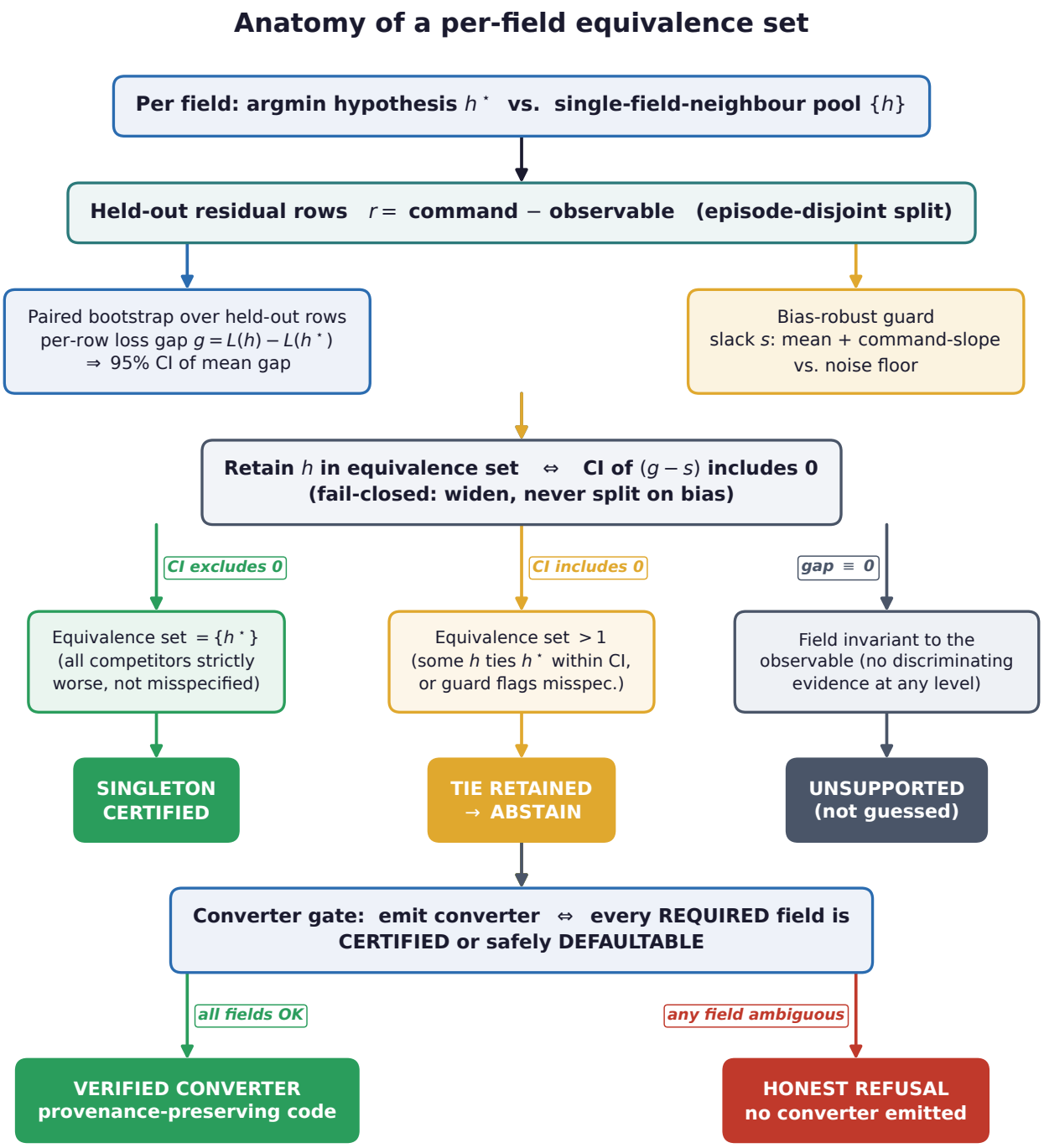

Figure 2: Anatomy of a per-field equivalence set. Held-out residuals feed a paired bootstrap over the loss gap g between the argmin $h^\star$ and each single-field neighbour, widened by a fail-closed bias slack s. A field is *certified* only when its equivalence set is a singleton, *abstained* when a tie or misspecification is detected, and *unsupported* when the observable is invariant to it. A converter is emitted only if every required field is certified or safely defaultable; otherwise ActionABI refuses.

cause: the paired bootstrap guards *sampling* noise but not a common, command-correlated *systematic* residual (the ∼0.6× scale attenuation of §5.2); with ∼180 held-out rows this small systematic gap is statistically resolvable, letting the set certify a biased argmin as unique while excluding the truth. *Fix (`experiments/bias_robust.py`):* a fail-closed guard estimates a per-channel systematic-bias bound (mean offset t_{mean} and command-correlated

Table 1: 100-case synthetic gate, frozen seed 20260718 (75 excited identifiable + 25 zero-excitation equivalence cases). Source: `reports/benchmark_sprint.md`.

Method	tgt/lag	cov.	false uniq.	abst.	Brier/ECE
Metadata	—	0.00	0	1.00	0.00
Forced argmin	0.82	1.00	**25**	0.00	0.25
ActionABI	1.00	1.00	**0**	0.25	0.00

slope t_{slope} vs. the noise floor, t_{crit}=4), inflates the equivalence threshold by the bias-explained Huber slack s so a candidate is split from the argmin only if worse by more than the estimated bias, and abstains entirely when any channel is flagged misspecified. *Regression test:* 8 new tests (`test_bias_robust.py`)—zero-mean noise stays unflagged with slack ≈0; constant offset and command-correlated bias are flagged; a fixed margin splits at s=0 but is retained at $s \geq d$. *Before/after:* labeled-sim false uniques 4→0, truth coverage 0.02→0.39; because the guard is a no-op under zero-mean noise, the synthetic matrix is **unchanged** (still 0 false uniques). Flag rate 0.60, mean slack 0.031.

Defect 3—gate false-equivalence on near-static resets. *Measurement:* the first full 35-dataset Hub run produced exactly one contradiction, on `berkeley_rpt`. *Root cause:* on a near-static reset trajectory, absolute and episode-relative targets fit *each other* (the affine intercept absorbs the static pose), so `real_dataset_gate.py`'s equivalence branch fired on absolute ≈ episode_relative *without* checking absolute was the best-fit target, certifying a 2-way set that excluded the strictly-better delta. ActionABI's own evidence ranks delta best (held-out delta = velocity 0.353 nRMSE vs. absolute 0.926; fixed C++ scorer delta 0.030 < velocity 0.047 < absolute 0.521)—so the documentation was correct and the contradiction was an ActionABI defect, not a doc error. *Fix:* require the overall best-fit target to be absolute or episode-relative before taking the equivalence branch. *Regression test:* `test_delta_control_with_fixed_resets_is_not_false_equivalence`. *Before/after:* `berkeley_rpt` now returns honest abstention; all six pinned outcomes preserved, contradiction removed.

Verification ledger: 8/8 Release CTests, 8/8 under AddressSanitizer/UBSan, CUDA parity pass (GPU 3, sm_120), and 32/32 Python experiment tests (24 prior + 8 new bias tests).

5 Evidence

5.1 Synthetic ambiguity gate

Forced argmin turns all 25 constructed equivalence cases into false unique certifications; ActionABI abstains on exactly those 25 and produces **zero** false uniques at full coverage (Table 1). The important result is not the point accuracy—it is that ActionABI reports the equivalence set on uninformative traces and refuses unique certification. Both scorer fixes (§4) re-ran this gate unchanged, because the bias guard is a no-op under zero-mean synthetic noise. This is finite-grammar synthetic evidence only.

Table 2: Pre-fix forced-argmin per-field accuracy (permutation/sign/scale are per-channel means over 6 channels). The 0.97/0.98 figure is the narrow favorable stratum, *not* the headline. Source: `reports/labeled_sim_traces.md`.

Stratum	n	perm	sign	scale	tgt	lag
Overall	90	0.63	0.76	0.24	0.74	0.39
lag = 0	22	**0.92**	**0.93**	0.36	0.68	0.77
lag = 1	28	0.58	0.74	0.19	0.71	0.29
lag = 2	40	0.49	0.69	0.22	0.80	0.25
policy exc.	45	0.79	0.85	0.27	0.93	0.31
random exc.	45	0.47	0.67	0.22	0.56	0.47
lag0, random	11	**0.97**	**0.98**	0.42	0.45	**1.00**
lag0, policy	11	0.88	0.88	0.30	0.91	0.55
lag1, policy	14	0.77	0.89	0.30	0.93	0.14
lag1, random	14	0.39	0.58	0.08	0.50	0.43
lag2, policy	20	0.74	0.81	0.23	0.95	0.30
lag2, random	20	0.24	0.57	0.20	0.65	0.20

Table 3: Pre-fix → post-fix forced-argmin accuracy after the lag-observable fix. **Post-fix overall** 0.80/0.84 **is the headline.** Source: `reports/scorer_fixes.md`.

Stratum	n	perm	sign	lag
Overall	90	0.63→**0.80**	0.76→**0.84**	0.39→**0.66**
lag = 0	22	0.92→0.72	0.93→0.75	0.77→0.45
lag = 1	28	0.58→**0.73**	0.74→**0.78**	0.29→**0.50**
lag = 2	40	0.49→**0.89**	0.69→**0.92**	0.25→**0.88**

5.2 Labeled real-sim traces: supervised accuracy

ActionABI is scored against **90 traces (45 contracts, hash-disjoint from the sibling benchmark)** generated by rolling out a frozen, competent PPO policy under sampled ground-truth contracts on real ManiSkill 3.0.1 GPU simulation (PickCube-v1, `pd_ee_delta_pose`, backbone `8131f330bd69aa6b`/SHA `3e6c95d6...`, dataset manifest SHA `aaee673a...`). Scale is drawn from ActionABI's own finite grid and lag from $\{0, 1, 2\}$, so supervised accuracy is well-defined over exactly the declared grammar; the label is never an inference input. C++/Python residual parity: max gap 2.8×10^{-16}.

These two accuracy claims are distinct and must not be conflated. The 0.97/0.98 figure is the *pre-fix* permutation/sign accuracy on the favorable lag=0, random-excitation stratum (n=11)—the strongest, narrowest slice, not the headline. After the lag fix, the *overall* forced-argmin accuracy across all 90 traces is Table 3: permutation 0.63→**0.80**, sign 0.76→**0.84**, lag 0.39→**0.66**, scale 0.24→0.31, target 0.74 unchanged. The lag>0 strata—the pre-fix failures—improve sharply (lag=2 permutation

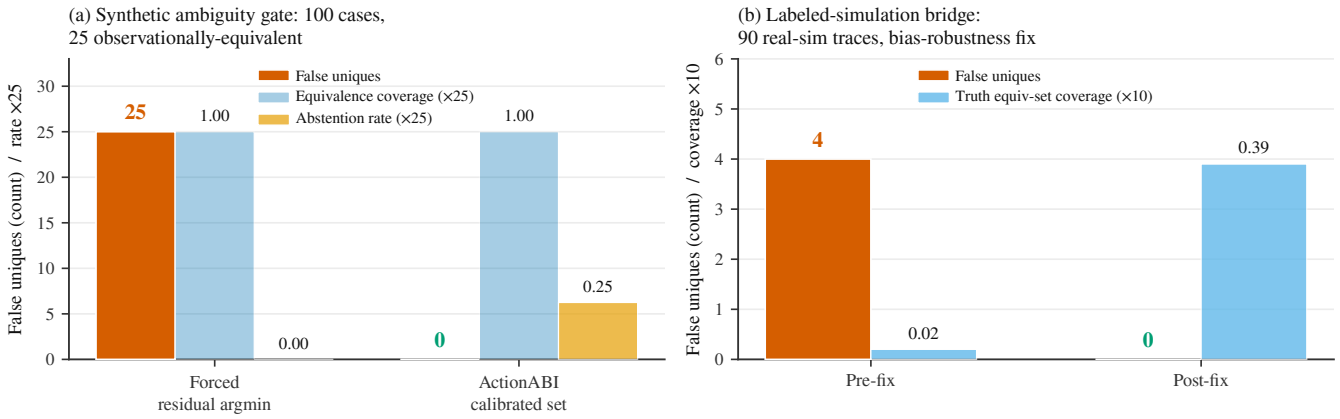

Figure 3: Calibrated abstention eliminates false uniques on the synthetic gate (left: 25→0 at coverage 1.00) and, after the bias-robustness fix, on the labeled-simulation bridge (right: false uniques 4→0, truth coverage 0.02→0.39).

Table 4: Calibrated comparator, pre-fix and after the bias fix. Sources: `labeled_sim_traces.md`, `scorer_fixes.md`.

Metric	Overall	lag0	lag1	lag2
false unique (pre)	4	**0**	1	3
truth coverage (pre)	0.02	0.09	0.00	0.00
median gap (truth−argmin)	0.30	**0.075**	0.25	0.49
false unique (post)	**0**	0	0	0
truth coverage (post)	**0.39**	0.27	0.29	0.53

Table 5: Hub audit over 139 field-labels. Source: `reports/lerobot_hub_audit.md`.

Metric	Base (6)	New (29)	All (35)
Contradictions	0	**0**	**0 / 139**
Unique certifications	1	4	5
documentation-correct	1	4	**5 / 5**
false uniques	0	**0**	**0**
Equivalence-consistent	1	3	4
Partial-consistent	1	11	12
Flagged partial (frame)	1	2	3
Abstention-consistent	23	92	115
Unlabeled / N/A	3	33	36

0.49→0.89). We report the **honest counter-movement**: the lag=0 stratum *regresses* (permutation 0.92→0.72) because once lag>0 hypotheses score correctly they compete fairly and a smooth passive trajectory does not strongly separate lag on a lag=0 trace. The net across strata is strongly positive; the post-fix overall **0.80/0.84** is the number to cite.

Findings. (1) *Structural fields are identifiable from real simulated responses*: permutation and sign reach 0.92/0.93 at lag 0, rising to 0.97/0.98 under random excitation (pre-fix)—attenuation from the weak controller preserves *which* channel and sign best explain the motion. (2) *Scale is not identifiable—a controller/response-model confound, not a grammar limit*: accuracy 0.24–0.42; diagnostically the least-squares scale is systematically attenuated to ≈0.6× truth (median 0.67×, 52% of channels below 0.7×, fit R^2 0.09–0.34), so the loss-optimal grid scale lands below truth. (3) *Target identifiability is excitation-dependent*: smooth policy excitation identifies delta-vs-absolute well (0.91–0.95); random excitation does not (0.45–0.65). (4) *Calibrated abstention avoids false uniques—after the bias fix*: pre-fix 4 false uniques at lag>0 with truth coverage 0.02; the guard drives this to 0 and coverage 0.39 (Table 4, Fig. 4).

5.3 35-dataset Hugging Face Hub audit

Pointed at **35 in-the-wild LeRobot datasets** (6 pinned + 29 newly audited; 383 MB of trajectory Parquet) across six ecosystems (OXE ports [17], native ALOHA [21], SO-100/101 teleop, LIBERO [16], MetaWorld [20], gym PushT [3]/xArm), ActionABI scores recovered conventions against documented ones per field, with documentation a *label only*, never an inference input (test-enforced). Over **139 documented field-labels** (Table 5): **0 contradictions**, 5 unique certifications (5/5 documentation-correct), **0 false uniques**, 115 honest abstentions on knowable fields. This reproduces the synthetic abstention property (§5.1) at ∼6× the dataset count, and it *required* the defect-3 gate fix. The full per-dataset ledger is Appendix Table 9.

"OXE" is not one convention. The 15 OXE ports do *not* share an action space: 9 delta, 3 velocity (`cable_routing`, `nyu_door`, `ucsd_pick`), 1 nominally absolute but evidence-delta (`taco_play`), 2 undocumented/mismatched (`stanford_kuka` ships a 7-dim tensor against a 4-dim absolute spec; `ucsd_kitchen`

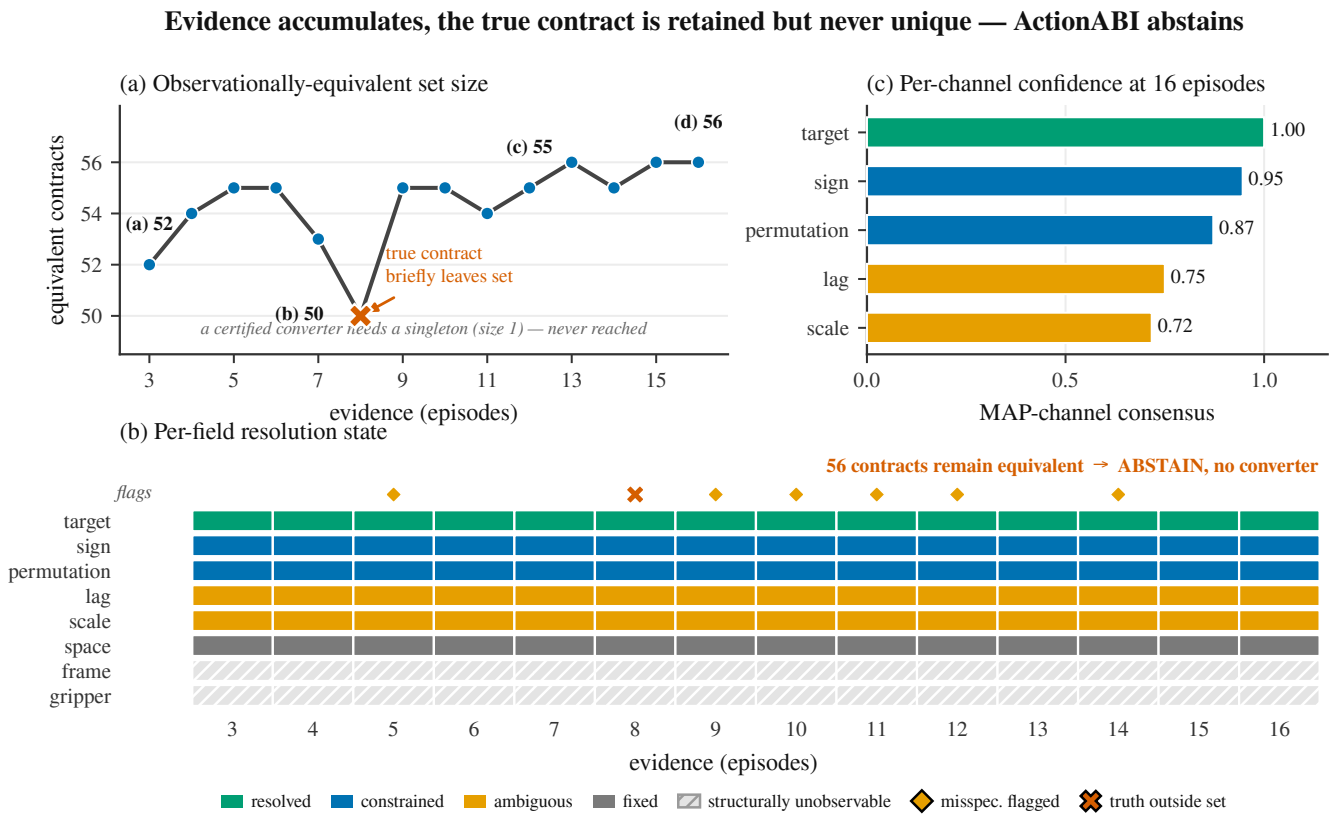

Figure 4: Evidence accumulation on one labeled policy trace (`c029_policy`), from real held-out Huber residuals; every plotted value is read from the committed trace. **(a)** The observationally-equivalent contract set never collapses to a singleton—it drifts over 52–56 contracts across episodes 3–16 (checkpoints (a)–(d)), so no unique converter is certifiable, and at 8 episodes the true contract *briefly leaves* the retained set, marked honestly. **(b)** Per-field resolution state: `target` is RESOLVED, `sign`/`permutation` stay *constrained*, `lag`/`scale` *ambiguous*, `space` is fixed, and `frame`/`gripper` are structurally unobservable; misspecification is flagged at six episodes and the episode-8 truth-exit marked. **(c)** Per-channel confidence at 16 episodes is high (0.72–1.00) yet the 56-contract set is still not unique, so ActionABI **abstains and emits no converter**. Source: `media/recovery_trace.json`.

leaves delta-vs-absolute unstated). Layouts range from `world_vector+rotation_delta` (ur5, roboturk, toto) to bare [*dx*..*dyaw*] (fanuc), delta joint (mvp, rpt), to mixed joint-velocity+EE-delta (nyu_franka, 15-dim). All 15 lost their semantic RLDS names to generic `motor_N` in the LeRobot v3.0 conversion—the config-level convention is uniformly erased.

≥ 5 **gripper-polarity conventions coexist** (Fig. 6, Table 6). Most starkly, ALOHA's 0=closed/1=open is *literally inverted* from OXE's `gripper_closedness_action` 1=close. ActionABI *abstains* on gripper for every real dataset, so this is a documentation-side cluster, reported honestly as such.

Table 6: Documented gripper-polarity conventions. Source: `reports/lerobot_hub_audit.md`.

Convention	Datasets	Meaning
ALOHA joint	6 ALOHA (+3 SO-100/101)	0=closed, 1=open
OXE closedness	ur5, jaco, roboturk, nyu_door	**1=close**, −1/0=open
OXE binary state	mvp, rpt	1=closed, 0=open
OXE `open_gripper`	toto	True=open (inverted naming)
taco gripper	taco_play	−1=open, 1=close
robosuite cont.	libero(_10), metaworld	continuous $[-1, 1]$

5.4 The pybind11 fusion: a load-bearing systems artifact

ActionABI's evidence-scoring core is not only a CLI. `bindings/` builds it as a Python extension (`actionabi_cells`, a dedicated CMake target discovering pybind11 [13] via its installed config with a FetchContent fallback) that runs as a **live identification backend**

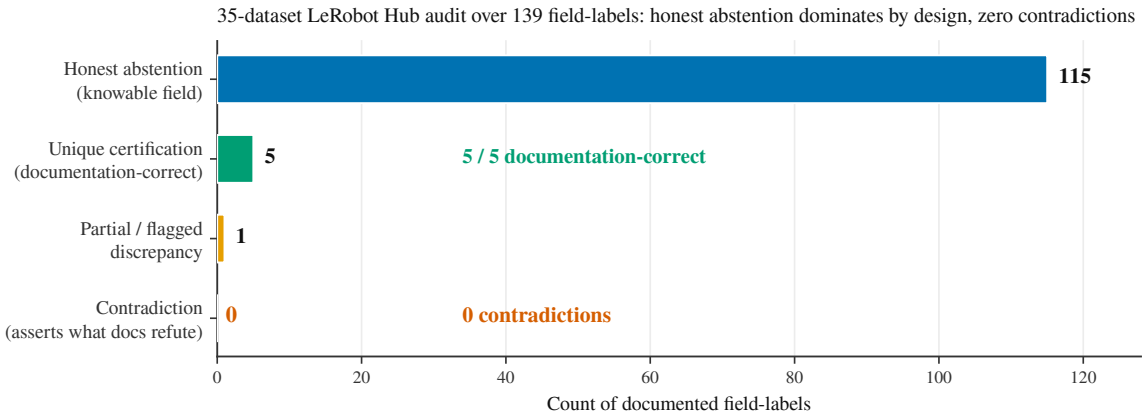

Figure 5: The audit ledger: of 139 documented field-labels, honest abstention (115) is the dominant, designed outcome; the 5 unique certifications are all documentation-correct; contradictions are zero.

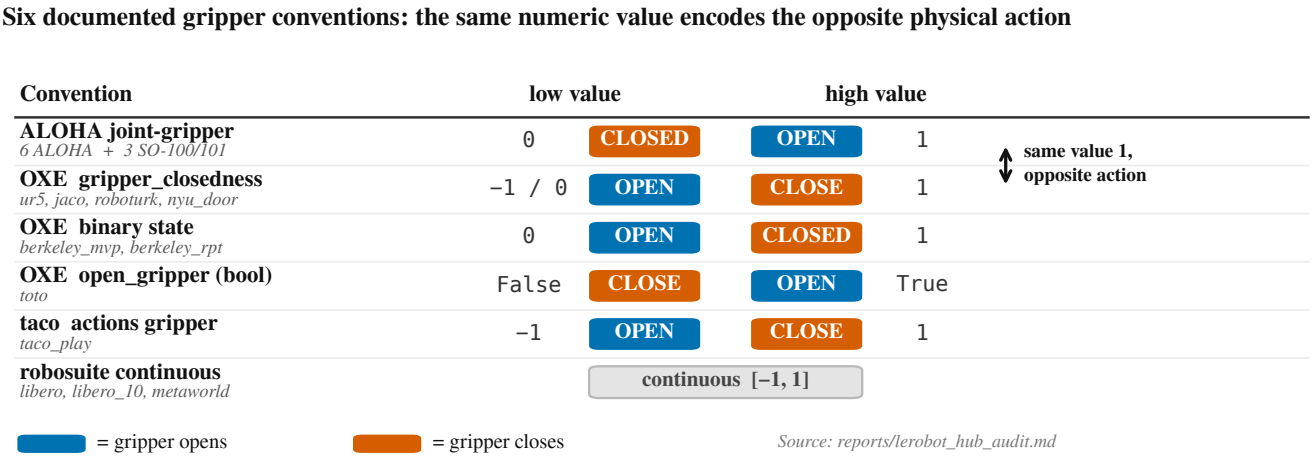

Figure 6: Six documented gripper-polarity conventions from the 35-dataset Hub audit; each row is a *documented* convention scored as a label, never an ActionABI inference. The physical meaning of the low and high numeric value is colour-coded (blue: gripper opens; vermillion: closes). The same numeric value 1 means *open* for ALOHA but *close* for OXE `gripper_closedness_action`—a direct polarity flip on the same channel across two live dataset families—while robosuite [22] datasets use an undirected continuous $[-1, 1]$ channel. ActionABI *abstains* on gripper for every real dataset, so this is a documentation-side cluster. Source: `reports/lerobot_hub_audit.md`.

inside the sibling ActionShift factorized-grammar Bayesian belief loop. It computes exactly the per-cell Gaussian log-evidence and pooled per-hypothesis log-likelihood ActionShift otherwise computes in torch,

$$\text{contribution}[b,i,j,s,k] = -\tfrac{1}{2}\left(\frac{o_{b,i} - \alpha_i\, \text{base}_{b,j}\, \sigma_s\, s_k}{\sigma_i}\right)^2,$$

as a single fused C++ pass with *no* intermediate tensors (the torch path allocates several broadcast temporaries per mode), using the fixed single-step-delayed lag semantics (§3.2). The seam is minimal: `FactorizedGrammarDriver` gained one optional `cell_scorer` argument and the inner scoring block was extracted verbatim; with no scorer the original torch path runs unchanged (all 13 factorized tests green), and only the evidence-scoring step crosses into C++—the MAP linear-assignment, history ring, masks, and encode stay in Python.

Parity methodology. Float64 per-cell evidence over a 40-step sequence across every grammar mode reproduces torch at max relative diff 5.7×10^{-14} (target $\leq 1 \times 10^{-6}$, met with $\sim$7 orders of margin); float32 (production) MAP decisions are *bit-identical* to torch (per-cell relative diffs $\sim 1.5 \times 10^{-5}$ on near-zero cells only, from accumulation order); the pool log-likelihood matches to $< 1 \times 10^{-6}$; and C++ CUDA vs C++ CPU agrees to 6.1×10^{-5} (float32 rounding). End-to-end on real PickCube (200 episodes, identical frozen checkpoint), success is within one episode of torch (torch 0.320, C++ 0.315)—a single float32 MAP flip, identical per-contract structure.

In the many-parallel-environment CPU regime, single-thread C++ beats 32-thread torch by $\sim$13× **at 8 envs**

Table 7: Per-step scoring latency (ms, median), the heavy factorized $(M, B, 6, 6, 2, 7)$ kernel, CPU. Single-thread C++ beats 32-thread torch at every size. Source: `actionshift/reports/cpp_fusion.md`.

envs	torch (32t)	cpp 1-thread	cpp 8-thread
8	0.186	**0.014**	0.212
64	1.308	**0.056**	0.238
256	1.822	**0.216**	0.261
1024	9.434	0.828	**0.337**

Table 8: RTX 5090 residual-scoring matrix. The transfer-inclusive CUDA path is the headline; kernel-only is diagnostic. Source: `README.md`.

Hyp.	Multicore CPU	CUDA (transfer)	CUDA kernel	Speedup
128	4.358	6.157	5.301	0.71×
1,024	4.105	2.963	2.376	1.39×
8,192	4.377	2.338	1.149	1.87×
65,536	6.571	10.170	0.470	0.65×

and ∼11× at 1024 envs on the heavy kernel (Table 7), by avoiding per-op dispatch and per-mode broadcast temporaries; multi-thread C++ carries a fixed ∼0.2 ms spawn cost so it only pays off at the largest batch. This is stated as *consistent with*, never *beating*, the documented small-batch CPU/GPU crossover envelope. The defensible claim is the CPU scaling regime for future training-time multi-env identification, not a single-GPU eval where scoring is not the bottleneck.

5.5 Systems benchmark: an honest negative on CUDA

Hardware: AMD Ryzen Threadripper PRO 7975WX (32c/64t), RTX 5090, Release, GNU 12.x; ∼10M residual evaluations per point, 5 warmups, 30 measurements.

The result is workload-dependent (Table 8): the expanded matrix reports transfer-inclusive CUDA speedup ranging from 7.133× at a small 2-D batch to 0.580× at 14 dimensions. **CUDA fails the preregistered general 5× transfer-inclusive promotion gate** and remains experimental. Kernel-only timings are reported diagnostically and never substituted for the transfer-inclusive headline; float32 is reported as *inapplicable* (the public scorer is float64-only), not filled with an invented result.

6 Limitations and Claim Boundary

Scale is unidentifiable under `pd_ee_delta_pose` as calibrated—a physical controller/response confound (≈0.6× attenuation, R^2 0.09–0.34), not a grammar limit. Frame is degenerate under the identity-rotation evaluation wrapper (base/tool observationally identical) and under-determined on real data—the three Hub frame flags point in inconsistent directions, evidence that passive diagonal-affine frame identification needs an active on-robot probe. Gripper inversion is unobservable from the tcp-pose response and reported unsupported, not guessed. The six real passive datasets carry no ground-truth latent-contract labels; we make no supervised accuracy claim on them—only documentation-*agreement* scoring, weaker than the labeled-simulation supervised accuracy, which is itself scoped to one task (PickCube), one backbone, and 90 traces. All dynamics evidence is simulated: no hardware validation. The grammar is finite. Kernel-only CUDA is not end-to-end speedup; the promotion verdict is negative.

DO NOT CLAIM. No hardware or real-robot result (ActionABI is sim-only). Not that ActionABI "identifies" real-world action semantics with measured accuracy—its real-data evidence is documentation-*agreement*, a weaker claim. Not a general C++ CUDA speedup—the transfer-inclusive path loses to torch-CUDA at every measured size; only the CPU many-environment regime is a defensible win. Not that scale is identifiable under `pd_ee_delta_pose`. No "first" claims; ORTF is cited descriptively as a commercial, non-peer-reviewed resource, never "beaten."

7 Reproducibility

Every reported number traces to a committed, hash-addressed artifact. The synthetic gate uses frozen seed 20260718 (`sprint_accuracy.py`). The six real datasets are pinned by exact HF revision hash with local Parquet SHA-256 (`run_case_studies.py`; a second run must reproduce a byte-identical `manifest.json`). The labeled real-sim traces are reproducible end-to-end (`export_labeled_traces.py` on the ActionShift side with the frozen PPO backbone, `score_labeled_traces.py` on the ActionABI side with the real C++ `actionabi infer`); C++/Python residual parity is verified to 2.8×10^{-16} for every lag. The 35-dataset Hub audit is reproducible via `run_hub_audit.py` against the revisions frozen in `results/hub_audit.json`. Fresh-verification gates at the time of writing: 8/8 Release CTests, 8/8 ASan/UBSan CTests, CUDA parity pass, 32/32 Python experiment tests. Figures regenerate from the committed report numbers and trace JSON via `media/make_media.py` (Figs. 2 and 4 included).

A Per-dataset audit ledger (35 datasets)

Table 9: Full 35-dataset Hub audit. `dim a/s` = action/state dimension; "CLI" = fixed C++ scorer min-held-out-loss target on square datasets. Source: `reports/lerobot_hub_audit.md`.

Dataset (repo@rev)	Eco	dim a/s	Documented target	ActionABI outcome	CLI	Verdict
Baseline (6 pinned)						
pusht@7628202	gym-pusht	2/2	absolute 2D EE (world)	unique_absolute	—	agreement
aloha_sim_insertion_scripted@8ab6609	ALOHA	14/14	absolute 14-DoF joint	abs/ep-rel equivalence	—	equiv.-consistent
berkeley_autolab_ur5@c4e26a6	OXE-port	7/8	delta EE cartesian	partial_cartesian (frame tool)	—	partial; frame flagged
droid_100@87301a2 [15]	OXE/DROID	7/7	6-DoF EE velocity (base)	report (no unique req.)	—	abstention-consistent
stanford_hydra_dataset@ff06383 [1]	OXE-port	7/8	delta EE (3 pos+3 orient)	report (no unique req.)	—	abstention-consistent
xarm_lift_medium@79efb0e	gym-xarm	4/4	delta EE displ. + gripper	report (no unique req.)	—	abstention-consistent
New Hub audit (29)						
aloha_mobile_cabinet@7a752b3	ALOHA	14/14	absolute (semantic)	abs/ep-rel equivalence	absolute	equiv.-consistent
aloha_sim_insertion_human@cc571a3	ALOHA	14/14	absolute (semantic)	abs/ep-rel equivalence	absolute	equiv.-consistent
aloha_sim_transfer_cube_human@6a43d50	ALOHA	14/14	absolute (semantic)	abs/ep-rel equivalence	absolute	equiv.-consistent
aloha_static_battery@06dc3da	ALOHA	14/14	absolute (motor_N)	report (no unique req.)	absolute	abstention-consistent
aloha_static_coffee@b144896	ALOHA	14/14	absolute (semantic)	report (no unique req.)	absolute	abstention-consistent
aloha_static_cups_open@d793c96	ALOHA	14/14	absolute (semantic)	report (no unique req.)	absolute	abstention-consistent
libero@a1aaacb	LIBERO	7/8	delta (opaque)	partial_cartesian	—	partial-consistent
libero_10@551d7d8	LIBERO	7/8	delta (semantic)	partial_cartesian	—	partial-consistent
metaworld_mt50@a59f742	MetaWorld	4/4	delta (semantic)	report (no unique req.)	delta	abstention-consistent
austin_buds_dataset@d9f9289	OXE-port	7/24	delta (motor_N)	report (no unique req.)	—	abstention-consistent
berkeley_cable_routing@20a7774	OXE-port	7/8	velocity (motor_N)	report (no unique req.)	—	abstention-consistent
berkeley_fanuc_manipulation@51bdefb	OXE-port	7/8	delta (motor_N)	report (no unique req.)	—	abstention-consistent
berkeley_mvp@076135b	OXE-port	8/15	delta (motor_N)	report (no unique req.)	—	abstention-consistent
berkeley_rpt@692eff9	OXE-port	8/8	delta (motor_N)	report (no unique req.)	delta	abstention-consistent
jaco_play@265773f	OXE-port	7/8	delta (motor_N)	partial_cartesian	—	partial-consistent
nyu_door_opening@4ae8986	OXE-port	7/8	velocity (motor_N)	partial_cartesian	—	partial-consistent
nyu_franka_play_dataset@3b2367d	OXE-port	15/13	delta (motor_N)	partial_cartesian	—	partial-consistent
roboturk@d38c919	OXE-port	7/8	delta (motor_N)	partial_cartesian	—	partial-consistent
stanford_kuka_multimodal@93927d1	OXE-port	7/7	None (motor_N)	report (no unique req.)	delta	unlabeled
taco_play@15f69c9	OXE-port	7/7	absolute (motor_N)	report (no unique req.)	delta	abstention-consistent
toto@51f5a8d	OXE-port	7/8	delta (motor_N)	report (no unique req.)	—	abstention-consistent
ucsd_kitchen_dataset@fd7751e	OXE-port	8/21	None (motor_N)	report (no unique req.)	—	unlabeled
ucsd_pick_and_place@5af7ee8	OXE-port	4/7	velocity (motor_N)	report (no unique req.)	—	abstention-consistent
utaustin_mutex@a880eae	OXE-port	7/8	delta (motor_N)	report (no unique req.)	—	abstention-consistent
svla_so100_pickplace@728583b	SO-100/101	6/6	absolute (semantic)	unique_absolute	absolute	agreement
svla_so100_sorting@13870ca	SO-100/101	6/6	absolute (semantic)	unique_absolute	absolute	agreement
svla_so101_pickplace@f641879	SO-100/101	6/6	absolute (semantic)	unique_absolute	absolute	agreement
pusht-subtask@184ff45	gym-pusht	2/2	absolute (motor_N)	unique_absolute	absolute	agreement
xarm_push_medium@50352b1	gym-xarm	3/4	delta (motor_N)	partial_cartesian	—	partial-consistent

References

[1] Suneel Belkhale, Yuchen Cui, and Dorsa Sadigh. HYDRA: Hybrid robot actions for imitation learning, 2023.

[2] Boyuan Chen, Robert Kwiatkowski, Carl Vondrick, and Hod Lipson. Full-body visual self-modeling of robot morphologies. *Science Robotics*, 7(68), 2022.

[3] Cheng Chi, Siyuan Feng, Yilun Du, Zhenjia Xu, Eric Cousineau, Benjamin Burchfiel, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion, 2023.

[4] Weidong Cui, Jayanthkumar Kannan, and Helen J. Wang. Discoverer: Automatic protocol reverse engineering from network traces. In *16th USENIX Security Symposium*, 2007.

[5] Fernando Díaz Ledezma and Sami Haddadin. Machine learning–driven self-discovery of the robot body morphology. *Science Robotics*, 8(85), 2023.

[6] Bradley Efron. Bootstrap methods: Another look at the jackknife. *The Annals of Statistics*, 7(1):1–26, 1979.

[7] Yuchun Feng, Jinliang Zheng, Zhihao Wang, Dongxiu Liu, Jianxiong Li, Jiangmiao Pang, Tai Wang, and Xianyuan Zhan. Demystifying action space design for robotic manipulation policies, 2026.

[8] Aditya Ganapathi, Pete Florence, Jake Varley, Kaylee Burns, Ken Goldberg, and Andy Zeng. Implicit kinematic policies: Unifying joint and cartesian action spaces in end-to-end robot learning. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2022.

[9] Gerra. ORTF v0.2: Open robot training format. `https://www.gerra.com/research/ortf`, 2025. Commercial data-vendor resource, not peer-reviewed.

[10] Yuhang Hu, Yunzhe Wang, Ruibo Liu, Zhou Shen, and Hod Lipson. Reconfigurable robot identification from motion data, 2024.

[11] Peter J. Huber. Robust estimation of a location parameter. *The Annals of Mathematical Statistics*, 35(1):73–101, 1964.

[12] Hugging Face. LeRobot action representations documentation. `https://huggingface.co/docs/lerobot/action_representations`.

[13] Wenzel Jakob, Jason Rhinelander, and Dean Moldovan. pybind11 – seamless operability between C++11 and Python. `https://github.com/pybind/pybind11`, 2017.

[14] Jiayi Jiang, Xiyuan Zhang, Chengcheng Wan, Haoyi Chen, Haiying Sun, and Ting Su. BinPRE: Enhancing field inference in binary analysis based protocol reverse engineering. In *ACM Conference on Computer and Communications Security (CCS)*, 2024.

[15] Alexander Khazatsky et al. DROID: A large-scale in-the-wild robot manipulation dataset, 2024.

[16] Bo Liu, Yifeng Zhu, Chongkai Gao, Yihao Feng, Qiang Liu, Yuke Zhu, and Peter Stone. LIBERO: Benchmarking knowledge transfer for lifelong robot learning. In *Advances in Neural Information Processing Systems (NeurIPS) Datasets and Benchmarks*, 2023.

[17] Open X-Embodiment Collaboration. Open x-embodiment: Robotic learning datasets and RT-X models, 2023.

[18] Jianwei Tai. Same weights, different robot: A deployment safety view of VLA policies, 2026. cs.CR preprint (unreviewed, single-author).

[19] Junhai Yang, Fenghua Li, Yixuan Zhang, Junhao Zhang, Liang Fang, and Yunchuan Guo. Automatic state machine inference for binary protocol reverse engineering, 2024.

[20] Tianhe Yu, Deirdre Quillen, Zhanpeng He, Ryan Julian, Karol Hausman, Chelsea Finn, and Sergey Levine. Meta-world: A benchmark and evaluation for multi-task and meta reinforcement learning. In *Conference on Robot Learning (CoRL)*, 2019.

[21] Tony Z. Zhao, Vikash Kumar, Sergey Levine, and Chelsea Finn. Learning fine-grained bimanual manipulation with low-cost hardware. In *Robotics: Science and Systems (RSS)*, 2023.

[22] Yuke Zhu, Josiah Wong, Ajay Mandlekar, Roberto Martín-Martín, Abhishek Joshi, Soroush Nasiriany, and Yifeng Zhu. robosuite: A modular simulation framework and benchmark for robot learning, 2020.