UEREBot: Learning Safe Quadrupedal Navigation under Unstructured Environments and High-Speed Dynamic Obstacles

Zihao Xu[1†] Runyu Lei[1] Zihao Li[2] Boxi Lin[3] Ce Hao[1,4] Jin Song Dong[1]
[1]National University of Singapore [2]ShanghaiTech University [3]Xi'an Jiaotong University
[4]Beijing Zhongguancun Academy
[†]corresponding to zihao.xu@u.nus.edu

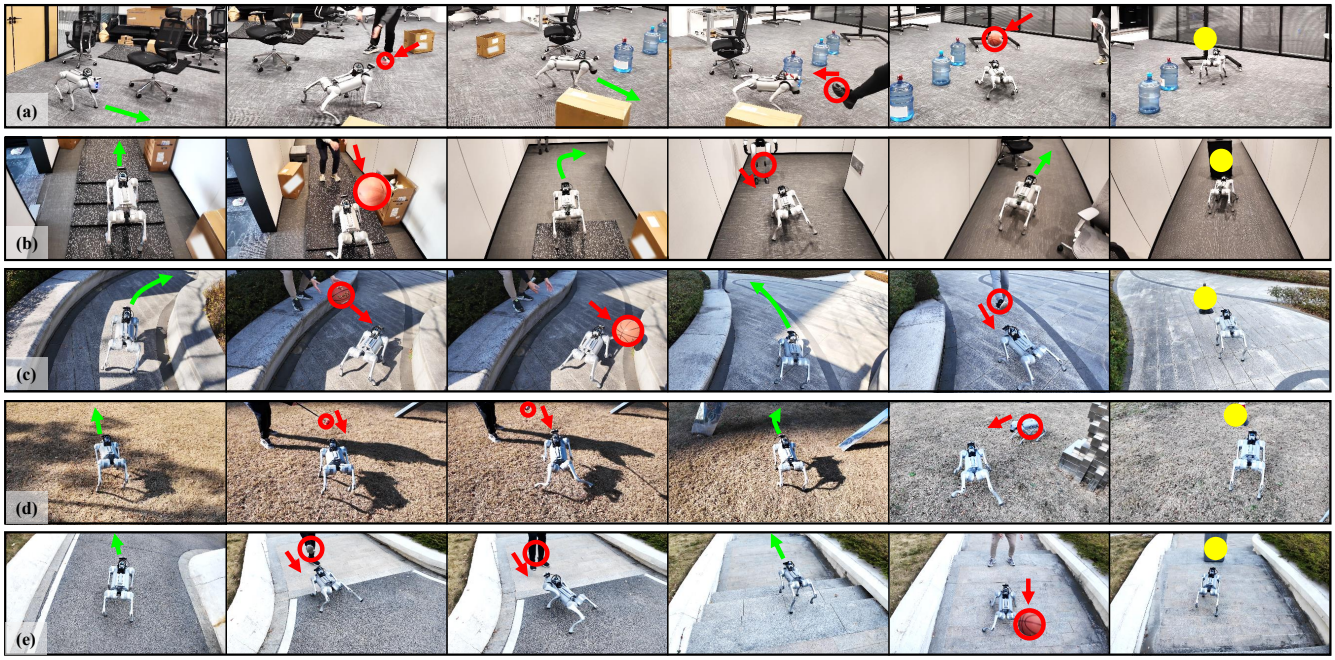

Fig. 1: **Overview of UEREBot in complex dynamic environments.** The robot tracks a reference path toward the goal region while responding to dynamic attacks under limited reaction time. **Green arrows** indicate the reference direction, **red arrows** indicate the attack direction of the dynamic obstacle, and **yellow circle** marks the goal location. (a) Indoor room. (b) Indoor corridor with steps. (c) Outdoor narrow path. (d) Outdoor grass. (e) Outdoor steps. The attacks include poke, hit, and kick, as well as a quadruped and a humanoid robot.

***Abstract*—Safe quadrupedal navigation in unstructured environments requires long-horizon goal progress, passability over uneven terrain and static constraints, and collision avoidance against high-speed dynamic obstacles under limited reaction time. Planning-based decisions can be too slow, while purely reactive decisions can sacrifice goal progress and passability. To resolve this conflict, we propose UEREBot (Unstructured-Environment Reflexive Evasion Robot), a hierarchical framework that separates slow planning from instantaneous reflexive evasion and coordinates them during execution. UEREBot adopts a spatial–temporal planner that provides reference guidance toward the goal, obstacle predictions, and threat signals. It then uses a threat-aware handoff to fuse navigation and reflex commands, and a control barrier function shield as an execution-time command filter. We evaluate UEREBot in Isaac Lab simulation and deploy it on a Unitree Go2 quadruped equipped with onboard perception. Across diverse environments with complex static structure and high-speed dynamic obstacles, UEREBot achieves higher avoidance success while maintaining goal progress than representative baselines, demonstrating improved safety–progress trade-offs. Project homepage: https://uerebot-2026.github.io/.**

I. Introduction

Quadruped robots are increasingly deployed in real-world settings such as industrial plants, warehouses, and outdoor inspection tasks [22, 28, 17, 46]. Their legged morphology enables mobility beyond the capabilities of wheeled platforms [36, 12, 27]. In unstructured environments, safe navigation involves three tasks: long-horizon goal progress, passability through uneven terrain and static constraints, and collision avoidance against high-speed dynamic obstacles under limited reaction time (Fig. 1). Prior work typically addresses these requirements with map-based planning [28], terrain- and constraint-aware navigation [48, 5], or fast reactive avoidance [40].

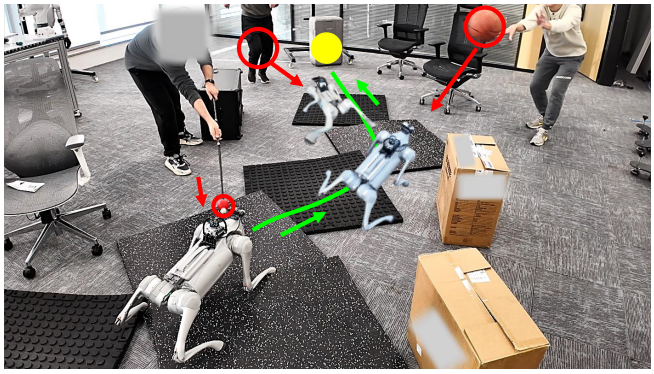

Fig. 2: **Task overview.** The robot tracks a reference path toward the goal through rough terrain and static obstacles, while responding to dynamic attacks such as poke, hit, and kick.

However, goal progress, passability, and high-speed collision avoidance are difficult to satisfy within a single decision mechanism under limited reaction time. Goal progress favors planning with long-horizon intent. Passability requires feasibility reasoning over non-convex terrain and static constraints. Collision avoidance against high-speed dynamic obstacles demands instantaneous reactions. These tasks require different decision speeds within the same real-time control loop: planning-based decisions can be too slow [21, 15], while reactive decisions can sacrifice goal progress and passability [40]. This calls for a spatial–temporal mechanism beyond planning or reactive avoidance.

To resolve this conflict, we propose **UEREBot** (Unstructured-Environment Reflexive Evasion Robot), a hierarchical framework for safe navigation in unstructured environments and high-speed dynamic obstacles. The key idea is to separate slow planning from instantaneous reflexive evasion and coordinate them during execution. UEREBot adopts a spatial–temporal planner that outputs a reference path, obstacle predictions, and a threat score. A navigation policy tracks the reference path, a reflexive evasion policy based on REBot [40] performs high-speed evasion, a threat-aware handoff fuses their commands, and a control barrier function (CBF) shield serves as the final command filter.

We evaluate UEREBot in Isaac Lab simulation and on a Unitree Go2 equipped with onboard perception [41], covering complex terrain, static obstacles, and high-speed dynamic obstacles. Compared with representative baselines, UEREBot achieves higher avoidance success and larger clearance margins while maintaining goal progress, demonstrating improved safety–progress trade-offs. Ablation studies further verify the contribution of the key components.

In summary, this paper makes the following contributions:

- We identify and formulate the safe quadrupedal navigation problem under unstructured environments and high-speed dynamic obstacles, where goal progress, environment passability, and dynamic-obstacle safety must be considered jointly.
- We propose UEREBot, a hierarchical framework that integrates spatial–temporal planning, learned navigation and reflexive evasion policies, threat-aware command handoff, and CBF-based execution-time filtering.
- We validate UEREBot in simulation and on real robots, demonstrating improved safety–progress trade-offs and task completion over representative baselines, with ablation studies supporting the key components.

II. Related Works

Navigation and passability in unstructured environments. Navigation and planning for goal progress are mature [3, 8, 37]. Model-based methods maintain environment representations and enable fast replanning in partially observed spaces [7, 49, 42]. Learning-augmented approaches improve unknown-environment navigation with compact context representations [35, 39, 9, 4] and hierarchical policies for path or subgoal selection [33, 13, 19, 32, 20, 34]. In unstructured environments, model-based approaches combine terrain maps, traversability models, foothold or contact planning, whole-body optimization, and MPC under collision and friction constraints [44, 38, 24, 16, 6]. Learning-based approaches train perceptive locomotion with reinforcement learning or imitation learning [28, 48, 23, 31, 43], often using 3D or context representations and hierarchical policies for generalization and Sim2Real transfer [29, 2, 11, 18]. These methods address goal progress and static passability, while high-speed dynamic obstacle avoidance remains less explored.

Dynamic obstacle avoidance in real time. Learning-based obstacle avoidance has been studied for UAVs, where single-rigid-body dynamics allow agile policies trained with end-to-end learning for rapid evasion [10, 26, 47, 25]. For dynamic environments, REASAN [45, 1] combines online tracking with reactive commands and safety filtering. For legged robots, ABS [14] combines locomotion with safety-aware correction to avoid collisions with slower moving obstacles, while REBot [40] learns instantaneous reflexive evasion actions and recovery behaviors to handle high-speed obstacles under limited reaction time. These methods address important parts of the problem, but do not jointly consider long-horizon goal progress, passability in unstructured environments, and high-speed dynamic obstacle avoidance under limited reaction time.

III. Preliminary

Problem setup. We consider a quadruped robot moving toward a goal region G in an unstructured environment with uneven terrain, static obstacles, and high-speed dynamic obstacles (Fig. 2). The robot is controlled through high-level body-frame velocity commands tracked by a low-level locomotion controller. We define the reduced state as $\mathbf{q}_t = (x_t, y_t, \theta_t) \in SE(2)$, where (x_t, y_t) is the planar position and θ_t is the yaw angle in the world frame $\mathcal{W}$. The robot position is $\mathbf{p}_t^R = (x_t, y_t)$, and the command is $\mathbf{u}_t = [v_{x,t}\ v_{y,t}\ \omega_t]^\top \in \mathcal{U}$, where $\mathcal{U}$ is the admissible command set. Learned policies use proprioceptive observations $\mathbf{o}_t$, such as base velocity, roll and pitch, and exteroceptive observations, including terrain features, reference path, and dynamic obstacles.

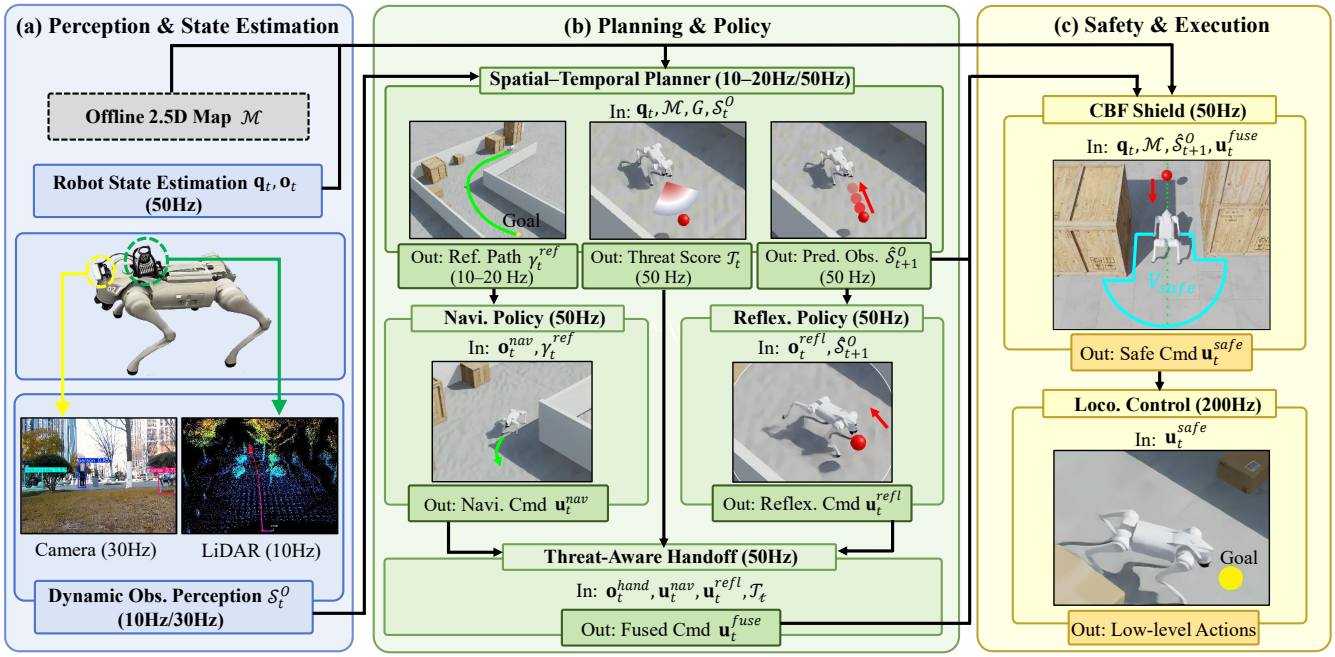

Fig. 3: **Overview of UEREBot framework.** Perception, state estimation, and an offline 2.5D map provide robot state, obstacle observations, and passability information. The spatial–temporal planner outputs a reference path, obstacle predictions, and a threat score. Navigation and reflex policies generate commands fused by a threat-aware handoff; the fused command is filtered by a CBF shield and executed by the low-level locomotion controller.

Environment and obstacles. The static environment is represented by an offline 2.5D map $\mathcal{M}$. For a planar position $\mathbf{p} \in \mathbb{R}^2$, the map defines a terrain passability field $\Phi_{\text{terr}}(\mathbf{p})$ and a static-obstacle passability field $\Phi_{\text{static}}(\mathbf{p})$. We use $\Phi_{\text{env}}(\mathbf{p}) = \min\{\Phi_{\text{terr}}(\mathbf{p}), \Phi_{\text{static}}(\mathbf{p})\}$ as the environment passability field, where $\Phi_{\text{env}}(\mathbf{p}) > 0$ indicates passability. Let $\mathcal{O}$ be the set of observed dynamic obstacles, with states $\mathcal{S}_t^O = \{(\mathbf{p}_{i,t}^O, \mathbf{v}_{i,t}^O, r_{\text{eff}}^{(i)})\}_{i\in\mathcal{O}}$, where $\mathbf{p}_{i,t}^O$, $\mathbf{v}_{i,t}^O$, and $r_{\text{eff}}^{(i)}$ are the position, velocity, and effective safety radius of obstacle i. The dynamic clearance field is $\Phi_{\text{dyn}}^{(i)}(\mathbf{p}, t) = \|\mathbf{p} - \mathbf{p}_{i,t}^O\| - r_{\text{eff}}^{(i)}$.

Constrained task formulation. Inspired by a constrained optimal control perspective, we describe the task by three requirements: $\mathbf{p}_t^R \to G$, $\Phi_{\text{env}}(\mathbf{p}_t^R) > 0$, and $\Phi_{\text{dyn}}^{(i)}(\mathbf{p}_t^R, t) > 0$ for all $i \in \mathcal{O}$. This formulation clarifies the constraints that motivate UEREBot, but we do not solve a global optimal control problem online. UEREBot instead uses a hierarchical framework with a spatial–temporal planner, learned command policies, threat-aware handoff, and safety filtering.

IV. Method

We present UEREBot, a hierarchical framework for safe navigation in unstructured environments with high-speed dynamic obstacles. The framework consists of a spatial–temporal planner (Sec. IV-B), a reference-tracking navigation policy (Sec. IV-C), a reflexive evasion policy (Sec. IV-D), a threat-aware handoff policy (Sec. IV-E), and a CBF safety shield (Sec. IV-F).

A. Framework overview

UEREBot is a hierarchical framework for safe navigation toward the goal region G (Fig. 3). At each control cycle, the spatial–temporal planner takes the reduced state $\mathbf{q}_t$, the map $\mathcal{M}$, the goal region G, and the observed dynamic-obstacle states $\mathcal{S}_t^O$, and outputs a reference path γ_t^{ref}, predicted obstacle states $\hat{\mathcal{S}}_{t+1}^O$, and a threat score $\mathcal{T}_t$. The planner provides downstream guidance and threat context, but it does not output executable commands.

Two learned policies then run in parallel and output commands in the same high-level command space $\mathcal{U}$. Let $\mathbf{o}_t^{\text{nav}}$, $\mathbf{o}_t^{\text{refl}}$, and $\mathbf{o}_t^{\text{hand}}$ denote the observations for navigation, reflexive evasion, and handoff, respectively. The navigation policy uses $\mathbf{o}_t^{\text{nav}}$ and γ_t^{ref} to output the navigation command $\mathbf{u}_t^{\text{nav}}$, while the reflexive evasion policy uses $\mathbf{o}_t^{\text{refl}}$ and $\hat{\mathcal{S}}_{t+1}^O$ to output the reflexive evasion command $\mathbf{u}_t^{\text{refl}}$. The threat-aware handoff uses $\mathbf{o}_t^{\text{hand}}$, $\mathbf{u}_t^{\text{nav}}$, $\mathbf{u}_t^{\text{refl}}$, and $\mathcal{T}_t$ to output the fused command $\mathbf{u}_t^{\text{fuse}}$, which is then filtered by the CBF safety shield using $\mathbf{q}_t$, $\mathcal{M}$, and $\hat{\mathcal{S}}_{t+1}^O$ to produce the safety-filtered command $\mathbf{u}_t^{\text{safe}}$ for the locomotion controller.

B. Spatial–temporal planner

The spatial–temporal planner provides reference guidance, obstacle prediction, and threat context for downstream modules. **Input:** reduced state $\mathbf{q}_t$, goal region G, map $\mathcal{M}$, and observed dynamic-obstacle states $\mathcal{S}_t^O$. **Output:** reference path γ_t^{ref}, predicted obstacle states $\hat{\mathcal{S}}_{t+1}^O$, and scalar threat score $\mathcal{T}_t$. The planner does not output executable commands: γ_t^{ref} guides the navigation policy, $\hat{\mathcal{S}}_{t+1}^O$ is used by the reflex policy and the CBF shield, and $\mathcal{T}_t$ is used by the threat-aware handoff.

The planner uses a shared encoder with three output heads for γ_t^{ref}, $\hat{\mathcal{S}}_{t+1}^O$, and $\mathcal{T}_t$. We use a multi-rate design: γ_t^{ref} is updated at 10–20 Hz, while $\hat{\mathcal{S}}_{t+1}^O$ and $\mathcal{T}_t$ are updated at 50 Hz for fast evasion, handoff, and safety filtering. The

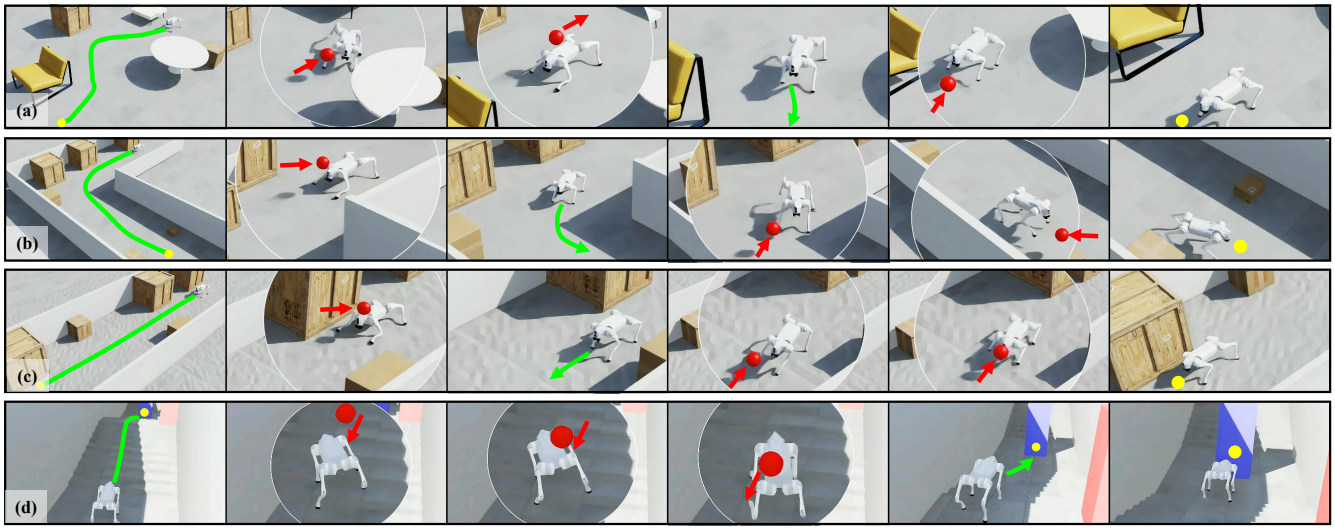

Fig. 4: **Representative simulation experiments.** The robot performs locomotion toward the goal region while traversing rough terrain, passing static obstacles in open and confined spaces, and avoiding high-speed dynamic obstacles under limited reaction time. (a) Open flat ground with static and dynamic obstacles. (b) Narrow corridor. (c) Rough terrain. (d) Stair traversal.

prediction head estimates one-step obstacle motion, and the threat head summarizes dynamic interaction urgency from predicted clearance and relative motion.

To obtain γ_t^{ref}, the planner samples candidate paths toward G on $\mathcal{M}$ and scores them using path progress, environment passability Φ_{env}, and predicted dynamic clearance from $\hat{\mathcal{S}}_{t+1}^{O}$. Candidates that violate passability or predicted clearance are discarded. The selected γ_t^{ref} is used as guidance rather than an executable trajectory with a formal tracking guarantee.

C. Reference-tracking navigation policy

The navigation policy converts the reference path into high-level command for goal progress. **Input:** navigation observation $\mathbf{o}_t^{\mathrm{nav}}$ and reference path γ_t^{ref}. **Output:** navigation command $\mathbf{u}_t^{\mathrm{nav}} \in \mathcal{U}$.

The policy outputs $\mathbf{u}_t^{\mathrm{nav}} = \pi_{\mathrm{nav}}(\mathbf{o}_t^{\mathrm{nav}}, \gamma_t^{\mathrm{ref}})$. The reference path provides tracking waypoint and desired tangential direction, which define the intended motion direction toward G. We train π_{nav} jointly with the locomotion controller in closed loop; π_{nav} outputs body-frame velocity commands, while the locomotion controller realizes them as whole-body motion during terrain traversal.

D. Reflexive evasion policy

The reflexive evasion policy provides a low-latency command for high-speed dynamic obstacles. **Input:** reflex observation $\mathbf{o}_t^{\mathrm{refl}}$ and predicted obstacle states $\hat{\mathcal{S}}_{t+1}^{O}$. **Output:** reflexive evasion command $\mathbf{u}_t^{\mathrm{refl}} \in \mathcal{U}$. The policy selects the most threatening obstacle $i_t^{\star}$ from $\hat{\mathcal{S}}_{t+1}^{O}$ using predicted reaction time under the current relative motion.

The policy network inputs the selected obstacle state and outputs $\mathbf{u}_t^{\mathrm{refl}} = \pi_{\mathrm{refl}}(\mathbf{o}_t^{\mathrm{refl}}, \hat{\mathcal{S}}_{t+1,i_t^{\star}}^{O})$. We train π_{refl} jointly with the low-level locomotion controller; π_{refl} outputs body-frame velocity commands, while the locomotion controller realizes them as whole-body evasive motions such as crouching or jumping-away. The reward is $r_t = r_{\mathrm{safe},t} + r_{\mathrm{reg},t} + r_{\mathrm{ene},t} + r_{\mathrm{rec},t}$, where the terms encourage dynamic-obstacle clearance, command regularity, energy efficiency, and recovery after evasion.

E. Threat-aware handoff

The threat-aware handoff learns how to combine the navigation and reflexive evasion commands when they conflict. **Input:** handoff observation $\mathbf{o}_t^{\mathrm{hand}}$, navigation command $\mathbf{u}_t^{\mathrm{nav}}$, reflexive evasion command $\mathbf{u}_t^{\mathrm{refl}}$, and threat score $\mathcal{T}_t$. **Output:** fused command $\mathbf{u}_t^{\mathrm{fuse}} \in \mathcal{U}$. Direct switching between $\mathbf{u}_t^{\mathrm{nav}}$ and $\mathbf{u}_t^{\mathrm{refl}}$ can cause abrupt command changes, so we use a learned fusion policy f_ψ to compute $\mathbf{u}_t^{\mathrm{fuse}} = f_\psi(\mathbf{o}_t^{\mathrm{hand}}, \mathbf{u}_t^{\mathrm{nav}}, \mathbf{u}_t^{\mathrm{refl}}, \mathcal{T}_t)$.

During handoff training, the pretrained navigation and reflex policies are fixed, and RL updates only the fusion policy f_ψ. The handoff reward is $r_t^{\mathrm{handoff}} = r_{\mathrm{coord},t} + r_{\mathrm{smooth},t} + r_{\mathrm{stable},t}$. The coordination term encourages $\mathbf{u}_t^{\mathrm{fuse}}$ to stay close to $\mathbf{u}_t^{\mathrm{nav}}$ when $\mathcal{T}_t$ is low and shift toward $\mathbf{u}_t^{\mathrm{refl}}$ when $\mathcal{T}_t$ is high. The smoothness term penalizes abrupt changes in $\mathbf{u}_t^{\mathrm{fuse}}$, and the stability term encourages stable execution.

F. CBF safety shield

The CBF safety shield is the final command filter before execution. **Input:** reduced state $\mathbf{q}_t$, fused command $\mathbf{u}_t^{\mathrm{fuse}}$, map $\mathcal{M}$, and predicted obstacle states $\hat{\mathcal{S}}_{t+1}^{O}$. **Output:** safety-filtered command $\mathbf{u}_t^{\mathrm{safe}} \in \mathcal{U}$. The shield does not make decisions; it filters $\mathbf{u}_t^{\mathrm{fuse}}$ with local barrier constraints.

We use a reduced kinematic model that maps the body-frame velocity command $\mathbf{u} \in \mathcal{U}$ to the planar state rate $\dot{\mathbf{q}}$. Given Φ_{env} and $\Phi_{\mathrm{dyn}}^{(i)}$, the shield computes $\mathbf{u}_t^{\mathrm{safe}} = \arg\min_{\mathbf{u}\in\mathcal{U}} \|\mathbf{u} - \mathbf{u}_t^{\mathrm{fuse}}\|^2$ subject to $\dot{\Phi}_{\mathrm{env}} + \alpha\Phi_{\mathrm{env}} \geq 0$ and $\dot{\Phi}_{\mathrm{dyn}}^{(i)} + \alpha\Phi_{\mathrm{dyn}}^{(i)} \geq 0$, for the dynamic obstacles considered by the shield, where $\alpha > 0$ and $\dot{\Phi}$ denotes the directional derivative induced by the reduced-order model and predicted obstacle motion. The shield is a reduced-order local safeguard over high-level commands; infeasible constraints may prevent

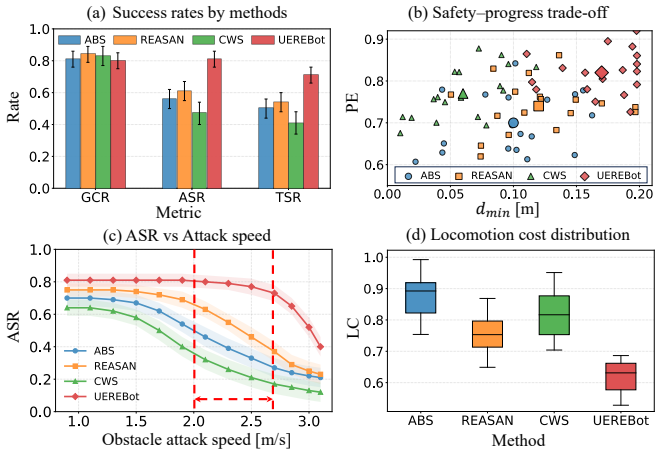

Fig. 5: Quantitative results comparing UEREBot against baselines in terms of overall success and its safety–progress trade-off, robustness under increasing attack speed, and the distribution of locomotion cost.

corrective protection, and the reduced-order constraints do not imply formal full-body safety guarantees under tracking error or model mismatch.

V. Simulation Experiments

We evaluate UEREBot in simulation under unstructured environments with high-speed dynamic obstacles, and later validate it on hardware in Sec. VI. Our experiments answer three questions: **Q1** Can UEREBot achieve higher task success and better safety–progress trade-offs than representative baselines? (Sec. V-C) **Q2** How do the threat-aware handoff, the CBF shield, and prediction affect UEREBot performance and execution safety? (Sec. V-D) **Q3** Can UEREBot be deployed on a real quadruped platform under onboard sensing, realistic noise, and latency? (Sec. VI)

A. Experiment setup

We conduct simulation experiments in Isaac Lab [30] using a quadruped model that matches the real-robot platform. All methods use the same robot model, high-level command space $\mathcal{U}$, and low-level locomotion controller. We use base motion and proprioceptive signals to construct policy observations, while the reduced state $\mathbf{q}_t$ is used for planning and safety filtering. Each episode requires the robot to move from a start region to the goal region G across uneven terrain, static obstacles, and high-speed moving balls.

B. Tasks, baselines and metrics

Tasks. We conduct simulation experiments in Isaac Lab [30] using a quadruped model that matches the real-robot platform. All methods use the same robot model, high-level command space $\mathcal{U}$, and low-level locomotion controller. We use base motion and proprioceptive signals to construct policy observations, while the reduced state $\mathbf{q}_t$ is used for planning and safety filtering. Each episode requires the robot to move from a start region to the goal region G across uneven terrain, static obstacles, and high-speed moving balls.

Baselines. We compare UEREBot with three learning-based baselines: 1) *ABS* [14], an agile/recovery dual-policy with a learned reach-avoid value function; 2) *REASAN* [45], a modular framework with navigation, reactive safety filtering, and locomotion policies; 3) *CWS* [29], a command-tracking policy using 3D scene representations for confined spaces. For fair comparison, all methods use identical task distributions and random seeds.

Metrics. We use system-level metrics to evaluate task completion, safety, and efficiency. Goal completion rate (GCR) is $\mathrm{GCR} = N_{\mathrm{goal}}/N_{\mathrm{total}}$, where N_{goal} counts episodes reaching G without static collision. Avoidance success rate (ASR) is $\mathrm{ASR} = N_{\mathrm{avoid}}/N_{\mathrm{total}}$, where N_{avoid} counts episodes without dynamic-obstacle collision. Overall task success rate (TSR) is $\mathrm{TSR} = N_{\mathrm{succ}}/N_{\mathrm{total}}$, where N_{succ} counts episodes satisfying both GCR and ASR criteria. Path efficiency (PE) is $\mathrm{PE} = L_{\mathrm{stat}}/L_{\mathrm{act}}$, the ratio of the shortest static-map path length to the actual path length. Minimum clearance is $d_{\min} = \min_{t,i\in\mathcal{O}} \Phi^{(i)}_{\mathrm{dyn}}(\mathbf{p}^R_t, t)$, the minimum clearance margin to any dynamic obstacle. Locomotion cost (LC) measures motion cost through goal distance, body roll–pitch deviation, and command variation: $\mathrm{LC} = \frac{1}{T}\sum_{t=0}^{T-1}(w_g \frac{d_G(t)}{d_0} + w_s \frac{\|\boldsymbol{\theta}_{rp}(t)\|^2}{\theta_0^2} + w_c \frac{\|\Delta\mathbf{u}_t\|^2}{u_0^2})$, where $d_G(t)$ is the distance to G, $\boldsymbol{\theta}_{rp}(t)$ is the roll–pitch vector, and $\Delta\mathbf{u}_t$ is the command change.

C. Main experiments and results

Fig. 4 shows representative simulation experiments. Across these environments, the robot follows the reference path toward the goal region across uneven terrain such as rough ground and stairs. When a dynamic obstacle approaches, the robot triggers reflexive evasion maneuvers, such as crouching down or jumping away. After evasion, it returns to tracking and completes the task.

UEREBot achieves the highest overall success, primarily driven by improved dynamic obstacle avoidance. Tab. I and Fig. 5(a) show that UEREBot improves ASR by about 20% over the baseline REASAN, with a similar gain in TSR. Meanwhile, GCR remains close across methods, indicating that the main performance gap comes from dynamic obstacle avoidance rather than goal reaching.

UEREBot improves both safety margin and progress efficiency over baselines, yielding a better safety–progress trade-off. Tab. I and Fig. 5(b) show that UEREBot increases $d_{\min}$ by $0.05\,\mathrm{m}$ over REASAN and improves PE by 0.05 over CWS. Moreover, UEREBot shows fewer samples at extremely small $d_{\min}$, suggesting that it maintains safety margins in the most safety-critical cases.

UEREBot remains more robust as the attack speed increases. Fig. 5(c) shows that ASR decreases as attack speed increases for all methods, but UEREBot sustains high ASR over a wider speed range and degrades more slowly in the high-speed regime.

UEREBot achieves lower locomotion cost with a tighter distribution. Tab. I and Fig. 5(d) show that UEREBot reduces LC by 0.13 compared with REASAN and by up to 0.26 compared with ABS. The distribution is also more concentrated,

TABLE I: Main experiment results.

Method	GCR ↑	ASR ↑	TSR ↑	PE ↑	$d_{\min}$ (m) ↑	LC ↓
ABS [14]	0.81 ± 0.05	0.56 ± 0.06	0.50 ± 0.06	0.70 ± 0.07	0.10 ± 0.04	0.88 ± 0.07
REASAN [45]	$\mathbf{0.84 \pm 0.05}$	0.61 ± 0.06	0.54 ± 0.06	0.74 ± 0.06	0.12 ± 0.04	0.75 ± 0.06
CWS [29]	0.83 ± 0.06	0.47 ± 0.07	0.41 ± 0.07	0.77 ± 0.06	0.06 ± 0.03	0.82 ± 0.07
UEREBot (ours)	0.80 ± 0.05	$\mathbf{0.81 \pm 0.05}$	$\mathbf{0.74 \pm 0.05}$	$\mathbf{0.82 \pm 0.05}$	$\mathbf{0.17 \pm 0.03}$	$\mathbf{0.62 \pm 0.05}$

Mean ± std across runs in coupled uneven terrain, static obstacles, and high-speed dynamic obstacles.
Higher is better for GCR/ASR/TSR/PE/$d_{\min}$; lower is better for LC. PE/$d_{\min}$/LC are computed over TSR-successful episodes.

TABLE II: Ablation study results.

Variant	GCR ↑	ASR ↑	TSR ↑	PE ↑	$d_{\min}$ (m) ↑	LC ↓
UEREBot	$\mathbf{0.80 \pm 0.05}$	$\mathbf{0.81 \pm 0.05}$	$\mathbf{0.74 \pm 0.05}$	$\mathbf{0.82 \pm 0.05}$	$\mathbf{0.17 \pm 0.03}$	$\mathbf{0.62 \pm 0.05}$
Hard handoff	0.74 ± 0.07	0.71 ± 0.07	0.63 ± 0.07	0.76 ± 0.06	0.13 ± 0.04	0.73 ± 0.07
w/o CBF	0.74 ± 0.07	0.79 ± 0.06	0.68 ± 0.07	0.80 ± 0.06	0.17 ± 0.04	0.65 ± 0.06
w/o pred.	0.78 ± 0.05	0.75 ± 0.06	0.69 ± 0.06	0.80 ± 0.05	0.14 ± 0.03	0.67 ± 0.06

Mean ± std across runs under the same coupled uneven terrain, static obstacles, and high-speed dynamic obstacles as the main experiments.

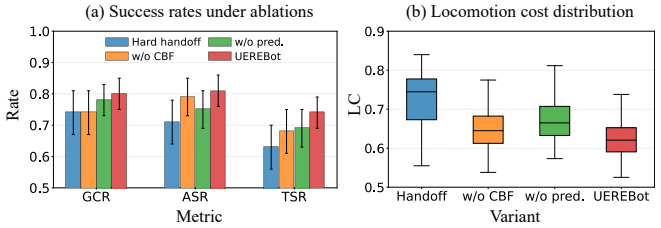

Fig. 6: Quantitative results of the ablation study comparing UEREBot against its variants in terms of success rates and the distribution of LC.

indicating less variation across successful trials.

D. Ablation studies

We conduct an ablation study on three key components of UEREBot: the threat-aware handoff that coordinates $\mathbf{u}_t^{\text{nav}}$ and $\mathbf{u}_t^{\text{refl}}$, the CBF safety shield, and the prediction module for dynamic obstacles. We construct three variants of the full system: *Hard handoff* replaces the fusion module with a hard switch between $\mathbf{u}_t^{\text{nav}}$ and $\mathbf{u}_t^{\text{refl}}$; *w/o CBF* removes the CBF shield; and *w/o pred.* disables prediction by holding the latest obstacle state.

Threat-aware handoff ensures smooth navigation–reflex coordination. As shown in Tab. II and Fig. 6, hard handoff yields the largest drop in TSR by about 10% and increases LC by roughly 0.1. This suggests that abrupt switching between $\mathbf{u}_t^{\text{nav}}$ and $\mathbf{u}_t^{\text{refl}}$ introduces command discontinuities that destabilize locomotion during handoff, making both goal progress and reflexive evasion less reliable.

CBF shield prevents reflexive evasion from colliding with static obstacles. As shown in Tab. II and Fig. 6, w/o CBF reduces GCR by about 5% and lowers TSR by around 6%. Without the CBF shield, reflexive evasion can collide with static obstacles or enter infeasible terrain under high-speed attacks. LC also increases slightly, consistent with more frequent corrective actions.

Prediction improves reflexive evasion safety margin. As shown in Tab. II and Fig. 6(a), w/o pred. reduces ASR by about 5% and leads to a clear decrease in $d_{\min}$. This drop is caused by delayed reflexive evasion under high-speed attacks when prediction is disabled, leaving less reaction time and tighter clearances.

VI. Real-robot Experiments

A. Real-robot setup

We deploy UEREBot on a Unitree Go2 using a body-frame velocity command interface. We equip the robot with a MID-360 LiDAR and a forward RGB-D camera for onboard perception. The LiDAR provides 360° coverage, while the RGB-D camera covers the frontal field of view. We run the full UEREBot stack onboard in closed loop with the same modules and interfaces as in simulation. We conduct 200 real-robot trials in both indoor and outdoor environments. Indoor scenarios include rooms and corridors with static obstacles such as boxes and chairs. Outdoor scenarios include grass, slopes, and steps with static obstacles in the environment. Dynamic obstacles include thrown balls, fast approaching pedestrians, and close-range stick pokes. In each trial, the robot performs locomotion toward a goal region G while avoiding dynamic obstacles. We evaluate the same metrics as in simulation using the same definitions.

B. Real-robot results and analysis

Quantitative results. We summarize quantitative results over 200 real-robot trials spanning indoor and outdoor settings. UEREBot achieves 75% GCR, 70% ASR, and 64% TSR over all trials. Compared to simulation, these success rates decrease on hardware as expected. LC averaged over TSR-successful trials is 0.74, reflecting a higher locomotion cost on hardware than in simulation. Overall, these results support effective onboard deployment of UEREBot in real-robot evaluation.

Qualitative observations and Sim2Real gap. Fig. 7 shows representative robot executions in the evaluated scenarios. Across these examples, the robot maintains smooth locomotion toward the goal region and performs reflexive evasion maneuvers, including crouching down, jumping away, and side stepping, under fast attacks. Compared to simulation, success rates decrease and LC increases on hardware due to three factors: onboard computation increases end-to-end latency and shortens the effective reaction time window; obstacle

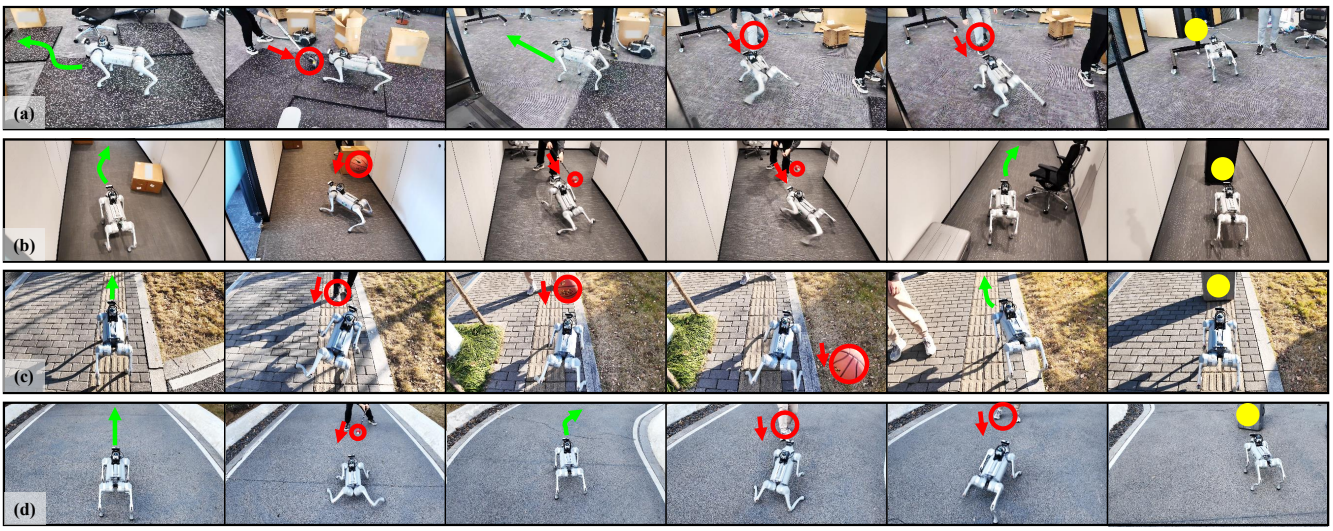

Fig. 7: **Real-robot experiments.** Real-robot trials are conducted in scenarios with varied terrain and layout. (a) Indoor room with rough terrain. (b) Indoor corridor. (c) Outdoor rough terrain. (d) Outdoor slope. (See real-robot videos)

perception is affected by noise, missing measurements, and brief occlusions, which reduce prediction reliability; and real-world terrain and contact conditions are more complex than in simulation, increasing tracking errors and corrective actions.

VII. Limitation and Future Works

UEREBot has several limitations. The CBF shield is a reduced-order local safeguard over high-level velocity commands and does not provide a formal full-body safety guarantee under tracking error, model mismatch, or infeasible constraints. The real-robot experiments demonstrate onboard deployment but do not constitute a standardized hardware benchmark, and are limited to one quadruped platform with a fixed sensing and computing setup. Our experiments also do not yet cover denser clutter, unknown spaces, stronger perception degradation, low-friction terrain, or repeated blind-spot attacks. Future work will improve full-body safety-aware filtering, expand hardware evaluation to more platforms and standardized scenarios, and test longer deployments with more complex environments and dynamic interactions.

VIII. Conclusion

In this paper, we presented UEREBot, a hierarchical framework for safe quadrupedal navigation in unstructured environments with uneven terrain and high-speed dynamic obstacles. UEREBot integrates a spatial–temporal planner, navigation and reflexive evasion policies, a threat-aware handoff module, and a CBF safety shield for execution-time safety filtering. We evaluated UEREBot in simulation and on a real quadruped robot. Compared with representative baselines, UEREBot achieves higher avoidance and task success rates, a better safety–progress trade-off, and lower locomotion cost across successful trials. These results support the effectiveness of UEREBot for safe quadrupedal navigation in complex dynamic environments.

References

[1] Ziyu Cao, William Talbot, and Kailai Li. Resple: Recursive spline estimation for lidar-based odometry. *IEEE Robotics and Automation Letters*, 2025.

[2] Jiaqi Chen, Jonas Frey, Ruyi Zhou, Takahiro Miki, Georg Martius, and Marco Hutter. Identifying terrain physical parameters from vision-towards physical-parameter-aware locomotion and navigation. *IEEE Robotics and Automation Letters*, 2024.

[3] Weinan Chen, Wenzheng Chi, Sehua Ji, Hanjing Ye, Jie Liu, Yunjie Jia, Jiajie Yu, and Jiyu Cheng. A survey of autonomous robots and multi-robot navigation: Perception, planning and collaboration. *Biomimetic Intelligence and Robotics*, 5(2):100203, 2025.

[4] Xuechao Chen, Yidong Du, Zishun Zhou, Zhicheng Yuan, Qingrui Zhao, Fei Meng, Zhangguo Yu, Peng Lu, and Qiang Huang. Online behavior-centric adaptation for bipedal robot sim-to-real transfer with unmodeled dynamics mismatch. *IEEE Transactions on Automation Science and Engineering*, 23:1533–1545, 2025.

[5] Yeke Chen, Ji Ma, Zeren Luo, Yimin Han, Yinzhao Dong, Bowen Xu, and Peng Lu. Learning autonomous and safe quadruped traversal of complex terrains using multi-layer elevation maps. *IEEE Robotics and Automation Letters*, 2025.

[6] Thomas Corbères, Carlos Mastalli, Wolfgang Merkt, Jaehyun Shim, Ioannis Havoutis, Maurice Fallon, Nicolas Mansard, Thomas Flayols, Sethu Vijayakumar, and Steve Tonneau. Perceptive locomotion through whole-body mpc and optimal region selection. *IEEE Access*, 2025.

[7] Daniel Louback da Silva Lubanco, Markus Pichler-Scheder, and Thomas Schlechter. A novel frontier-based exploration algorithm for mobile robots. In *2020 6th International Conference on Mechatronics and Robotics Engineering (ICMRE)*, pages 1–5. IEEE, 2020.

[8] Mihir Dharmadhikari, Tung Dang, Lukas Solanka, Jo-

hannes Loje, Huan Nguyen, Nikhil Khedekar, and Kostas Alexis. Motion primitives-based path planning for fast and agile exploration using aerial robots. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 179–185. IEEE, 2020.

[9] Hongbo Duan, Shangyi Luo, Zhiyuan Deng, Yanbo Chen, Yuanhao Chiang, Yi Liu, Fangming Liu, and Xueqian Wang. Causalnav: A long-term embodied navigation system for autonomous mobile robots in dynamic outdoor scenarios. *IEEE Robotics and Automation Letters*, 2026.

[10] Xiyu Fan, Minghao Lu, Bowen Xu, and Peng Lu. Flying in highly dynamic environments with end-to-end learning approach. *IEEE Robotics and Automation Letters*, 2025.

[11] Jonas Frey, David Hoeller, Shehryar Khattak, and Marco Hutter. Locomotion policy guided traversability learning using volumetric representations of complex environments. In *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5722–5729. IEEE, 2022.

[12] Yimin Han, Jiahui Zhang, Zeren Luo, Yingzhao Dong, Jinghan Lin, Liu Zhao, Shihao Dong, and Peng Lu. Omninet: Omnidirectional jumping neural network with height-awareness for quadrupedal robots. *IEEE Robotics and Automation Letters*, 2025.

[13] Chengyang He, Tianze Yang, Tanishq Duhan, Yutong Wang, and Guillaume Sartoretti. Alpha: Attention-based long-horizon pathfinding in highly-structured areas. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 14576–14582. IEEE, 2024.

[14] Tairan He, Chong Zhang, Wenli Xiao, Guanqi He, Changliu Liu, and Guanya Shi. Agile but safe: Learning collision-free high-speed legged locomotion. *arXiv preprint arXiv:2401.17583*, 2024.

[15] Hasitha Sanjeewa Hewawasam, M Yousef Ibrahim, and Gayan Kahandawa Appuhamillage. Past, present and future of path-planning algorithms for mobile robot navigation in dynamic environments. *IEEE Open Journal of the Industrial Electronics Society*, 3:353–365, 2022.

[16] Han Jiang, Teng Chen, Guoteng Zhang, Xuewen Rong, and Yibin Li. A nonlinear mpc and hybrid whole-body control framework for optimizing agile motions in quadruped robots. *International Journal of Control, Automation and Systems*, 23(9):2666–2677, 2025.

[17] Hyeongjun Kim, Hyunsik Oh, Jeongsoo Park, Yunho Kim, Donghoon Youm, Moonkyu Jung, Minho Lee, and Jemin Hwangbo. High-speed control and navigation for quadrupedal robots on complex and discrete terrain. *Science Robotics*, 10(102):eads6192, 2025.

[18] Joonho Lee, Marko Bjelonic, Alexander Reske, Lorenz Wellhausen, Takahiro Miki, and Marco Hutter. Learning robust autonomous navigation and locomotion for wheeled-legged robots. *Science Robotics*, 9(89): eadi9641, 2024.

[19] Jingsong Liang, Zhichen Wang, Yuhong Cao, Jimmy Chiun, Mengqi Zhang, and Guillaume Adrien Sartoretti. Context-aware deep reinforcement learning for autonomous robotic navigation in unknown area. In *Conference on Robot Learning*, pages 1425–1436. PMLR, 2023.

[20] Jingsong Liang, Yuhong Cao, Yixiao Ma, Hanqi Zhao, and Guillaume Sartoretti. Hdplanner: Advancing autonomous deployments in unknown environments through hierarchical decision networks. *IEEE Robotics and Automation Letters*, 2024.

[21] Hao Liu, Yi Shen, Chang Zhou, Yuelin Zou, Zijun Gao, and Qi Wang. Td3 based collision free motion planning for robot navigation. In *2024 6th International Conference on Communications, Information System and Computer Engineering (CISCE)*, pages 247–250. IEEE, 2024.

[22] Hongyi Liu and Quan Yuan. Safe and robust motion planning for autonomous navigation of quadruped robots in cluttered environments. *IEEE Access*, 12:69728–69737, 2024.

[23] Jianwei Liu, Shirui Lyu, Denis Hadjivelichkov, Valerio Modugno, and Dimitrios Kanoulas. Vit-a*: Legged robot path planning using vision transformer a. In *2023 IEEE-RAS 22nd International Conference on Humanoid Robots (Humanoids)*, pages 1–6. IEEE, 2023.

[24] Guanglin Lu, Teng Chen, Xuewen Rong, Guoteng Zhang, Jian Bi, Jingxuan Cao, Han Jiang, and Yibin Li. Whole-body motion planning and control of a quadruped robot for challenging terrain. *Journal of Field Robotics*, 40(6): 1657–1677, 2023.

[25] Minghao Lu, Han Chen, and Peng Lu. Perception and avoidance of multiple small fast moving objects for quadrotors with only low-cost rgbd camera. *IEEE Robotics and Automation Letters*, 7(4):11657–11664, 2022.

[26] Minghao Lu, Xiyu Fan, Han Chen, and Peng Lu. Fapp: Fast and adaptive perception and planning for uavs in dynamic cluttered environments. *IEEE Transactions on Robotics*, 2024.

[27] Yidan Lu, Yinzhao Dong, Jiahui Zhang, Ji Ma, and Peng Lu. Fr-net: Learning robust quadrupedal fall recovery on challenging terrains through mass-contact prediction. *IEEE Robotics and Automation Letters*, 2025.

[28] 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.

[29] Takahiro Miki, Joonho Lee, Lorenz Wellhausen, and Marco Hutter. Learning to walk in confined spaces using 3d representation. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 8649–8656. IEEE, 2024.

[30] Mayank Mittal, Pascal Roth, James Tigue, Antoine Richard, Octi Zhang, Peter Du, Antonio Serrano-Muñoz, Xinjie Yao, René Zurbrügg, Nikita Rudin, et al. Isaac lab: A gpu-accelerated simulation framework for multi-modal robot learning. *arXiv preprint arXiv:2511.04831*,

2025.

[31] Xue Bin Peng, Erwin Coumans, Tingnan Zhang, Tsang-Wei Lee, Jie Tan, and Sergey Levine. Learning agile robotic locomotion skills by imitating animals. *arXiv preprint arXiv:2004.00784*, 2020.

[32] Mingyo Seo, Ryan Gupta, Yifeng Zhu, Alexy Skoutnev, Luis Sentis, and Yuke Zhu. Learning to walk by steering: Perceptive quadrupedal locomotion in dynamic environments. *arXiv preprint arXiv:2209.09233*, 2022.

[33] Kuankuan Sima, Longbin Tang, Haozhe Ma, and Lin Zhao. Macronav: Multi-task context representation learning enables efficient navigation in unknown environments. *arXiv preprint arXiv:2511.04320*, 2025.

[34] Yuting Tao, Mingyang Li, Xiao Cao, and Peng Lu. Mobile robot collision avoidance based on deep reinforcement learning with motion constraints. *IEEE Transactions on Intelligent Vehicles*, 2024.

[35] Erik Wijmans, Abhishek Kadian, Ari Morcos, Stefan Lee, Irfan Essa, Devi Parikh, Manolis Savva, and Dhruv Batra. Dd-ppo: Learning near-perfect pointgoal navigators from 2.5 billion frames. *arXiv preprint arXiv:1911.00357*, 2019.

[36] Erdong Xiao, Yinzhao Dong, James Lam, and Peng Lu. Learning stable bipedal locomotion skills for quadrupedal robots on challenging terrains with automatic fall recovery. *npj Robotics*, 3(1):22, 2025.

[37] Xuesu Xiao, Bo Liu, Garrett Warnell, and Peter Stone. Motion planning and control for mobile robot navigation using machine learning: a survey. *Autonomous Robots*, 46(5):569–597, 2022.

[38] Kang Xu, Yanqun Lu, Lei Shi, Jianyong Li, Shoukun Wang, and Tao Lei. Whole-body stability control with high contact redundancy for wheel-legged hexapod robot driving over rough terrain. *Mechanism and machine theory*, 181:105199, 2023.

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

[40] 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, 2025.

[41] Zihao Xu, Kuankuan Sima, Junhao Deng, Zixuan Zhuang, Chunzheng Wang, Ce Hao, and Jin Song Dong. Aprebot: Active perception system for reflexive evasion robot. *arXiv preprint arXiv:2509.24733*, 2025.

[42] Fan Yang, Chao Cao, Hongbiao Zhu, Jean Oh, and Ji Zhang. Far planner: Fast, attemptable route planner using dynamic visibility update. In *2022 ieee/rsj international conference on intelligent robots and systems (iros)*, pages 9–16. IEEE, 2022.

[43] Yuxiang Yang, Ken Caluwaerts, Atil Iscen, Tingnan Zhang, Jie Tan, and Vikas Sindhwani. Data efficient reinforcement learning for legged robots. In *Conference on Robot Learning*, pages 1–10. PMLR, 2020.

[44] Jiyu Yu, Dongqi Wang, Zhenghan Chen, Ci Chen, Shuangpeng Wu, Yue Wang, and Rong Xiong. A fast motion and foothold planning framework for legged robots on discrete terrain. In *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 10678–10685. IEEE, 2024.

[45] Qihao Yuan, Ziyu Cao, Ming Cao, and Kailai Li. Reasan: Learning reactive safe navigation for legged robots. *arXiv preprint arXiv:2512.09537*, 2025.

[46] Chong Zhang, Nikita Rudin, David Hoeller, and Marco Hutter. Learning agile locomotion on risky terrains. In *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 11864–11871. IEEE, 2024.

[47] Yuying Zhang, Na Fan, Haowen Zheng, Junning Liang, Zongliang Pan, Qifeng Chen, and Ximin Lyu. Threat-aware uav dodging of human-thrown projectiles with an rgb-d camera. *IEEE Robotics and Automation Letters*, 11(2):1178–1185, 2025.

[48] Zewei Zhang, Chenhao Li, Takahiro Miki, and Marco Hutter. Motion priors reimagined: Adapting flat-terrain skills for complex quadruped mobility. *arXiv preprint arXiv:2505.16084*, 2025.

[49] Hongbiao Zhu, Chao Cao, Yukun Xia, Sebastian Scherer, Ji Zhang, and Weidong Wang. Dsvp: Dual-stage viewpoint planner for rapid exploration by dynamic expansion. In *2021 IEEE/RSJ international conference on intelligent robots and systems (IROS)*, pages 7623–7630. IEEE, 2021.

APPENDIX

A. Details of the spatial–temporal planner

The spatial–temporal planner is implemented as a multi-head learning module with asynchronous outputs. The reference path γ_t^{ref} is updated at a lower rate (10–20 Hz), while the dynamic obstacle prediction $\hat{\mathcal{S}}_{t+1}^{O}$ and the threat score $\mathcal{T}_t$ are updated at a higher rate (50 Hz). Downstream modules always consume the most recent outputs, with γ_t^{ref} held constant between consecutive low-rate updates.

The reference path γ_t^{ref} represents a locally feasible geometric intent toward the goal region and is selected from a fixed set of candidate paths. The planner uses a local 2.5D environment representation constructed offline prior to deployment, from which the environment passability field Φ_{env} is derived. Candidate paths are instantiated as motion primitives conditioned on the current reduced state $\mathbf{q}_t$. Candidates that violate $\Phi_{\mathrm{env}}(\cdot) > 0$ at any sampled location along the path are discarded. Candidates that would immediately violate dynamic clearance under the current obstacle prediction are also removed. Among the remaining candidates, a learned scoring head ranks the candidates and selects the highest-scoring one as γ_t^{ref}. The scoring head is trained with supervision constructed from simulation rollouts, where candidate quality is evaluated based on goal progress, environment passability margins, and motion efficiency. Learning is used here to capture preference among feasible candidates rather than to enforce safety constraints.

The prediction head outputs $\hat{\mathcal{S}}_{t+1}^{O}$, a one-step prediction of dynamic obstacle states aligned with the fast update interval. This prediction is used to compensate for limited perception update rates by providing a time-aligned obstacle state for execution-time modules. The prediction head is trained in simulation using rollout data, where ground-truth next-step obstacle states are available. On the real system, the trained model is used directly to generate time-aligned predictions between perception updates. The prediction output is consumed by the reflexive policy and the CBF shield, and does not affect reference path generation.

The threat head outputs a scalar threat score $\mathcal{T}_t$ that summarizes the urgency of dynamic interactions under limited reaction time. The threat score is computed from the shared planner representation and takes the predicted one-step obstacle states as input. Supervision for the threat head is derived from a pressure signal that combines dynamic clearance margin, relative motion between the robot and obstacles, and the tendency of the interaction to deteriorate within the next control step. As a result, the threat score increases not only when clearance becomes small, but also when obstacles are approaching rapidly. The threat score is updated at the same fast rate as the prediction and is used solely for continuous modulation in the threat-aware handoff; it does not directly determine control actions or override feasibility checks.

B. Details of navigation and reflex policies

The navigation policy π_{nav} is trained independently to output nominal commands that track the reference path γ_t^{ref} while remaining terrain-executable and cost-efficient. Its per-step reward is defined as

$$r_t^{\mathrm{nav}} = r_{\mathrm{prog},t} + r_{\mathrm{track},t} + r_{\mathrm{stable},t}^{\mathrm{nav}} + r_{\mathrm{effort},t}. \tag{1}$$

Path progress. Let $\Pi(\mathbf{p})$ denote the projection of planar position $\mathbf{p}$ onto the reference path γ_t^{ref}, and let $s(\mathbf{p})$ be the arc-length coordinate of the projected point along the path. We reward incremental advancement along γ_t^{ref} toward the goal region G:

$$r_{\mathrm{prog},t} = s\big(\mathbf{p}_t^R\big) - s\big(\mathbf{p}^R(t-\Delta t)\big)\,. \tag{2}$$

Reference tracking. We discourage lateral deviation from the path and encourage heading alignment with the local path direction using exponential shaping of lateral and heading errors:

$$r_{\mathrm{track},t} = \exp\Big(-\alpha_p\big\|\mathbf{p}_t^R - \Pi(\mathbf{p}_t^R)\big\|^2 - \alpha_\psi\big(\Delta\psi_t\big)^2\Big), \tag{3}$$

where $\Delta\psi_t$ is the yaw difference between the robot heading and the path tangent at $\Pi(\mathbf{p}_t^R)$. This term encourages geometric consistency with γ_t^{ref} so that forward progress is achieved near the reference path rather than through large lateral drift.

Terrain-executable stability. We promote terrain-executable body motion on uneven terrain by shaping roll–pitch deviation and base angular motion:

$$r_{\mathrm{stable},t}^{\mathrm{nav}} = \exp\Big(-\alpha_{\mathrm{rp}}\|\boldsymbol{\theta}_{\mathrm{rp},t}\|^2 - \alpha_\omega\|\boldsymbol{\omega}_t^R\|^2\Big). \tag{4}$$

This term discourages excessive body tilt and aggressive rotations, which are indicative of poorly conditioned locomotion on uneven terrain. By constraining these body-level quantities, the nominal navigation behavior remains terrain-executable.

Cost efficiency. We discourage unnecessarily high actuation during nominal navigation by shaping instantaneous mechanical effort:

$$r_{\mathrm{effort},t} = \exp\Big(-\alpha_e\sum_j\big|\tau_{j,t}\dot{q}_{j,t}\big|\Big). \tag{5}$$

This term penalizes strategies that rely on large joint power to obtain progress or tracking, and encourages cost-efficient nominal motion with less aggressive actuation.

The reflexive evasion policy π_{refl} is trained to generate rapid, short-horizon commands that create sufficient separation from high-speed dynamic obstacles under limited reaction time, while remaining physically executable and enabling subsequent recovery. Its per-step reward follows the reflexive evasion reward decomposition, and the individual terms are specified as follows.

Safety. The primary objective of reflexive evasion is to increase separation from the most threatening dynamic obstacle. We therefore shape the safety reward using the dynamic clearance field $\Phi_{\mathrm{dyn}}^{(i_t^\star)}(\mathbf{p}^R(t),t)$ defined in the main text:

$$r_{\mathrm{safe},t} = \tanh\big(\alpha_\Phi\,\Phi_{\mathrm{dyn}}^{(i_t^\star)}(\mathbf{p}_t^R,t)\big) - \alpha_v\big(\max(0,v_{\mathrm{app},t})\big)^2. \tag{6}$$

The first term provides a continuous incentive to increase clearance, with mild saturation to avoid unbounded scaling. The second term penalizes positive approaching speed $v_{\mathrm{app},t}$, which denotes the radial component of the relative velocity between the robot and the selected threatening obstacle, discouraging evasive actions that continue to reduce separation.

Regularization. To ensure that aggressive evasive maneuvers remain physically executable, we include a body-level regularization term:

$$r_{\mathrm{reg},t} = \frac{1}{1 + \alpha_{\mathrm{rp}}^{\mathrm{reg}} \|\boldsymbol{\theta}_{\mathrm{rp},t}\|^2 + \alpha_{\omega}^{\mathrm{reg}} \|\boldsymbol{\omega}_t^R\|^2}. \tag{7}$$

This term favors small roll–pitch deviation and moderate base rotational motion, providing a lightweight executability prior during reflexive evasion.

Energy. To avoid achieving evasive separation through unnecessarily high actuation, we penalize instantaneous mechanical effort:

$$r_{\mathrm{ene},t} = \exp\Big(- \alpha_{\mathrm{ene}} \sum_j \big|\tau_{j,t} \dot{q}_{j,t}\big| \Big). \tag{8}$$

This term promotes energy-efficient reflexive actions and suppresses impulsive control strategies.

Recovery. Finally, we encourage the robot to return to a stable, upright configuration after aggressive evasive maneuvers using a system-level recovery term:

$$r_{\mathrm{rec},t} = \exp\Big(- \alpha_h \big(p_{z,t}^R - h^{\mathrm{nom}}\big)^2 - \alpha_{\mathrm{rp}}^{\mathrm{rec}} \|\boldsymbol{\theta}_{\mathrm{rp},t}\|^2 \Big). \tag{9}$$

where $p_{z,t}^R$ denotes the robot base height. This term promotes restoration of nominal base height and upright posture, facilitating a smooth transition from reflexive evasion back to normal locomotion.

C. Details of threat-aware handoff

This section provides reward details for training the threat-aware handoff. In training, we optimize only the handoff module while treating the pretrained navigation and reflex policies as fixed controllers. The handoff reward is defined as

$$r_t^{\mathrm{handoff}} = r_{\mathrm{coord},t} + r_{\mathrm{smooth},t} + r_{\mathrm{stable},t}. \tag{10}$$

Coordination. We encourage the fused command to stay close to the navigation output under low threat and to the reflexive output under high threat. Let $g(\mathcal{T}_t) \in [0,1]$ be a monotone mapping of the threat score. We use exponential shaping of the control discrepancy:

$$\begin{aligned} r_{\mathrm{coord},t} = \big(1 - g(\mathcal{T}_t)\big) \exp\Big(-\alpha_{\mathrm{nav}} \big\|\mathbf{u}_t^{\mathrm{fuse}} - \mathbf{u}_t^{\mathrm{nav}}\big\|^2\Big) \\ + g(\mathcal{T}_t) \exp\Big(-\alpha_{\mathrm{refl}} \big\|\mathbf{u}_t^{\mathrm{fuse}} - \mathbf{u}_t^{\mathrm{refl}}\big\|^2\Big). \end{aligned} \tag{11}$$

This term provides continuous, threat-aware guidance for fusion by smoothly biasing $\mathbf{u}_t^{\mathrm{fuse}}$ toward the appropriate source as $\mathcal{T}_t$ increases.

Smoothness. We stabilize blending by discouraging abrupt changes in the fused command over time. Define the control increment $\Delta \mathbf{u}_t^{\mathrm{fuse}} = \mathbf{u}_t^{\mathrm{fuse}} - \mathbf{u}_{t-\Delta t}^{\mathrm{fuse}}$. We apply a bounded shaping term:

$$r_{\mathrm{smooth},t} = \tanh\Big(\alpha_{\mathrm{smooth}} \big/ \big(\|\Delta \mathbf{u}_t^{\mathrm{fuse}}\| + \varepsilon\big)\Big), \tag{12}$$

where $\varepsilon > 0$ is a small constant for numerical stability. This term assigns higher reward to smaller command increments and saturates smoothly to avoid excessive scaling.

Stability. We regularize posture during fusion to maintain well-conditioned body configurations. Using the base roll–pitch vector $\boldsymbol{\theta}_{\mathrm{rp},t}$, we define

$$r_{\mathrm{stable},t} = \exp\Big(-\alpha_{\mathrm{rp}}^{\mathrm{stab}} \, \|\boldsymbol{\theta}_{\mathrm{rp},t}\|^2\Big). \tag{13}$$

This term promotes stable posture throughout fusion, which supports reliable execution of the blended commands.

We find that hard handoff often causes unstable transitions and may lead to falls (Fig. 8).

D. Details of the CBF shield

Command filtering. We use the reduced kinematic model $\dot{\mathbf{q}} = g(\mathbf{q})\mathbf{u}$ to compute directional derivatives over high-level commands. Without this shield, the robot may collide with nearby static obstacles during reflexive evasion, as shown in Fig. 9. We construct the environment barrier as $h_{\mathrm{env}}(\mathbf{q}_t) = \Phi_{\mathrm{env}}(\mathbf{p}_t^R)$, where $\mathbf{p}_t^R = (x_t, y_t)$, and the dynamic-obstacle barrier as $h_{\mathrm{dyn}}^{(i)}(\mathbf{q}_t, t) = \Phi_{\mathrm{dyn}}^{(i)}(\mathbf{p}_t^R, t)$ using the one-step predicted obstacle state. The barrier values depend on the state, while their directional derivatives depend on the command through the reduced kinematic model. For each active barrier, we impose the CBF condition $\dot{h} + \alpha h \geq 0$, where $\dot{h}$ is the directional derivative induced by the reduced-order model and, for dynamic obstacles, the predicted obstacle motion. The resulting execution-time admissible command set is

$$\begin{aligned} V_{\mathrm{safe}}(t) = \{\mathbf{u} \in \mathcal{U} \mid \; & \dot{h}_{\mathrm{env}}(\mathbf{q}_t, \mathbf{u}) + \alpha_{\mathrm{env}} h_{\mathrm{env}}(\mathbf{q}_t) \geq 0, \\ & \dot{h}_{\mathrm{dyn}}^{(i)}(\mathbf{q}_t, t, \mathbf{u}) + \alpha_{\mathrm{dyn}} h_{\mathrm{dyn}}^{(i)}(\mathbf{q}_t, t) \geq 0, \\ & \forall i \in \mathcal{A}_t\}. \end{aligned} \tag{14}$$

where $\mathcal{A}_t$ is the active set of dynamic obstacles considered by the shield.

Given the fused command $\mathbf{u}_t^{\mathrm{fuse}}$, the shield computes

$$\mathbf{u}_t^{\mathrm{safe}} = \arg \min_{\mathbf{u} \in V_{\mathrm{safe}}(t)} \big\|\mathbf{u} - \mathbf{u}_t^{\mathrm{fuse}}\big\|^2. \tag{15}$$

If $V_{\mathrm{safe}}(t)$ is empty, the shield may fail to provide corrective protection. This reduced-order filter does not imply formal full-body safety guarantees under tracking error or model mismatch.

E. Onboard perception system

The real-robot system adopts the onboard perception pipeline from APREBot [41] and runs it fully on Jetson Orin NX. We use a 360° LiDAR together with a forward-facing RGB-D camera in a complementary way: the LiDAR stream provides all-around coverage and continuous tracking, while the RGB-D stream refines obstacle-state estimates when the

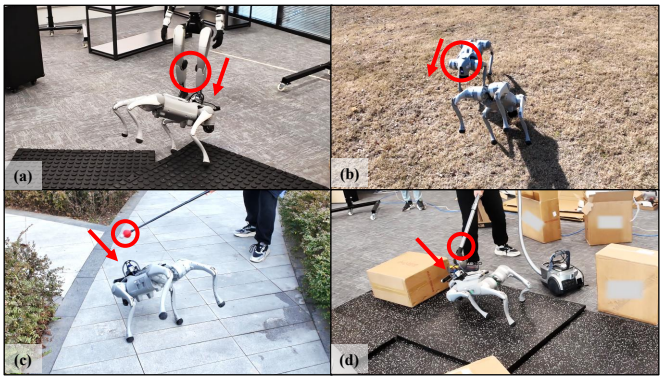

Fig. 8: Failure examples with hard handoff. Each subfigure (a–d) shows a representative trial where the hard handoff between navigation and reflex commands causes an unstable transition, leading to loss of balance and a fall.

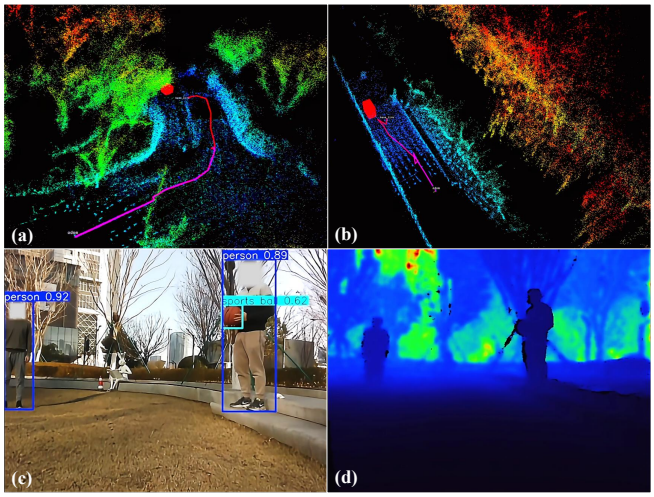

Fig. 10: Onboard perception examples on the real robot. (a)(b) LiDAR point clouds from the omnidirectional 3D LiDAR scans. (c) RGB image. (d) Depth image.

obstacle is observable in the frontal view. For each dynamic obstacle, the perception module outputs a metrically consistent state estimate, including position, velocity, and effective radius, for downstream modules. Fig. 10 shows example onboard sensing outputs from LiDAR point clouds, RGB images, and depth images.

LiDAR-based omnidirectional sensing. Livox MID-360 is used to monitor moving objects around the robot. Each scan is aligned to a common world frame using LiDAR–inertial localization and then simplified by basic filtering, including region-of-interest selection and downsampling. We identify obstacle point groups by DBSCAN clustering and keep their identities consistent over time through Hungarian matching with track initialization and termination. A lightweight 3D Kalman filter is applied to suppress jitter and to produce stable obstacle position, velocity, and effective-radius estimates.

RGB-D-based frontal refinement. Intel RealSense D435 provides refined estimates when an obstacle falls inside the camera field of view. We run YOLO on RGB images to obtain detections and use ByteTrack to connect detections across

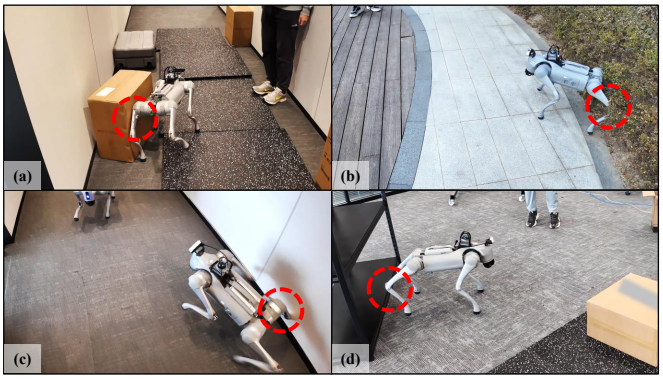

Fig. 9: Failure examples without the CBF safeguard. During evasive motion, the robot collides with a nearby static obstacle, highlighting the role of the safe shield in preventing static collisions.

time. For each tracked object, we extract a compact depth region by growing from the track center on the depth image, and then lift the selected depth pixels into a local 3D point set using calibrated camera parameters. The refined 3D centroid and effective radius are computed from this point set, and the obstacle velocity is obtained from consecutive refined 3D positions, complementing the LiDAR-based tracking.

F. Simulation randomization settings

We apply domain randomization during simulation training to improve robustness to state-estimation noise, obstacle-observation noise, contact variation, and latency. Tab. III summarizes the randomized factors and their ranges.

The randomization includes bounded perturbations on robot state estimates and obstacle observations, episode-level variations in obstacle initial position and speed, and system-level variations in ground contact and end-to-end latency.

G. Additional experimental illustrations

We present additional qualitative results from simulation and real-robot experiments to showcase a wider range of evaluation scenarios.

In simulation, we include extended scenarios that vary both terrain and spatial constraints, covering open rooms and confined corridors with flat ground, rough terrain, and slopes (Fig. 11).

On the real robot, we further present additional indoor trials in open rooms and corridors under different terrain conditions, including flat ground, step terrain, and rough surfaces (Fig. 12).

We also include extended outdoor trials across diverse terrains, including riverside trails, stone-paved paths, wooden floors, slopes, and grass fields (Fig. 13).

TABLE III: Domain randomization used in simulation training.

Randomized factor	Value
Base position noise (per axis)	$\mathcal{U}(-0.03, 0.03)$ m
Base yaw noise	$\mathcal{U}(-2, 2)$ deg
Base linear velocity noise (per axis)	$\mathcal{U}(-0.10, 0.10)$ m/s
Base angular velocity noise (per axis)	$\mathcal{U}(-0.15, 0.15)$ rad/s
Obstacle initial position (per axis)	$\mathcal{U}(-3.0, 3.0)$ m
Obstacle speed	$\mathcal{U}(0.5, 4.0)$ m/s
Obstacle position noise (per axis)	$\mathcal{U}(-0.05, 0.05)$ m
Obstacle velocity noise (per axis)	$\mathcal{U}(-0.20, 0.20)$ m/s
Observation dropout	$p = 0.05$, hold 1–3 steps
Ground friction factor	$\mathcal{U}(0.6, 1.2)$
End-to-end latency	$\mathcal{U}(0.00, 0.06)$ s

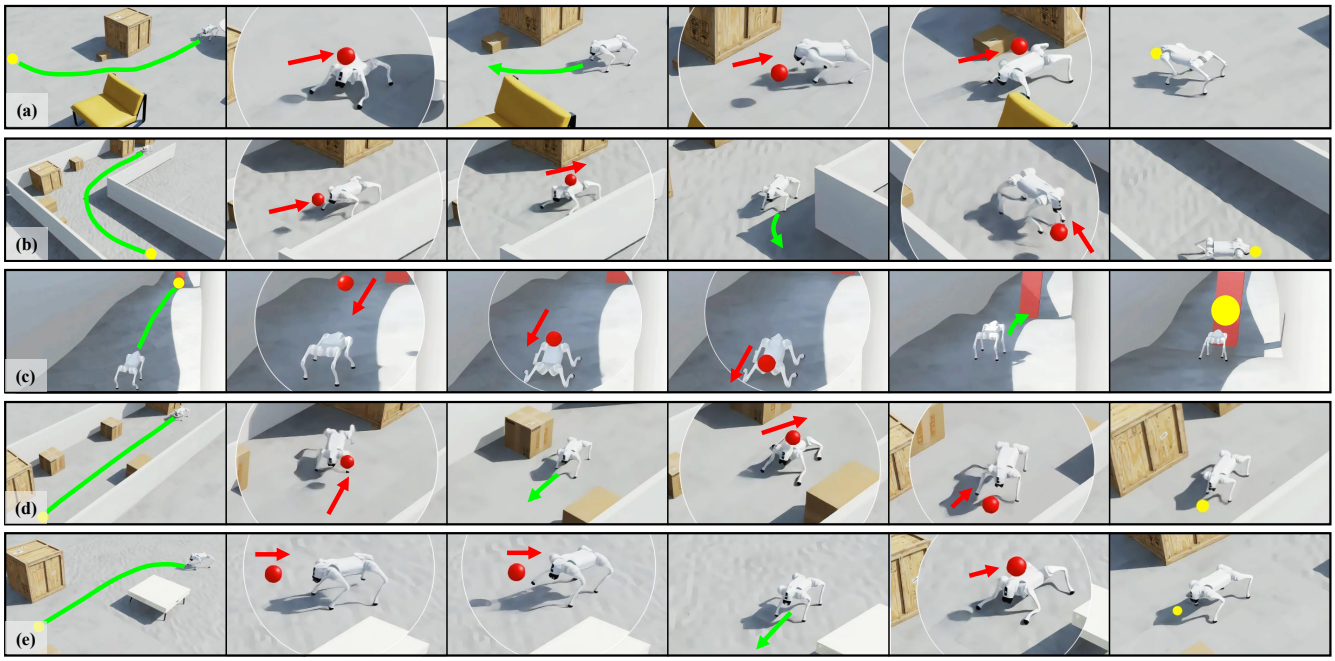

Fig. 11: **Additional simulation scenarios.** Extended qualitative results in simulation across diverse terrain and space constraints. (a) Open flat ground with static obstacles. (b) Confined corridor with rough terrain. (c) Slope terrain. (d) Confined corridor on flat ground. (e) Open space combining flat and rough terrain.

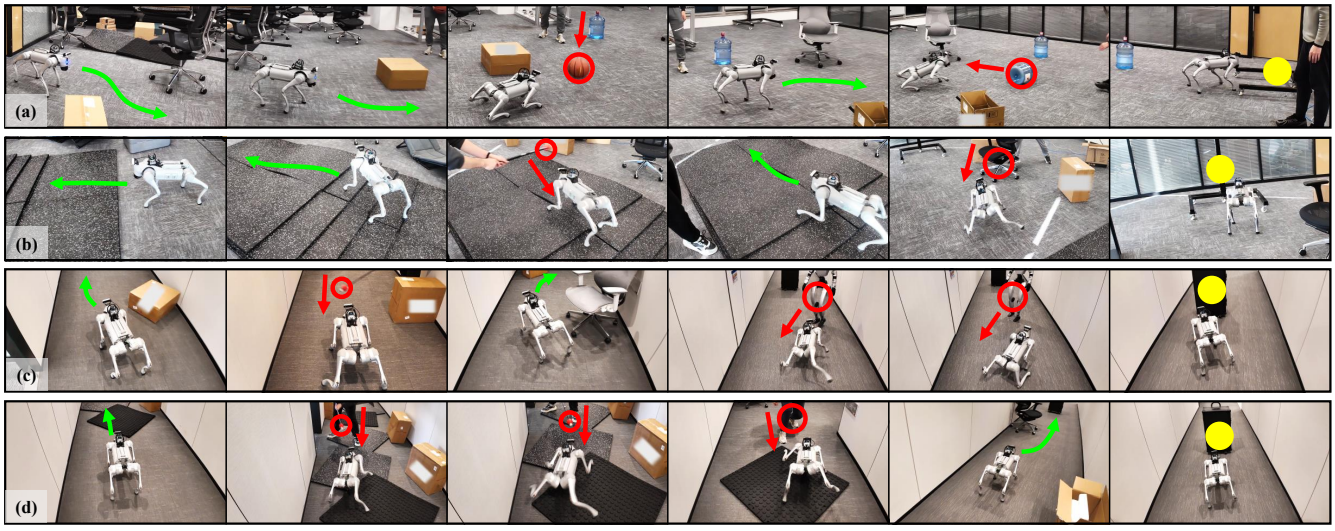

Fig. 12: **Additional indoor real-robot experiments.** Extended indoor trials across open rooms and corridors with different terrain conditions. (a) Open indoor room on flat ground. (b) Open indoor room with step terrain. (c) Indoor corridor on flat ground. (d) Indoor corridor with rough terrain.

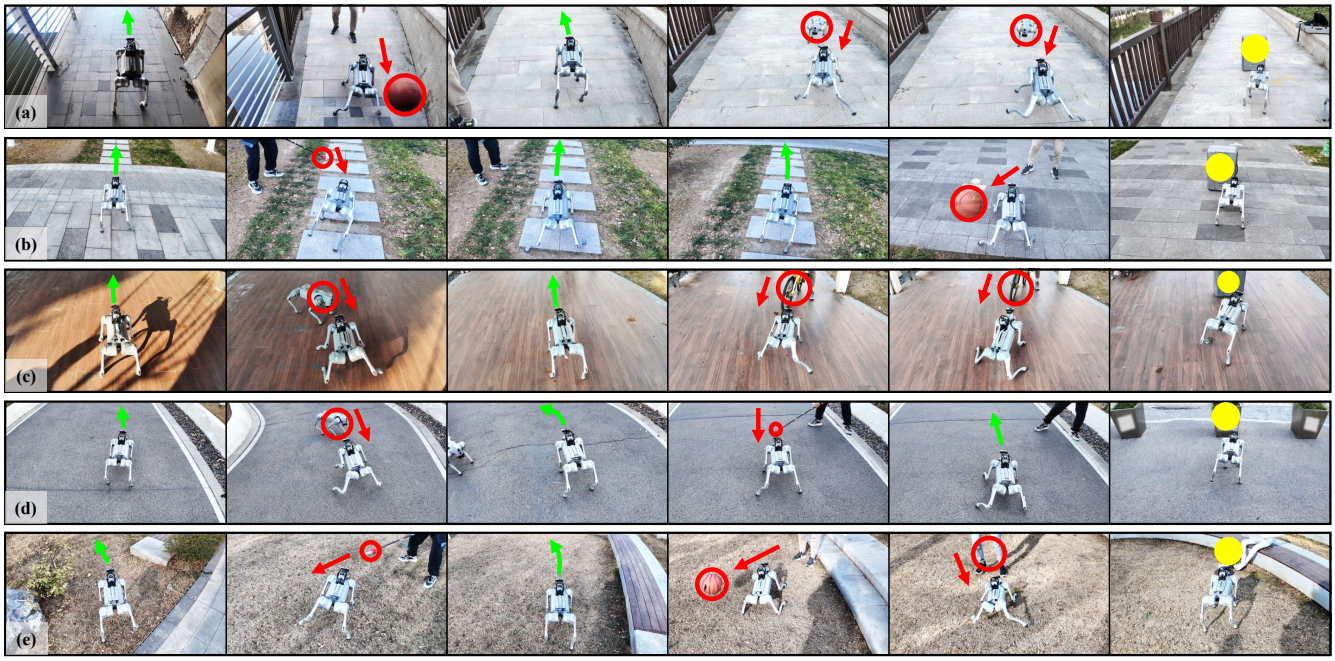

Fig. 13: **Additional outdoor real-robot experiments.** Extended outdoor trials across diverse terrains and ground profiles. (a) Riverside trail. (b) Stone-paved path. (c) Wooden floor. (d) Slope terrain. (e) Grass field.