SEA-Nav: Learning Quadruped Navigation in Cluttered Environments in One Hour

Anonymous Author(s)
Affiliation
Address
email

sea-nav.github.io

Figure 1: SEA-Nav is trained within one hour and deployed zero-shot in a previously unseen maze. The robot successfully escapes using the out-of-distribution built-in MPC controller and the onboard sparse LiDAR, demonstrating strong generalization to real-world scenarios.

Abstract: Efficiently training quadruped robots for navigation in cluttered environments remains a major challenge. Prior methods either guarantee safety and agility only in simple layouts or suffer from poor scalability and excessive training time in complex scenarios. We propose SEA-Nav (**S**afe, **E**fficient, and **A**gile **Nav**igation), a reinforcement learning framework for quadruped navigation that achieves strong real-world performance with minimal training cost. Our method combines a reward structure that encourages safe exploration, a goal-dwell termination mechanism that reinforces successful task completion, and action constraints that softly regulate velocity outputs for safety. With only one hour of training on a single RTX 4090 GPU, SEA-Nav enables zero-shot deployment to previously unseen static and dynamic obstacle settings. Extensive simulation and real-world experiments show that SEA-Nav achieves superior navigation success and agile locomotion across diverse environments.

Keywords: Reinforcement learning, Quadruped Navigation, Collision Avoidance

1 Introduction

Reinforcement learning (RL) has enabled substantial progress in quadruped robot control, supporting robust locomotion across diverse terrains [1, 2, 3, 4] and facilitating complex motor behaviors [5, 6, 7, 8]. One contributing factor to this success is that locomotion environments are typically designed with dense, task-relevant terrain variations, resulting in frequent and meaningful agent–environment interactions that enhance sample efficiency during training.

In contrast, navigation tasks pose a different set of challenges. Agent–environment interactions are significantly sparser, particularly in obstacle-dense or occluded environments, limiting the effec-

Submitted to the 9th Conference on Robot Learning (CoRL 2025). Do not distribute.

tiveness of reward signals and increasing the cost of policy learning [9, 10, 11]. This challenge is further exacerbated in highly cluttered scenes, where dense negative feedback from frequent collisions, coupled with long-horizon sparse rewards, significantly impedes training efficiency.

To mitigate these limitations, learning-based approaches have evolved into two principal paradigms. The first focuses on path-centric planning, where a policy generates intermediate trajectories or waypoints, which are subsequently tracked by low-level controllers. These methods often leverage expert demonstrations [12, 13, 14], self-supervised data labeling [15, 16], or cost maps [10, 17] to approximate near-optimal paths. While offering modularity, such methods suffer from delayed feedback and limited adaptability in dynamic or cluttered scenarios due to the decoupling between planning and control. The second paradigm learns motion commands directly from perception, enabling end-to-end control [18, 19]. Although these methods demonstrate agile behaviors in constrained settings, they typically lack robustness in obstacle-dense environments and offer limited safety guarantees, particularly under distribution shift.

To this end, this work proposes SEA-Nav (**S**afe, **E**fficient, and **A**gile **Nav**igation), a RL-based framework that addresses these challenges through an exploration-driven training paradigm. The key contributions are summarized as follows:

Exploration–Safety Coupled Training. SEA-Nav introduces an exploration-enhancing reward formulation and environment interaction strategy that jointly promote safe exploration and improve sampling efficiency during training.

Reliable Navigation via Action Constraints. SEA-Nav incorporates velocity regularization and smoothness constraints to suppress unsafe behaviors while preserving exploratory capability, thereby enhancing real-world deployability.

High-Performance Navigation Policy. SEA-Nav achieves zero-shot deployment after only one hour of training on a single RTX 4090 GPU. Compared to previous state-of-the-art (SOTA) method, SEA-Nav demonstrates superior local navigation performance across diverse environments.

2 Related Work

Existing navigation approaches can be broadly categorized based on output granularity into two paradigms: *path-centric planning* and *direct motion generation*, which differ significantly in control architecture, computational complexity, and safety performance.

Path-Centric Planning. These methods first generate discrete waypoints or trajectories that are subsequently tracked by a low-level controller, and are commonly adopted for long-range navigation. Classical approaches include search-based algorithms such as A* [20], sampling-based methods like RRT [21], PRM [22, 23], as well as heuristic optimization methods [24]. While effective in structured environments, these methods often require solving non-convex optimization problems online, resulting in high computational overhead [25] and poor adaptability to dynamic obstacles [10]. Recent advances incorporate supervised learning or reinforcement learning to predict paths or intermediate waypoints [10, 16, 26, 27, 28]. However, they typically depend on extensive prior data collection and training cycles. Furthermore, the decoupling between planning and control introduces latency and limits agility—a critical requirement for legged platforms. Additionally, the lack of explicit modeling of system dynamics makes them vulnerable to collisions in densely cluttered environments or during sharp maneuvers [29].

Direct Motion Generation. Rather than producing waypoints, direct methods compute low-level actions such as joint angles or velocity commands. Model-based optimization approaches [30, 31, 32, 33] aim to generate dynamically feasible, collision-free motions in configuration space using analytical or learned models. Although theoretically grounded, these methods are highly sensitive to model inaccuracies [34] and computationally expensive. RL-based approaches, such as ABS [18], map perception directly to control commands, achieving highly agile behaviors but often at the cost of degraded safety under complex obstacle distributions. To improve formal safety guarantees,

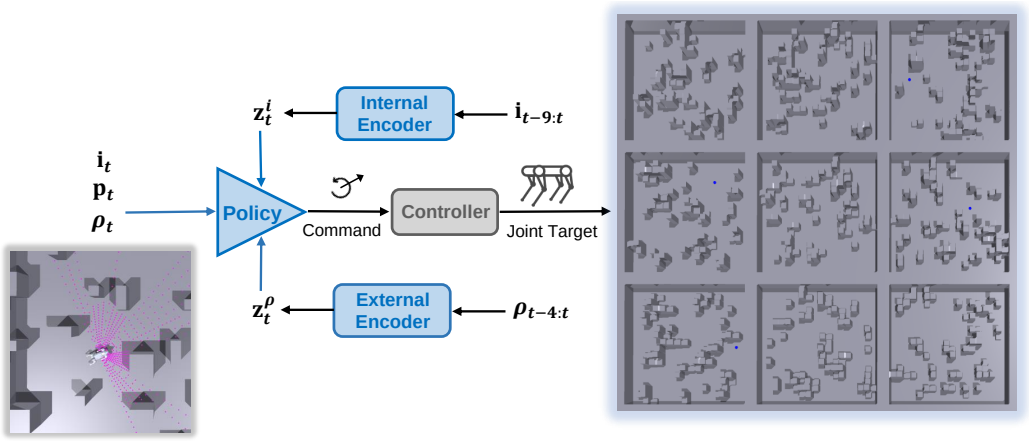

Figure 2: Overview of the SEA-Nav framework. All neural networks in the system are implemented using a standard MLP architecture with hidden layer sizes of [512, 256, 128].

certificate-based frameworks [35, 36, 37, 38] constrain policy outputs using reachability-based value functions or optimal control-theoretic safety filters. However, these methods typically rely on the offline availability of accurate certificate functions and dynamics models, limiting their robustness under model uncertainty.

While these frameworks provide structured solutions to motion planning and control, they often involve considerable architectural complexity and require careful system integration. In contrast, by improving sampling efficiency through efficient environment and training design, we demonstrate that a simple single-stage pipeline can achieve competitive or even superior navigation performance, without relying on additional network architecture tuning or sophisticated learning strategies.

3 Method

Navigating in obstacle-dense environments requires simultaneously balancing safety, agility, and training efficiency. To address these challenges, we propose SEA-Nav, a single-stage RL-based framework, as illustrated in Figure 2. This section first outlines the main challenges of the task, then presents the overall framework and key design components.

3.1 Key Challenges & Overview

Exploration–Exploitation Tradeoff. In cluttered environments, reward shaping must simultaneously promote exploration and guide goal completion. Dense obstacles increase collision risk, leading to overly conservative behaviors, while long-horizon goals exacerbate reward sparsity. SEA-Nav addresses these by incorporating active exploration incentives and employing a goal-dwell termination strategy, which increases the density of useful experiences and improves training efficiency.

Robot Dynamics Constraints. Sharp turns and high-speed maneuvers pose stability risks for legged robots. While traditional smoothing or action clipping imposes rigid constraints that limit exploration, SEA-Nav introduces soft, differentiable penalties on unsafe actions, promoting safe behaviors while preserving motion diversity.

Safety Margin for Real-World Deployment. Sensor latency, actuation delay, and perception noise in real-world settings introduce discrepancies between observations and true states. SEA-Nav enhances safety by enlarging the collision shape and injecting noise into observations and actions, fostering more cautious behaviors and improving generalization to safety-critical scenarios.

3.2 Observation & Action Space

Observation Space. As illustrated in Figure 2, the policy receives observations from external ray distances, internal proprioceptive states, their temporal encodings, and the relative goal position. At each timestep t, the robot observes 31 log-scaled ray distances $\boldsymbol{\rho}_t \in \mathbb{R}^{31}$, spanning angles $[-\frac{\pi}{2}, \frac{\pi}{2}]$ with an angular resolution of $\frac{\pi}{30}$, clipped between 0.1 m and 5.0 m. The internal observation $\mathbf{i}_t \in \mathbb{R}^{12}$ includes gravity direction in robot frame $\mathbf{g}_t \in \mathbb{R}^3$, previous action command $\mathbf{a}_{t-1} \in \mathbb{R}^3$, linear velocity $\mathbf{v}_t \in \mathbb{R}^3$, and angular velocity $\boldsymbol{\omega}_t \in \mathbb{R}^3$. The relative goal position $\mathbf{p}_t \in \mathbb{R}^2$ is provided in robot frame. To capture temporal dependencies, a 5-step history of ray observations $\boldsymbol{\rho}_{t-4:t}$ and a 10-step history of proprioceptive observations $\mathbf{i}_{t-9:t}$ are processed by separate external and internal encoders, producing latent embeddings $\mathbf{z}_t^{\rho}, \mathbf{z}_t^{i} \in \mathbb{R}^{16}$. Ray and goal observations are updated at 10 Hz, while proprioceptive inputs are updated at 50 Hz.

Action Space. The policy outputs a velocity command $\mathbf{a}_t = (c_x, c_y, c_{\text{yaw}}) \in \mathbb{R}^3$, which specifies the desired linear and angular velocities. These actions are executed by the low-level controller at 50 Hz to ensure responsive and agile motion in real-time navigation scenarios.

3.3 Exploration-Driven Reward Design

To enable efficient navigation in dense obstacle environments, we design a structured reward function composed of three categories: obstacle avoidance, navigation guidance and locomotion stability. These components together form the total reward: $r_t = r_t^{\text{obst}} + r_t^{\text{nav}} + r_t^{\text{loc}}$. In this section, we focus specifically on the exploration-driven subset of r_t^{obst} and r_t^{nav}, which play a critical role in overcoming conservative behaviors and sparse feedback signals in cluttered scenarios. The full reward formulation is provided in Appendix Table 4.

Directional Exploration Reward. To encourage task-directed motion and enhance exploration, we introduce a velocity reward that directs the robot's forward movement towards a target direction:

$$r_{\text{direct}} = \min(v_x, v_\sigma) \cdot \langle \hat{\mathbf{d}}, \mathbf{f} \rangle \cdot \mathbb{1}(d \geq d_\sigma) + w \cdot \mathbb{1}(d < d_\sigma) \tag{1}$$

where $\mathbf{f} \in \mathbb{R}^2$ is the robot's forward unit vector, and $\hat{\mathbf{d}} \in \mathbb{R}^2$ denotes the normalized target direction. The operator $\langle \cdot, \cdot \rangle$ represents the inner product, and $\mathbb{1}(\cdot)$ is the binary indicator function. The constant v_σ limits the contribution of forward velocity to prevent overly aggressive movements. The variables d and d_σ denote the current and threshold distances to the goal, respectively, and w is a fixed reward applied when the robot is within the goal neighborhood.

This reward appears in both the obstacle avoidance and navigation guidance terms, instantiated as $r_{\text{clear}} \subset r_t^{\text{obst}}$ and $r_{\text{guide}} \subset r_t^{\text{nav}}$, respectively. In r_{clear}, the target direction $\hat{\mathbf{d}}$ is set to the unit vector $\hat{\mathbf{u}}_{i^*}$ of the longest ray, where $i^* = \arg\max_i \boldsymbol{\rho}_t^i$. In r_{guide}, the direction $\hat{\mathbf{d}}$ is defined as $\hat{\mathbf{p}} = \frac{\mathbf{p}_{\text{goal}} - \mathbf{p}_{\text{base}}}{\|\mathbf{p}_{\text{goal}} - \mathbf{p}_{\text{base}}\|}$, which points from the robot's base to the goal.

Stuck Penalty. To discourage behavioral stagnation, we impose a penalty when the robot fails to make meaningful spatial progress. A state is considered stagnated if the maximum displacement over the past 10 steps falls below a threshold d_{stuck}. The corresponding reward term is defined as:

$$r_{\text{stuck}} = -\frac{1}{1 + \bar{v}_{\text{yaw}}^2} \cdot \mathbb{1}(\text{stuck}) \cdot \mathbb{1}(d \geq d_\sigma) - \|\mathbf{v}\| \cdot \mathbb{1}(d < d_\sigma) \tag{2}$$

where $\bar{v}_{\text{yaw}}$ denotes the average yaw velocity over the past 10 steps, and $\|\mathbf{v}\|$ denotes the magnitude of the velocity vector. This reward encourages the agent to replan and escape from localized deadlocks when far from the goal, while encouraging smooth deceleration at the goal.

3.4 Action Constraints

To improve the safety of velocity commands while retaining sufficient exploration capacity, we introduce two soft action losses that penalize excessive or risky actions.

Velocity Regularization Loss. We define a unified velocity safety loss consisting of two components: **(a)** a penalty for violating safe velocity bounds, and **(b)** a penalty for unsafe combinations of

velocity dimensions. The total loss is defined as:

$$\mathcal{L}_{\text{reg}} = \underbrace{\sum_{j=1}^{3} \left(a_t^j - \text{clip}(a_t^j, a_j^{\min}, a_j^{\max}) \right)^2}_{\text{(a) range penalty}} + \underbrace{\sum_{(i,j)\in\mathcal{C}} \text{ReLU}(|a_t^i| - \tau_i) \cdot \text{ReLU}(|a_t^j| - \tau_j) \cdot |a_t^i \cdot a_t^j|}_{\text{(b) combination penalty}} \tag{3}$$

where the clipping bounds are $a^{\min} = [-0.5, -0.8, -1.2]$ and $a^{\max} = [1.5, 0.8, 1.2]$, corresponding to the safe ranges for $[c_x, c_y, c_{\text{yaw}}]$. The safety thresholds are $\tau_x = 1.25$, $\tau_y = 0.6$, and $\tau_{\text{yaw}} = 1.0$. The set $\mathcal{C} = \{(x, y), (x, \text{yaw}), (y, \text{yaw})\}$ specifies velocity component pairs considered risky when both magnitudes are large. This loss softly constrains action outputs without resorting to hard clipping, thereby allowing meaningful exploration while maintaining compatibility with higher-speed locomotion, as evidenced by the results in Table 2.

Lipschitz Continuity Loss. To promote smooth transitions in both action and value predictions, we adopt a Lipschitz continuity constraint [39]:

$$\mathcal{L}_{\text{smth}} = \lambda_\pi \cdot D\left(\pi_\theta(s_t), \pi_\theta(\bar{s}_t)\right) + \lambda_V \cdot D\left(V_\phi(s_t), V_\phi(\bar{s}_t)\right) \tag{4}$$

where $\bar{s}_t = s_t + u \cdot (s_{t+1} - s_t)$ is an interpolated state with $u \sim \mathcal{U}(-1, 1)$, and $D(\cdot, \cdot)$ denotes the mean squared error (MSE). Here, λ_π and λ_V are weighting factors for the policy and value network smoothness losses, respectively. This constraint improves deployment safety by suppressing abrupt changes in action and value outputs, thereby reducing the risk of falls and motor overheating.

3.5 Training in Simulation

SEA-Nav is trained using PPO [40] within the IsaacGym simulation framework [41]. The low-level controller is adopted from SLR [4]. To improve training efficiency and promote effective exploration, several targeted strategies are integrated into the learning process.

Efficient Environment Design. We construct a cluttered navigation environment to maximize obstacle interaction and goal-directed exploration. Specifically, a 10×10 multi-room layout is designed, with each room containing obstacles of varying densities and spatial distributions (a subset is shown in Figure 2). These compact environments are critical for efficient training by ensuring frequent agent-environment interactions and maintaining strong task relevance throughout episodes. Additionally, the mixture of wide and narrow passages facilitates ray-based perception learning, obviating the extra need for complex encoder optimization.

Goal-Dwell Termination. In navigation tasks, sparse goal rewards often arise because agents rarely reach the target region during early exploration, resulting in episodes terminating prematurely without sufficient goal-related experience. To address this, we introduce a goal-dwell-based termination strategy: once the agent enters a small neighborhood around the goal, it is allowed to remain in proximity for a short period rather than triggering immediate reset. This increases the likelihood of accumulating valuable reaching experiences and effectively mitigates the sparsity of the learning signal. By converting rare goal encounters into consistent training signals, the strategy improves sample efficiency and accelerates the acquisition of reliable goal-reaching behaviors.

Perception and Safety Margin Enhancement. To improve obstacle awareness and promote safer motion, we expand the agent's collision shape (Figure 3). This enlarges the perceived danger zone and encourages earlier avoidance behaviors. Additionally, we inject noise into both the controller and the navigation policy following domain randomization practices (Table 5), forcing the agent to execute slightly perturbed actions. These modifications increase collision risk during training, thereby encouraging safer planning behaviors.

Figure 3: **Top**: the original Unitree Go2 URDF model. **Bottom**: the expanded collision shapes.

4 Simulation Experiments

4.1 Navigation Task Evaluation

To comprehensively evaluate SEA-Nav, we construct three navigation environments—*Easy*, *Medium*, and *Hard*—with increasing obstacle density (Figure 4). Each mode is tested across 100 randomized trials with different start-goal pairs and initial headings.

We conduct comparisons against the prior SOTA method ABS [18], along with ablation studies. Evaluation metrics include **Success Rate (SR)**, **Collision Rate (CR)** and **Timeout Rate (TR)**. A trial is considered successful if the robot reaches within 0.5 m of the goal. Trials exceeding 20 seconds without reaching the goal are marked as timeouts, reflecting cases of being stuck or trapped in local minima. Results are summarized in Table 1.

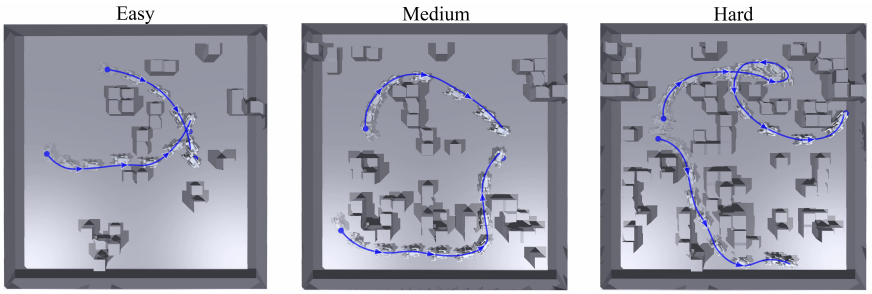

Figure 4: SEA-Nav trajectories in three navigation scenarios of increasing difficulty. Each subfigure shows two distinct start-goal trials. SEA-Nav successfully traverses narrow passages and performs timely maneuvering adjustments to avoid entrapment in cluttered environments.

Table 1: Different difficulty levels are evaluated, with the mean and standard deviation reported over 3 runs. For termination strategy ablation, we compare immediate reset (*w/o Goal-Dwell*) and fixed timeout termination (*w/ Fixed Timeout*). For observation input ablation, we assess the effect of removing proprioceptive input (*w/o Proprio*) and historical encoding (*w/o History*).

Method	Easy			Medium			Hard		
	SR (%) ↑	CR (%) ↓	TR (%) ↓	SR (%) ↑	CR (%) ↓	TR (%) ↓	SR (%) ↑	CR (%) ↓	TR (%) ↓
(a) Exploration Rewards									
SEA-Nav w/o r_{guide}	99.67 (±0.47)	0.33 (±0.47)	0.33 (±0.47)	79.67 (±2.62)	2.67 (±1.70)	20.33 (±2.62)	74.67 (±4.92)	3.33 (±1.25)	25.33 (±4.92)
SEA-Nav w/o r_{clear}	98.33 (±0.47)	1.67 (±0.47)	1.67 (±0.47)	82.67 (±3.86)	1.67 (±0.94)	17.33 (±3.86)	45.67 (±2.05)	6.33 (±1.25)	54.33 (±2.05)
SEA-Nav w/o r_{stuck}	99.67 (±0.47)	0.00 (±0.00)	0.33 (±0.47)	94.67 (±2.05)	**1.33** (±0.94)	5.33 (±2.05)	86.33 (±3.09)	5.67 (±1.70)	13.67 (±3.09)
(b) Termination Strategy									
SEA-Nav w/o Goal-Dwell	47.67 (±3.51)	2.33 (±4.15)	52.33 (±3.51)	42.00 (±3.23)	3.67 (±2.05)	58.00 (±3.23)	41.33 (±5.91)	6.67 (±6.68)	58.67 (±5.91)
SEA-Nav w/ Fixed Timeout	34.33 (±4.92)	0.33 (±0.47)	65.67 (±4.92)	23.00 (±2.94)	1.33 (±1.25)	77.00 (±2.94)	14.67 (±4.99)	**1.67** (±1.70)	85.33 (±4.99)
(c) Action Constraints									
SEA-Nav w/o $\mathcal{L}_{\text{reg}}$	**100.00** (0.00)	1.00 (±0.82)	**0.00** (±0.00)	88.67 (±1.25)	7.33 (±2.05)	11.33 (±1.25)	73.67 (±3.40)	24.67 (±5.44)	26.33 (±3.40)
SEA-Nav w/o $\mathcal{L}_{\text{smth}}$	**100.00** (0.00)	1.00 (±0.82)	**0.00** (±0.00)	93.67 (±2.05)	4.00 (±0.82)	6.33 (±2.05)	84.67 (±2.49)	3.33 (±1.25)	15.33 (±2.49)
(d) Observation Input									
SEA-Nav w/o Proprio	94.00 (±0.82)	3.00 (±0.82)	6.00 (±0.82)	67.33 (±2.49)	31.33 (±2.87)	32.67 (±2.49)	52.33 (±2.87)	45.00 (±3.74)	47.67 (±2.87)
SEA-Nav w/o History	**100.00** (0.00)	1.33 (±0.47)	**0.00** (0.00)	83.33 (±3.77)	4.67 (±0.47)	16.67 (±3.77)	77.67 (±4.19)	14.33 (±1.89)	22.33 (±4.19)
(e) Collision Shape Expansion									
ABS [18] w/ Expansion	98.67 (±0.47)	3.67 (±0.47)	2.33 (±0.47)	71.33 (±3.09)	10.33 (±1.89)	28.67 (±2.16)	66.33 (±4.62)	31.33 (±3.68)	33.67 (±4.62)
SEA-Nav w/o Expansion	98.67 (±1.25)	24.67 (±2.49)	1.33 (±1.25)	95.67 (±0.94)	68.67 (±4.11)	4.33 (±1.63)	90.67 (±1.70)	69.00 (±2.45)	9.33 (±1.70)
(f) ABS [18]	95.67 (±1.25)	18.67 (±1.70)	4.33 (±1.25)	69.67 (±2.87)	52.00 (±2.16)	30.33 (±2.87)	61.00 (±3.68)	57.00 (±4.08)	39.00 (±3.68)
(g) SEA-Nav	**100.00** (±0.00)	**0.00** (±0.00)	**0.00** (±0.00)	**97.00** (±1.63)	1.33 (±1.25)	**3.00** (±1.63)	**95.67** (±2.87)	2.33 (±1.25)	**4.33** (±2.87)

4.2 Results and Analysis

SEA-Nav achieves an excellent balance between safety and task completion across all difficulty levels. This improvement results from several key design choices:

Safe Exploration Induced by Reward Design. The designed rewards promote cautious exploration, enabling the robot to safely traverse cluttered environments while maintaining sufficient progress toward goals. By encouraging timely adjustments in narrow or risky regions, the policy reduces the likelihood of collision without becoming overly conservative.

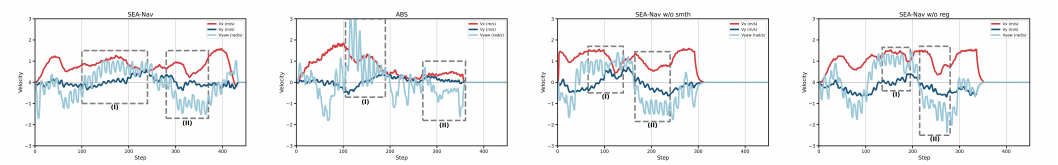

Figure 6: The velocity-step curves demonstrate that the action constraints reduce unsafe velocity magnitudes, suppress risky combinations, and mitigate abrupt changes, while implicitly encouraging anticipatory and safer behaviors.

Goal-Dwell Termination Enhances Learning Efficiency. The goal-dwell termination mechanism allows the agent to reinforce successful reaching behaviors while avoiding unnecessary idle time, thereby improving training efficiency and accelerating convergence. Moreover, by making the final objective more salient during training, it encourages the agent to pursue goal completion more deliberately, leading to safer navigation and fewer collision penalties.

Action Constraints Enhance Execution Safety. Soft action constraints discourage risky inertia-driven motions while maintaining sufficient exploratory behavior. This reduces the likelihood of high-speed collisions and improves real-world deployability.

Proprioceptive Encoding Enhances State Awareness. Proprioceptive feedback improves the policy's understanding of locomotion execution, while history encoding aids in detecting stagnation through temporal motion patterns.

Collision Shape Enlargement Enhances Hazard Perception. Expanding the collision shape, particularly around the head, increases the effective risk sensing area. This adjustment encourages earlier deceleration and safer avoidance behaviors when approaching obstacles.

4.3 Case Study: Curved Corridor Navigation

To analyze the role of velocity constraints, we conduct a case study on a track with continuous tight turns, comparing SEA-Nav, ABS [18], and two ablations: SEA-Nav without velocity regularization (*w/o reg*) and without smoothness constraint (*w/o smth*). In *w/o reg*, the soft velocity penalty is replaced by hard clipping; in *w/o smth*, smoothness loss is removed. As shown in Figure 5, SEA-Nav initiates earlier turning and maintains safer clearance, resulting in a smoother, collision-free trajectory, whereas ABS [18] fails due to unsafe sharp turns at corners.

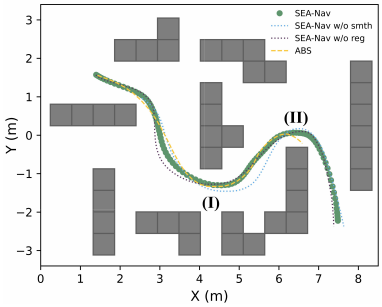

Figure 5: Trajectories between SEA-Nav and other baselines.

Velocity profiles (Figure 6) reveal that ABS [18] produces abrupt and hazardous spikes, while SEA-Nav suppresses excessive lateral velocities and discourages greedy forward acceleration, preventing sudden braking and unsafe sharp turns near obstacles. Without smoothness constraints, *w/o smth* exhibits noticeable velocity jumps, posing potential risks to joint actuators. Although action constraints slightly delay goal reaching, they significantly improve safety margins and dynamic stability, which are crucial for real-world deployment.

5 Real-World Experiments

5.1 Hardware Setup

We deploy SEA-Nav on a Unitree Go2 quadruped robot and evaluate its performance in real-world environments featuring diverse obstacle configurations, including static and dynamic obstacles (see Figure 7). Two deployment setups are considered: a **Built-in Setup** using the onboard sparse LiDAR and the robot's built-in model-based MPC controller (out-of-distribution relative to training), and an

Enhanced Setup employing a high-resolution RPLIDAR A2 and the trained SLR policy [4] for low-level control. The point cloud sparsity differences between the two LiDAR sensors are shown in Appendix Figure 9. To assess generalization across different motion regimes, SEA-Nav is tested under two maximum speed limits using the enhanced setup. Additionally, Unitree's built-in SLAM system is included as a global navigation baseline. This evaluation framework enables testing SEA-Nav's adaptability to sensor sparsity, controller mismatch, and dynamic task variations.

5.2 Results and Analysis

Table 2 summarizes the quantitative results. SEA-Nav demonstrates superior navigation success rates and safety across a variety of real-world scenarios. Although ABS [18] achieves higher average speeds in simple obstacle environments, it lacks robustness in out-of-distribution settings. In particular, it frequently stops prematurely or collides at unsafe velocities when navigating narrow or curved passages. Meanwhile, the SLAM-based baseline performs well in static, pre-mapped scenes but depends heavily on time-consuming and highly accurate map construction, which limits its applicability in dynamic or fast-paced environments.

Figure 7: Real-world experimental environments. 10 trials are conducted in each environment.

Table 2: Real-world deployment results. AS: average speed (m/s); SEA-Nav-h: high-speed setup (1.75 m/s limit); SEA-Nav-l: low-speed setup (0.75 m/s limit); SEA-Nav-b: built-in MPC and sparse LiDAR.

Method	Cluttered Room			Dynamic Obstacle			Obstacle Course			S Blend Track		
	SR ↑	CR ↓	AS ↑	SR ↑	CR ↓	AS ↑	SR ↑	CR ↓	AS ↑	SR ↑	CR ↓	AS ↑
SEA-Nav-h	90	10	1.6	90	20	1.5	**100**	20	**1.2**	**100**	10	**1.4**
SEA-Nav-l	**100**	**0**	0.6	90	10	0.6	90	10	0.6	**100**	**0**	0.5
SEA-Nav-b	**100**	10	0.9	**100**	20	0.8	90	10	0.7	**100**	10	0.5
ABS [18]	70	50	**2.1**	90	10	**1.7**	0	100	0.6	80	20	0.8
SLAM	**100**	**0**	0.5	**100**	**0**	0.4	**100**	**0**	0.4	80	**0**	0.3

6 Conclusion

We present SEA-Nav, a reinforcement learning framework that enables safe, efficient, and agile quadruped navigation in cluttered environments. By integrating reward shaping for safe exploration, a goal-dwell termination strategy to enhance sample efficiency, and action constraints to promote stable locomotion, SEA-Nav can be trained within one hour on a single GPU and deployed zero-shot to unseen scenarios. Relying on efficient environment and training design, SEA-Nav avoids the need for additional model optimization and achieves higher success rates and lower collision risks compared to prior SOTA method, supporting agile deployment in diverse real-world settings.

A Limitations

While SEA-Nav demonstrates strong local navigation capabilities across diverse environments, several limitations remain. First, due to the absence of explicit path planning, the method can only perform limited replanning in relatively simple local minima scenarios. In complex mazes or narrow cul-de-sacs, the agent often fails to escape, as illustrated in Figure 8. Second, SEA-Nav is currently designed for planar navigation and does not handle three-dimensional terrains such as stairs, slopes, or uneven surfaces.

Addressing these challenges represents a promising direction for future work. One potential solution is to incorporate lightweight global guidance, such as subgoal proposals from high-level planners, to complement SEA-Nav's local agility. Furthermore, extending the observation space and refining the reward structure to account for vertical geometry may enable adaptation to multi-level environments, thereby enhancing the method's applicability in real-world robotic deployments.

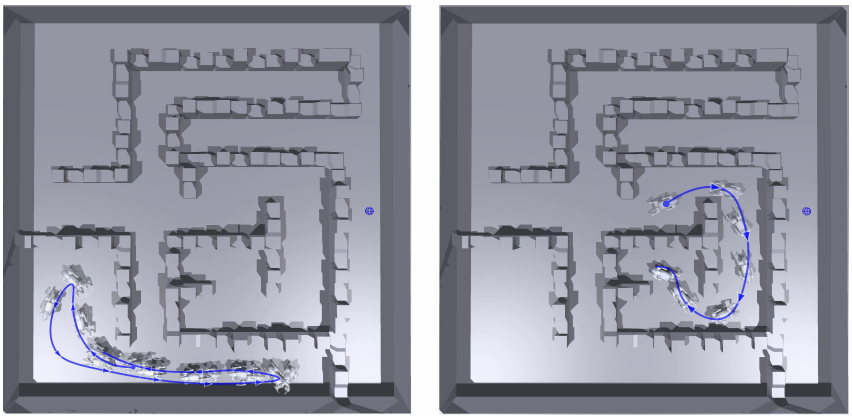

Figure 8: Failure cases in maze environments. **Left:** The robot trajectory forms a loop, becoming trapped in a local cycle due to suboptimal exploration. **Right:** The robot enters a dead-end corridor, highlighting the limitation of lacking explicit path planning.

B Appendix

B.1 LIDAR Point Cloud Comparison

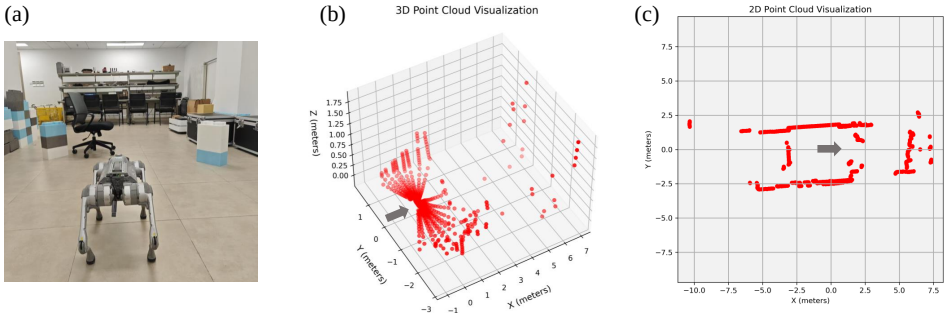

Figure 9: Comparison of LIDAR point clouds. The robot's position and orientation are indicated by the gray arrow. **(a):** The robot's actual environment. **(b):** Sparse point cloud from the Unitree Go2 robot's onboard LIDAR, consisting of approximately 1,000 3D points per frame. **(c):** Denser point cloud from the RPLIDAR A2, consisting of approximately 2,000 2D points per frame.

B.2 Performance Across Training Durations

Table 3: Simulation results complementing Section 4.1, evaluating SEA-Nav performance across different training durations (15–60 minutes). The results show that SEA-Nav achieves strong performance even with limited training time, and progressively improves as training proceeds.

Training duration	Easy			Medium			Hard		
	SR (%) ↑	CR (%) ↓	TR (%) ↓	SR (%) ↑	CR (%) ↓	TR (%) ↓	SR (%) ↑	CR (%) ↓	TR (%) ↓
SEA-Nav									
15 mins	98.67 (±0.94)	1.33 (±1.25)	1.33 (±0.47)	96.33 (±1.70)	1.67 (±0.94)	3.67 (±1.70)	87.67 (±2.05)	11.00 (±2.94)	12.33 (±2.05)
30 mins	99.67 (±0.47)	1.00 (±1.41)	0.33 (±0.47)	**97.67** (±1.25)	2.67 (±1.70)	2.33 (±1.25)	92.33 (±1.89)	6.33 (±1.25)	7.67 (±1.89)
45 mins	**100.00** (±0.00)	**0.00** (±0.00)	**0.00** (±0.00)	97.33 (±1.25)	1.67 (±1.70)	**2.67** (±1.25)	93.67 (±2.36)	3.67 (±0.94)	6.33 (±1.25)
60 mins	**100.00** (±0.00)	**0.00** (±0.00)	**0.00** (±0.00)	97.00 (±1.63)	**1.33** (±1.25)	3.00 (±1.63)	**95.67** (±2.87)	**2.33** (±1.25)	**4.33** (±2.87)

B.3 Reward Function

Table 4: Reward function components used in training. Obstacle avoidance and navigation guidance rewards incorporate exploration-enhancing terms, while locomotion rewards regularize motion stability.

Term	Expression	Weight	Description
(a) r^{obst}		–	Obstacle Avoidance Reward
r_{clear}	$\min(v_x, v_\sigma) \cdot \langle \hat{\mathbf{u}}_{i^*}, \mathbf{f} \rangle \cdot \mathbb{1}(d \geq d_\sigma) + w \cdot \mathbb{1}(d < d_\sigma)$	6	Directional exploration along clear paths
$r_{\text{collision}}$	$-\mathbb{1}(\text{collision}) \cdot (1 + v_x)$	300	Penalize collisions, scaled by speed
(b) r^{nav}		–	Navigation Guidance Reward
r_{guide}	$\min(v_x, v_\sigma) \cdot \langle \hat{\mathbf{p}}, \mathbf{f} \rangle \cdot \mathbb{1}(d \geq d_\sigma) + w \cdot \mathbb{1}(d < d_\sigma)$	2	Guide forward progress toward goal
r_{stuck}	$-\frac{1}{1+\bar{v}_{\text{yaw}}^2} \cdot \mathbb{1}(\text{stuck}) \cdot \mathbb{1}(d \geq d_\sigma) - \|\mathbf{v}\| \cdot \mathbb{1}(d < d_\sigma)$	1	Escape from stagnation and stop at goal
r_{reach}	$\frac{1}{1+d^2} \cdot \mathbb{1}(d < d_\sigma)$	50	Reward reaching goal proximity
(c) r^{loc}		–	Locomotion Stability Reward
$r_{\text{lin\_vel\_z}}$	$-v_z^2$	10	Penalize vertical motion
$r_{\text{ang\_vel\_xy}}$	$-\|\omega_{\mathbf{xy}}\|^2$	0.5	Penalize planar angular velocity
$r_{\text{joint\_rate}}$	$-\|\mathbf{q}_t - \mathbf{q}_{t-1}\|^2$	0.01	Penalize abrupt joint changes
r_{track}	$-\|\mathbf{c} - \mathbf{v}\|^2$	0.1	Penalize command-velocity deviation

B.4 Domain Randomization

Table 5: Domain-randomization parameters used during training.

Term	Value
Gravity noise	$\mathcal{U}(-0.05, 0.05)$
Linear velocity noise	$\mathcal{U}(-0.1, 0.1)\,\text{m/s}$
Angular velocity noise	$\mathcal{U}(-0.2, 0.2)\,\text{rad/s}$
Ray distance noise	$\mathcal{U}(-0.2, 0.2)\,\text{m}$
Linear velocity command noise	$\mathcal{U}(-0.2, 0.2)\,\text{m/s}$
Angular velocity command noise	$\mathcal{U}(-0.3, 0.3)\,\text{rad/s}$
Friction coefficient factor	$\mathcal{U}(-0.2, 1.25)$
Mass perturbation	$\mathcal{U}(-1.5, 1.5)\,\text{kg}$

B.5 Hyper Parameters

Table 6: Hyper Parameters for Training

Term	Value
λ_π	1.0
λ_V	0.1
Number of Environments	1280
Batch Size	1280×48
Episode Duration	60 s
Goal-Dwell Time T_{stay}	1 s
Goal Proximity Threshold d_σ	0.5 m
Stagnation Threshold d_{stuck}	0.2 m
Minimum Forward Velocity v_σ	0.5 m/s

References

[1] N. Rudin, D. Hoeller, P. Reist, and M. Hutter. Learning to walk in minutes using massively parallel deep reinforcement learning. In *Conference on Robot Learning*, pages 91–100. PMLR, 2022.

[2] A. Kumar, Z. Fu, D. Pathak, and J. Malik. Rma: Rapid motor adaptation for legged robots. *arXiv preprint arXiv:2107.04034*, 2021.

[3] J. Long, W. Yu, Q. Li, Z. Wang, D. Lin, and J. Pang. Learning h-infinity locomotion control. *arXiv preprint arXiv:2404.14405*, 2024.

[4] S. Chen, Z. Wan, S. Yan, C. Zhang, W. Zhang, Q. Li, D. Zhang, and F. U. D. Farrukh. Slr: Learning quadruped locomotion without privileged information. In P. Agrawal, O. Kroemer, and W. Burgard, editors, *Proceedings of The 8th Conference on Robot Learning*, volume 270 of *Proceedings of Machine Learning Research*, pages 3212–3224. PMLR, 06–09 Nov 2025. URL https://proceedings.mlr.press/v270/chen25e.html.

[5] D. Hoeller, N. Rudin, D. Sako, and M. Hutter. Anymal parkour: Learning agile navigation for quadrupedal robots. *Science Robotics*, 9(88):eadi7566, 2024.

[6] X. Cheng, K. Shi, A. Agarwal, and D. Pathak. Extreme parkour with legged robots. *arXiv preprint arXiv:2309.14341*, 2023.

[7] Z. Zhuang, Z. Fu, J. Wang, C. Atkeson, S. Schwertfeger, C. Finn, and H. Zhao. Robot parkour learning. In *Conference on Robot Learning (CoRL)*, 2023.

[8] R. Huang, S. Zhu, Y. Du, and H. Zhao. Moe-loco: Mixture of experts for multitask locomotion, 2025. URL https://arxiv.org/abs/2503.08564.

[9] K. E. R. H. J. S. A. H. R. G. A. K. L. W. Kuo-Hao Zeng, Zichen Zhang. Poliformer: Scaling on-policy rl with transformers results in masterful navigators. *arXiv*, 2024.

[10] F. Yang, C. Wang, C. Cadena, and M. Hutter. iPlanner: Imperative Path Planning. In *Proceedings of Robotics: Science and Systems*, Daegu, Republic of Korea, July 2023. doi:10.15607/RSS.2023.XIX.064.

[11] J. Ye, D. Batra, A. Das, and E. Wijmans. Auxiliary tasks and exploration enable objectnav, 2021.

[12] D. Shah, A. Sridhar, A. Bhorkar, N. Hirose, and S. Levine. Gnm: A general navigation model to drive any robot. *arXiv preprint arXiv:2210.03370*, 2022.

[13] A. Sadat, S. Casas, M. Ren, X. Wu, P. Dhawan, and R. Urtasun. Perceive, predict, and plan: Safe motion planning through interpretable semantic representations. In *European Conference on Computer Vision*, pages 414–430. Springer, 2020.

[14] M. Pfeiffer, M. Schaeuble, J. Nieto, R. Siegwart, and C. Cadena. From perception to decision: A data-driven approach to end-to-end motion planning for autonomous ground robots. In *IEEE International Conference on Robotics and Automation (ICRA)*, page 1527–1533. IEEE, 2017.

[15] G. Kahn, P. Abbeel, and S. Levine. Badgr: An autonomous self-supervised learning-based navigation system. *IEEE Robotics and Automation Letters*, 6(2):1312–1319, 2021.

[16] Y. Kim, C. Kim, and J. Hwangbo. Learning forward dynamics model and informed trajectory sampler for safe quadruped navigation. In *Proceedings of Robotics: Science and Systems*, New York City, NY, USA, June 2022. doi:10.15607/RSS.2022.XVIII.069.

[17] P. Roth, J. Nubert, F. Yang, M. Mittal, and M. Hutter. Viplanner: Visual semantic imperative learning for local navigation. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5243–5249. IEEE, 2024.

[18] T. He, C. Zhang, W. Xiao, G. He, C. Liu, and G. Shi. Agile but safe: Learning collision-free high-speed legged locomotion. In *Robotics: Science and Systems (RSS)*, 2024.

[19] Y. Zhong, C. Zhang, T. He, and G. Shi. Bridging adaptivity and safety: Learning agile collision-free locomotion across varied physics. *arXiv preprint arXiv:2501.04276*, 2025.

[20] P. E. Hart, N. J. Nilsson, and B. Raphael. A formal basis for the heuristic determination of minimum cost paths. *IEEE Transactions on Systems Science and Cybernetics*, 4(2):100–107, 1968. doi:10.1109/TSSC.1968.300136.

[21] S. M. LaValle. Rapidly-exploring random trees : a new tool for path planning. *The annual research report*, 1998. URL https://api.semanticscholar.org/CorpusID:14744621.

[22] L. Kavraki, P. Svestka, J.-C. Latombe, and M. Overmars. Probabilistic roadmaps for path planning in high-dimensional configuration spaces. *IEEE Transactions on Robotics and Automation*, 12(4):566–580, 1996. doi:10.1109/70.508439.

[23] S. Karaman and E. Frazzoli. Sampling-based algorithms for optimal motion planning. *The international journal of robotics research*, 30(7):846–894, 2011.

[24] O. Khatib. Real-time obstacle avoidance for manipulators and mobile robots. *The international journal of robotics research*, 5(1):90–98, 1986.

[25] A. Loquercio, E. Kaufmann, R. Ranftl, M. Müller, V. Koltun, and D. Scaramuzza. Learning high-speed flight in the wild. *Science Robotics*, 6(59):eabg5810, 2021.

[26] M. Bojarski, D. Del Testa, D. Dworakowski, B. Firner, B. Flepp, P. Goyal, L. D. Jackel, M. Monfort, U. Muller, J. Zhang, et al. End to end learning for self-driving cars. *arXiv preprint arXiv:1604.07316*, 2016.

[27] M. Pfeiffer, S. Shukla, M. Turchetta, C. Cadena, A. Krause, R. Siegwart, and J. Nieto. Reinforced imitation: Sample efficient deep reinforcement learning for mapless navigation by leveraging prior demonstrations. *IEEE Robotics and Automation Letters*, 3(4):4423–4430, 2018.

[28] X. Xiao, T. Zhang, K. Choromanski, E. Lee, A. Francis, J. Varley, S. Tu, S. Singh, P. Xu, F. Xia, et al. Learning model predictive controllers with real-time attention for real-world navigation. *arXiv preprint arXiv:2209.10780*, 2022.

[29] A. Thirugnanam, J. Zeng, and K. Sreenath. Safety-critical control and planning for obstacle avoidance between polytopes with control barrier functions. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 286–292, 2022. doi:10.1109/ICRA46639.2022.9812334.

[30] D. Kim, D. Carballo, J. Di Carlo, B. Katz, G. Bledt, B. Lim, and S. Kim. Vision aided dynamic exploration of unstructured terrain with a small-scale quadruped robot. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2464–2470. IEEE, 2020.

[31] T. Dudzik, M. Chignoli, G. Bledt, B. Lim, A. Miller, D. Kim, and S. Kim. Robust autonomous navigation of a small-scale quadruped robot in real-world environments. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3664–3671. IEEE, 2020.

[32] M. Gaertner, M. Bjelonic, F. Farshidian, and M. Hutter. Collision-free mpc for legged robots in static and dynamic scenes. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 8266–8272. IEEE, 2021.

[33] J.-R. Chiu, J.-P. Sleiman, M. Mittal, F. Farshidian, and M. Hutter. A collision-free mpc for whole-body dynamic locomotion and manipulation. In *2022 international conference on robotics and automation (ICRA)*, pages 4686–4693. IEEE, 2022.

[34] D. Hoeller, L. Wellhausen, F. Farshidian, and M. Hutter. Learning a state representation and navigation in cluttered and dynamic environments. *IEEE Robotics and Automation Letters*, 6 (3):5081–5088, 2021.

[35] A. Lin, S. Peng, and S. Bansal. One filter to deploy them all: Robust safety for quadrupedal navigation in unknown environments. *arXiv preprint arXiv:2412.09989*, 2024.

[36] W. Xiao, T. He, J. Dolan, and G. Shi. Safe deep policy adaptation. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 17286–17292. IEEE, 2024.

[37] R. Cheng, G. Orosz, R. M. Murray, and J. W. Burdick. End-to-end safe reinforcement learning through barrier functions for safety-critical continuous control tasks. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pages 3387–3395, 2019.

[38] Z. Xu, X. Han, H. Shen, H. Jin, and K. Shimada. Navrl: Learning safe flight in dynamic environments. *arXiv preprint arXiv:2409.15634*, 2024.

[39] T. Kobayashi. L2c2: Locally lipschitz continuous constraint towards stable and smooth reinforcement learning. In *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4032–4039. IEEE, 2022.

[40] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

[41] V. Makoviychuk, L. Wawrzyniak, Y. Guo, M. Lu, K. Storey, M. Macklin, D. Hoeller, N. Rudin, A. Allshire, A. Handa, et al. Isaac gym: High performance gpu-based physics simulation for robot learning. *arXiv preprint arXiv:2108.10470*, 2021.