LOAD-AWARE QUADRUPED NAVIGATION WITH ONLINE CONTACT-BASED GEOMETRY ESTIMATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Navigating cluttered environments while carrying objects remains a challenging problem for quadruped robots. Compared with works that focused on load-aware locomotion adaptation to maintain gait stability, existing autonomous navigation systems generally ignore the influence of carried objects' geometries and use fixed footprint inflation, leading to overly conservative navigation. To fill this knowledge gap, we propose a two-phase load-aware navigation learning framework. In Phase 1, we train LOADNav, a load-aware navigation policy via reinforcement learning, enabling the robot to adapt its trajectory and clearance behavior according to the geometric size of the carried object. In Phase 2, we train CoGE, an online geometry predictor via imitation learning, which estimates unknown payload geometry from occasional contact events. We evaluate the framework in cluttered environments with loads of varying sizes and shapes. The results show that our navigation policy exhibits clear load-aware behaviors, and the online geometry predictor estimates object geometry from occasional contact events, refining its estimate and adapting the navigation behavior accordingly. Without requiring any prior knowledge of the payload, it narrows the gap to geometry-aware policies.

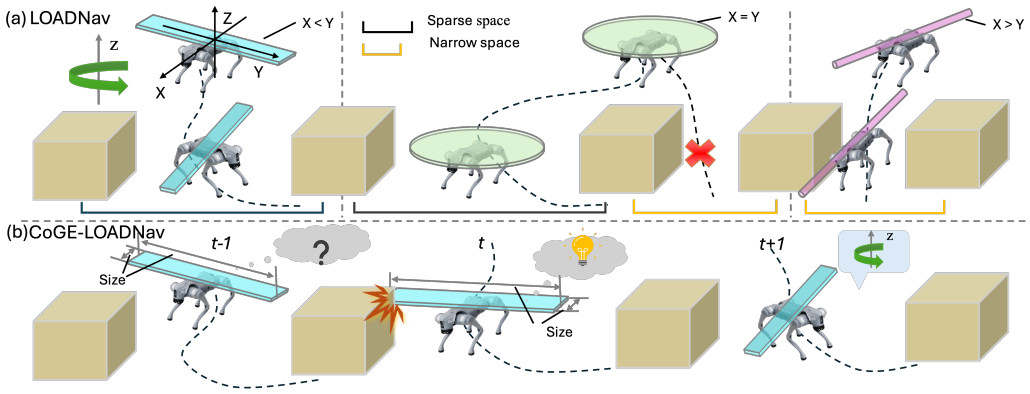

Figure 1: The illustration of showing our proposed two-phase load-aware navigation learning framework. (a) LOADNav (Phase 1): with privileged object size information, the policy learns load-aware navigation strategies, adapting its behavior across sparse and narrow spaces according to different object geometries ($X < Y$, $X = Y$, $X > Y$); (b) CoGE-LOADNav (Phase 2): the robot infers object geometry from occasional contact events, enabling adaptive navigation without size inputs.

1 INTRODUCTION

Quadruped robots have demonstrated promising potential in object transportation tasks (Hutter et al., 2016; Unitree Robotics, 2026a;b). Recent research studies have investigated the effect of carried objects on quadruped locomotion, focusing on maintaining stable gaits under varying payload conditions (Sombolestan & Nguyen, 2024; Amanzadeh et al., 2024; Xu et al., 2025a; Zhi et al., 2025), and improving controller robustness by modeling or predicting the dynamic properties of the payload (Chang et al., 2025; Martinez et al., 2025).

However, the impacts on navigation when the carried objects significantly exceed the robot's size remain unexplored. Transporting large objects substantially enlarges and irregularizes the robot's

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

geometric boundary, fundamentally altering the geometric constraints imposed on the navigation system. Although many previous studies have been made in autonomous navigation for quadruped robots (Wang et al., 2025b; Liu et al., 2025), existing approaches typically assume a fixed and compact robot size (Fu et al., 2022; Xu et al., 2025b; Wellhausen & Hutter, 2023), and therefore struggle to handle the irregular geometry introduced by large carried objects. Moreover, the geometric boundary of the robot varies with the shape and size of the carried object, requiring the navigation system to adapt to diverse geometric constraints. This challenge is further compounded by the fact that the geometry of the payload, mounted on the robot's back, lies in the blind spot of onboard visual and LiDAR sensors. Consequently, the system cannot directly perceive the object's geometry, necessitating inference through proprioceptive feedback.

To address these challenges, we propose a **two-phase load-aware navigation learning** framework. In Phase 1, we train a navigation policy called **LOADNav** via reinforcement learning that enabled the robot to avoid obstacles and reach target goals while carrying objects of varying geometries. The policy demonstrated emergent load-aware behaviors. For instance, a board mounted perpendicularly to the robot rotated sideways to pass through narrow passages, circular objects that uniformly enlarged the robot's bounding contour detoured around overly narrow regions, and tubes aligned with the robot's body shape sought the shortest traversable path, as illustrated in Fig. 1(a). In Phase 2, we train an online geometry predictor called **CoGE-LOADNav** via imitation learning, enabling the robot to estimate the geometry of the carried object during deployment without prior knowledge, as illustrated in Fig. 1(b). Results indicate that the predictor effectively estimated object size through occasional contact events. Importantly, our method does not intentionally induce collisions but leverages contact events when they occur to refine its geometry estimate and avoid future collisions. Our main contributions are as follows:

- We investigate how payload geometry affects navigation behavior and show that geometry-conditioned policies adapt their trajectories and orientations to different payloads.
- We analyze how payload geometry is identifiable only through occasional contact events. Unlike dynamic parameters that continuously affect proprioception, payload geometry becomes observable only through contact-induced motion deviations. Based on this property, we train CoGE, a contact-based online geometry estimator.
- We identify forgetting in finite-memory estimators. Once contact information leaves the observation window, the estimator inevitably loses the acquired geometry. We address this with a recurrent estimator that preserves contact information over time.
- We validate our framework in simulation and real-world experiments, demonstrating improved navigation performance over baselines and effective online geometry estimation.

2 Related Works

Load-Aware Locomotion. Payload carrying significantly affects the locomotion stability and control of quadrupedal robots, and existing work broadly addresses this challenge along three directions: model-based adaptive control (Sombolestan & Nguyen, 2024; Amanzadeh et al., 2024; Zeng et al., 2024; Minniti et al., 2021), online inertial parameter estimation (Martinez et al., 2025; Sobanbabu et al., 2025), and learning-based adaptation (Chang et al., 2025; Singh et al., 2024). On the model-based side, Sombolestan & Nguyen (2024) develop adaptive force-based controllers that leverage proprioceptive feedback to compensate for payload-induced disturbances, while Amanzadeh et al. (2024) integrate an indirect adaptive law with MPC to enable robust payload transport under terrain and load uncertainty. For online estimation, Martinez et al. (2025) demonstrate that inertial parameters of an unknown payload can be identified in real time on a Unitree Go1 and fed back into a full-dynamics MPC controller, yielding measurable improvements in posture stability and foot tracking. On the learning side, Chang et al. (2025) train an RL policy capable of inferring and stabilizing unknown dynamic loads using proprioception alone. Despite these advances, existing work focuses predominantly on how payload affects locomotion, while its impact on navigation remains largely unexplored, particularly when the carried object extends beyond the robot's own footprint. Moreover, existing payload estimation methods primarily estimate dynamic properties relevant to locomotion, such as mass and inertia, rather than the payload's geometry required for navigation.

Planning-based Navigation. Navigation algorithms for robots can be categorized into planning-based and learning-based approaches. Planning-based methods explicitly search for paths over a spatial representation of the environment by computing traversability or motion costs (Fu et al., 2022; Ren et al., 2025; Scheidemann et al., 2025; Wellhausen & Hutter, 2023; 2021; Luan et al., 2024; Han et al., 2025; Garg et al., 2025; Yin & Shi, 2026), whereas learning-based methods use neural networks to implicitly map perception inputs to navigation commands (Xu et al., 2025b; Zhang et al., 2024; Xu et al., 2025c; Yao et al., 2025; Huang et al., 2024; Wang et al., 2025b; Liu et al., 2025; Wang et al., 2025a; Zhang et al., 2025). Existing navigation approaches universally treat robot footprint as a fixed, compact constraint in the algorithm. For instance, VP-Nav (Fu et al., 2022) computes a distance-based penalty map that keeps the robot at a fixed clearance from obstacles, and ArtPlanner (Wellhausen & Hutter, 2023) assigns motion costs based on a fixed bounding volume enclosing the robot. On the learning-based side, NavRL (Xu et al., 2025b) enforces safety by constraining the action space through a fixed safety shield. However, this assumption of a fixed robot footprint breaks down when a carried object enlarges and irregularizes the robot's geometry.

Two-phase Learning. Two-phase learning has been widely adopted to facilitate sim-to-real transfer in quadruped robots (Kumar et al., 2021; Miki et al., 2022; Cheng et al., 2024; Loquercio et al., 2023; Lee et al., 2020; Agarwal et al., 2023; Nahrendra et al., 2023). In the first phase, the policy is trained with privileged information to accelerate learning, while in the second phase, imitation learning is employed to distill the policy into a deployable model that estimates the privileged features from limited proprioceptive inputs. In legged locomotion, RMA (Kumar et al., 2021) regresses a latent encoding of environment factors from the robot's state-action history, Miki et al. (2022) train robots to handle noisy height map inputs through a two-phase framework, and Agarwal et al. (2023) enable height feature estimation from depth image inputs. Beyond locomotion, this paradigm has also been applied in manipulation (Lum et al., 2025; Torne et al., 2024; Chen et al., 2023; 2025; Fuchioka et al., 2024; Yoo et al., 2025), where DextrAH-G (Lum et al., 2025) estimates object pose from depth images. However, in these studies, the privileged information continuously affects the robot's dynamics or perception, such that each step of the proprioceptive history contains information about them. In contrast, the payload geometry considered in our work does not continuously affect proprioception and is revealed only through intermittent contact with the environment.

3 METHOD

3.1 PROBLEM FORMULATION

We consider a quadruped robot navigating from a start position to a goal position while carrying a payload mounted on its back. The navigation task is defined as a 2D point navigation problem: given a planar (xy) goal position $\mathbf{g} \in \mathbb{R}^2$, the robot must reach $\mathbf{g}$ while avoiding collisions with obstacles in the environment. The payload may change the robot's planar physical boundary, introducing geometric constraints that vary with the payload's planar geometric shape and size. We define the robot's local coordinate frame with the heading direction as the x-axis and the lateral direction as the y-axis. The payload is assumed to be rigidly and centrally symmetric attached to the robot's back without relative motion or rotation. The payload geometry is characterized by two parameters: $\mathbf{s} = (s^x, s^y) \in \mathbb{R}^2$, where s_x and s_y denote the absolute extent of the payload along the x and y axes in the robot's local frame, respectively. Based on the geometric relationship between the payload and the robot's heading direction, we define six representative payload types, as illustrated in Fig. 2. These payload types impose distinct geometric constraints on navigation, motivating our proposed two-phase load-aware navigation framework detailed in the following sections.

3.2 OVERVIEW FRAMEWORK

We propose a two-phase load-aware navigation learning framework, as illustrated in Fig. 2. The policy network takes as input the robot's proprioceptive states and a local occupancy map constructed from onboard sensors. In the first phase, we train a load-aware navigation policy via reinforcement learning, enabling the robot to acquire distinct navigation strategies for different payload types, where the payload geometry $\mathbf{s}$ is provided as privileged information. In the second phase, to further enhance deployment flexibility, the trained policy network was frozen, and a contact-based online geometry predictor is trained via imitation learning to estimate $\mathbf{s}$ from onboard proprioceptive sensing, enabling load-aware navigation without prior geometric knowledge.

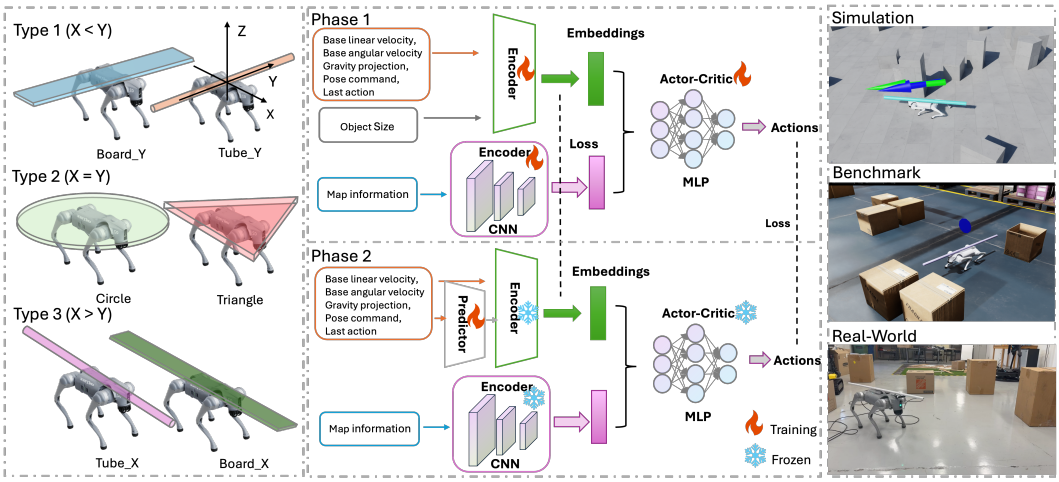

Figure 2: Overview of our proposed navigation framework. (Left) Three object geometry types categorized by their X and Y dimensions. (Middle) Two-phase training pipeline: Phase 1 trains the load-aware navigation policy with privileged object size input via Reinforcement Learning, while Phase 2 trains the contact-based predictor via Imitation Learning with the main policy network frozen. (Right) Evaluation across simulation, benchmark, and real-world environments.

3.3 Phase 1: Load-Aware Navigation Policy

Observation. The observation space consists of three components: $\mathbf{o} = (\mathbf{o}^p, \mathbf{o}^s, \mathbf{o}^m)$. The proprioceptive state $\mathbf{o}_p$ includes the robot's base linear and angular velocities over the past 5 frames, the current projected gravity vector, the relative goal position $\mathbf{p}_b^{g,xy} \in \mathbb{R}^2$, and the previous action. The payload geometry $\mathbf{o}^s = (s^x, s^y) \in \mathbb{R}^2$ encodes the geometric extent of the payload along the robot's local x and y axes. The map observation $\mathbf{o}_m$ is a local occupancy map centered on the robot, spanning $[-2.5, 2.5]$ m along both axes with a resolution of 0.05 m, resulting in a 100×100 grid flattened into a 10000-dimensional vector. The total observation dimension is 10040.

Action. The action space consists of a 3-dimensional velocity command $\mathbf{a} = (v^x, v^y, \omega) \in \mathbb{R}^3$, representing the desired linear velocities along the x axis, y axis, and the angular velocity. These commands are executed by our pre-trained flat locomotion controller based on the default velocity tracking task from Isaac Lab (NVIDIA, 2026).

Network Architecture. The policy network adopts a dual-branch actor-critic architecture. The proprioceptive branch processes $(\mathbf{o}^p, \mathbf{o}^s)$ through a two-layer MLP encoder, while the map branch processes $\mathbf{o}^m$ through a three-layer CNN followed by a linear projection. The outputs of both branches are concatenated and fed into independent actor and critic heads, outputting the velocity command $\mathbf{a}$ and state value, respectively. The entire network is trained using Proximal Policy Optimization (PPO) (Schulman et al., 2017), with detailed hyperparameters provided in the Appendix Table 4.

Reward Functions. The reward function consists of two components: terminal rewards r_{terminal} triggered by episode termination conditions, and dense rewards r_{dense} that provide continuous feedback during navigation:

$$r = r_{\text{terminal}} + r_{\text{dense}} \tag{1}$$

The terminal rewards are defined as:

$$r_{\text{terminal}} = \lambda_1 r_{\text{succ}} + \lambda_2 r_{\text{timeout}} + \lambda_3 r_{\text{robot}} + \lambda_4 r_{\text{stuck}}, \tag{2}$$

The dense rewards are defined as:

$$r_{\text{dense}} = \lambda_5 r_{\text{obj}} + \lambda_6 r_{\text{vel}} + \lambda_7 r_{\text{head}} + \lambda_8 r_{\text{step}}, \tag{3}$$

The terminal rewards encode four termination conditions: a success reward r_{succ} upon goal reaching, and penalties for timeout r_{timeout}, robot body contact with the environment r_{robot}, and becoming stuck r_{stuck}, which is triggered when both the planar velocity and angular velocity of the robot fall below their respective thresholds within a single timestep. The dense rewards include a velocity projection

reward r_{vel} that encourages the robot to move toward the goal, a heading alignment reward r_{head} that encourages the robot to move along its forward heading direction, avoiding backward or sideways locomotion, an object collision penalty r_{obj} that penalizes contacts between the loaded object and the environment, and a per-step time penalty r_{step} that encourages the robot to complete the task in fewer steps. Detailed formulations and reward weights are provided in the Appendix Section A.1.

Navigation Curriculum. We design a Navigation Curriculum to progressively expose the robot to increasingly challenging navigation scenarios. The training terrain is organized as a 10×30 grid, where each terrain cell contains obstacles at random positions. Obstacle count increases linearly along the row direction, with each row corresponding to a distinct difficulty level. At the beginning of each episode, the robot's initial position and goal are randomly sampled within the current terrain cell, subject to a minimum clearance constraint. Each robot independently maintains its curriculum level l_i within its assigned column. A sliding window of length N tracks the robot's recent navigation trials, and when the window is full, the curriculum level is updated according to:

$$l_i' = \begin{cases} l_i + 1, & s_i \geq \tau_{\text{succ}} \\ l_i - 1, & s_i \leq \tau_{\text{fail}} \\ l_i, & \tau_{\text{fail}} < s_i < \tau_{\text{succ}} \end{cases}, \tag{4}$$

where s_i is the success rate within the current window, and τ_{succ} and τ_{fail} are the promotion and demotion thresholds respectively. Following each level transition, a cooldown period is enforced during which the curriculum level remains fixed, preventing boundary oscillation. Detailed curriculum parameters are provided in the Appendix Table 5.

3.4 Phase 2: Contact-based Online Geometry Predictor

Before introducing CoGE, we first clarify how the payload geometry s differs from the privileged information estimated in existing two-phase methods, and why this distinction calls for event-driven estimation. Let q_t denote the robot pose, O the obstacle set, and $B(q_t, s) = q_t \oplus (B_0 \cup F(s))$ the effective footprint formed by the robot body B_0 and payload $F(s)$. Contact introduces a disturbance to the nominal locomotion, such that $q_{t+1} = f(q_t, a_t) + d_t$, where $d_t \neq 0 \iff B(q_t, s) \cap O \neq \emptyset$. We assume that the payload is rigidly attached, has negligible dynamic effects (mass fixed at $0.1\,\text{kg}$), and lies outside the sensor field of view; thus, o_t^m is independent of s.

Proposition. At any contact-free time step t, for arbitrary s and s', $p(o_t^p \mid h_{t-1}, a_{t-1}, s) = p(o_t^p \mid h_{t-1}, a_{t-1}, s')$. Hence, the Fisher information of o_t^p about s is zero, i.e., $\mathcal{I}_t(s) = 0$.

Proof. Without contact, $d_t = 0$, making the transition dynamics independent of s. Since o_t^p consists of velocity, projected gravity, goal command, and the previous action, it is also independent of s. Therefore, the proposition holds. Consequently, the cumulative information about s along a trajectory satisfies

$$\sum_t \mathcal{I}_t(s) = \sum_{t \in \mathcal{T}_c} \mathcal{I}_t(s), \tag{5}$$

where $\mathcal{T}_c$ denotes the set of contact times. Information about s therefore increases only upon contact. Before the first contact along an axis, its estimation error cannot improve beyond the prior variance, e.g.,

$$\mathbb{E}\left[(\hat{s}_x - s_x)^2\right] \geq \text{Var}(s_x). \tag{6}$$

In existing two-phase methods, privileged information ξ continuously affects the transition dynamics $f(q_t, a_t, \xi)$, providing information at every time step. In contrast, information about s arises only from contact events. The geometry estimator must therefore detect contact and update accordingly. To enable deployment flexibility without requiring prior knowledge of the object geometry, we introduce a contact-based online geometry predictor CoGE that estimates the object size $\hat{o}_t^s$ from occasional contact events. The predicted geometry $\hat{o}_t^s$ is then concatenated with the proprioceptive state and fed into the frozen policy network to produce the action $\hat{a}_t = \pi_\theta(o_t^p | \hat{o}_t^s | o_t^m)$. The predictor is trained via imitation learning with the Phase 1 load-aware navigation policy π_θ frozen. Two MSE losses are jointly optimized $\mathcal{L} = \mathcal{L}_{\text{obj}} + \mathcal{L}_a$: a geometry loss $\mathcal{L}_{\text{obj}} = |\hat{o}_t^s - o_t^s|^2$ that supervises the predicted object size against the ground truth, and an action loss $\mathcal{L}_a = |\hat{a}_t - a_t|^2$ that encourages the Phase 2 policy to reproduce actions of Phase 1 $a_t = \pi_\theta(o_t^p | o_t^s | o_t^m)$. The predictor is optimized using the Adam optimizer, with all other network weights frozen.

4 EXPERIMENTAL RESULTS

4.1 EXPERIMENTAL SETTING

All policies are trained in NVIDIA Isaac Lab using 512 parallel Unitree Go2 robots on a single RTX 4090 GPU. Phase 1 trains the load-aware navigation policy on a 10×30 terrain grid with the Navigation Curriculum for 3000 episodes, requiring approximately 10 hours. Phase 2 trains the contact-based online geometry predictor on a 10×10 terrain grid with fixed obstacle density of 80 for 1000 episodes, requiring approximately 4 hours. The loaded object is rigidly attached to the robot and comprises six shape categories with randomly sampled dimensions; detailed size ranges are provided in the Appendix Table 6. The robot's linear velocity was clipped to $[-1.5, 1.5]$ m/s and angular velocity to $[-1.0, 1.0]$ rad/s. We evaluated all methods using two metrics: 1) **Success Rate (SR)**, defined as the percentage of episodes in which the robot reaches within 0.5m of the goal; 2) **Average Collision Times (ACT)**, measuring the average number of collisions involving both the robot body and the loaded object per successful trial. We compared against the following methods: 1) VP-Nav (Fu et al., 2022): A planning-based navigation policy with occupancy map inflation and the Obstacle Distance Map design. 2) VP-Nav-I (inflated): the same planner with the inflation radius enlarged to cover the payload. 3) NavRL (Xu et al., 2025b): A reinforcement learning-based navigation policy with safety rewards and Policy Action Safety Shield. 4) LOADNav (with given object size): Our load-aware policy that requires object size as input. 5) LOADNav-B (size-blind): LOADNav trained without payload geometry input. 6) CoGE-LOADNav: Our full method equipped with a contact-based online geometry predictor with finite memory. 7) CoGE-LOADNav+ (optimized): an improved variant with recurrent temporal training.

4.2 EVALUATION IN THE SIMULATOR

4.2.1 LOAD-AWARE NAVIGATION PERFORMANCE

This experiment serves two purposes: (1) to show that LOADNav exhibits load-aware navigation behavior, adapting its strategy to the geometry of the carried payload, and (2) to verify that LOADNav performs consistently across payload types and obstacle densities, and that CoGE-LOADNav approaches this performance without requiring the geometry input.

Load-aware behavior. The trajectories and velocity profiles in Fig. 3 demonstrate that LOADNav exhibits load-aware navigation behavior. Given the same environment, start, and goal, the policy adapts its strategy according to the geometry of the carried payload. For Tube_X, which is aligned with the robot's heading, the robot selects the shortest path through relatively narrow passages, as the elongated payload imposes minimal lateral clearance constraints. For Circle, whose extent is uniform in all directions, the robot avoids narrow spaces and takes a wider detour. When carrying Board_Y, oriented perpendicular to the heading, the robot likewise detours; however, the velocity profile shows an additional angular-velocity peak when passing between obstacles, indicating that the robot actively rotates its body to avoid collision and prevent the board from getting stuck.

Performance across payload types and densities. We evaluate LOADNav and CoGE-LOADNav across six payload types and their mixture under obstacle densities from $N = 20$ to $N = 80$ on a 5×5 terrain grid with 50 robots over 200 episodes; full results are given in Appendix Table 7. LOADNav achieves an average success rate of **91.26%** across all settings and remains robust under dense obstacles. Without any geometry input, CoGE-LOADNav achieves **89.74%**, within 1.6 percentage points of LOADNav; its higher ACT reflects the predictor's reliance on contact feedback and is analyzed in Section 4.2.2.

4.2.2 CONTACT-BASED ONLINE GEOMETRY ESTIMATION ANALYSIS

Figure 4 illustrates the geometry estimation process of CoGE-LOADNav on a representative episode. Initially, the predictor has no prior knowledge of the board geometry in the perpendicular direction, resulting in large estimation errors. Upon each contact event, the predictor receives contact feedback and progressively refines its size estimates, with errors dropping significantly after each contact event, as shown in Fig. 4(f). With improved geometry estimates, the policy is able to make more informed decisions, such as escaping stuck situations and planning appropriate paths

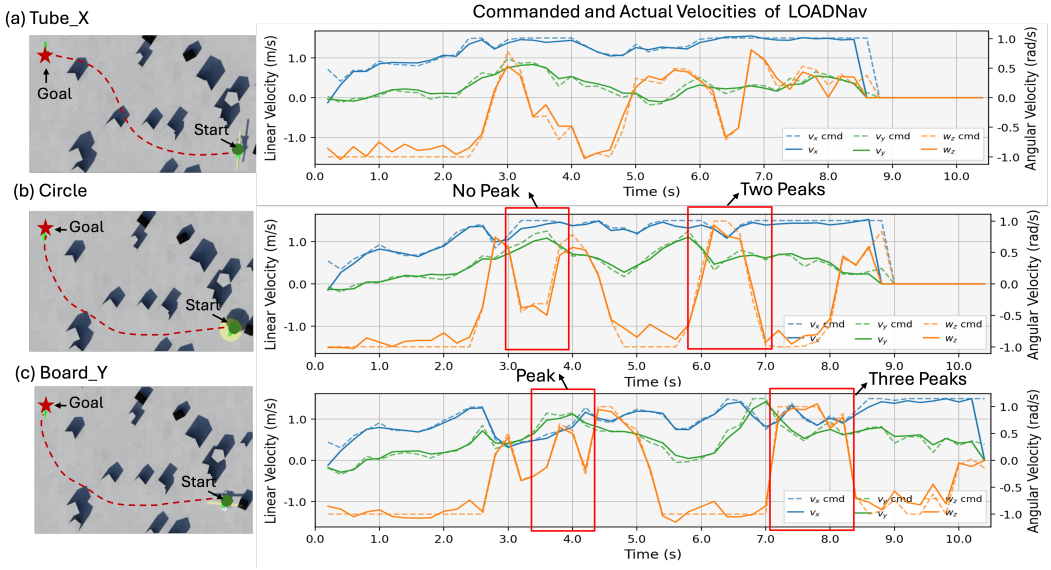

Figure 3: LOADNav trajectories and velocity profiles for different carried objects (N=80). (a) Tube_X (Length = 1.6 m). (b) Circle (Radius = 0.5 m). (c) Board_Y (Length = 1.6 m).

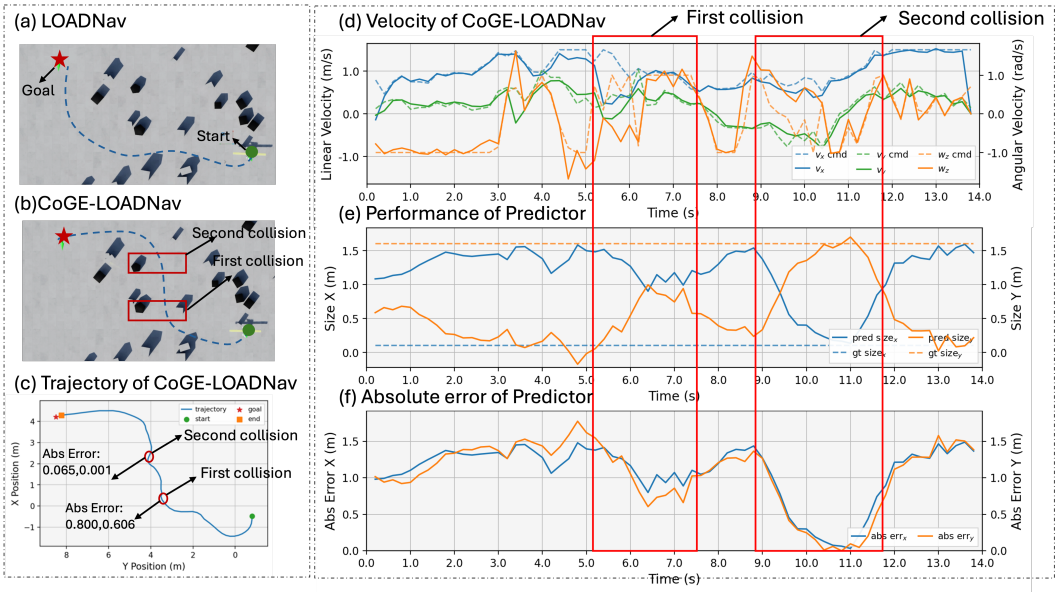

Figure 4: Visualization of LOADNav and CoGE-LOADNav in simulation, illustrating how CoGE-LOADNav infers geometry through collisions. (a-c) Navigation trajectories of LOADNav and CoGE-LOADNav. (d) Commanded and actual velocities of CoGE-LOADNav. (e-f) Predicted object size and absolute errors.

and load-aware actions suited to the identified payload shape. The predictor identifies contact events by detecting discrepancies between commanded and actual velocities, as shown in Fig. 4(d).

4.3 Memory-enhanced estimation

As shown in Fig. 4(e)(f), the estimation error decreases rapidly after contact but increases again after approximately $K = 5$ steps (1 s), corresponding to the predictor's input window length. This indicates that the contact information is forgotten as it moves out of the window. This behavior is not simply due to insufficient training, but reflects an inherent limitation of finite-window estimation: when the window contains no contact, the optimal window estimator $\hat{\mathbf{s}}_t^* = \mathbb{E}[\mathbf{s} \mid \mathbf{o}^p_{t-K+1:t}]$ collapses

to a constant independent of $\mathbf{s}$, since the window carries no evidence to condition on. Therefore, contact-driven geometry estimation does not satisfy K-step decodability for any finite K (Efroni et al., 2022), and the estimator tends to return toward its prior once the contact signal leaves the window. To retain contact information over longer horizons, the estimator must maintain a state across steps. We therefore train a recurrent memory model whose hidden state is carried across steps within each episode. This temporal training reduces the average size prediction MSE by 15.67% and the action imitation MSE by 25.09%. As shown in Fig. 5, after two contact events, the estimation error is corrected and remains stable rather than returning to its pre-contact level.

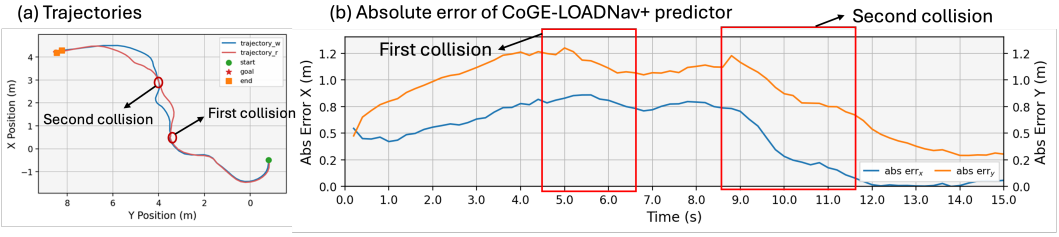

Figure 5: Memory-enhanced geometry estimation with CoGE-LOADNav+. (a) Trajectories of CoGE-LOADNav and CoGE-LOADNav+. (b) Absolute estimation error of the CoGE-LOADNav+.

4.4 Evaluation in the Benchmark

To evaluate navigation performance, we construct a simulated warehouse environment featuring shelves and 100 boxes, as shown in Fig. 10. Real-time occupancy maps are built using RTABMap (Labbé & Michaud, 2019) from RGB-D images. Each method is evaluated over 20 trials per object category. As shown in Table 1, LOADNav achieves the highest average SR of 91.67%, compared with 41.67% for VP-Nav and 30.00% for NavRL. Increasing the inflation radius in VP-Nav-I does not improve performance, with its SR dropping to 28.33% as few traversable paths remain. CoGE-LOADNav achieves an SR of 86.67%, while the optimized CoGE-LOADNav+ further improves the SR to 88.33%. In comparison, LOADNav-B, which does not use payload geometry, achieves only 63.33% SR, with the largest drop on Board_Y (85.00% to 45.00%), due to its limited geometry-aware navigation behavior. The higher ACT of CoGE-LOADNav+ results from a few trials where it recovered from collisions, whereas CoGE-LOADNav became stuck, with collisions in failed trials excluded from ACT.

Table 1: Navigation Performance in the Benchmark.

Method	Average		Board_Y		Circle		Tube_X	
	SR↑	ACT↓	SR↑	ACT↓	SR↑	ACT↓	SR↑	ACT↓
NavRL	30.00	0.81	30.00	0.33	50.00	0.60	10.00	1.50
VP-Nav	41.67	1.20	20.00	1.50	55.00	0.91	50.00	1.20
VP-Nav-I	28.33	0.46	20.00	0.50	45.00	0.89	20.00	**0.00**
CoGE-LOADNav	86.67	0.39	80.00	0.50	**90.00**	0.56	90.00	0.11
CoGE-LOADNav+	88.33	0.60	75.00	0.93	**90.00**	0.56	**100.00**	0.30
LOADNav-B	63.33	0.67	45.00	0.89	80.00	0.50	65.00	0.62
LOADNav	**91.67**	**0.32**	**85.00**	**0.24**	**90.00**	**0.11**	**100.00**	0.60

4.5 Real-World Deployment

To validate the proposed framework in real-world settings, we deploy both LOADNav and CoGE-LOADNav on the Unitree Go2 platform equipped with a default L1 LiDAR providing a local occupancy map. We evaluate the framework in three scenarios: carrying a 0.03×1.2 m board aligned along the Y-axis, a 1.2×0.09 m tube aligned along the X-axis and a 1.0×0.03 m tube aligned along the Y-axis in cluttered environments. Each scenario is evaluated over 20 trials. As shown in Table 2, LOADNav achieves the highest average SR of 98.33%, while CoGE-LOADNav achieves 81.67% without prior knowledge of the payload geometry. The optimized CoGE-LOADNav+ further improves the SR to 90.00%, reducing the gap to LOADNav. In comparison, VP-Nav achieves only 45.00% SR. As demonstrated in Fig. 6, when carrying the board

Table 2: Navigation Performance in Real-World.

Method	Average		Board_Y		Tube_X		Tube_Y	
	SR↑	ACT↓	SR↑	ACT↓	SR↑	ACT↓	SR↑	ACT↓
VP-Nav	45.00	1.23	30.00	2.00	50.00	0.60	55.00	1.09
CoGE-LOADNav	81.67	0.33	80.00	0.25	80.00	0.38	85.00	0.35
CoGE-LOADNav+	90.00	0.39	85.00	0.35	95.00	0.37	90.00	0.44
LOADNav	**98.33**	**0.04**	**95.00**	**0.11**	**100.00**	**0.00**	**100.00**	**0.00**

aligned along the Y-axis with known geometry, the robot exhibits clear load-aware behaviors, such as rotating sideways to pass through relatively open areas. When deploying CoGE-LOADNav without prior knowledge of the payload geometry, the robot may experience slight collisions during early navigation; however, the online geometry predictor rapidly reduces estimation error and corrects its geometry estimate, ultimately guiding the robot to successfully reach the goal.

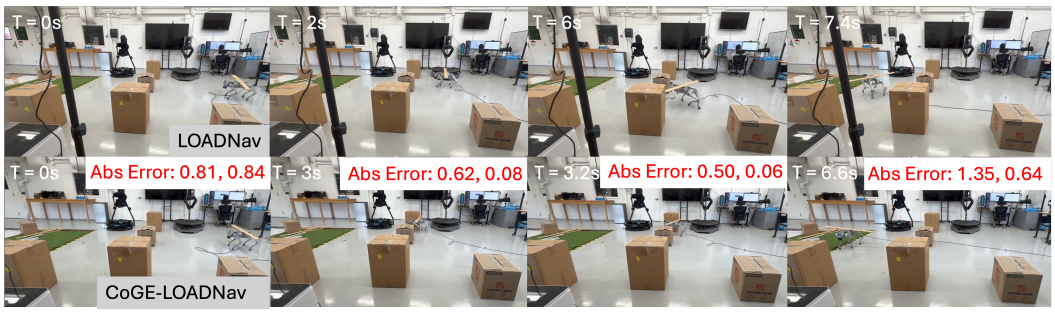

Figure 6: Real-world examples of LOADNav and CoGE-LOADNav.

4.6 Ablation Studies

To further evaluate the contribution of each component, we conduct ablation studies in simulation under three settings: w/o NavC., where the navigation curriculum is replaced by randomly sampled terrains with random reset positions; w/o $\mathcal{L}obj$, where Phase 2 only uses $\mathcal{L}a$; and w/o $\mathcal{L}a$, where Phase 2 only uses $\mathcal{L}obj$. As shown in Fig. 7, the navigation curriculum significantly improves performance in dense environments, and both loss terms are essential for effective online geometry estimation and navigation.

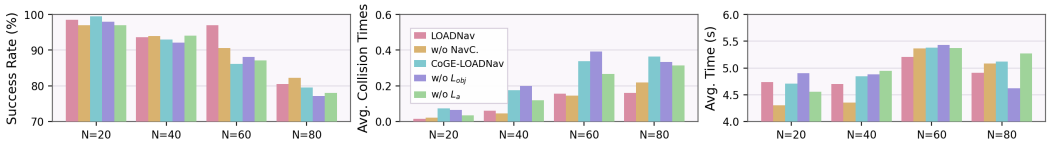

Figure 7: The results of ablation studies comparing LOADNav, CoGE-LOADNav, w/o NavC., w/o $\mathcal{L}_{obj}$, and w/o $\mathcal{L}_a$, in terms of success rate, average collision times, and average time.

5 Discussion and Limitations

In this paper, we propose a two-phase load-aware navigation learning framework that enables quadruped robots to achieve autonomous navigation in cluttered environments while carrying objects of varying shapes and sizes. Phase 1 trains a load-aware navigation policy that adapts its behavior according to the spatial geometry of the carried object, while Phase 2 enables online estimation of unknown payload geometry through occasional contact events feedback. A temporal training optimization further mitigates geometry estimation forgetting and reduces the performance gap caused by estimation errors. Extensive experiments in both simulation and real-world environments demonstrate clear load-aware navigation behaviors, such as sideways rotation detour planning, and the effectiveness of the online geometry estimation.

Limitations. Despite the promising results, there are still some limitations. First, real-world deployments reveal that uneven terrain negatively affects navigation performance. In particular, on surfaces such as grass and sand, the policy fails to accurately perceive the robot's state, resulting in persistent oscillatory behavior, as the policy is trained exclusively on flat terrain in simulation, causing out-of-distribution effects on velocity tracking. Future work will incorporate uneven terrain during training to enhance the robustness of the proposed framework (Zhang et al., 2024). Second, while LOADNav, CoGE-LOADNav and CoGE-LOADNav+ achieve comparable performance in simulation, CoGE-LOADNav and CoGE-LOADNav+'s success rate does drop in the real world. Based on our observations, this drop is primarily caused by the gap in collision-force resolution between the simulator and the real world, which degrades the performance of online geometry predictor. In our future work, we will strengthen sim-to-real transfer to improve performance in the real world.

AI USE STATEMENT

In this work, we used generative AI tools only for editing the manuscript to improve readability and identifying relevant literature. All AI-suggested edits and references were reviewed and verified by the authors. We take responsibility for the final content of this work.

REFERENCES

Ananye Agarwal, Ashish Kumar, Jitendra Malik, and Deepak Pathak. Legged locomotion in challenging terrains using egocentric vision. In *Conference on robot learning*, pp. 403–415. PMLR, 2023.

Leila Amanzadeh, Taizoon Chunawala, Randall T Fawcett, Alexander Leonessa, and Kaveh Akbari Hamed. Predictive control with indirect adaptive laws for payload transportation by quadrupedal robots. *IEEE Robotics and Automation Letters*, 9(11):10359–10366, 2024.

Leixin Chang, Yuxuan Nai, Hua Chen, and Liangjing Yang. Beyond robustness: Learning unknown dynamic load adaptation for quadruped locomotion on rough terrain, 2025. URL https://arxiv.org/abs/2507.07825.

Tao Chen, Megha Tippur, Siyang Wu, Vikash Kumar, Edward Adelson, and Pulkit Agrawal. Visual dexterity: In-hand reorientation of novel and complex object shapes. *Science Robotics*, 8(84): eadc9244, 2023.

Zeyuan Chen, Qiyang Yan, Yuanpei Chen, Tianhao Wu, Jiyao Zhang, Zihan Ding, Jinzhou Li, Yaodong Yang, and Hao Dong. Clutterdexgrasp: A sim-to-real system for general dexterous grasping in cluttered scenes. In *Conference on Robot Learning*, pp. 885–905. PMLR, 2025.

Xuxin Cheng, Kexin Shi, Ananye Agarwal, and Deepak Pathak. Extreme parkour with legged robots. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 11443–11450. IEEE, 2024.

Yonathan Efroni, Chi Jin, Akshay Krishnamurthy, and Sobhan Miryoosefi. Provable reinforcement learning with a short-term memory. In *International Conference on Machine Learning*, pp. 5832–5850. PMLR, 2022.

Zipeng Fu, Ashish Kumar, Ananye Agarwal, Haozhi Qi, Jitendra Malik, and Deepak Pathak. Coupling vision and proprioception for navigation of legged robots. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 17273–17283, 2022.

Yuni Fuchioka, Cristian C Beltran-Hernandez, Hai Nguyen, and Masashi Hamaya. Robotic object insertion with a soft wrist through sim-to-real privileged training. In *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 9159–9166. IEEE, 2024.

Sourav Garg, Dustin Craggs, Vineeth Bhat, Lachlan Mares, Stefan Podgorski, Madhava Krishna, Feras Dayoub, and Ian Reid. Objectreact: Learning object-relative control for visual navigation, 2025. URL https://arxiv.org/abs/2509.09594.

Ruihua Han, Shuai Wang, Shuaijun Wang, Zeqing Zhang, Jianjun Chen, Shijie Lin, Chengyang Li, Chengzhong Xu, Yonina C Eldar, Qi Hao, et al. Neupan: Direct point robot navigation with end-to-end model-based learning. *IEEE transactions on robotics*, 2025.

Zhehui Huang, Zhaojing Yang, Rahul Krupani, Baskın Şenbaşlar, Sumeet Batra, and Gaurav S Sukhatme. Collision avoidance and navigation for a quadrotor swarm using end-to-end deep reinforcement learning. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 300–306. IEEE, 2024.

Marco Hutter, Christian Gehring, Dominic Jud, Andreas Lauber, C. Dario Bellicoso, Vassilios Tsounis, Jemin Hwangbo, Karen Bodie, Peter Fankhauser, Michael Bloesch, Remo Diethelm, Samuel Bachmann, Amir Melzer, and Mark Hoepflinger. Anymal - a highly mobile and dynamic quadrupedal robot. In *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 38–44, 2016. doi: 10.1109/IROS.2016.7758092.

Ashish Kumar, Zipeng Fu, Deepak Pathak, and Jitendra Malik. Rma: Rapid motor adaptation for legged robots. *arXiv preprint arXiv:2107.04034*, 2021.

Mathieu Labbé and François Michaud. Rtab-map as an open-source lidar and visual simultaneous localization and mapping library for large-scale and long-term online operation. *Journal of field robotics*, 36(2):416–446, 2019.

Joonho Lee, Jemin Hwangbo, Lorenz Wellhausen, Vladlen Koltun, and Marco Hutter. Learning quadrupedal locomotion over challenging terrain. *Science robotics*, 5(47):eabc5986, 2020.

Wei Liu, Huihua Zhao, Chenran Li, Yuchen Deng, Joydeep Biswas, Soha Pouya, and Yan Chang. Compass: Cross-embodiment mobility policy via residual rl and skill synthesis. *arXiv preprint arXiv:2502.16372*, 2025.

Antonio Loquercio, Ashish Kumar, and Jitendra Malik. Learning visual locomotion with cross-modal supervision. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 7295–7302. IEEE, 2023.

Yunfei Luan, Muhang He, Yudong Tian, Chengjie Lin, Yunhan Fang, Zihao Zhao, Jianxin Yang, and Yao Guo. Intelligent disinfection robot with high-touch surface detection and dynamic pedestrian avoidance. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 3595–3601. IEEE, 2024.

Tyler Ga Wei Lum, Martin Matak, Viktor Makoviychuk, Ankur Handa, Arthur Allshire, Tucker Hermans, Nathan D Ratliff, and Karl Van Wyk. Dextrah-g: Pixels-to-action dexterous arm-hand grasping with geometric fabrics. In *Conference on Robot Learning*, pp. 3182–3211. PMLR, 2025.

Sergi Martinez, Robert Griffin, and Carlos Mastalli. Multi-contact inertial parameters estimation and localization in legged robots. *IEEE Robotics and Automation Letters*, 10(5):4730–4737, 2025. doi: 10.1109/LRA.2025.3550845.

Takahiro Miki, Joonho Lee, Jemin Hwangbo, Lorenz Wellhausen, Vladlen Koltun, and Marco Hutter. Learning robust perceptive locomotion for quadrupedal robots in the wild. *Science robotics*, 7(62):eabk2822, 2022.

Maria Vittoria Minniti, Ruben Grandia, Farbod Farshidian, and Marco Hutter. Adaptive clf-mpc with application to quadrupedal robots. *IEEE Robotics and Automation Letters*, 7(1):565–572, 2021.

I Made Aswin Nahrendra, Byeongho Yu, and Hyun Myung. Dreamwaq: Learning robust quadrupedal locomotion with implicit terrain imagination via deep reinforcement learning. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 5078–5084. IEEE, 2023.

NVIDIA. Isaac lab. `https://isaac-sim.github.io/IsaacLab/main/index.html`, 2026. Accessed: 2026-05-28.

Junli Ren, Yikai Liu, Yingru Dai, Junfeng Long, and Guijin Wang. Top-nav: Legged navigation integrating terrain, obstacle and proprioception estimation. In *Conference on Robot Learning*, pp. 3454–3473. PMLR, 2025.

Carmen Scheidemann, Lennart Werner, Victor Reijgwart, Andrei Cramariuc, Joris Chomarat, Jia-Ruei Chiu, Roland Siegwart, and Marco Hutter. Obstacle-avoidant leader following with a quadruped robot. In *2025 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1407–1413. IEEE, 2025.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms, 2017. URL `https://arxiv.org/abs/1707.06347`.

Kartikeya Singh, Yash Turkar, Christo Aluckal, Charuvarahan Adhivarahan, and Karthik Dantu. Panos: Payload-aware navigation in offroad scenarios, 2024. URL `https://arxiv.org/abs/2409.16566`.

Nikhil Sobanbabu, Guanqi He, Tairan He, Yuxiang Yang, and Guanya Shi. Sampling-based system identification with active exploration for legged robot sim2real learning, 2025. URL https://arxiv.org/abs/2505.14266.

Mohsen Sombolestan and Quan Nguyen. Adaptive-force-based control of dynamic legged locomotion over uneven terrain. *IEEE Transactions on Robotics*, 40:2462–2477, 2024.

Marcel Torne, Anthony Simeonov, Zechu Li, April Chan, Tao Chen, Abhishek Gupta, and Pulkit Agrawal. Reconciling reality through simulation: A real-to-sim-to-real approach for robust manipulation. *arXiv preprint arXiv:2403.03949*, 2024.

Unitree Robotics. Go1 quadruped robot. https://www.unitree.com/products/go1, 2026a. Accessed: 2026-05-28.

Unitree Robotics. Go2 quadruped robot. https://www.unitree.com/go2, 2026b. Accessed: 2026-05-28.

Haitong Wang, Aaron Hao Tan, Angus Fung, and Goldie Nejat. X-nav: Learning end-to-end cross-embodiment navigation for mobile robots. *IEEE Robotics and Automation Letters*, 11(1):698–705, 2025a.

Zifan Wang, Teli Ma, Yufei Jia, Xun Yang, Jiaming Zhou, Wenlong Ouyang, Qiang Zhang, and Junwei Liang. Omni-perception: Omnidirectional collision avoidance for legged locomotion in dynamic environments. *arXiv preprint arXiv:2505.19214*, 2025b.

Lorenz Wellhausen and Marco Hutter. Rough terrain navigation for legged robots using reachability planning and template learning. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 6914–6921. IEEE, 2021.

Lorenz Wellhausen and Marco Hutter. Artplanner: Robust legged robot navigation in the field. *Field Robotics*, 3:413–434, 2023.

Botian Xu, Haoyang Weng, Qingzhou Lu, Yang Gao, and Huazhe Xu. Facet: Force-adaptive control via impedance reference tracking for legged robots, 2025a. URL https://arxiv.org/abs/2505.06883.

Zhefan Xu, Xinming Han, Haoyu Shen, Hanyu Jin, and Kenji Shimada. Navrl: Learning safe flight in dynamic environments. *IEEE Robotics and Automation Letters*, 2025b.

Zihao Xu, Ce Hao, Chunzheng Wang, Kuankuan Sima, Fan Shi, and Jin Song Dong. Rebot: Reflexive evasion robot for instantaneous dynamic obstacle avoidance. *IEEE Robotics and Automation Letters*, 11(2):1746–1753, 2025c.

Jianpeng Yao, Xiaopan Zhang, Yu Xia, Zejin Wang, Amit Roy-Chowdhury, and Jiachen Li. Towards generalizable safety in crowd navigation via conformal uncertainty handling. In *Conference on Robot Learning*, pp. 4206–4225. PMLR, 2025.

Qi Yin and Yangming Shi. Mitigating multi-module errors for reliable navigation in dynamic environments via online trajectory refinement. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, pp. 1077–1086, June 2026.

Uksang Yoo, Jonathan Francis, Jean Oh, and Jeffrey Ichnowski. Kinesoft: Learning proprioceptive manipulation policies with soft robot hands, 2025. URL https://arxiv.org/abs/2503.01078.

Xuanqi Zeng, Hongbo Zhang, Linzhu Yue, Zhitao Song, Lingwei Zhang, and Yun-Hui Liu. Adaptive model predictive control with data-driven error model for quadrupedal locomotion. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 5731–5737. IEEE, 2024.

Chong Zhang, Jin Jin, Jonas Frey, Nikita Rudin, Matías Mattamala, Cesar Cadena, and Marco Hutter. Resilient legged local navigation: Learning to traverse with compromised perception end-to-end. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 34–41, 2024. doi: 10.1109/ICRA57147.2024.10611254.

Zewei Zhang, Chenhao Li, Takahiro Miki, and Marco Hutter. Motion priors reimagined: Adapting flat-terrain skills for complex quadruped mobility, 2025. URL https://arxiv.org/abs/2505.16084.

Peiyuan Zhi, Peiyang Li, Jianqin Yin, Baoxiong Jia, and Siyuan Huang. Learning a unified policy for position and force control in legged loco-manipulation, 2025. URL https://arxiv.org/abs/2505.20829.

A Appendix

A.1 Reward Functions

The terminal rewards encode four episode termination conditions. A positive reward r_{succ} is granted upon successful goal reaching, defined as:

$$r_{\text{succ}} = \mathbb{I}\left(\|\mathbf{p}_b^{g,xy}\|_2 \leq \delta\right) \tag{7}$$

where $\mathbf{p}_b^g$ is the relative goal position in the robot body frame and δ is the goal tolerance. The remaining three terminal rewards impose penalties for undesired terminations: r_{timeout} penalizes the robot for failing to reach the goal within the maximum episode steps; r_{robot} penalizes physical contact between the robot body and the environment; r_{stuck} penalizes the robot when it becomes immobile, defined as:

$$r_{\text{stuck}} = \mathbb{I}\left(\|\mathbf{v}_b^{xy}\|_2 < v_{\text{th}} \wedge |\omega_b^z| < \omega_{\text{th}}\right) \tag{8}$$

where v_{th} and ω_{th} are the velocity thresholds below which the robot is considered stuck. The dense rewards provide continuous feedback throughout the navigation episode. r_{vel} encourages the robot to move toward the goal by rewarding the projection of the robot's planar velocity onto the goal direction:

$$r_{\text{vel}} = \mathbf{v}_b^{xy} \cdot \frac{\mathbf{p}_b^{g,xy}}{\|\mathbf{p}_b^{g,xy}\|_2 + \varepsilon} \tag{9}$$

where $\mathbf{v}_b^{xy}$ is the robot's planar velocity. r_{head} encourages the robot to move along its heading direction by rewarding the cosine similarity between the forward axis and the current velocity direction:

$$r_{\text{head}} = \begin{cases} \dfrac{v_b^x}{\|\mathbf{v}_b^{xy}\|_2 + \varepsilon}, & \|\mathbf{v}_b^{xy}\|_2 > 10^{-3} \\ 0, & \text{otherwise} \end{cases} \tag{10}$$

where the zero branch avoids meaningless heading estimates when the robot is nearly stationary. r_{obj} penalizes collisions between the loaded object and the environment. r_{step} applies a constant penalty at every timestep, encouraging the robot to complete the task in fewer steps. The reward weights are provided in Table 3.

Table 3: Reward Weights.

Category	Reward Term	Weight
Termination	Success reward r_{succ}	400.0
	Timeout penalty r_{timeout}	-40.0
	Robot body contact penalty r_{robot}	-10.0
	Stuck penalty r_{stuck}	-50.0
Dense	Velocity projection reward r_{vel}	6.0
	Heading alignment reward r_{head}	3.0
	Object collision penalty r_{obj}	-30.0
	Per-step time penalty r_{step}	-0.5

The PPO hyperparameters used in Phase 1 training are summarized in Table 4.

A.2 Navigation Curriculum

The terrain and curriculum parameters adopted in the two-phase training are summarized in Table 5. The training environments are illustrated in Figure 8.

A.3 Loaded Objects

During training, each robot is assigned a randomly sampled carried object rigidly attached to the base joint. Six object categories are used, each with randomized dimensions sampled from the ranges specified in Table 6. All six categories are sampled with equal probability.

Table 4: PPO Hyperparameters.

Parameter	Value
PPO clip parameter	0.2
GAE-λ	0.95
Reward discount factor	0.99
Learning rate	3×10^{-4}
Learning rate schedule	adaptive
Value loss coefficient	1.0
Use clipped value loss	True
Entropy coefficient	0.003
Desired KL divergence	0.01
Max gradient norm	1.0
Number of environments	512
Number of env steps per batch	24
Learning epochs per batch	4
Number of mini batches per epoch	8

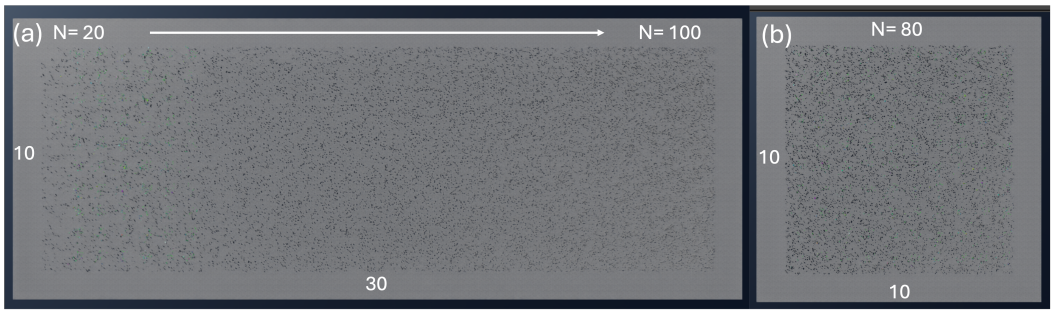

Figure 8: Training environments. (a) Phase 1 with navigation curriculum (N = 20 to 100). (b) Phase 2 without navigation curriculum (N = 80).

A.4 Navigation Performance

LOADNav Navigation Performance Across Payload Type. Table 7 reports the complete results summarized in Section 4.2.1. LOADNav and CoGE-LOADNav are evaluated on each of the six payload types and their mixture under obstacle densities $N = 20$, 40, 60, and 80, on a 5×5 terrain grid with 50 robots over 200 episodes per setting. The ACT of CoGE-LOADNav is higher because its estimate is refined through contact. Board_Y and Tube_Y, placed perpendicular to the heading, are the most challenging types; Board_X and Tube_X, aligned with the elongated body, are the easiest.

CoGE-LOADNav Navigation Performance. To further evaluate CoGE-LOADNav, we compare LOADNav and CoGE-LOADNav for Tube_X and Board_Y across varying obstacle densities in simulation, as shown in Fig. 9. For Tube_X, CoGE-LOADNav produces trajectories nearly identical to those of LOADNav: since the payload is aligned with the robot's heading and adds little lateral extent, the robot navigates without collision and the predicted geometry remains stable throughout the episode. Comparing Board_Y LOADNav with Tube_X LOADNav, the load-aware navigation policy learned in Phase 1 consistently drives the robot to take wider detours when carrying an object with a larger Y dimension, demonstrating stable and geometry-aware path selection. Finally, for Board_Y, CoGE-LOADNav starts without knowledge of the payload's lateral extent; once a collision occurs, the geometry estimate is corrected, which helps the robot escape and informs its subsequent navigation decisions.

Evaluation in the Simulation Benchmark. Fig. 10 shows the benchmark environment used in Section 4.4. The warehouse contains shelves and 100 randomly placed box obstacles and communicates with the Unitree Go2 via ROS2. Occupancy maps are built online with RTAB-Map from

Table 5: Navigation Curriculum Parameters.

Category	Parameter	Value
Terrain	Size	16×16
Box obstacles	Length range	$0.25 \sim 0.55$
	Width range	$0.25 \sim 0.55$
	Height range	$0.4 \sim 1.0$
Cylinder obstacles	Radius range	$0.15 \sim 0.35$
	Height range	$0.4 \sim 1.0$
Phase 1	Number rows	30
	Number columns	10
	Number objects start	20
	Number objects end	100
Phase 2	Number rows	10
	Number columns	10
	Number objects	80
Curriculum window	Size	10
	Success rate	0.8
	Fail rate	0.5
	Cooldown	20

Table 6: Object Parameters.

Object	Parameter	Value
Board_Y	Length range	$0.40 \sim 1.60$
	Width range	$0.04 \sim 0.20$
	Thickness	0.30
Tube_Y	Length range	$0.40 \sim 1.60$
	Radius range	$0.01 \sim 0.08$
Circle	Radius range	$0.20 \sim 0.50$
	Thickness	$0.01 \sim 0.04$
Triangle	Side range	$0.40 \sim 1.00$
	Thickness	$0.01 \sim 0.04$
Board_X	Length range	$0.80 \sim 1.60$
	Width range	$0.04 \sim 0.20$
	Thickness	0.30
Tube_X	Length range	$0.80 \sim 1.60$
	Radius range	$0.01 \sim 0.08$

the onboard RGB-D camera, and each method is evaluated over 20 trials per payload with the three payloads shown in Fig. 10(c).

Real-World Navigation Performance. Due to page limitations, we present more real-world examples here, as shown in Fig. 11. For Tube_X, both LOADNav and CoGE-LOADNav follow similar trajectories without any collision, and the absolute prediction error of CoGE-LOADNav remains stable throughout. For Board_Y, LOADNav initially attempts to pass through a narrow space but pauses upon approach and reroutes to a wider passage. CoGE-LOADNav of Board_Y encounters a collision in the narrow space, after which the robot actively rotates its body and successfully passes through to reach the goal. The absolute prediction error decreases significantly upon collision, reflecting improved geometry estimation through slight collision.

Table 7: Navigation Performance Across Payload Types and Obstacle Densities. ACT is averaged over all trials.

Numbers of Obstacles	Method	Mix		Board_Y		Tube_Y		Circle		Triangle		Board_X		Tube_X	
		SR↑	ACT↓	SR↑	ACT↓	SR↑	ACT↓	SR↑	ACT↓	SR↑	ACT↓	SR↑	ACT↓	SR↑	ACT↓
N=20	LOADNav	98.50	0.015	97.50	0.035	97.01	0.035	99.50	0.005	97.00	0.015	98.00	0.030	99.01	0.015
	CoGE-LOADNav	99.50	0.074	96.02	0.179	98.00	0.140	98.00	0.045	98.50	0.020	100.00	0.020	99.00	0.020
N=40	LOADNav	93.60	0.059	92.54	0.090	93.00	0.070	95.50	0.035	95.50	0.045	95.00	0.060	94.50	0.035
	CoGE-LOADNav	93.00	0.175	91.54	0.234	93.50	0.235	92.50	0.070	93.00	0.085	89.00	0.070	89.55	0.050
N=60	LOADNav	87.00	0.155	88.62	0.158	88.50	0.150	88.50	0.060	90.50	0.080	91.63	0.118	90.10	0.069
	CoGE-LOADNav	86.14	0.337	87.00	0.450	86.50	0.330	87.50	0.105	89.00	0.120	89.00	0.085	90.55	0.065
N=80	LOADNav	80.50	0.160	81.09	0.234	84.50	0.230	81.50	0.170	85.57	0.114	85.57	0.139	85.50	0.130
	CoGE-LOADNav	79.50	0.365	76.00	0.500	78.50	0.480	80.50	0.180	81.00	0.150	85.00	0.120	85.50	0.090
Average	LOADNav	**89.90**	**0.097**	**89.94**	**0.129**	**90.75**	**0.121**	**91.25**	**0.080**	**92.14**	**0.064**	**92.55**	0.089	**92.28**	0.062
	CoGE-LOADNav	89.54	0.238	87.64	0.341	89.13	0.296	89.63	0.100	90.38	0.094	90.75	**0.074**	91.15	**0.056**

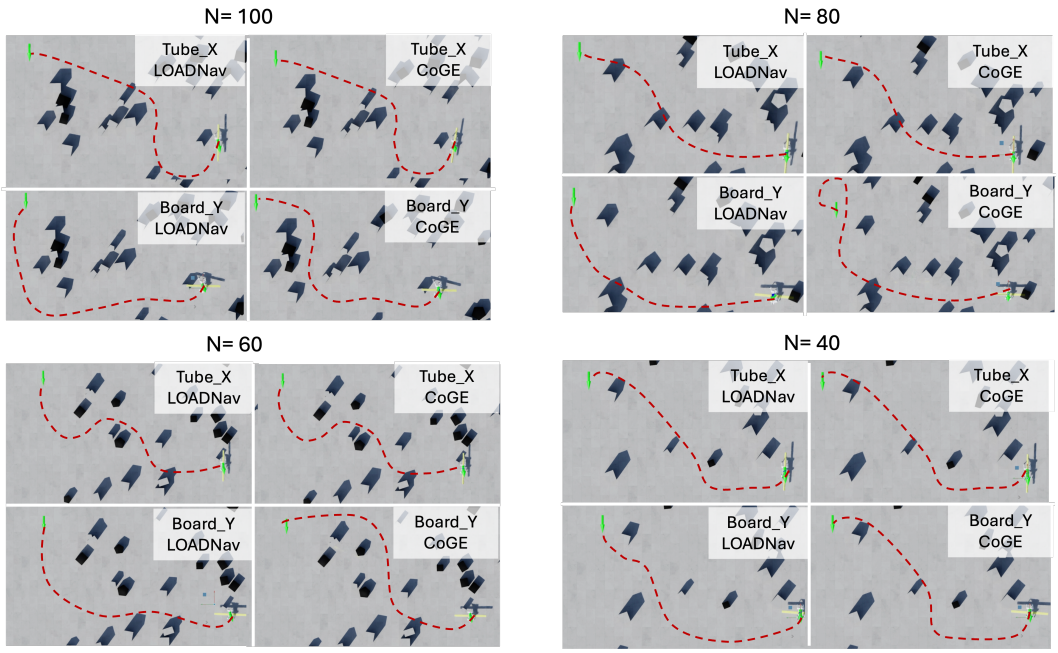

Figure 9: Simulation navigation trajectories of LOADNav and CoGE-LOADNav for Tube_X and Board_Y under varying obstacle densities (N = 40, 60, 80, 100).

A.5 Ablation

Fig. 12 compares the Phase 1 training of LOADNav with that of LOADNav-B (the size-blind policy) used in Section 4.4. The mean reward decreases over training because the navigation curriculum raises the obstacle density as the policy improves. Throughout training, LOADNav attains both a higher reward and a higher curriculum level.

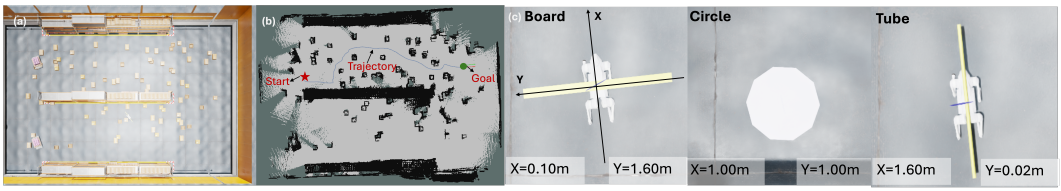

Figure 10: Warehouse benchmark environment. (a) Simulated warehouse with shelves and 100 randomly placed box obstacles, in which the Unitree Go2 is controlled through ROS2. (b) Occupancy map built online by RTAB-Map and a representative trajectory of LOADNav carrying a Circle payload. (c) The three payloads used in Table 1: Board_Y (X = 0.10 m, Y = 1.60 m), Circle (X = 1.00 m, Y = 1.00 m), and Tube_X (X = 1.60 m, Y = 0.02 m).



Figure 11: Real-world navigation examples of LOADNav and CoGE-LOADNav for Tube_X and Board_Y.

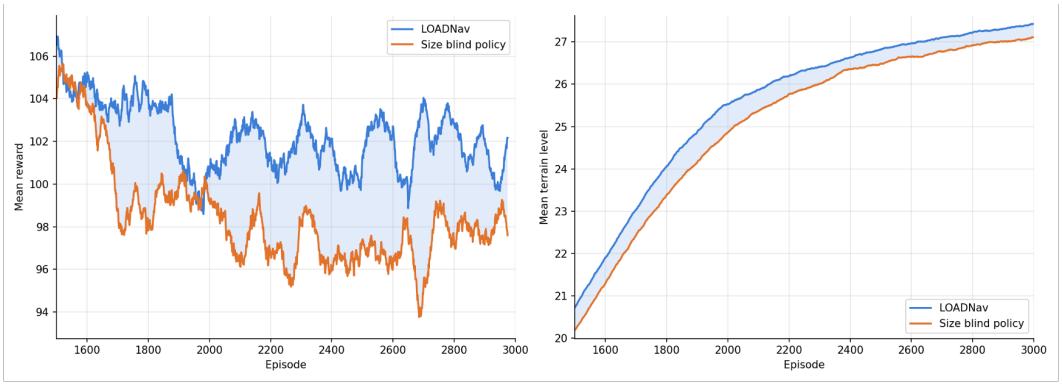

Figure 12: Phase 1 training curves of LOADNav and the LOADNav-B (size-blind policy) (episodes 1500–3000). Left: mean episode reward. Right: mean curriculum level, where a higher level corresponds to a denser obstacle field. Both policies share the curriculum, payload distribution, and PPO settings and differ only in the geometry input.