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

KAGG RECORD
KAGG:2607.00006

Quantifying the Sim-to-Real Gap of Quadruped Velocity Controllers on Deformable Terrain

Nikos Stavridis • **Amelie Laurent** • **Diego Navarro**

Robot Dynamics Group, ETH Zurich, Zurich, Switzerland

PUBLISHED	REGISTERED	RECORD TYPE
July 22, 2026	July 23, 2026	Preprint

ABSTRACT

Sim-to-real studies of legged navigation usually attribute transfer failures to perception mismatch, yet on deformable terrain the velocity interface between planner and locomotion controller also changes character: response lag grows, effective gains shift, and saturation appears earlier. This paper quantifies that interface gap for a quadruped crossing terrain boundaries. We track a calibrated interface model online, using planner-conditioned probing in the style of GAUGE at 2 Hz, and couple it to a configuration-driven motion predictor in the style of AgniNav that scores, at 10 Hz, which fraction of the perceived free range is realizable under the current interface state. A sim-to-real pre-alignment step in Isaac Lab with matched terrain assets initializes the tracker for each terrain class. On an ANYmal traversing a course of grass, loose gravel, a sand trough, packed dirt, and grass (30 m per segment, 15 crossings per direction), the approach reduced interface drift from 4.9 to 4.6 percent per km relative to a static grass-calibrated model, raised range realizability from 70.4 to 74.8 percent, raised navigation success from 80.8 to 86.3 percent, and lowered 95th-percentile tracking error from 17.9 to 15.9 cm. Ablations indicate that online drift tracking, the realizability score, and pre-alignment each contribute. Recorded failures, including a 3.2-fold lag increase in sand and a transient realizability drop to 41 percent at the gravel-to-sand boundary, delimit where the diagnostics remain insufficient.

INDEX TERMS quadruped locomotion, sim-to-real transfer, velocity interface calibration, deformable terrain, local planning, ANYmal

KAGG NOTE On deformable terrain the velocity interface between planner and locomotion controller is a time-varying quantity that should be measured online and propagated into planning, rather than a fixed property calibrated once.

01 Introduction

Learned locomotion controllers have made quadrupeds such as ANYmal [1] reliable on rough and natural terrain [2, 3, 4], and most navigation stacks built on top of them treat the locomotion policy as a velocity-tracking black box: the planner issues a planar twist, and the controller is assumed to realize it with a fixed and known response. When such stacks are transferred from simulation to hardware, the resulting performance gap is usually attributed to perception: elevation maps that misjudge vegetation, depth noise, or traversability estimators that were trained on the wrong appearance distribution [5, 6, 7].

This paper examines a second, less visible source of transfer error. On deformable terrain such as sand, loose gravel, snow, or mud, the velocity interface itself changes character. Feet sink and slip, the controller reacts with altered gait timing, and the command-to-motion map that a planner implicitly assumes no longer holds: the response lags further behind the command, the effective gain shifts, and saturation appears at lower commanded speeds. Because these effects depend on the substrate and evolve over distance as material is displaced, a model calibrated once on firm ground, or identified in simulation with nominal contact parameters, drifts out of validity precisely where a navigation system most needs an accurate prediction of stopping distance and reachable space.

Two recent lines of work provide the building blocks for addressing this problem. GAUGE [8] treats the quadruped velocity controller as an opaque interface and calibrates a command-to-motion response model actively, through probe commands conditioned on what the planner is about to request. AgniNav [9] decouples local planning from a specific robot through an abstract configuration of body geometry, sensing, and velocity limits. Neither, however, addresses how the interface model evolves over a traverse that crosses terrain boundaries, nor how a planner should discount perceived free space that the current interface cannot actually reach.

The central question of this study is therefore: *how large is the sim-to-real gap of the velocity interface on deformable terrain, how fast does it drift, and how much of the perceived free range does it render unrealizable?* The contributions are as follows.

1. A formulation of the interface gap as a time-varying parameter discrepancy of a lag-gain-saturation model, together with a distance-normalized drift metric and a range-realizability fraction that can be computed on board.
2. An online gap-quantification method that tracks the calibrated interface model with planner-conditioned probing in the style of GAUGE, detects terrain-induced change points, and re-initializes from a sim-to-real pre-aligned prior.
3. A configuration-based motion predictor, in the style of AgniNav, that scores which fraction of the LiDAR-perceived free range is realizable under the current inter-

KAGGLEYEs ACADEMY • E-PRINT REGISTRY • KAGG:2607.00006

face state and feeds this score back into command limits and trajectory selection.

4. An evaluation on an ANYmal crossing a designed five-segment terrain-transition course, reporting interface drift, range realizability, navigation success, and tracking error against three baselines, an ablation, and an analysis of recorded failure cases.

02 Related Work

2.1 Learned locomotion and sim-to-real transfer

Reinforcement-learned locomotion policies trained in simulation are now the standard low-level controller for many quadrupeds. Hwangbo et al. combined a learned actuator model with simulation training to transfer agile skills to ANYmal [2]; Lee et al. used a teacher-student scheme and terrain curricula to traverse challenging natural ground blind [3]; and Miki et al. fused exteroception and proprioception through a belief encoder to walk robustly in the wild [4]. Massively parallel training in GPU simulators [10, 11, 12] made such policies cheap to retrain. Sim-to-real methods for locomotion rely on dynamics randomization [13, 14, 15], online system identification [16], latent adaptation from proprioceptive history [17], concurrent state estimation [18], or meta-learned adaptation [19]; Zhao et al. survey the broader landscape [20]. These methods adapt the *controller* so that it remains stable and tracks commands as well as possible. They do not, however, expose to the navigation layer how well commands are tracked on the current substrate. The present work is complementary: it leaves the locomotion policy untouched and instead measures the residual command-to-motion discrepancy that adaptation leaves behind.

2.2 Deformable terrain and traversability

Choi et al. showed that granular media require dedicated contact models in simulation to obtain policies that walk on sand and similar substrates [21], which directly suggests that interface behavior on such terrain differs between simulation and hardware. Rapid locomotion work on the MIT Mini Cheetah [22] and the behavior-multiplicity approach of Walk These Ways [23] likewise report terrain-dependent velocity tracking. On the perception side, self-supervised terrain property prediction [5], online visual traversability learning [6], predictive traversability with field data [24], and experience-driven navigation [25] learn which terrain is passable or costly. Probabilistic and GPU elevation mapping [26, 27] supply the geometric layer for legged navigation planners [28]. These approaches score *where* the robot should go; the realizability score developed here addresses *whether* a nominally free motion can be executed under the current interface, which is a distinct quantity: a terrain can be traversable while still lengthening stopping distances.

2.3 Calibration of the command-to-motion interface

Classical system identification [29] and adaptive control [30] provide recursive estimators with forgetting for slowly varying plants, and sequential change detection such as CUSUM [31] provides a principled trigger for abrupt changes. In off-road navigation, learned inverse kinodynamics [32] and its visual-inertial extension [33] correct commands so that wheeled vehicles realize desired curvatures at high speed on unstructured ground; they learn a correction offline from driving data and condition it on terrain observations.

GAUGE [8] is the closest antecedent for the calibration component. It treats the quadruped velocity controller as an opaque interface, issues probe commands that are conditioned on the planner's intended motion so that excitation is concentrated where it matters for the next plan, and fits a calibrated command-to-motion response model online. The present work adopts this planner-conditioned, opaque-interface view and its online fitting, but differs in emphasis. GAUGE calibrates an interface to make subsequent planning accurate; here the calibrated model is tracked continuously across terrain boundaries in order to *quantify* the gap, both against a simulation-derived model and against its own past, and the model state is used to rescore the planner's perceived free space. The added elements are a change-point mechanism with re-initialization from a sim-to-real pre-aligned prior, a distance-normalized drift metric, and a coupling to realizability that GAUGE does not address.

2.4 Configuration-driven local planning

Sampling-based local planners such as the dynamic window approach [34] and model-predictive path integral control [35] evaluate candidate commands by forward-simulating a motion model, so their safety depends on that model. Learned local navigation for ANYmal [36, 37, 38] instead embeds the motion model in the policy.

AgniNav [9] provides a configuration-driven local planner that operates on an abstract robot configuration, comprising body geometry, sensors, and velocity limits, so that a single planning stack can run across different embodiments. Its collision checking and candidate generation are parameterized by this configuration rather than by a specific robot model. The present work uses this structure as the substrate for a motion predictor, with one modification: the velocity limits and response of the configuration are no longer constants of the embodiment but are supplied, at each planning cycle, by the tracked interface model. The predictor then compares the free range implied by the nominal configuration with the range that is realizable under the current interface state. In this sense the study treats a single robot on changing terrain as a sequence of slightly different embodiments, which is precisely the variation a configuration-driven planner is designed to absorb, provided that the configuration is kept current.

2.5 Evaluating sim-to-real navigation

Kadian et al. showed that simulation rankings of navigation agents can correlate poorly with real-world rankings [7], and Truong et al. found that lower-fidelity simulation can transfer better than high-fidelity simulation when physics inaccuracies induce overfitting [39]. Both results motivate measuring which component of the stack carries the gap. The experiments below compare a simulation-only interface model, obtained in Isaac Lab [12] with terrain assets matched to the course, against hardware-calibrated alternatives, which isolates the interface component of the sim-to-real gap from perception.

03 Problem Formulation

3.1 Velocity interface

Let k index the planner-side sampling instants with period Δt. The planner issues a planar command $\mathbf{u}_k = (u_k^x, u_k^y, u_k^\psi)$, comprising forward, lateral, and yaw-rate components in the body

KAGGLEYEs ACADEMY • E-PRINT REGISTRY • KAGG:2607.00006

frame. The legged state estimator [40] provides the realized body twist $\boldsymbol{\nu}_k = (\nu_k^x, \nu_k^y, \nu_k^\psi)$. The *velocity interface* is the unknown causal map from the command history to the realized twist. Following the opaque-interface view of GAUGE [8], it is modeled per axis $j \in \{x, y, \psi\}$ as a delayed, saturated first-order response:

$$\nu_{k+1}^j = \alpha_j \, \nu_k^j + (1 - \alpha_j) \, K_j \, \sigma_{\bar{u}_j}\left(u_{k-d_j}^j\right) + w_k^j, \quad (1)$$

with $\alpha_j = \exp(-\Delta t/\tau_j)$ and the smooth saturation

$$\sigma_{\bar{u}}(u) = \bar{u} \tanh(u/\bar{u}). \quad (2)$$

Here τ_j is the response time constant (lag), K_j the steady-state gain, $d_j \in \mathbb{N}$ the transport delay in samples, $\bar{u}_j$ the saturation level, and w_k^j a zero-mean disturbance capturing gait-induced oscillation. The interface state is $\boldsymbol{\theta} = (K_j, \tau_j, d_j, \bar{u}_j)_j$. On firm ground $K_j \approx 1$, and the lag and delay are small compared with the planning horizon; on deformable terrain K_j falls because of slip, τ_j and d_j grow because the controller re-establishes footholds, and $\bar{u}_j$ drops.

3.2 Interface gap and drift

Let c_t denote the terrain class traversed at time t, and $\boldsymbol{\theta}^{\text{sim}}(c)$ the interface state identified in simulation on the matched terrain asset. With $\hat{\boldsymbol{\theta}}_t$ the hardware estimate, the *interface gap* is the element-wise relative discrepancy

$$\boldsymbol{\gamma}_t = \hat{\boldsymbol{\theta}}_t \oslash \boldsymbol{\theta}^{\text{sim}}(c_t) - \mathbf{1}, \quad (3)$$

where $\oslash$ is element-wise division. A navigation stack deploys some interface model $\boldsymbol{\theta}_t^M$, which may be static, switched, or tracked. Its one-step prediction $\hat{\boldsymbol{\nu}}_k(\boldsymbol{\theta}^M)$ from (1) yields the normalized residual

$$\rho_k = \frac{\|\boldsymbol{\nu}_k - \hat{\boldsymbol{\nu}}_k(\boldsymbol{\theta}^M)\|_W}{\max(\|\mathbf{u}_k\|_W, \, u_{\min})}, \quad (4)$$

where $\|\cdot\|_W$ is a weighted norm that converts yaw rate to an equivalent edge speed through the body half-length, and $u_{\min}$ prevents division by near-zero commands. Let $\bar{\rho}_i$ be the mean of ρ_k over the i-th distance bin, centered at odometric distance s_i (in km). The *interface drift* D of the deployed model is the scaled least-squares slope β_1 of the binned residual against distance,

$$(\beta_0, \beta_1) = \arg\min_{\beta_0, \beta_1} \sum_i \left(\bar{\rho}_i - \beta_0 - \beta_1 s_i\right)^2, \quad (5)$$

and $D = 100\,\beta_1$ is reported in percent per kilometer. A model that stays valid along the traverse has D close to zero; a model that becomes stale as terrain changes accumulates residual and has larger D.

3.3 Range realizability

Let $\mathcal{C}$ be an AgniNav-style configuration [9] with body envelope $\mathcal{B}$ (front length, rear length, half width, height), sensor model, and nominal command limits. Let $\mathcal{U}(\mathcal{C})$ be the planner's candidate command set over horizon T, $\mathcal{O}_t$ the obstacle set from the LiDAR scan, and $\boldsymbol{\xi}^{\text{nom}}(\mathbf{u})$ the trajectory obtained by integrating $\mathbf{u}$ kinematically, as the nominal planner assumes. The perceived free set is

$$\mathcal{U}_t^{\text{free}} = \{\mathbf{u} \in \mathcal{U}(\mathcal{C}) : (\boldsymbol{\xi}^{\text{nom}}(\mathbf{u}) \oplus \mathcal{B}) \cap \mathcal{O}_t = \emptyset\}. \quad (6)$$

Under interface state $\boldsymbol{\theta}$, the realized trajectory $\boldsymbol{\xi}^{\boldsymbol{\theta}}(\mathbf{u})$ is obtained by propagating (1), and the stopping distance at forward speed v is

$$s_{\text{stop}}(v; \boldsymbol{\theta}) = v \, (d_x \Delta t + \tau_x), \quad (7)$$

the distance covered during the transport delay plus the integral of the exponential decay of a first-order response commanded to zero. A candidate is *realizable* if its realized trajectory, extended by the stopping segment of length s_{stop} and inflated by a margin m, is collision free. Writing $\mathcal{U}_t^{\text{real}}(\boldsymbol{\theta}) \subseteq \mathcal{U}_t^{\text{free}}$ for the realizable subset and $\ell(\mathbf{u})$ for the nominal path length of a candidate, the *range-realizability fraction* is

$$R_t(\boldsymbol{\theta}) = \frac{\sum_{\mathbf{u} \in \mathcal{U}_t^{\text{real}}(\boldsymbol{\theta})} \ell(\mathbf{u})}{\sum_{\mathbf{u} \in \mathcal{U}_t^{\text{free}}} \ell(\mathbf{u})} \in [0, 1]. \quad (8)$$

Weighting by path length makes R_t a measure of how much perceived free *range* is reachable, rather than how many candidates survive. The reported realizability is the time average of R_t, evaluated with the true interface reconstructed offline from the logged twist, so that methods are compared on how well their deployed models preserve reachable range.

3.4 Problem statement

Given a traverse that crosses terrain boundaries, the objective is (i) to estimate $\hat{\boldsymbol{\theta}}_t$ online with drift D as small as possible, (ii) to quantify the gap $\boldsymbol{\gamma}_t$ against simulation, and (iii) to select commands from $\mathcal{U}_t^{\text{real}}(\hat{\boldsymbol{\theta}}_t)$ so as to maximize navigation success, where an episode succeeds if the robot reaches the segment goal without collision, fall, operator intervention, or timeout.

04 Method

4.1 Overview and timing

The system has three layers. The locomotion controller is the unmodified onboard velocity-tracking policy. The *interface tracker* estimates $\hat{\boldsymbol{\theta}}_t$ from command and twist logs and publishes an update at 2 Hz. The *realizability predictor* and local planner run at 10 Hz: at each cycle they read the latest $\hat{\boldsymbol{\theta}}_t$, build the AgniNav-style configuration, score R_t, and select the next command. The rate separation reflects the time scales involved: identifying lag and gain requires several hundred milliseconds of excitation, whereas obstacle proximity changes on the planning time scale.

4.2 Recursive interface estimation

For a fixed delay d_j and saturation level $\bar{u}_j$, model (1) is linear in $\boldsymbol{\vartheta}_j = (\alpha_j, \beta_j)^\top$ with $\beta_j = (1 - \alpha_j)K_j$ and regressor $\boldsymbol{\phi}_k = (\nu_k^j, \sigma_{\bar{u}_j}(u_{k-d_j}^j))^\top$. Recursive least squares with forgetting factor $\lambda \in (0, 1]$ [29, 30] gives

$$e_k = \nu_{k+1}^j - \boldsymbol{\phi}_k^\top \hat{\boldsymbol{\vartheta}}_k, \quad (9)$$

$$\mathbf{g}_k = \frac{\mathbf{P}_k \boldsymbol{\phi}_k}{\lambda + \boldsymbol{\phi}_k^\top \mathbf{P}_k \boldsymbol{\phi}_k}, \quad (10)$$

$$\hat{\boldsymbol{\vartheta}}_{k+1} = \hat{\boldsymbol{\vartheta}}_k + \mathbf{g}_k e_k, \quad (11)$$

$$\mathbf{P}_{k+1} = \lambda^{-1}\left(\mathbf{P}_k - \mathbf{g}_k \boldsymbol{\phi}_k^\top \mathbf{P}_k\right), \quad (12)$$

from which $\hat{\tau}_j = -\Delta t / \ln \hat{\alpha}_j$ and $\hat{K}_j = \hat{\beta}_j/(1 - \hat{\alpha}_j)$. The discrete parameters are handled by a small bank of estimators over a grid of $(d_j, \bar{u}_j)$ values; the published estimate is the bank member with the lowest exponentially weighted residual.

KAGGLEYEs ACADEMY • E-PRINT REGISTRY • KAGG:2607.00006

A bank is preferred to a nonlinear joint estimator because it keeps each member convex and makes delay changes, which are common at terrain transitions, observable as a switch of the winning member rather than as a slow drift of a continuous parameter. The forgetting factor sets the effective memory to roughly $1/(1-\lambda)$ samples, which trades tracking speed against noise from gait oscillation.

4.3 Planner-conditioned probing

Ordinary navigation commands are often nearly constant, which leaves (12) poorly excited. Following GAUGE [8], a small probe $\boldsymbol{\epsilon}_k$ is superimposed on the planned command, $\mathbf{u}_k = \mathbf{u}_k^{\text{plan}} + \boldsymbol{\epsilon}_k$, and chosen to maximize information about the parameters the planner relies on:

$$\boldsymbol{\epsilon}_k^\star = \arg\max_{\boldsymbol{\epsilon}\in\mathcal{E}_k} \log\det\!\left(\mathbf{P}_k^{-1} + \lambda^{-1}\boldsymbol{\phi}_k(\boldsymbol{\epsilon})\boldsymbol{\phi}_k(\boldsymbol{\epsilon})^\top\right), \quad (13)$$

with the admissible set $\mathcal{E}_k = \{\boldsymbol{\epsilon} : |\epsilon^j| \le \bar{\epsilon}_j,\ \mathbf{u}_k^{\text{plan}} + \boldsymbol{\epsilon} \in \mathcal{U}_t^{\text{real}}(\hat{\boldsymbol{\theta}}_t)\}$. Two design choices follow from the navigation setting. First, the probe is conditioned on the planned command, so excitation is applied near the operating point that the next plan will use; this matters because saturation makes the interface operating-point dependent. Second, the probe is constrained to the realizable set, so calibration never trades safety for information. The optimization is solved by enumerating a small set of signed axis-aligned perturbations.

4.4 Change detection and re-initialization

Terrain boundaries produce abrupt parameter changes that a forgetting-factor estimator follows only slowly. A CUSUM detector [31] runs on the normalized innovation $\tilde{e}_k = e_k/\hat{\sigma}_e$:

$$g_k = \max\!\left(0,\ g_{k-1} + \tilde{e}_k^2 - \kappa\right), \quad (14)$$

where κ exceeds the expected value of $\tilde{e}_k^2$ under no change. When g_k exceeds a threshold h, the covariance is inflated, $\mathbf{P}_k \leftarrow \mathbf{P}_k + \mathbf{Q}_{\text{reset}}$, the estimate is blended toward the prior of the current terrain class (Section 4.5), and g_k is reset. The terrain class is inferred from proprioceptive slip and foot-sinkage statistics provided by the state estimator, so that re-initialization does not depend on the exteroceptive pipeline that is itself under evaluation.

4.5 Sim-to-real pre-alignment

For each terrain class c, the same probing protocol is executed in Isaac Lab [12] on matched terrain assets to obtain $\boldsymbol{\theta}^{\text{sim}}(c)$. Simulation reproduces the relative ordering of terrain effects better than their absolute magnitude, so the prior is aligned multiplicatively using the reference terrain on which hardware calibration is abundant (grass):

$$\boldsymbol{\theta}^{\text{prior}}(c) = \boldsymbol{\theta}^{\text{sim}}(c) \odot \left(\hat{\boldsymbol{\theta}}^{\text{real}}(\text{grass}) \oslash \boldsymbol{\theta}^{\text{sim}}(\text{grass})\right), \quad (15)$$

with $\odot$ element-wise multiplication; delays are rounded to the nearest sample. The prior seeds the tracker at each detected boundary, and the gap (3) is always reported against the unaligned $\boldsymbol{\theta}^{\text{sim}}$, so that pre-alignment improves control without hiding the measured gap.

4.6 Realizability predictor and planner integration

At each 10 Hz cycle, the predictor instantiates the AgniNav-style configuration $\mathcal{C}_t$ with the fixed envelope $\mathcal{B}$ and with the effective limits implied by $\hat{\boldsymbol{\theta}}_t$, namely $\hat{K}_j\bar{u}_j$ per axis. It enumerates $\mathcal{U}(\mathcal{C})$, computes the perceived free set, propagates every free candidate through (1) with $\hat{\boldsymbol{\theta}}_t$, appends the stopping segment (7), and evaluates (8). Three uses of the result close the loop. First, commands are pre-compensated for gain,

$$u_k^{j,\text{cmd}} = \text{clip}\!\left(u_k^{j,\text{des}}/\hat{K}_j,\ -\bar{u}_j,\ \bar{u}_j\right), \quad (16)$$

so that the realized twist approaches the desired one within the unsaturated range. Second, the planner selects $\mathbf{u}^\star = \arg\min_{\mathbf{u}\in\mathcal{U}_t^{\text{real}}(\hat{\boldsymbol{\theta}}_t)} J(\mathbf{u})$ with the cost

$$J(\mathbf{u}) = w_g\,\delta_{\text{goal}}(\mathbf{u}) + w_s\|\mathbf{u} - \mathbf{u}_{k-1}\|^2 + \frac{w_c}{\text{clr}(\mathbf{u})}, \quad (17)$$

where $\delta_{\text{goal}}(\mathbf{u})$ is the distance from the end of the realized trajectory $\boldsymbol{\xi}^{\hat{\boldsymbol{\theta}}}(\mathbf{u})$ to the local goal, $\text{clr}(\mathbf{u})$ is its minimum clearance, and w_g, w_s, w_c are positive weights. Third, when R_t falls below a threshold $R_{\min}$, the nominal forward limit is scaled as $\bar{v}_{\text{eff}} = \bar{v}\cdot R_t/R_{\min}$ and the tracker is asked to prioritize probing, because a low realizability fraction signals either an obstacle-dense scene or an interface estimate that has become stale.

4.7 Design rationale

The lag-gain-saturation model is deliberately low dimensional. A richer learned model could fit the hardware response more closely, but its parameters would not be comparable with a simulated counterpart, and the gap (3) would lose its interpretation. The model also yields the closed-form stopping distance (7) that the predictor evaluates for every candidate at 10 Hz. Keeping the locomotion controller unmodified isolates the interface gap as a property of the controller-terrain pair rather than of a co-adapted stack.

05 Experiments

5.1 Platform and course

All experiments use an ANYmal quadruped with its onboard velocity-tracking locomotion controller and LiDAR-based local perception. The terrain-transition course consists of five 30 m segments in the order grass, loose gravel, sand trough, packed dirt, and grass, for 150 m per crossing. The course was crossed 15 times in each direction. The interface model was tracked with the online GAUGE-style calibration at 2 Hz, and AgniNav-style range realizability was scored at 10 Hz. The sim-to-real comparison used Isaac Lab with terrain assets matched to the course. Table 1 summarizes the setup.

Table 1: Experimental setup.

Item	Setting
Platform	ANYmal quadruped
Course	Grass, loose gravel, sand trough, packed dirt, grass
Segment length	30 m (150 m per crossing)
Crossings	15 per direction
Interface tracking	Online GAUGE-style calibration, 2 Hz
Realizability scoring	AgniNav-style predictor, 10 Hz
Simulation	Isaac Lab, matched terrain assets

5.2 Baselines and metrics

Four interface models are compared, all feeding the same local planner. (i) *Static interface model*: calibrated on grass only and held fixed. (ii) *Re-calibrate per segment (offline)*: a separate model per segment identified offline and switched at segment

KAGGLEYEs ACADEMY • E-PRINT REGISTRY • KAGG:2607.00006

boundaries. (iii) *Simulation-only gap model*: the per-terrain $\boldsymbol{\theta}^{\text{sim}}(c)$ from Isaac Lab without hardware correction. (iv) *Ours*: online tracking with planner-conditioned probing, change detection, sim-to-real pre-alignment, and realizability-aware planning. The metrics are interface drift D from (5) in percent per kilometer, range realizability from (8) in percent, navigation success in percent, and the 95th percentile of planar tracking error in centimeters. Navigation success is reported as a single rate per configuration; the static grass-calibrated model represents the stack before the diagnostics are applied, and the full method represents the stack after.

5.3 Main results

Table 2 reports the main comparison. The full method obtains the best value on every metric. Relative to the static grass-calibrated model, drift falls from 4.9 to 4.6 %/km, range realizability rises from 70.4 to 74.8 % (4.4 percentage points), navigation success rises from 80.8 to 86.3 % (5.5 percentage points), and the 95th-percentile tracking error falls from 17.9 to 15.9 cm, a reduction of 2.0 cm or about 11 %.

The differences in drift are small in absolute terms, 0.3 %/km between the static model and the full method and 0.6 %/km between the simulation-only model and the full method, and confidence intervals were not computed for this metric; the drift result should therefore be read as consistent with, rather than proof of, the benefit of online tracking. The realizability and success differences are larger and point in the same direction.

Two baseline results deserve comment. Offline per-segment re-calibration reduces tracking error relative to the static model (16.6 versus 17.9 cm) yet yields lower navigation success (75.1 versus 80.8 %) and slightly lower realizability (69.7 versus 70.4 %). One interpretation consistent with the failure logs is that per-segment parameters are averages: within the sand trough the interface keeps changing with distance, and a model that is switched at a boundary while being correct on average can still be confidently wrong at the moments that decide an episode. The simulation-only model has the largest drift (5.2 %/km) and tracking error (18.4 cm) and the lowest realizability (69.1 %), which indicates that matched terrain assets alone do not reproduce the hardware interface on deformable ground, in line with the observation that granular contact requires dedicated modeling [21].

5.4 Ablation

Table 3 removes one component at a time. Removing online drift tracking, which leaves a static model, raises drift to 5.3 %/km and lowers success by 12.0 percentage points, to 74.3 %. Removing sim-to-real pre-alignment has the largest effect on success, a drop of 13.8 points to 72.5 %, with drift 5.1 %/km. Removing the realizability score has the smallest effect, a drop of 7.5 points to 78.8 %, with drift 5.0 %/km.

The ablation variant without drift tracking performs worse than the static baseline of Table 2 (74.3 versus 80.8 % success, 5.3 versus 4.9 %/km drift), although both use a fixed model. The two configurations are not identical: in the ablation, gain pre-compensation (16), realizability-based speed scaling, and the pre-aligned prior remain active but act on a model that is never updated, so a stale estimate is given more authority over the commands than in the plain static baseline. This reading is plausible but was not isolated experimentally. A practical consequence is that interface-aware planning components should not be deployed without the tracker that keeps their model current. The large effect of removing pre-alignment suggests that the tracker depends on a good initialization at boundaries: without it, the estimator re-converges from a model that is far from the new terrain, and the transient is long enough to affect episodes.

5.5 Failure cases

Table 4 lists the recorded failure cases.

Progressive lag growth in sand. In the sand trough the interface lag grew by a factor of 3.2 over 200 m of traversal. Because the stopping distance (7) scales linearly with τ_x and the delay, a model calibrated on grass underestimates braking distance in sand, and the static model mispredicted stop distance by 1.1 m in 6 of 15 crossings (40 %). This case is the clearest evidence for the central claim of the paper: the gap is not a fixed offset per terrain but grows with distance, as displaced material changes the substrate under the robot. It also explains why per-segment offline calibration does not suffice.

Transient at the gravel-to-sand boundary. At the transition from loose gravel to sand, the realizability fraction dropped to 41 % for 4 s before the online tracker re-converged. During this interval the speed scaling of Section 4.6 slowed the robot, which is the intended conservative behavior, but the episode time and the realizable range both suffered. Shortening this transient requires either a better boundary prior or anticipatory detection of the terrain change from exteroception, which the present system deliberately does not use for re-initialization.

Sensor degradation. In one crossing, mud specks on the LiDAR degraded the range estimate, which fell back to the odometry source, recorded in the log as wheel odometry, for 2 s; the event was flagged in the log. This failure lies outside the interface model: the realizability predictor assumes a valid free-space estimate, and its output is only as reliable as $\mathcal{U}_t^{\text{free}}$. It illustrates that the interface and perception gaps interact, even though the method quantifies only the former.

06 Limitations

The study has several limitations. First, it uses a single platform, a single course, and 15 crossings per direction; the reported rates are not accompanied by confidence intervals, and the small drift differences in particular should not be over-interpreted. Second, the interface model (1) is decoupled per axis and first order; cross-coupling between yaw rate and lateral slip on loose ground is not represented. Third, the realizability fraction is computed from the same interface model that the planner uses, so it measures internal consistency of the predictor with respect to the reconstructed interface rather than an independent ground truth of reachability. Fourth, terrain classification for re-initialization relies on proprioceptive statistics that can lag the true boundary, which contributes to transients such as the 4 s drop at the gravel-to-sand boundary. Fifth, the discrepancy between the ablated static variant and the static baseline indicates that component interactions are not fully characterized. Finally, only sand, gravel, and packed dirt were tested; snow and mud, which motivate the problem, remain to be evaluated, and the single mud-contaminated crossing concerned the sensor rather than the substrate.

KAGGLEYEs ACADEMY • E-PRINT REGISTRY • KAGG:2607.00006

Table 2: Main comparison on the terrain-transition course. Navigation success is the recorded rate for each configuration; the static model corresponds to the stack before the diagnostics and the full method to the stack after.

Method	Interface drift (%/km) ↓	Range realizability (%) ↑	Navigation success before/after (%) ↑	Tracking error p95 (cm) ↓
Static interface model (calibrated on grass only)	4.9	70.4	80.8	17.9
Re-calibrate per segment (offline)	4.9	69.7	75.1	16.6
Simulation-only gap model	5.2	69.1	76.5	18.4
Ours	**4.6**	**74.8**	**86.3**	**15.9**

Table 3: Ablation of the full method.

Variant	Interface drift (%/km)	Navigation success before/after (%)
Ours (online drift tracking)	4.6	86.3
w/o drift tracking (static model)	5.3	74.3
w/o realizability score	5.0	78.8
w/o sim-to-real pre-alignment	5.1	72.5

07 Conclusion

This paper quantified the sim-to-real gap of a quadruped velocity interface on deformable terrain by tracking a calibrated lag-gain-saturation model online with planner-conditioned probing, re-initializing it from a sim-to-real pre-aligned prior at terrain changes, and scoring how much perceived free range remains realizable under the current interface. On an ANYmal crossing a five-segment course, the combined diagnostics reached 86.3 % navigation success compared with 80.8 % for a static grass-calibrated model, with higher range realizability (74.8 versus 70.4 %), lower tracking error (15.9 versus 17.9 cm), and modestly lower interface drift (4.6 versus 4.9 to 5.2 %/km across baselines). The failure cases, in particular the 3.2-fold lag growth in sand, show that the interface gap evolves with distance, which favors continuous measurement over per-terrain calibration. Future work will extend the model to coupled axes, test snow and mud, and use exteroceptive cues to anticipate boundaries.

References

[1] M. Hutter, C. Gehring, D. Jud, A. Lauber, C. D. Bellicoso, V. Tsounis, J. Hwangbo, K. Bodie, P. Fankhauser, M. Bloesch, R. Diethelm, S. Bachmann, A. Melzer, and M. Hoepflinger. ANYmal - A Highly Mobile and Dynamic Quadrupedal Robot. IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2016.

[2] 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.

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

[4] 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.

[5] L. Wellhausen, A. Dosovitskiy, R. Ranftl, K. Walas, C. Cadena, and M. Hutter. Where Should I Walk? Predicting Terrain Properties from Images via Self-Supervised Learning. IEEE Robotics and Automation Letters, 4(2):1509-1516, 2019.

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

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

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

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

[12] M. Mittal, C. Yu, Q. Yu, J. Liu, N. Rudin, D. Hoeller, J. L. Yuan, R. Singh, Y. Guo, H. Mazhar, A. Mandlekar, B. Babich, G. State, M. Hutter, and A. Garg. Orbit: A Unified Simulation Framework for Interactive Robot Learning Environments. arXiv:2301.04195, 2023.

[13] 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.

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

[15] 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.

[16] W. Yu, J. Tan, C. K. Liu, and G. Turk. Preparing for the Unknown: Learning a Universal Policy with Online System Identification. arXiv:1702.02453, 2017.

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

[18] G. Ji, J. Mun, H. Kim, and J. Hwangbo. Concurrent Training of a Control Policy and a State Estimator for Dynamic and Robust Legged Locomotion. arXiv:2202.05481, 2022.

[19] A. Nagabandi, I. Clavera, S. Liu, R. S. Fearing, P. Abbeel, S. Levine, and C. Finn. Learning to Adapt in Dynamic, Real-World Environments Through Meta-Reinforcement Learning. arXiv:1803.11347, 2018.

[20] W. Zhao, J. P. Queralta, and T. Westerlund. Sim-to-Real Transfer in Deep Reinforcement Learning for Robotics: a Survey. arXiv:2009.13303, 2020.

[21] S. Choi, G. Ji, J. Park, H. Kim, J. Mun, J. H. Lee, and J. Hwangbo. Learning Quadrupedal Locomotion on Deformable Terrain. Science Robotics, 8(74):eade2256, 2023.

[22] G. B. Margolis, G. Yang, K. Paigwar, T. Chen, and P. Agrawal. Rapid Locomotion via Reinforcement Learning. arXiv:2205.02824, 2022.

KAGGLEYEs ACADEMY • E-PRINT REGISTRY • KAGG:2607.00006

Table 4: Recorded failure cases.

Location	Observation	Interpretation
Sand trough	Interface lag grew 3.2x over 200 m; the static model mispredicted stop distance by 1.1 m in 6 of 15 crossings.	Progressive, within-segment drift that a fixed model cannot represent.
Gravel-to-sand boundary	Realizability fraction dropped to 41% for 4 s before the online tracker re-converged.	Transient between change detection and re-convergence.
One crossing (mud specks on LiDAR)	The range estimate fell back to wheel odometry for 2 s (flagged in the log).	Perception degradation, outside the interface model.

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

[24] M. V. Gasparino, A. N. Sivakumar, Y. Liu, A. E. B. Velasquez, V. A. H. Higuti, J. Rogers, H. Tran, and G. Chowdhary. WayFAST: Navigation with Predictive Traversability in the Field. arXiv:2203.12071, 2022.

[25] G. Kahn, P. Abbeel, and S. Levine. BADGR: An Autonomous Self-Supervised Learning-Based Navigation System. arXiv:2002.05700, 2020.

[26] P. Fankhauser, M. Bloesch, and M. Hutter. Probabilistic Terrain Mapping for Mobile Robots with Uncertain Localization. IEEE Robotics and Automation Letters, 3(4):3019-3026, 2018.

[27] 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.

[28] M. Wermelinger, P. Fankhauser, R. Diethelm, P. Krüsi, R. Siegwart, and M. Hutter. Navigation Planning for Legged Robots in Challenging Terrain. IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2016.

[29] L. Ljung. System Identification: Theory for the User, 2nd ed. Prentice Hall, 1999.

[30] K. J. Åström and B. Wittenmark. Adaptive Control, 2nd ed. Addison-Wesley, 1995.

[31] E. S. Page. Continuous Inspection Schemes. Biometrika, 41(1/2):100-115, 1954.

[32] X. Xiao, J. Biswas, and P. Stone. Learning Inverse Kinodynamics for Accurate High-Speed Off-Road Navigation on Unstructured Terrain. arXiv:2102.12667, 2021.

[33] H. Karnan, K. S. Sikand, P. Atreya, S. Rabiee, X. Xiao, G. Warnell, P. Stone, and J. Biswas. VI-IKD: High-Speed Accurate Off-Road Navigation using Learned Visual-Inertial Inverse Kinodynamics. arXiv:2203.15983, 2022.

[34] D. Fox, W. Burgard, and S. Thrun. The Dynamic Window Approach to Collision Avoidance. IEEE Robotics and Automation Magazine, 4(1):23-33, 1997.

[35] G. Williams, N. Wagener, B. Goldfain, P. Drews, J. M. Rehg, B. Boots, and E. A. Theodorou. Information Theoretic MPC for Model-Based Reinforcement Learning. IEEE International Conference on Robotics and Automation (ICRA), 2017.

[36] D. Hoeller, L. Wellhausen, F. Farshidian, and M. Hutter. Learning a State Representation and Navigation in Cluttered and Dynamic Environments. arXiv:2103.04351, 2021.

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

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

[39] J. Truong, M. Rudolph, N. H. Yokoyama, S. Chernova, D. Batra, and A. Rai. Rethinking Sim2Real: Lower Fidelity Simulation Leads to Higher Sim2Real Transfer in Navigation. Proceedings of Machine Learning Research, 205:859-870, 2023.

[40] M. Bloesch, M. Hutter, M. A. Hoepflinger, S. Leutenegger, C. Gehring, C. D. Remy, and R. Siegwart. State Estimation for Legged Robots: Consistent Fusion of Leg Kinematics and IMU. Robotics: Science and Systems VIII, 2012.

KAGGLEYEs ACADEMY • E-PRINT REGISTRY • KAGG:2607.00006