

MOTIONBENCH: A Goal-Referenced Framework for Evaluating Robot Performance from Trajectory Data

Design, Open-Source Implementation, and Preliminary Experiments

[Student author 1] [Student author 2] [other authors, if any]

Independent researchers [contact email]

<https://motionbench.org> <https://github.com/motionbenchmark/mb-core>

Draft v0.2, October 2026

Abstract

Robot performance is hard to compare across agents, environments and control methods: a student driving a robot, the team’s autonomous code and a commercial product are rarely measured the same way. We propose MOTIONBENCH, a framework that scores any attempt at a task as the share of the task’s stated goal it achieved, computed from the attempt’s recorded trajectory alone, without access to the agent’s software. We describe the scoring model, an open implementation (a deterministic scorer, a trajectory format, phone-video capture tools, and a simulator that runs unmodified VEX AIM Python code), and an evaluation protocol with public and unseen layouts. In a preliminary simulated experiment on a classroom sorting task, the best of four routines on the public layout outscores a parameterized simulated-driver baseline by 1.10 points out of 6 (paired 95% interval 1.00 to 1.25), but on ten unseen layouts its score falls by 1.20 points and it shows no detectable difference from that baseline, while a camera-based routine keeps its score. On synthetic video, tag-free capture places the robot within about 3 cm (median); it has not yet been validated on real footage, and the simulated drivers are not calibrated against real people. These two gaps, and a paired comparison of real drivers with their own code, are the next research objectives.

1 Introduction

Students in robot competitions open every match with a period in which the robot runs its own code, followed by a period of human driving, and they argue about which of the two is better. The question generalizes: how much of a job does a given agent — a person, team-written code, AI-written code, a person with robot assistance, or a commercial robot — actually get done? Answering it fairly is difficult. Match scores mix in opponents and luck; product claims are best cases; and the strongest research benchmarks evaluate learned control policies on standardized robots and scenes [5, 9, 10, 16], which excludes most of the agents above.

MOTIONBENCH starts from the other end: *it fixes the task and its goal, and measures whatever attempts it*. This paper is a framework paper with preliminary feasibility experiments. Its contributions are:

1. a **goal-referenced scoring model** that reports, for every kind of agent, the share of the task’s goal achieved, beside the level reached by chance, with comparisons to people restricted to tasks where people use the same means (Section 3);

Built and released	Shown (preliminary)	Not yet shown
Scorer, trajectory format and task specifications (Apache-2.0 / CC BY 4.0)	The scorer is deterministic and its boards are reproducible from published traces	That task specifications and scoring rules capture what practitioners mean by “done”
Phone-video capture, with and without printed tags	On synthetic video: ≈ 1 cm median error with tags, ≈ 3 cm without	Accuracy on real footage (reflections, blur, occlusion, camera motion)
VEX AIM simulator that runs unmodified routines; parameterized simulated drivers	Public-layout advantage of a routine can vanish on unseen layouts (simulation)	That simulated drivers resemble real students; that simulated gaps match real ones
Protocol: tiers, run counts, unseen layouts, paired verdicts	Rankings by the secondary index are stable under local weight changes (synthetic)	Repeatability of real sessions; a controlled comparison of real drivers and their code

Table 1: Status of the work. All quantitative results in this paper come from simulation, synthetic video, one real capture or one open dataset.

2. an **open implementation** in which the only interface between an agent and the benchmark is its recorded trajectory (Section 4);
3. an **evaluation protocol** with public and unseen layouts, paired comparisons and explicit outcomes for “no difference found” versus “equivalent” (Section 5);
4. a **preliminary experiment** showing that a routine’s advantage on a public layout can disappear on unseen layouts (Section 6).

Table 1 separates what is built, what our experiments show, and what remains a hypothesis.

2 Related work

Benchmarks for robot policies. Meta-World [16], CALVIN [10] and LIBERO [9] evaluate learned manipulation policies in simulation; Habitat [14] does so for embodied navigation; SimplerEnv [8] studies how well simulated evaluation predicts real-robot results. RoboDojo [5] combines simulated and real tasks on standardized bimanual hardware, scores real trials by blinded human evaluators with partial credit, and verifies submissions on hidden layouts. RoboArena [3] distributes real-world evaluation across institutions through pairwise comparisons. These benchmarks standardize the robot and setting, evaluate the policy, and require integration with their software. MOTIONBENCH standardizes the task, evaluates the whole agent (robot, software and any human operator), and uses only the recorded outcome. We adopt from this literature explicit protocols, hidden layouts, repeatability reporting and a separation between verified and unverified results.

Human reference points. Human-normalized scores were popularized by the Arcade Learning Environment [4, 11]. Section 3 explains why MOTIONBENCH moved from normalizing to people to a goal-referenced score.

Reporting under uncertainty. Agarwal et al. [2] show that point estimates over few runs mislead and recommend interval estimates; we use bootstrap intervals [6], paired designs and an equivalence margin.

Task	Platform	Goal	Arena	Budget	People comparable
MB-VEX-NAV@1.0	VEX V5	6 waypoints, 5 cm / 5°	3.66 m square	20 s	yes
MB-VEX-PICK@1.0	VEX V5	deliver 6 objects	3.66 m square	60 s	yes
MB-AIM-NAV@0.1	VEX AIM	5 waypoints, 3 cm / 10°	1.83 × 1.22 m	20 s	yes
MB-AIM-SORT@0.1	VEX AIM	sort 6 barrels by colour	1.83 × 1.22 m	45 s	yes
MB-COVER@0.1	floor robots	cover free floor, 30 cm tool	4 × 3 m	180 s	no
MB-FIELD-HALLWAY@0.1	robot vacuum	cover visible floor	1.3 × 3.0 m	137 s	no

Table 2: Current tasks (specification 0.2). “People comparable”: a person can do the same job with the same means, for example by driving the same robot.

Product test methods. IEC 62929 [1] specifies laboratory methods for household cleaning robots. MOTIONBENCH is complementary: it measures robots where they are used, from ordinary video.

3 Scoring model

Tasks. A task specifies an arena (size, obstacles, zones), the robot footprint, a time budget, public layouts, layout rules for generating unseen ones, and a goal of one of three kinds: *waypoints* (reach each in order within position and heading tolerances), *delivery* (each object rests in its target zone at the end) or *coverage* (share of free floor swept by a tool of given width). Table 2 lists the current tasks; they use competition hardware but never copy a season’s game.

Versions. Three version numbers appear and are independent. A *task* carries its own version after the @ in its name; the two VEX V5 tasks are at 1.0 because their field and rules are settled, the others at 0.1 because they are experimental. The *specification* (0.2) versions the trace and board formats and the scoring rules. The *scorer* (0.1.0) versions the software. Every report records all three, and results under an older specification remain valid under that specification.

Primary outcome: goal score. For a run with success S in task units and maximum $S_{\max}$, the goal score is $g = S/S_{\max} \in [0, 1]$. For each agent we report the mean over its runs with a 95% bootstrap interval, the worst run, and the *completion rate* (share of runs that achieved the whole goal). Time to goal is reported over runs that achieved it.

Chance. Each task reports the goal score of a randomly moving robot beside the agents’ scores, without subtracting it. On coverage, a random walk already covers about half a room; readers should see that rather than have it absorbed into a transformation.

Comparison with people (optional). Our first specification normalized every agent to the median person on the task, following [11]. That turned differences in equipment into apparent differences in skill when means differed (a robot vacuum against a person with a stick vacuum), left tasks without a human baseline unscored, and saturated where most people finish. Specification 0.2 reports $\text{vs_people} = (S - S_{\text{chance}})/(S_{\text{people}} - S_{\text{chance}})$ only on tasks declared people-comparable, only when people’s median beats chance by at least 10% of $S_{\max}$ (otherwise the ratio is unstable and is not reported), and never as the headline.

Secondary measures and an experimental index. Boards also report path efficiency e (shortest sensible path over path driven, clamped to $[0, 1]$), consistency $c = \text{clamp}(1 - \sigma_S/\mu_S, 0, 1)$ with $c = 0$ when $\mu_S = 0$ and σ_S the sample standard deviation of success over runs, and safety $s = 1/(1 + \bar{v}) \in (0, 1]$ with $\bar{v}$ the mean number of contact or rule violations per run. A single index ranks agents within one task:

$$\text{MBI} = 0.50g + 0.20e + 0.15c + 0.15s \in [0, 1].$$

The weights are a design choice, not an estimate; we treat the index as experimental and secondary to g , never average it across tasks, and report its sensitivity to the weights (Section 7).

Ranks. For display, agents share a rank while their intervals overlap the leader’s. This is a presentation convention, not a test of equivalence.

4 Implementation

Trajectory traces and the scorer. A run is a directory with a manifest (task, layout, agent and its class, capture method, tier, file hashes), timestamped poses (t, x, y, θ) , optional events and the final positions of objects. This is the only interface between an agent and the benchmark, so closed products can be evaluated from video alone. The scorer is implemented once in Rust and called from Python and the command line; it validates traces (hashes, monotonic time, sampling gaps), computes run reports and builds boards. An independent browser page recomputes waypoints, deliveries and coverage from raw poses and flags disagreements. *Determinism is not accuracy*: the scorer returning the same report for the same trace says nothing about whether the trace, the task specification or the scoring rules are right, which is why capture accuracy, task design and the protocol are examined separately.

Capture. With printed AprilTags [12] in the arena corners and on the robot, a per-frame homography maps the image to the floor. Without tags, frames of a fixed or hand-held camera are aligned to the first frame; the per-pixel median gives an empty-floor image and the robot is the region that differs from it. The floor plane comes from four image points outlining a floor rectangle [7]; focal length and scale come from a measured width and length, or from the robot’s known size by searching for the camera parameters at which its silhouettes match. The robot’s position is taken half a robot behind the lowest point of its silhouette; floor the camera cannot see is excluded from coverage and safety.

Simulation: games and drivers. A *game* is defined once (field, zones, wall-mounted fiducials, objects, layouts, layout rules, match length, scoring) and then simulated many times; a game may link to a task with the same layouts, in which case simulated runs are also scored by the standard scorer and the two scores must agree. The first platform is VEX AIM, a small holonomic classroom robot with a magnetic pick-up, a kicker and a camera. The simulator runs VEXcode AIM Python files unmodified: the import header receives stand-ins for the robot, every API call advances simulated time, speeds follow the published API limits (200 mm/s, 150°/s), and each run draws its own wheel slip, drift, turn overshoot, kick scatter and camera noise. The *parameterized simulated-driver baseline* consists of controller policies with explicit traits (reaction time with partial anticipation, share of full speed used, steering shake, placement precision, choice of next object, mistake and hesitation rates); six named characters span a plausible classroom, and the *typical*

simulated driver is the median member of a simulated class of twenty. These traits are chosen to be plausible; they are not fitted to real students.

5 Evaluation protocol

Tiers. Results are *simulated* (S), *field* (F: a real robot captured and submitted by its owner, reviewed before publication) or *verified* (V). They are never mixed without labels.

Runs and layouts. An agent needs at least five runs on a task to be ranked (ten recommended); runs are consecutive attempts and every attempt is reported. Public layouts are published; unseen layouts are generated from each task’s layout rules with an unpublished seed.

Comparisons. When two agents are compared in simulation, run i of each uses the same seed (the same layout and the same draw of robot imperfections), and the comparison is the mean of paired differences with a 95% bootstrap interval. The resampling unit is the paired run on a single layout; across unseen layouts the bootstrap is hierarchical (layouts, then runs within each chosen layout), so intervals reflect variation across layouts. Outcomes are *better* or *worse* when the interval excludes zero, *equivalent* when it lies within a margin of ± 0.5 points, and *no difference found* otherwise. No difference found does not mean equivalent.

Verified runs. A verified session requires a layout sent at the start; one uncut video from a fixed camera showing the arena, a code word sent at the start and every run; a trace from tags or from the converter run by a reviewer; all attempts reported; the agent described (for code, the routine’s version, shared privately on request); and a second reviewer re-scoring at least one run from raw video. Reviewers are project maintainers or volunteers who have re-scored three reference runs within tolerance; they declare conflicts and may not review a team, school, family member or product maker they are connected to. If the two reviewers disagree, a third decides, and reviewer identities are recorded with the result.

Disputes, versions, minors. Disputes are re-scored by a reviewer who did not score the original; every change to a published result is logged with versions and date. Published results use trajectories, not footage; videos showing people are not published without permission (for minors, a parent’s or guardian’s).

6 Preliminary experiment: does a public-layout advantage generalize?

Setup. One VEX AIM robot must sort three orange and three blue barrels into the zone of their colour on a $1.83\text{ m} \times 1.22\text{ m}$ table within 45 s (one point per barrel; six maximum). Four routines, written for the public layout, span common approaches: *dead reckoning* (a fixed plan, no sensors); *vision* (the camera finds the nearest barrel, wheel odometry returns it to the zone); *vision + tags* (as vision, then homing on the AprilTag behind each zone); and *planner* (the barrel positions of the public layout, an efficient order, full speed, the camera only for the last few centimetres, and spread drop positions). Each routine, each of the six driver characters and the typical simulated driver play 20 paired runs on the public layout, and 20 paired runs spread over ten unseen layouts (same start, barrels placed at random away from the zones, the start and each other).

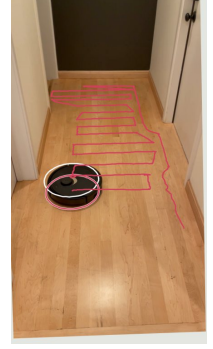
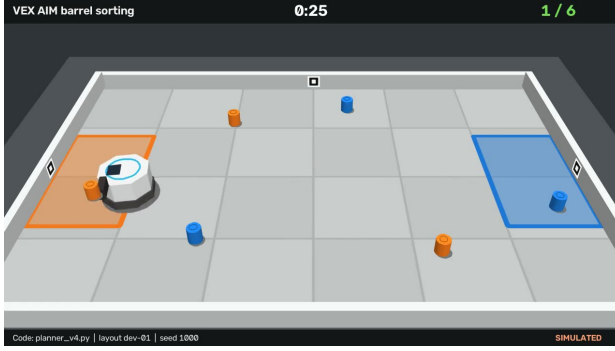


Figure 1: Left: the simulated AIM sorting game (labelled as simulation in all outputs). Right: the tag-free capture of a household robot vacuum from one hand-held phone video (Section 7).

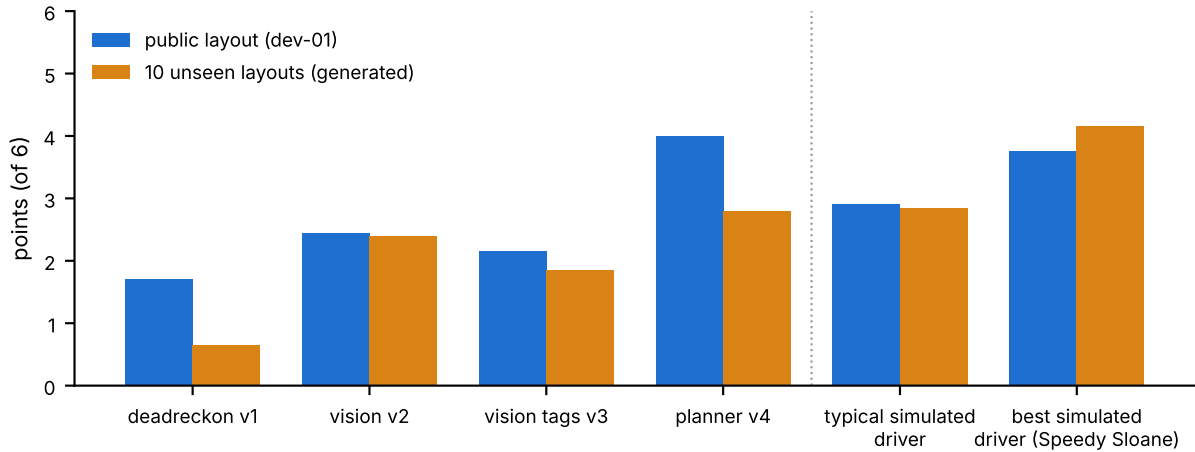


Figure 2: Simulated AIM sorting: mean points on the public layout and on ten unseen layouts (20 paired runs per agent). The drivers are a parameterized, uncalibrated baseline.

Results. Figure 2 and Table 3 summarize the outcome. On the public layout the planner scores 4.00 points, ahead of the typical simulated driver (2.90) by a paired difference of +1.10 (95% interval +1.00 to +1.25), and ahead of every driver character, including the two strongest (+0.25 and +0.50). On unseen layouts it scores 2.80: a drop of 1.20 points (−1.60 to −0.85), no detectable difference from the typical simulated driver (−0.05; −0.65 to +0.60), and below the two strongest characters (−1.35 and −1.00). The vision routine scores 2.45 on the public layout and 2.40 on unseen layouts, a change that is equivalent to zero within the margin (−0.05; −0.50 to +0.40). The dead-reckoning routine, which also encodes the public layout, drops by 1.05.

Interpretation. The planner’s public-layout advantage comes largely from knowing where the barrels are. This conclusion rests on a comparison of the routine with itself across layouts and does not depend on the simulated drivers; the comparisons with drivers additionally depend on the uncalibrated baseline and should be read as “outperforms a parameterized simulated-driver baseline”, not as a statement about real students. Two further observations would plausibly carry over to real robots: the tag-homing routine loses time searching for the tag, and routines that drop every barrel at the same spot pick up the barrel already there. At AIM’s top speed no agent sorts

Routine	Public layout			Unseen layouts		
	Mean	vs typical driver		Mean	vs typical driver	
Dead reckoning	1.70	-1.20	$[-1.50, -0.90]$ worse	0.65	-2.20	$[-2.75, -1.55]$ worse
Vision	2.45	-0.45	$[-0.80, -0.10]$ worse	2.40	-0.45	$[-1.05, +0.20]$ n.d.f.
Vision + tags	2.15	-0.75	$[-1.20, -0.35]$ worse	1.85	-1.00	$[-1.50, -0.50]$ worse
Planner	4.00	+1.10	$[+1.00, +1.25]$ better	2.80	-0.05	$[-0.65, +0.60]$ n.d.f.
Typical simulated driver	2.90			2.85		

Table 3: Paired differences in points (of 6) with 95% bootstrap intervals; n.d.f. = no difference found. Values are exact to two decimals; full results, including comparisons with each driver character, are in `results/aim-sort-verdict-v0.1.json`. The typical driver’s paired runs average 2.90; the class median of per-driver means, used only to select that driver, is 2.83.

Capture	Scale from	Median error	90th percentile	Frames without position
Printed tags	tag geometry	1.2 cm	1.5 cm	0%
No tags	measured floor	3.2 cm	5.2 cm	8.2%
No tags	robot size only	3.2 cm	5.4 cm	8.2%

Table 4: Capture accuracy on rendered video with known ground truth (medians over three render seeds; `paper/analysis/capture_accuracy.py`).

all six barrels in 45 s (best runs score 4 or 5), which suggests the task’s budget should be revisited.

7 Feasibility demonstrations

Capture accuracy on synthetic video. Table 4 reports per-frame errors over three render seeds. With tags the robot is found in every frame with a median error of about 1 cm; without tags the median is about 3 cm whether scale comes from a measured floor or from the robot’s size, with about 8% of frames left without a position. Synthetic video lacks reflections, blur, occlusion and lighting changes; real-world validation is the first planned pilot.

A real robot vacuum. A household robot vacuum was filmed for 137 s with a hand-held phone, without tags; scale came from the robot’s size and unseen floor was excluded. It covered 85% of 3.24 m² of visible floor with a 30 cm cleaning width, driving 18.6 m. This demonstrates the pipeline end to end on real footage; without ground truth it does not establish accuracy.

People in an open dataset. We scored 120 motion-capture walks along one route (about 7 m) from THÖR-MAGNI [15]. All 120 reached the goal (median 6.8 s), so the goal score could not separate walkers and secondary measures did; with a 60 s budget a random walk reached the goal in 40% of runs, showing that budgets should follow typical completion times.

Index sensitivity. On the synthetic boards whose simulated agents respect their robots’ limits (two VEX V5 tasks and room coverage), we recomputed the index for 5,000 weight vectors drawn near the defaults (Dirichlet with concentration 50 w) and uniformly from the simplex (Table 5). Under local changes, rankings are stable (median Kendall’s $\tau \geq 0.94$) and the top agent almost

Task	Agents	Local weights		Uniform weights	Goal only
		median τ	top agent same	median τ (5th pct.)	τ
MB-VEX-NAV@1.0	19	0.99	100%	0.95 (0.88)	0.91
MB-VEX-PICK@1.0	19	0.99	100%	0.90 (0.74)	0.81
MB-COVER@0.1	9	0.94	99.6%	0.83 (0.33)	0.83

Table 5: Kendall’s τ between the default index ranking and rankings under perturbed weights (synthetic agents; `paper/analysis/index_sensitivity.py`).

never changes; under arbitrary weights the coverage ranking is the least stable. Ranking by goal score alone agrees with the index at $\tau = 0.81$ to 0.91 .

Extension: multi-robot matches. The toolkit also includes a seeded 2-versus-2 match simulator, a renderer of broadcast-style video from a camera position measured on a real event broadcast, a per-robot report and a tracking benchmark. A baseline tracker finds 90–97% of robot positions with a median error of 7–9 cm on three simulated matches but switches identities 25–44 times per match (IDF1 [13] 48–66%). We present this as infrastructure for future work.

8 Research roadmap

Three pilots would test the hypotheses in Table 1. Each needs equipment or participants we do not have and is designed so that a host lab would also obtain a research output.

1. **Real-world capture validation.** The same runs recorded with tags and without, against a calibrated reference (for example motion capture), across rooms, lighting and camera placements; report median and 90th-percentile error and failure rates.
2. **Real drivers versus their own code.** Student drivers and their teams’ routines on the same robot and layouts, paired by layout, at least ten runs per agent, on public and unseen layouts. Involving minors requires parental consent, the host institution’s youth-protection rules and an ethics/IRB determination.
3. **Simulation to reality.** The routines of Section 6 on a real AIM robot and table with the same layouts; report the simulator’s prediction error per routine, then fit the driver baseline to filmed runs and re-check the verdicts.

9 Limitations

Most results are simulated and encode our assumptions (robot size, accelerations, camera field of view, friction, every driver trait). Simulated agents are less variable than real ones — some score the same in every run — so intervals are likely narrower than they will be on real robots. The synthetic VEX AIM boards published with specification 0.2 contain simulated agents faster than the AIM robot’s top speed; they are marked invalid on the project site and excluded from this paper, pending regeneration. Tag-free capture needs a visible floor rectangle and has been validated only on synthetic video. Coverage scores model cleaning rather than measure it. Goal scores depend on how goals and budgets are set, which is why tasks and protocol change only through public RFCs.

10 Conclusion

MOTIONBENCH proposes measuring how much of a task’s goal any agent achieves, from its trajectory alone, with an open scorer and an explicit protocol. The framework and tools are built and released; a preliminary simulated experiment shows why unseen layouts belong in any verdict. Whether the method measures real robots and real people accurately is the open question, and the roadmap above is how we intend to answer it. Comments are welcome at <https://motionbench.org/paper>.

Author contributions

[Describe who did what, for example using the CRediT roles: conceptualization; software (scorer, capture, simulator); experiments; writing; supervision. State the contribution of each student author and of any adult mentor.]

Use of AI tools

[Describe accurately how AI assistance was used, for example: “Parts of the software and of this manuscript were drafted with an AI coding assistant under the authors’ direction; the authors designed the study, reviewed and tested all code, checked every reported number against the released results, and take responsibility for the content.”]

Ethics, data and code availability

No new data from human participants were collected for this paper; the human trajectories come from a published dataset [15]. The robot-vacuum video was recorded by the authors in a private home. Footage from a public event broadcast was used only to measure camera geometry and is not redistributed. Code (Apache-2.0), task specifications and data (CC BY 4.0) are at <https://github.com/motionbenchmark/mb-core>; `paper/reproduce.sh` regenerates every number and figure (Appendix A).

References

- [1] IEC 62929: Cleaning robots for household use – dry cleaning: Methods of measuring performance. Technical report, International Electrotechnical Commission, 2014.
- [2] Rishabh Agarwal, Max Schwarzer, Pablo Samuel Castro, Aaron Courville, and Marc G. Bellemare. Deep reinforcement learning at the edge of the statistical precipice. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021. arXiv:2108.13264.
- [3] Pranav Atreya et al. RoboArena: Distributed real-world evaluation of generalist robot policies. In *Proceedings of the Conference on Robot Learning (CoRL)*, volume 305 of *PMLR*, 2025. arXiv:2506.18123.
- [4] Marc G. Bellemare, Yavar Naddaf, Joel Veness, and Michael Bowling. The arcade learning environment: An evaluation platform for general agents. *Journal of Artificial Intelligence Research*, 47:253–279, 2013.

- [5] Tianxing Chen, Yue Chen, Zixuan Li, et al. RoboDojo: A unified sim-and-real benchmark for comprehensive evaluation of generalist robot manipulation policies. *arXiv preprint arXiv:2607.04434*, 2026.
- [6] Bradley Efron. Bootstrap methods: Another look at the jackknife. *The Annals of Statistics*, 7(1):1–26, 1979.
- [7] Richard Hartley and Andrew Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2 edition, 2004.
- [8] Xuanlin Li et al. Evaluating real-world robot manipulation policies in simulation. In *Proceedings of the Conference on Robot Learning (CoRL)*, volume 270 of *PMLR*, 2024. arXiv:2405.05941.
- [9] 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 Track*, 2023. arXiv:2306.03310.
- [10] Oier Mees, Lukas Hermann, Erick Rosete-Beas, and Wolfram Burgard. CALVIN: A benchmark for language-conditioned policy learning for long-horizon robot manipulation tasks. *IEEE Robotics and Automation Letters*, 2022. arXiv:2112.03227.
- [11] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, et al. Human-level control through deep reinforcement learning. *Nature*, 518:529–533, 2015.
- [12] Edwin Olson. AprilTag: A robust and flexible visual fiducial system. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2011.
- [13] Ergys Ristani, Francesco Solera, Roger Zou, Rita Cucchiara, and Carlo Tomasi. Performance measures and a data set for multi-target, multi-camera tracking. In *European Conference on Computer Vision (ECCV) Workshops*, 2016.
- [14] Manolis Savva, Abhishek Kadian, Oleksandr Maksymets, Yili Zhao, Erik Wijmans, Bhavana Jain, Julian Straub, Jia Liu, Vladlen Koltun, Jitendra Malik, Devi Parikh, and Dhruv Batra. Habitat: A platform for embodied AI research. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019. arXiv:1904.01201.
- [15] Tim Schreiter et al. THÖR-MAGNI: A large-scale indoor motion capture recording of human movement and robot interaction, 2024. arXiv:2403.09285.
- [16] 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 *Proceedings of the Conference on Robot Learning (CoRL)*, 2019. arXiv:1910.10897.

A Reproducing the results

Environment used: Python 3.13, NumPy 2.5, SciPy 1.18, OpenCV 5.0, FFmpeg 6.1, Rust 1.97, motionbench 0.1.0 (specification 0.2); exact versions are written to `paper/results/environment.txt` on each run.

```
git clone https://github.com/motionbenchmark/mb-core && cd mb-core
pip install maturin
(cd python && maturin build --release)
pip install "target/wheels/motionbench-*.whl[tracker]"
bash paper/reproduce.sh      # ~5 min; --full adds the match benchmark
```

Expected outputs: the synthetic board is byte-identical to `results/leaderboard-synthetic-v0.1.json`; `paper/results/aim-sort-verdict.json` reproduces Table 3; `index-sensitivity.json` reproduces Table 5; `capture-accuracy.json` reproduces Table 4 (small differences are possible across OpenCV/FFmpeg versions). SHA-256 checksums of the deterministic outputs are in `paper/results/SHA256SUMS` and are checked by the script.

B Index definition: edge cases

g uses the task’s maximum score; e is clamped to $[0, 1]$; $c = 0$ when mean success is zero, and $c = 1 - \sigma_S / \mu_S$ clamped to $[0, 1]$ otherwise, with σ_S the sample standard deviation ($\sigma_S = 0$ for a single run, which is why agents with fewer than five runs are unranked); $s = 1 / (1 + \bar{v})$ lies in $(0, 1]$. All four terms therefore lie in $[0, 1]$, and so does the index.