KAGGLEYEs ACADEMY E-PRINT REGISTRY
ARCHIVAL PREPRINT / OPEN SCHOLARLY RECORD

KAGG RECORD
KAGG:2610.00001

A Unified Evaluation Harness for Quadruped Navigation across Interfaces, Terrain, and Sensors

Marcus Hill • **Rachel Kim** • **Yating Feng** • **Ivan Sokolov**

Robotics Institute, Georgia Institute of Technology, Atlanta, GA, USA

PUBLISHED	REGISTERED	RECORD TYPE
October 1, 2026	October 6, 2026	Preprint

ABSTRACT

Reported performance of quadruped navigation planners is rarely comparable across papers, because each evaluation couples the planner with a particular velocity interface, terrain definition, and sensor rig. This paper presents a unified evaluation harness that isolates these three axes. The interface axis spans an ideal velocity tracker, a calibrated-opaque interface identified by planner-conditioned active calibration, and two native platform interfaces; the terrain axis comprises six standardized courses (flat, mixed, stairs, narrow, cluttered, rough); and the sensor axis comprises LiDAR-only, LiDAR+vision, and degraded rigs. A fixed set of four planners, with a configuration-driven cross-embodiment planner as the reference stack, is executed on Go2-class, B2-class, and ANYmal-class assets, and every run emits a standardized 214-field results record. Across 6480 standardized runs, the harness reaches 95.7% cross-platform run parity and 100.0% record schema completeness at 2.3 s of overhead per run and measures nine confounding factors, compared with 82.9% parity for raw-score ranking and 87.0% when only the interface axis is held fixed, both of which measure eight. Ablations show that removing the interface, terrain, or sensor-rig axis lowers parity to 74.5%, 80.3%, and 86.4%, respectively. A case study shows that a planner ranked first on raw scores falls to third once the interface axis is held fixed, because its advantage stemmed from a favorable interface on one platform. Recorded failure cases, including sanity-check quarantines and gait-mode switching delays, are reported as harness observations rather than discarded.

INDEX TERMS quadruped navigation, evaluation harness, velocity interface calibration, benchmarking, cross-embodiment local planning, reproducibility

KAGG NOTE A ranking of quadruped navigation planners is only interpretable when the velocity interface, terrain course, and sensor rig are held fixed and recorded as explicit axes; otherwise the ranking can reflect interface luck rather than planning quality.

01 Introduction

Quadruped robots have moved from laboratory demonstrations of robust locomotion [1, 2, 3] to field deployments in which a navigation stack must choose where the robot goes, not only how it places its feet [4, 5]. As the number of navigation methods for legged platforms has grown, so has the number of published comparisons among them. Those comparisons, however, are difficult to interpret. One paper evaluates on a small quadruped through the vendor's high-level velocity interface, another on a larger industrial platform through a different locomotion manager, a third in simulation with an ideal velocity tracker; the terrain is called "rough" in all three but means different step heights and obstacle densities; and the sensing ranges from a single LiDAR to a fused LiDAR and camera rig. When a planner is reported to outperform another, it is rarely possible to say whether the margin is due to the planner, the interface, the course, or the sensors.

The velocity interface is the least visible of these factors and arguably the most consequential. A navigation planner emits body-frame twist commands; the locomotion controller beneath it converts those commands into realized motion with its own gain, latency, saturation, and gait-mode logic. Learned locomotion controllers in particular are trained to track commands only approximately [6, 7], and the resulting command-to-motion map differs from platform to platform. Recent work on planner-conditioned active calibration, GAUGE [8], shows that this map can be identified online by treating the controller as an opaque interface and probing it with commands drawn from the planner's own operating regime. Configuration-driven planning, exemplified by AgniNav [9], shows that a single local planner can be deployed across embodiments by exposing body geometry, sensors, and velocity limits as an abstract configuration. Neither line of work, however, addresses the evaluation question directly: how should a comparison among planners be structured so that interface, terrain, and sensor differences are measured rather than silently absorbed into planner scores?

This paper presents a unified evaluation harness that answers this question by making each confounding source an explicit, controlled axis. The harness evaluates a fixed set of planners on three quadruped platforms, varies the velocity interface, the terrain course, and the sensor rig independently, and writes every run to a standardized results record whose schema is checked automatically. Runs that fail a sanity check are quarantined rather than dropped, and platform-specific events such as gait-mode switches are recorded as interface-axis observations rather than as planner failures. The contributions are as follows:

1. A formulation of cross-platform planner comparison in which the velocity interface, terrain course, and sensor rig are separate axes, together with two record-level quality

KAGGLEYEs ACADEMY • E-PRINT REGISTRY • KAGG:2610.00001

measures, cross-platform run parity and schema completeness, that quantify whether a comparison is commensurable (Section 3).

2. An evaluation harness that implements the three axes: a parametric interface layer with ideal, calibrated-opaque (GAUGE-identified), and native platform variants; six standardized terrain courses; and three sensor rigs including a degraded rig, with a 214-field results record and a quarantine-based sanity check (Section 4).
3. A measurement of the confounding factors present on Go2-class, B2-class, and ANYmal-class assets over 6480 standardized runs, with an ablation that attributes the loss of comparability to each axis (Section 5).
4. A case study in which a planner ranked first on raw cross-platform scores falls to third once the interface axis is held fixed, together with a discussion of the recorded failure cases (Sections 5.4 and 5.5).

The harness is not a new planner and does not claim to improve navigation performance. Its purpose is to make published-style comparisons reproducible and to expose when a reported ranking depends on factors other than the planner itself.

02 Related Work

2.1 Legged locomotion as the layer beneath navigation

Reinforcement-learned locomotion controllers now provide robust blind and perceptive walking on quadrupeds. Hwangbo et al. [1] trained policies with learned actuator models on ANYmal, Lee et al. [2] extended this to challenging natural terrain with a teacher-student scheme, and Miki et al. [3] fused exteroception and proprioception for robust perceptive locomotion in the wild. Massively parallel training [6, 10] made such controllers inexpensive to produce, and adaptation modules such as RMA [11] and egocentric-vision policies [12] widened the range of terrain they handle. Agile skills such as parkour [13, 14, 15] blur the boundary between locomotion and navigation. Model-based alternatives, such as perceptive nonlinear model predictive control [16], remain common on industrial platforms, and the survey by Ha et al. [17] reviews both families. For navigation evaluation, the key observation is that all of these controllers expose a commanded-velocity interface whose tracking behavior is a design outcome rather than a specification: Walk These Ways [7] explicitly exposes gait parameters that change how a velocity command is realized. The harness therefore treats the locomotion controller as part of the interface axis instead of as a fixed property of the robot.

2.2 Opaque velocity interfaces and their calibration

GAUGE [8] addresses the fact that a quadruped's velocity controller is usually an opaque interface: the navigation stack can send commands and observe the resulting motion but cannot inspect the controller. GAUGE issues planner-conditioned probe commands, that is, excitation drawn from the distribution of commands the planner actually produces, and fits a calibrated model of the command-to-motion response online. Its objective is better closed-loop navigation on a given robot. The present work uses the same idea for a different purpose. The calibrated-opaque level of the interface axis is obtained with a GAUGE-style identification, but the identified model is used to present every platform to the planners through a common reference response, so that differences in interface behavior are held fixed and can be measured separately. GAUGE thus supplies the mechanism that makes "holding the interface fixed" operational on platforms whose controllers cannot be modified. Classical sim-to-real work addressed related mismatches through dynamics randomization [18, 19] and identified actuator models [1, 20]; those methods change the policy so that it tolerates mismatch, whereas the harness records the mismatch as a measured factor.

2.3 Cross-embodiment and configuration-driven local planning

AgniNav [9] is a configuration-driven local planner: rather than retraining or re-engineering the planner for each robot, it reads an abstract robot configuration describing body geometry, sensor placement, and velocity limits, and uses that configuration to parameterize collision checking and command generation so that one planning stack runs on different embodiments. We adopt AgniNav as the reference stack of the harness for exactly this reason: because its embodiment dependence is confined to a declared configuration, any residual cross-platform variation in its results can be attributed to the axes the harness controls rather than to per-platform tuning. CeRLP [21] pursues a related goal for visual navigation by converting heterogeneous observations into a unified geometric representation before local planning. Learned navigation approaches also address embodiment: Truong et al. [22] learn navigation skills for legged robots with learned embodiment models, Fu et al. [23] couple vision and proprioception for legged navigation, Rudin et al. [24] learn locomotion and local navigation end-to-end, and ArtPlanner [5] combines sampling-based planning with learned reachability for field deployment. General navigation models such as GNM [25], ViNT [26], and NoMaD [27] train across many robots with a normalized action space. These works aim to transfer a method across embodiments; the harness instead aims to evaluate methods across embodiments without letting embodiment-specific interfaces bias the comparison. The two goals are complementary, and the configuration abstraction of AgniNav is what allows the reference stack to be held constant across platforms.

2.4 Terrain and sensor representation

Legged navigation stacks typically consume a robot-centric elevation map [28] built from LiDAR or depth sensing, often with LiDAR-inertial odometry such as FAST-LIO2 [29]. Self-supervised traversability estimation [30, 31] adds a visual channel that distinguishes geometrically similar but physically different terrain. Field systems such as CERBERUS [4] show that sensor degradation, from dust, darkness, or reflective surfaces, is a regular operating condition rather than an exception. These works motivate the sensor axis of the harness: rigs differ in modality and in effective range, and both must be fixed and recorded for scores to be comparable.

2.5 Benchmarks and evaluation methodology

Embodied-AI platforms such as Habitat [32] standardized simulated navigation evaluation, and Anderson et al. [33] proposed shared metrics such as success weighted by path length. Yokoyama et al. [34] argued that such metrics ignore dynamics and introduced success weighted by completion time, a point that is sharper for legged robots whose achievable

KAGGLEYEs ACADEMY • E-PRINT REGISTRY • KAGG:2610.00001

speed depends on the interface. Kadian et al. [35] showed that simulation rankings may fail to predict real-world rankings unless the simulator is tuned for predictivity. For ground robots, BARN [36] provides standardized obstacle courses of graded difficulty, and the BARN Challenge [37] documented how differences in deployment details affected outcomes; Xiao et al. [38] survey learned navigation more broadly. Orbit [39] and Isaac Gym [10] supply the simulation substrate on which harnesses of this kind can be built. Outside robotics, Henderson et al. [40] showed that implementation details and seeds can dominate reported differences between reinforcement learning methods. The present harness applies the same concern to quadruped navigation and identifies the velocity interface as a confound that existing benchmarks for wheeled or simulated agents do not need to model.

03 Problem Formulation

Let $\mathcal{P}$ denote the set of platforms, Π the fixed set of planners, $\mathcal{I}$ the set of velocity-interface levels, $\mathcal{C}$ the set of terrain courses, and $\mathcal{S}$ the set of sensor rigs. In this study $|\mathcal{P}| = 3$, $|\Pi| = 4$, $|\mathcal{C}| = 6$, and $|\mathcal{S}| = 3$, and $\mathcal{I} = \{\iota_{\text{id}}, \iota_{\text{cal}}, \iota_{\text{A}}, \iota_{\text{B}}\}$ contains the ideal, calibrated-opaque, and two native platform interfaces. A run is indexed by the tuple

$$\rho = (\pi, p, \iota, c, s, \sigma) \in \Pi \times \mathcal{P} \times \mathcal{I} \times \mathcal{C} \times \mathcal{S} \times \Sigma, \qquad (1)$$

where σ is a seed that fixes the start pose, goal sequence, and stochastic disturbances. During a run the planner observes a local map M_t and goal g and emits a commanded body twist $u_t^* = (v_x, v_y, \omega_z) \in \mathbb{R}^3$. The platform realizes a twist ν_t through the interface map

$$\nu_{0:T} = \Phi_{p,\iota}\left(u^*_{0:T}\right), \qquad (2)$$

which is a causal, possibly stochastic operator that depends on both the platform and the interface level. Each run produces a results record $r(\rho) \in \mathcal{R}$ with $F = 214$ fields, which include the outcome metrics, the axis descriptors, interface observations, and sanity flags.

Scores and rankings. For an outcome metric m, the score of planner π under a condition is the mean over seeds, $S(\pi \mid p, \iota, c, s) = \frac{1}{N}\sum_{\sigma} m(r(\rho))$. A published-style raw ranking aggregates over platforms using each platform's native conventions,

$$\bar{S}_{\text{raw}}(\pi) = \frac{1}{|\mathcal{P}|}\sum_{p\in\mathcal{P}} S\left(\pi \mid p, \iota_p^{\text{nat}}, c_p, s_p\right), \qquad (3)$$

where ι_p^{nat}, c_p, and s_p are the interface, course definition, and rig that happen to be used on platform p. A controlled ranking instead holds the axes fixed and averages over the full design,

$$\bar{S}_{\text{ctrl}}(\pi) = \frac{1}{|\mathcal{P}||\mathcal{C}||\mathcal{S}|}\sum_{p,c,s} S\left(\pi \mid p, \iota_{\text{cal}}, c, s\right). \qquad (4)$$

A *confounding factor* is a variable z that differs across platforms under the raw protocol and affects m; it is *measured* if the record contains an observable from which its value on every run can be recovered.

Commensurability. A comparison is meaningful only if the records being compared describe the same experiment up to the platform. Let $t = (\pi, \iota, c, s, \sigma)$ be a matched tuple and $\chi : \mathcal{R} \times \mathcal{R} \to \{0, 1\}$ a consistency predicate that checks that two records share the course specification, goal set, rig descriptor, interface normalization, and the mandatory fields. Cross-platform run parity is

$$\text{Par} = \frac{1}{|\mathcal{T}|}\sum_{t\in\mathcal{T}} \prod_{p,p'\in\mathcal{P}} \chi\left(r(t,p), r(t,p')\right), \qquad (5)$$

the fraction of matched tuples whose records are pairwise consistent across all platforms. Record schema completeness is the mean fraction of fields that are present and valid,

$$\text{Comp} = \frac{1}{|\mathcal{R}|}\sum_{r\in\mathcal{R}} \frac{1}{F}\sum_{f=1}^{F} \mathbb{1}\left[r_f \text{ present and valid}\right]. \qquad (6)$$

The design problem is to construct $\Phi_{p,\iota}$, the course and rig definitions, and the record so that Par and Comp are high, the set of measured confounding factors is as large as possible, and the per-run overhead of the harness remains small relative to the run itself.

04 Harness Design

4.1 Architecture

The harness wraps each run in four stages. A *scenario compiler* expands a tuple ρ into a concrete course instance, rig configuration, and interface configuration, and hashes each specification. An *interface layer* sits between the planner and the platform's locomotion controller and implements $\Phi_{p,\iota}$. A *runner* executes the episode, with the locomotion controller and interface layer stepping at a command period of $\Delta t = 20$ ms (50 Hz) and the planner at 10 Hz with zero-order hold onto the command stream. A *recorder* assembles the results record, validates it against the schema, and applies the sanity checks. The planners never see which interface level, course, or rig is active beyond what their own inputs reveal, and every planner receives the same map representation and goal sequence for a given ρ.

4.2 Interface axis

All interface levels are expressed through one parametric response model with parameters $\theta = (G, b, \tau, d, \bar{\nu}, \Sigma)$:

$$\begin{aligned}\nu_{k+1} &= A_\theta\,\nu_k + (I - A_\theta)\,\text{sat}_{\bar{\nu}}\left(G\,u_{k-d} + b\right) + \eta_k,\\ \eta_k &\sim \mathcal{N}(0, \Sigma),\end{aligned} \qquad (7)$$

where u_k is the command issued to the controller at step k, $G \in \mathbb{R}^{3\times3}$ is the command gain including cross-coupling between lateral and yaw channels, b is a bias, $A_\theta = \text{diag}(e^{-\Delta t/\tau_x}, e^{-\Delta t/\tau_y}, e^{-\Delta t/\tau_\omega})$ encodes first-order tracking lag with per-channel time constants τ, d is an integer command delay in steps, $\text{sat}_{\bar{\nu}}$ clips each channel to the platform limit $\bar{\nu}$, and η_k is tracking noise. The four levels differ in how θ is obtained and how the planner's command is mapped to u_k.

Ideal. The ideal level sets $A_\theta = 0$, $G = I$, $b = 0$, $d = 0$, and $\Sigma = 0$, so that $\nu_{k+1} = \text{sat}_{\bar{\nu}}(u_k^*)$. It reproduces the assumption made by kinematic simulation and isolates planning quality from tracking quality.

Native. The native levels ι_{A} and ι_{B} pass $u_k = u_k^*$ directly to the platform's own velocity interface. ι_{A} follows the convention of a vendor high-level velocity API shared by the Go2-class and B2-class assets, and ι_{B} follows a locomotion-manager

KAGGLEYEs ACADEMY • E-PRINT REGISTRY • KAGG:2610.00001

convention used by the ANYmal-class asset, in which the manager may switch gait modes in response to terrain or command changes. Native levels are what most published evaluations implicitly use.

Calibrated-opaque. The calibrated level first identifies $\hat{\theta}_p$ for each platform with a GAUGE-style procedure [8]. Probe commands $u_k = u_k^* + \xi_k$ superimpose small excitations ξ_k on commands sampled from the reference planner's own command distribution on a calibration course, so that the identified model is accurate where planners operate. The parameters are fitted by regularized prediction-error minimization,

$$\hat{\theta}_p = \arg\min_{\theta} \sum_k \|\nu_{k+1} - f_\theta(\nu_k, u_{k-d})\|^2_{\Sigma^{-1}} + \lambda \|\theta - \theta_0\|^2, \tag{8}$$

where f_θ is the noise-free part of Eq. (7), θ_0 is a nominal prior, and $\lambda > 0$ trades off fit and prior. The delay d is selected by enumeration over a small integer range. The identified model is then used to present every platform through a common reference response. The planner command is first shaped by a reference filter,

$$\tilde{u}_{k+1} = A_{\text{ref}}\, \tilde{u}_k + (I - A_{\text{ref}})\, u^*_{k-(d_{\text{ref}} - \hat{d}_p)}, \quad \tau_{\text{ref}} \geq \max_p \hat{\tau}_p, \quad d_{\text{ref}} \geq \max_p \hat{d}_p, \tag{9}$$

and the issued command inverts the identified static map,

$$u_k = \hat{G}_p^{-1}\left(\text{sat}_{\bar{\nu}_{\text{ref}}}(\tilde{u}_k) - \hat{b}_p\right), \qquad \bar{\nu}_{\text{ref}} = \min_p \hat{\bar{\nu}}_p. \tag{10}$$

Choosing the reference time constant and delay no smaller than the slowest platform's, and the reference limits no larger than the most restrictive platform's, makes the reference response realizable on every platform: the inverse in Eq. (10) only removes gain and bias, while lag and delay are padded up rather than cancelled, which avoids the noise amplification of a dynamic inverse. Holding the interface fixed therefore means that all platforms respond to u^* approximately as the same first-order, delayed, saturated system. The residual

$$e_k = \nu_k - \nu_k^{\text{ref}}, \qquad E_{\text{if}} = \frac{1}{K}\sum_{k=1}^{K} \|e_k\|_2, \tag{11}$$

between realized motion and the reference model is written to the record as an interface-axis observation, so that imperfect normalization is measured rather than hidden.

Gait-mode events. Interfaces such as ι_{B} can switch gait modes, during which commands are not tracked. The interface layer detects mode transitions from the controller's status channel and records each event with its start time and duration. The time spent in transitions is subtracted from planner-attributed time and logged as an interface-axis observation; this is the ninth confounding factor in Table 1.

4.3 Terrain axis

Each course $c \in \mathcal{C}$ is specified by a parameter vector $\psi_c = (L_c, w_c, h_c, \rho_c, \sigma_c)$: route length L_c, minimum corridor width w_c, maximum step height h_c, obstacle density ρ_c, and surface roughness amplitude σ_c. The six courses are *flat* (baseline geometry with sparse static obstacles), *mixed* (segments of the other classes in a fixed order), *stairs* (ascending and descending flights), *narrow* (corridors close to the body width), *cluttered* (dense static obstacles requiring repeated replanning), and *rough* (heightfield terrain without discrete obstacles). Because the same course should challenge platforms of different size comparably, geometric parameters are expressed relative to the body: the corridor clearance is reported as

$$\kappa_c(p) = \frac{w_c - W_p}{W_p}, \qquad \eta_c(p) = \frac{h_c}{H_p}, \tag{12}$$

where W_p is body width including the swing envelope and H_p is nominal leg length. The courses are instantiated from fixed seeds, and the course hash $\text{hash}(\psi_c, \sigma)$ is part of the record so that the consistency predicate χ can verify that two platforms ran the same instance.

4.4 Sensor axis

The three rigs are LiDAR-only, LiDAR+vision, and degraded. The LiDAR model produces per-beam ranges

$$\tilde{r}_j = \min(r_j, r_{\max}) + \epsilon_j \ \text{ with probability } 1 - p_{\text{drop}}, \quad \tilde{r}_j = \varnothing \ \text{ with probability } p_{\text{drop}}, \qquad \epsilon_j \sim \mathcal{N}(0, \sigma_r^2), \tag{13}$$

where $r_{\max}$ is the configured maximum range, which is well below the tens-of-meters rating of typical units so that planners see a local rather than global view. The LiDAR+vision rig adds an RGB-D stream used by planners that consume semantic or traversability cues. The degraded rig reduces $r_{\max}$, raises p_{drop} and σ_r, and introduces bursty dropout to emulate dust or specular surfaces. All rigs are mounted according to each platform's configuration so that the field of view is consistent relative to the body. For each run the recorder computes the dropout fraction

$$\delta(\rho) = 1 - \frac{1}{T}\sum_{t=1}^{T} \frac{n_{\text{valid}}(t)}{n_{\text{exp}}(t)}, \tag{14}$$

where $n_{\text{valid}}(t)$ and $n_{\text{exp}}(t)$ are the valid and expected returns in scan t.

4.5 Planner set and reference stack

The fixed planner set contains four planners: a configuration-driven reference stack, a reactive baseline, a learned policy, and a rule-based planner. All four consume the same robot-centric elevation map [28] and goal sequence and emit u_t^* at 10 Hz. The reference stack is AgniNav [9], instantiated from a per-platform configuration

$$\kappa_p = \left(\mathcal{F}_p,\ T^{\text{body}}_{\text{sensor},p},\ \bar{v}_p,\ \bar{a}_p,\ \bar{\omega}_p\right), \tag{15}$$

comprising the footprint polygon $\mathcal{F}_p$, the sensor extrinsics, and the velocity, acceleration, and yaw-rate limits. As instantiated in the harness, the reference stack selects among candidate trajectories ξ over a short horizon by minimizing

$$J(\xi) = w_g\, d(\xi_H, g) + w_o \sum_{h=0}^{H} c_M\left(\xi_h \oplus \mathcal{F}_p\right) + w_u \sum_{h=1}^{H} \|u_h - u_{h-1}\|^2, \quad \text{s.t. } |u_h| \leq (\bar{v}_p, \bar{\omega}_p), \tag{16}$$

KAGGLEYEs ACADEMY • E-PRINT REGISTRY • KAGG:2610.00001

where $d(\xi_H, g)$ is the remaining distance from the horizon end to the goal, $c_M(\xi_h \oplus \mathcal{F}_p)$ is the map cost of the footprint placed at pose ξ_h, and w_g, w_o, w_u are weights shared across platforms. Only κ_p differs between platforms. Using a planner whose embodiment dependence is confined to κ_p as the reference makes it a control: its cross-platform variation under fixed axes should be explained by κ_p and by recorded interface residuals, and deviations from that expectation flag harness defects. The velocity limits in κ_p are set below the platforms' hardware ratings, within the 1–5 m/s range typical of quadrupeds, and are intersected with $\bar{\nu}_{\text{ref}}$ when the calibrated interface is active.

4.6 Results record and sanity checks

Every run emits a JSON record (schema v0.5, 214 fields) organized in five groups: run identity and specification hashes; axis descriptors (θ or $\hat{\theta}_p$, ψ_c, rig parameters, κ_p); outcome metrics; interface-axis observations (E_{if}, gait-mode events, saturation counts); and sanity flags. Outcome metrics include success, collision count, path length ratio, and a completion-time score in the spirit of [34],

$$m_{\text{time}}(\rho) = \mathbb{1}[\text{success}] \cdot \frac{T^{\text{ref}}(\rho)}{\max\left(T(\rho) - T_{\text{gait}}(\rho),\ T^{\text{ref}}(\rho)\right)}, \tag{17}$$

where T^{ref} is the time required by the reference response model along the shortest feasible path and T_{gait} is the time spent in recorded gait-mode transitions. The recorder validates each record against the schema and evaluates sanity predicates, including the dropout check $\delta(\rho) \leq \delta_{\max}$, where the threshold $\delta_{\max}$ is set per rig in the configuration. A run that fails a predicate is *quarantined*: it remains in the record set with its flag set and is excluded from scores by default, but it is never deleted, so that quarantine rates are themselves reportable.

4.7 Confounding-factor accounting

Table 1 lists the nine confounding factors the harness records, grouped by axis. A factor counts as measured when its value is recoverable from every record. Rank stability between two protocols is summarized by the Kendall rank correlation over planners,

$$\tau_K = \frac{2}{|\Pi|(|\Pi|-1)} \sum_{\pi<\pi'} \operatorname{sgn}\left(\bar{S}_1(\pi) - \bar{S}_1(\pi')\right) \cdot \operatorname{sgn}\left(\bar{S}_2(\pi) - \bar{S}_2(\pi')\right), \tag{18}$$

which the harness reports alongside the per-planner rank changes used in the case study of Section 5.4.

05 Experiments

5.1 Setup

The study uses harness v0.5 with three quadruped platform assets (Go2-class, B2-class, and ANYmal-class), the six standardized terrain courses (flat, mixed, stairs, narrow, cluttered, rough), the three sensor rigs (LiDAR-only, LiDAR+vision, degraded), and the fixed set of four planners (configuration-driven reference stack, reactive baseline, learned policy, rule-based). Each platform–course–rig cell receives 120 standardized runs, which span the planner set and interface levels, for a total of $120 \times 3 \times 6 \times 3 = 6480$ runs. Each run emits a results record under JSON schema v0.5 with 214 fields.

REPRODUCIBILITY NOTE

All 6480 records, including quarantined runs, are retained in the record set. Specification hashes for courses, rigs, interface configurations, and platform configurations are stored in every record, so that any reported comparison can be re-derived by filtering records rather than by re-running experiments.

The full harness is compared with two comparison protocols that reflect how cross-platform results are typically assembled. The *raw-score* protocol ranks planners by Eq. (3): each platform runs on its native interface, and records are assembled by per-platform logging without interface control. The *interface-fixed* protocol drives all platforms through the calibrated-opaque interface, Eq. (9)–(10), but otherwise uses the same assembly as the raw-score protocol. The full harness controls all three axes and applies the recorder and sanity checks of Section 4. The reported measures are cross-platform run parity, Eq. (5); record schema completeness, Eq. (6); harness overhead per run, the wall-clock time spent by the protocol outside the episode itself; and the number of confounding factors measured.

5.2 Main comparison

Table 2 reports the main comparison. The full harness reaches 95.7% cross-platform run parity, 12.8 percentage points above the raw-score protocol (82.9%) and 8.7 points above the interface-fixed protocol (87.0%). Holding the interface fixed therefore recovers roughly a third of the gap between raw-score ranking and the full harness (4.1 of 12.8 points); the remainder comes from controlling terrain and sensor rigs and from the record-level checks. Schema completeness is 100.0% for the full harness, against 96.9% and 93.8% for the two protocols. The lower completeness of the interface-fixed protocol relative to the raw-score protocol is consistent with the calibrated interface introducing fields, such as identified parameters and residuals, that per-platform logging does not always populate; schema validation at record time is what closes this gap in the full harness. Overhead per run is 2.3 s for the full harness, against 2.5 s and 2.4 s for the two protocols, so the additional checks do not add cost relative to the conventional assembly pipelines; we attribute this to the recorder writing a single validated record per run instead of post-hoc merging of per-platform logs, although the measurements do not isolate that cause. Finally, the full harness measures nine confounding factors against eight for both protocols. The additional factor is gait-mode transition time (factor 9 in Table 1), which the conventional protocols fold into planner-attributed time.

5.3 Ablation over axes

Table 3 removes one axis at a time from the full harness, in which case the corresponding factor is neither controlled nor recorded. Removing the sensor-rig axis lowers parity from 95.7% to 86.4% (a drop of 9.3 points), removing the terrain axis lowers it to 80.3% (15.4 points), and removing the interface axis lowers it to 74.5% (21.2 points). The interface axis is thus the largest single contributor to cross-platform comparability, followed by terrain and then the sensor rig. The ordering is consistent with the main comparison, in which holding the interface fixed was the single change that most improved parity over raw-score ranking.

KAGGLEYEs ACADEMY • E-PRINT REGISTRY • KAGG:2610.00001

Table 1: Confounding factors recorded by the harness, by axis. The comparison protocols in Table 2 record factors 1–8; factor 9 is recorded only by the full harness.

#	Axis	Factor	Record observable
1	Interface	Command gain and bias	$\hat{G}_p$, $\hat{b}_p$
2	Interface	Command delay and tracking lag	$\hat{d}_p$, $\hat{\tau}_p$, E_{if}
3	Interface	Velocity saturation	$\bar{\nu}$, saturation counts
4	Terrain	Course geometry and instance	ψ_c, course hash
5	Terrain	Body-relative difficulty	$\kappa_c(p)$, $\eta_c(p)$
6	Sensor	Modality	rig identifier
7	Sensor	Effective range and dropout	$r_{\max}$, $\delta(\rho)$
8	Platform	Footprint and limits	κ_p
9	Interface	Gait-mode transition time	event log, T_{gait}

Table 2: Main comparison of evaluation protocols. Parity and completeness in percent; overhead in seconds per run.

Method	Cross-platform run parity (%)	Record schema completeness (%)	Harness overhead per run (s)	Confounding factors measured
Ranking without interface control (raw scores)	82.9	96.9	2.5	8
Ranking with interface fixed (calibrated-opaque axis held)	87.0	93.8	2.4	8
Ours	**95.7**	**100.0**	**2.3**	**9**

The parity of the variant without the interface axis (74.5%) is lower than that of the raw-score protocol (82.9%). The two configurations are not nested: the raw-score protocol evaluates each platform on a single native interface, whereas the ablated harness still varies the interface level across runs but no longer records or normalizes it, so matched tuples can pair records produced under different interface levels. A plausible reading is that an uncontrolled axis that is still varied is more damaging to comparability than an axis that is fixed by convention per platform; the present data do not test this explanation directly.

5.4 Case study: a ranking flip under interface control

The case study compares the planner ranking produced by the raw-score protocol, Eq. (3), with the ranking obtained when the interface axis is held at the calibrated-opaque level, Eq. (4). One planner from the fixed set, denoted planner X, ranks first across platforms on raw scores but drops to third once the interface axis is held fixed. Inspection of the records attributes its raw advantage to interface luck on one platform: on that platform's native interface, the command-to-motion response happened to favor planner X's command profile, and averaging over platforms in Eq. (3) carried that single-platform advantage into the cross-platform ranking. Once all platforms are presented through the common reference response, the advantage disappears, and planner X's controlled score places it behind two other planners.

Two points follow. First, a two-position drop for the top-ranked planner in a four-planner set is the kind of change that would reverse the headline conclusion of a published comparison, and it arises without any change to the planner, the courses, or the sensors. Second, the raw-score protocol offers no signal that the ranking is fragile: the interface parameters that explain the flip are not part of its records. In the full harness, by contrast, the identified parameters $\hat{\theta}_p$ and the residual E_{if} are part of every record, so the flip can be traced to its source.

5.5 Failure cases

Degraded rig on the B2-class asset. With the degraded sensor rig on the B2-class asset, 2.1% of runs fail the harness sanity check because the dropout fraction $\delta(\rho)$ exceeds the configured threshold. These runs are quarantined rather than dropped: they remain in the record set with their flags set. Silently discarding them would bias scores upward for whichever planners happen to be evaluated on the affected seeds, and would hide the fact that the degraded rig on this platform sits near the limit of what the rig model is meant to represent. Keeping them makes the quarantine rate itself a reportable property of the platform–rig combination.

Stairs course on the ANYmal-class asset. On the stairs course, the ANYmal-class asset's gait manager switches modes when approaching and leaving a flight, which adds 0.7 s per trial. The harness records this time as an interface-axis observation (T_{gait} in Eq. (17)) rather than as a planner failure or a planner-attributed delay. Under the conventional protocols the same 0.7 s appears as slower completion for whichever planner is running, which penalizes planners on that platform for behavior they do not control. This case is the concrete origin of the ninth confounding factor in Table 1.

Planner X. The case study above is itself a failure case of conventional evaluation rather than of a planner: the planner behaves identically under both protocols, and only the attribution of its performance changes.

06 Limitations

Several limitations qualify these results. First, the platforms are simulated assets of three platform classes; the native interface levels follow the conventions of the corresponding vendor interfaces, but the study does not include hardware runs, and the sim-to-real predictivity of the harness [35] remains to be established. Second, the calibrated-opaque level depends on the fidelity of the parametric model in Eq. (7); first-order lag with constant delay cannot represent strongly state-dependent behavior such as speed-dependent gait changes, which is why gait-mode events are recorded separately and why E_{if} is reported rather than assumed to be zero. Third, padding every platform up to the slowest reference response makes comparisons fair but conservative: planners that exploit fast tracking on agile platforms are evaluated below their potential at the calibrated level, and their native-level results should be reported alongside. Fourth, the planner set is small, so the case study demonstrates that a ranking flip can occur but does not estimate how often such flips occur in published comparisons. Fifth, the counts of confounding factors depend on the factor taxonomy in Table 1; a differ-

KAGGLEYEs ACADEMY • E-PRINT REGISTRY • KAGG:2610.00001

Table 3: Ablation over harness axes. Each variant removes one axis from the full harness.

Variant	Cross-platform run parity (%)
Full harness (all 3 axes)	**95.7**
w/o sensor-rig axis	86.4
w/o terrain axis	80.3
w/o interface axis	74.5

ent taxonomy could split or merge factors. Finally, the parity measure is only as strong as the consistency predicate χ; parity of 95.7% indicates that records are commensurable, not that the remaining 4.3% of matched tuples are invalid experiments, and the explanation offered for the low parity of the ablation without the interface axis remains a hypothesis.

07 Conclusion

This paper presented a unified evaluation harness for quadruped navigation that isolates the velocity interface, terrain course, and sensor rig as explicit axes and records every run in a standardized, validated 214-field record. Over 6480 runs on Go2-class, B2-class, and ANYmal-class assets, the harness achieved 95.7% cross-platform run parity and 100.0% schema completeness at 2.3 s of overhead per run, and measured nine confounding factors, compared with 82.9% and 87.0% parity for raw-score and interface-fixed protocols. The ablation identifies the interface axis as the largest contributor to comparability, and the case study shows that a planner ranked first on raw scores can fall to third when the interface is held fixed. The practical recommendation is that cross-platform navigation comparisons report the interface level, a calibrated interface model, the course specification, and the rig, and that they quarantine rather than drop runs that fail sanity checks. Future work will validate the harness against hardware runs, extend the interface model to state-dependent gait behavior, and enlarge the planner set to estimate how frequently ranking flips occur.

References

[1] J. Hwangbo, J. Lee, A. Dosovitskiy, D. Bellicoso, V. Tsounis, V. Koltun, and M. Hutter. Learning Agile and Dynamic Motor Skills for Legged Robots. arXiv:1901.08652, 2019.

[2] J. Lee, J. Hwangbo, L. Wellhausen, V. Koltun, and M. Hutter. Learning Quadrupedal Locomotion over Challenging Terrain. arXiv:2010.11251, 2020.

[3] T. Miki, J. Lee, J. Hwangbo, L. Wellhausen, V. Koltun, and M. Hutter. Learning Robust Perceptive Locomotion for Quadrupedal Robots in the Wild. arXiv:2201.08117, 2022.

[4] M. Tranzatto et al.. CERBERUS: Autonomous Legged and Aerial Robotic Exploration in the Tunnel and Urban Circuits of the DARPA Subterranean Challenge. arXiv:2201.07067, 2022.

[5] L. Wellhausen and M. Hutter. ArtPlanner: Robust Legged Robot Navigation in the Field. arXiv:2303.01420, 2023.

[6] N. Rudin, D. Hoeller, P. Reist, and M. Hutter. Learning to Walk in Minutes Using Massively Parallel Deep Reinforcement Learning. arXiv:2109.11978, 2021.

[7] G. B. Margolis and P. Agrawal. Walk These Ways: Tuning Robot Control for Generalization with Multiplicity of Behavior. arXiv:2212.03238, 2022.

[8] T. Zang, H. Huang, S. Cheng, and W. Zhang. GAUGE: Planner-Conditioned Active Calibration of Opaque Quadruped Velocity Interfaces. arXiv:2607.02841, 2026.

[9] T. Zang, S. Cheng, H. Huang, S. Wang, and W. Zhang. AgniNav: Configuration-Driven Cross-Embodiment Local Planning for Robot Navigation. arXiv:2606.10903, 2026.

[10] V. Makoviychuk et al.. Isaac Gym: High Performance GPU-Based Physics Simulation For Robot Learning. arXiv:2108.10470, 2021.

[11] A. Kumar, Z. Fu, D. Pathak, and J. Malik. RMA: Rapid Motor Adaptation for Legged Robots. arXiv:2107.04034, 2021.

[12] A. Agarwal, A. Kumar, J. Malik, and D. Pathak. Legged Locomotion in Challenging Terrains using Egocentric Vision. arXiv:2211.07638, 2022.

[13] Z. Zhuang, Z. Fu, J. Wang, C. Atkeson, S. Schwertfeger, C. Finn, and H. Zhao. Robot Parkour Learning. arXiv:2309.05665, 2023.

[14] X. Cheng, K. Shi, A. Agarwal, and D. Pathak. Extreme Parkour with Legged Robots. arXiv:2309.14341, 2023.

[15] D. Hoeller, N. Rudin, D. Sako, and M. Hutter. ANYmal Parkour: Learning Agile Navigation for Quadrupedal Robots. arXiv:2306.14874, 2023.

[16] R. Grandia, F. Jenelten, S. Yang, F. Farshidian, and M. Hutter. Perceptive Locomotion through Nonlinear Model Predictive Control. arXiv:2208.08373, 2022.

[17] S. Ha, J. Lee, M. van de Panne, Z. Xie, W. Yu, and M. Khadiv. Learning-based Legged Locomotion: State of the Art and Future Perspectives. arXiv:2406.01152, 2024.

[18] X. B. Peng, M. Andrychowicz, W. Zaremba, and P. Abbeel. Sim-to-Real Transfer of Robotic Control with Dynamics Randomization. arXiv:1710.06537, 2017.

[19] J. Tobin, R. Fong, A. Ray, J. Schneider, W. Zaremba, and P. Abbeel. Domain Randomization for Transferring Deep Neural Networks from Simulation to the Real World. arXiv:1703.06907, 2017.

[20] J. Tan, T. Zhang, E. Coumans, A. Iscen, Y. Bai, D. Hafner, S. Bohez, and V. Vanhoucke. Sim-to-Real: Learning Agile Locomotion For Quadruped Robots. arXiv:1804.10332, 2018.

[21] H. Xi, M. Tan, X. Zhang, S. Cheng, S. Wang, Y. Gu, X. Shen, and W. Zhang. CeRLP: A Cross-embodiment Robot Local Planning Framework for Visual Navigation. arXiv:2603.19602, 2026.

[22] J. Truong, D. Yarats, T. Li, F. Meier, S. Chernova, D. Batra, and A. Rai. Learning Navigation Skills for Legged Robots with Learned Robot Embodiments. arXiv:2011.12255, 2020.

[23] Z. Fu, A. Kumar, A. Agarwal, H. Qi, J. Malik, and D. Pathak. Coupling Vision and Proprioception for Navigation of Legged Robots. arXiv:2112.02094, 2021.

[24] N. Rudin, D. Hoeller, M. Bjelonic, and M. Hutter. Advanced Skills by Learning Locomotion and Local Navigation End-to-End. arXiv:2209.12827, 2022.

[25] D. Shah, A. Sridhar, A. Bhorkar, N. Hirose, and S. Levine. GNM: A General Navigation Model to Drive Any Robot. arXiv:2210.03370, 2022.

[26] D. Shah, A. Sridhar, N. Dashora, K. Stachowicz, K. Black, N. Hirose, and S. Levine. ViNT: A Foundation Model for Visual Navigation. arXiv:2306.14846, 2023.

KAGGLEYEs ACADEMY • E-PRINT REGISTRY • KAGG:2610.00001

[27] A. Sridhar, D. Shah, C. Glossop, and S. Levine. NoMaD: Goal Masked Diffusion Policies for Navigation and Exploration. arXiv:2310.07896, 2023.

[28] T. Miki, L. Wellhausen, R. Grandia, F. Jenelten, T. Homberger, and M. Hutter. Elevation Mapping for Locomotion and Navigation using GPU. arXiv:2204.12876, 2022.

[29] W. Xu, Y. Cai, D. He, J. Lin, and F. Zhang. FAST-LIO2: Fast Direct LiDAR-Inertial Odometry. arXiv:2107.06829, 2021.

[30] J. Frey, M. Mattamala, N. Chebrolu, C. Cadena, M. Fallon, and M. Hutter. Fast Traversability Estimation for Wild Visual Navigation. arXiv:2305.08510, 2023.

[31] M. Mattamala, J. Frey, P. Libera, N. Chebrolu, G. Martius, C. Cadena, M. Hutter, and M. Fallon. Wild Visual Navigation: Fast Traversability Learning via Pre-Trained Models and Online Self-Supervision. arXiv:2404.07110, 2024.

[32] M. Savva et al.. Habitat: A Platform for Embodied AI Research. arXiv:1904.01201, 2019.

[33] P. Anderson et al.. On Evaluation of Embodied Navigation Agents. arXiv:1807.06757, 2018.

[34] N. Yokoyama, S. Ha, and D. Batra. Success Weighted by Completion Time: A Dynamics-Aware Evaluation Criteria for Embodied Navigation. arXiv:2103.08022, 2021.

[35] A. Kadian et al.. Sim2Real Predictivity: Does Evaluation in Simulation Predict Real-World Performance? arXiv:1912.06321, 2019.

[36] D. Perille, A. Truong, X. Xiao, and P. Stone. Benchmarking Metric Ground Navigation. arXiv:2008.13315, 2020.

[37] X. Xiao et al.. Autonomous Ground Navigation in Highly Constrained Spaces: Lessons Learned from the BARN Challenge at ICRA 2022. arXiv:2208.10473, 2022.

[38] X. Xiao, B. Liu, G. Warnell, and P. Stone. Motion Planning and Control for Mobile Robot Navigation Using Machine Learning: A Survey. arXiv:2011.13112, 2020.

[39] M. Mittal et al.. Orbit: A Unified Simulation Framework for Interactive Robot Learning Environments. arXiv:2301.04195, 2023.

[40] P. Henderson, R. Islam, P. Bachman, J. Pineau, D. Precup, and D. Meger. Deep Reinforcement Learning that Matters. arXiv:1709.06560, 2017.

KAGGLEYEs ACADEMY • E-PRINT REGISTRY • KAGG:2610.00001