Quadruped robot traversing 3D complex environments with limited perception

Yi Cheng*[1], Hang Liu*[2], Guoping Pan[1], Linqi Ye†[3], Houde Liu†[1]
Quad-Traverse-Go2.github.io

Fig. 1: We tested the passability of the quadruped robot in more than 20 kinds of 3D complex environments, indoor and outdoor, without any external sensing devices such as radar, camera, etc., and our method showed good performance. In the lower part of the image is an example of a quadruped robot traversing highland. It is worth noting that the obstacles are unstructured and the robot can still traverse normally.

***Abstract*— Traversing 3-D complex environments has always been a significant challenge for legged locomotion. Existing methods typically rely on external sensors such as vision and lidar to preemptively react to obstacles by acquiring environmental information. However, in scenarios like nighttime or dense forests, external sensors often fail to function properly, necessitating robots to rely on proprioceptive sensors to perceive diverse obstacles in the environment and respond promptly. This task is undeniably challenging. Our research finds that methods based on collision detection can enhance a robot's perception of environmental obstacles. In this work, we propose an end-to-end learning-based quadruped robot motion controller that relies solely on proprioceptive sensing. This controller can accurately detect, localize, and agilely respond to collisions in unknown and complex 3D environments, thereby improving the robot's traversability in complex environments. We demonstrate in both simulation and real-world experiments that our method enables quadruped robots to successfully traverse challenging obstacles in various complex environments.**

I. INTRODUCTION

The natural environment is extraordinarily complex, characterized by irregular terrains and unstructured obstacles in three-dimensional spaces. Humans and animals rely on their robust limbs to locomote through complex environments by running, climbing, jumping, and altering their body postures. This often necessitates real-time visual perception and depth

* Equal Contributions
† corresponding author
Research supported by the National Natural Science Foundation of China under grants No.92248304 and Shenzhen Science Fund for Distinguished Young Scholars under Grant RCJC20210706091946001
[1] Tsinghua University, 100084 Beijing, China
[2] University of Michigan, Ann Arbor, MI 48109, USA
[3] Shanghai University, 200444 Shanghai, China.

of the surrounding environment to coordinate limbs to avoid obstacles. However, in the real world, both humans and animals have a limited field of vision, yet our limbs can move well beyond our visual perception range. When limbs move beyond the visual range or in environments where visual perception is ineffective (such as at night or in dense forests), perceiving environmental obstacles through proprioception becomes crucial. Designing a controller for robots that can navigate locomote through complex 3D environments and effectively avoid obstacles without relying on visual or lidar like external sensors presents a significant challenge.

In practical situations, humans and animals can rely on the sense of touch on their skin to detect the presence of obstacles around them. In robotic systems, this capability is referred to as collision detection. However, tactile sensors in robots are typically installed only in specific locations (such as the soles of the feet or fingertips), which results in the majority of the robot's body being unable to directly sense obstacles. Therefore, implementing collision detection for each limb becomes critically important. By relying on collision detection, robots can perceive unknown obstacles and guide their limbs movements, such as navigating through narrow spaces or finding paths in darkness. Ensuring that robots can effectively detect and respond to collisions not only enhances their autonomy but also ensures safety in interactions and operations within these unpredictable environments.

For quadruped robots, perceiving obstacles in complex three-dimensional environments poses a challenge. It requires three phases: detecting collisions, perceiving potential obstacles through the collisions, and replanning actions accordingly. Regarding collision detection, the current literature categorizes the entire collision process of robots into seven stages, referred to as the "collision event pipeline," including pre-collision, detection, isolation, identification, classification, reaction, and post-collision phases[1]. Among these, collision isolation specifically refers to locating the collision segment on the robot and its contact point. Most mature collision detection and isolation methods are designed for fixed-base robotic arms[2], [3], [4], [5], based on models or preset thresholds. Their primary aim is to halt the robotic arm in emergency situations to protect the operated object, the robotic arm itself, or human safety during human-robot interactions. However, these conventional methods are incapable of sensing potential obstacles generated from collisions, thus failing to redirect the robot's motion planning. Currently, end-to-end quadruped robot controllers based on reinforcement learning perceive complex terrains through proprioceptive observations, yet such methods are limited to terrain perception and do not extend to obstacle perception in three-dimensional environments[8], [9], [21], [22], [23], [30], [33], [34], [35], [44], [45]. Recently, some studies have used reinforcement learning to train quadruped robots for extreme parkour[40], [41], [24], [25], [26], [27], [28], [29], [31], [32], navigating through complex obstacles in three-dimensional spaces. However, these studies rely on external sensors, such as cameras, and have not proposed a solution that relies solely on proprioceptive sensing.

A. Related Works

Existing collision detection methods can be primarily classified into model-based methods or model-free methods. [2], [3], [4], [5] detect collision by comparing the estimated torques with predefined threshold. Among these, [2] proposes a momentum-based disturbance observer method. In comparison to the GM(generalized momenta-based) method, the MESO (modified extended state observer) method introduced by [5] demonstrates improved collision force estimation under similar noise levels in practical systems. Both methods can locate the robot link with collision and provide directional information on the Cartesian collision force. However, in practical robot applications, these model-based methods require high precision in the robot's model and are sensitive to disturbances and loads.

During the collision isolation phase [1], achieving more accurate localization of collision points/contact points relies on external sensors, such as those based on acoustics [6]. In model-free methods, approaches based on learning have also been proposed, but they necessitate external sensors such as laser rangefinders, cameras, and the collection of datasets [7]. The work discussed above is based on robotic arms with fixed bases. However, on robots with floating bases, the processes of collision detection, isolation, and identification become significantly more complex. This increased complexity arises because of more DOF, and the movement of these robots relies on continuous collision between their feet and the ground.

Recently, methods for collision detection, isolation, and identification in humanoid robots have been introduced [11], [12], [13]. In [13], relying solely on proprioceptive sensing, the authors combine and compare the GM approach with an Force/Torque-sensor-based approach. However, these methodologies have predominantly been tested in simulated environments with flat terrain. In real-world applications with robots, additional challenges such as communication delays, sensor synchronization, and noise must be taken into account. In the realm of body collision detection for quadruped robots, prior model-based efforts have proposed various state estimation techniques for the robot[14], [15], [16], attempting to fuse different sensors for more accurate state estimation [17]. Under conditions where only onboard perception is available, Fink et al. propose a sensor-less model based on kinematics to estimate the location of a single contact point at the shin level in real robot[10].

In the domain of model-free methods, collision/contact detection primarily involves processing sensor signals in the frequency domain [18] or constructing classifiers using machine learning approaches [19]. In [20], a deep learning-based contact estimator is developed using only onboard perception. However, these methods are predominantly applied to estimate foot forces for quadruped robots and do not consider the possibility of collisions at non-foot locations. [8], [9] have trained implicit estimators within their policies, treating external forces which applied to the robot as privi-

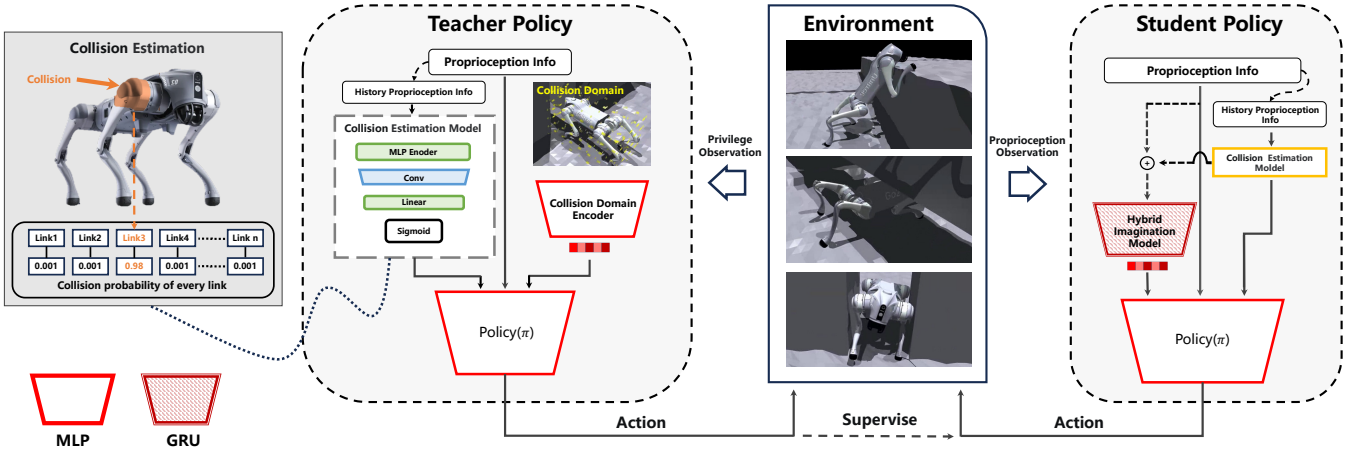

Fig. 2: Teacher-student based two-stage training framework, where on the left is the schematic of the collision estimation.

leged observations. This enables the use of historical data to inform subsequent movements. To the best of the authors' knowledge, there has yet to be research that trains an explicit estimator for collision detection and isolation, evaluates its accuracy in observing collisions, and utilizes this estimator to guide the robot's movement across varying environments.

B. Contributions

To further enhance the locomotive capabilities of quadruped robots without external perception, we have developed an end-to-end trained adaptive motion controller for quadruped robots, extending their mobility into complex three-dimensional environments. Our controller integrates proprioceptive sensing and collision estimation to implicitly imagine the features of obstacles in 3D spaces, enabling precise collision detection, localization of collision points, and agile collision responses and movements.

Our contributions are as follows:

- We introduced a collision estimator that utilizes historical proprioceptive data to precisely estimate the likelihood of collisions on each body part, guiding the robot to develop targeted response strategies through neural networks.
- We developed the concept of a Collision Domain and a Hybrid Imagination Model to capture and estimate the characteristics of three-dimensional obstacles, enhancing our method's ability to classify diverse obstacles in complex settings.
- We established a two-phase end-to-end training framework for quadruped robots, employing a simple linear velocity reward with directional constraints. This framework, relying solely on proprioceptive sensing, enables agile movement in complex 3D environments and demonstrates more robust locomotive performance in simulations and real-world scenarios compared to baseline methods.

II. Method

A. Task Formulation

We define the process of a quadruped robot traversing a 3D complex environment without the assistance of external sensors as four stages: 1) encountering and colliding with obstacles, 2) locating the specific position of the collision, 3) estimating the encountered obstacle, and 4) making a swift response to overcome the obstacle. When traversing in 3D complex environments, quadruped robots may unexpectedly collide with obstacles. Due to the uncertainty of the contact points, we first developed a collision detection estimator to determine whether a collision has occurred and its specific location. Our method of estimating encountered obstacles distinguishes us from other studies[21], [22]: unlike the traditional method of encoding terrain with elevation maps, we introduced the concept of a 3D collision domain, aimed at detecting obstacles that are about to come into contact with the robot's body. By combining the results of the collision detection estimator and the robot's proprioceptive abilities, we can effectively predict obstacles within the collision domain during actual movement. Based on this, the robot can take appropriate actions through real-time estimation of obstacles, effectively traversing various types of barriers.

B. Base Set

We model the environment as a Markov Decision Process (MDP). An MDP is defined by a tuple(S, A, P_a, R_a), where S represents the set of all possible states, and A represents the action space, $P(s_{t+1}|s_t, a_t)$ is the state transition function, indicating the probability of transitioning to state s_{t+1} after taking action a_t in state s_t, and $R(s_{t+1}|s_t, a_t)$ is the immediate reward. The agent selects an action a_t from the policy based on the current state s_t. For the state s_t and action a_t, the MDP calculates the next state s_{t+1} and reward r_t, and then provides feedback to the agent. The goal is to select a policy that maximizes the cumulative sum of discounted rewards, expressed as:

$$J(\pi) = \mathbb{E}_{\pi}\left[\sum_{t=0}^{\infty}\gamma^t r_t\right] \tag{1}$$

Policy Network For the training of the policy $\pi_\phi(a_t|s_t)$, we employ an actor-critic architecture. In this framework, the actor improves its decision-making policy by maximizing the expected return estimated by the critic. We utilize the Proximal Policy Optimization algorithm(PPO) to efficiently optimize the policy.

Action Space: The output of the policy is a 12-dimensional tensor. This tensor, once multiplied by a specific action scale coefficient a_{scale}, is amalgamated with a predetermined array of initial standing joint angles θ_{default}. This process culminates in the formation of the targeted joint angles configuration:

$$\theta_{\text{desired}} = \theta_{\text{default}} + a_t \cdot a_{scale} \tag{2}$$

Finally, proportional derivative (PD) controller is used for converting joint desired angle to joint torque.

State Space: For teacher policy, the state space s_t is composed of proprioceptive observation o_t, explicit observations $\hat{c}_t$, $\hat{v}_t$, where $\hat{c}_t$ is the estimate of collision information, $\hat{v}_t$ is the estimate of body linear velocity, and implicit observations p_t, e_t, refer to the latent variables obtained after the 3D collision domain and privileged information are processed through the encoders. Privileged implicit observation e_t includes encoded body mass, center of mass, friction and motor strength. The state space s_t is shown in the following equation:

$$s_t = \begin{bmatrix} o_t & \hat{c}_t & \hat{v}_t & p_t & e_t \end{bmatrix}^{\mathrm{T}} \tag{3}$$

The structure of student policy is kept consistent with the teacher policy, where e_t and p_t are estimated using proprioceptive observations. Specifically, in the first phase, we employ Regularized Online Adaptation (ROA)[42] to train a privileged information estimation module. In the second phase, we obtain estimates of privileged implicit information $\hat{e}_t$. Furthermore, in the second phase, we also train a hybrid imagination module to obtain estimates of $\hat{p}_t$, as detailed in section II.D.

Reward: Without the need for prior knowledge from external sensors, previous work's reward mechanisms [38], [40], [41] often cannot be directly applied to the tasks in this study. Our method primarily relies on observing the collision domain to master strategies for overcoming various types of obstacles. To ensure that the robot can smoothly traverse in the direction that obstacles are placed, we have designed an linear velocity tracking reward mechanism with heading constraints:

$$r_{vel} = L_{d_v} | \frac{min(v \cdot cos(\theta_{yaw}^{cmd} - \theta_{yaw}), v^{cmd})}{v^{cmd}} | \tag{4}$$

L_{d_v} is a weight factor that represents the positive and negative aspects of velocity, as follows:

$$L_{d_v} = \begin{cases} K_{positive} & , v \cdot \cos(\theta_{yaw}^{cmd} - \theta_{yaw}) > 0 \\ K_{negative} & , v \cdot \cos(\theta_{yaw}^{cmd} - \theta_{yaw}) < 0 \end{cases} \tag{5}$$

The linear velocity reward with heading constraint ensures that the robot moves along the direction of the obstacle placement, thus preventing the robot from choosing to bypass the obstacle. This design of L_{d_v} stems from observations made during the training process: when faced with difficult obstacles, robots tend to adopt a policy of quickly bouncing back to their original position, then approaching the obstacle at a set speed and bouncing back again, which easily leads to the policy falling into a local optimum. Therefore, we included a penalty for negative velocity in the speed tracking reward to prevent the robot from adopting a rebound policy.Where both $k_{positive}$ and $k_{negative}$ are constant quantities, positive reward coefficient and negative penalty coefficient respectively, the detailed values are in the appendix. To enable the robot to sidestep through narrow spaces and maintain a natural walking posture in normal environments, we introduced the following reward mechanism:

$$\begin{aligned} r_{Pos} &= W_1 \cdot r_{GuidePos} + W_2 \cdot r_{NaturalPos} \\ &= W_1 \cdot (|q_{hip}^{right} + q_{hip}^{left}|^2 + |q_{hip}^{front} - q_{hip}^{behind}|^2) \\ &\quad + W_2 \cdot |q_{dof}^{default} - q_{dof}|^2 \end{aligned} \tag{6}$$

The pos reward consists of two parts, the GuidePos reward does not directly limit the angle of the robot's hip joint but penalizes the sum of the angles of the left and right leg hips to be zero and the difference in angles between the front and back leg hips, ensuring that the robot's limbs can always walk perpendicular to the ground under any circumstances. This not only facilitates the generation of sidestepping actions but the NaturalPos reward ensures that the robot maintains a natural walking posture in normal environments. W_1, W_2 corresponds to the weights of the two-part rewards, the detailed values are in the appendix.

Although our method relies on collisions to perceive environmental obstacles, we have found that adding a collision penalty during actual training can accelerate the convergence of the policy. This is because, although collisions are necessary for perceiving obstacles, they should be avoided as much as possible during the traversal of obstacles. Therefore we refer to the collision penalty as follows, $\mathcal{F}_{collision}$ is the set of forces applied to the center of mass of each link.

$$r_{collision} = \mathcal{F}_{collision} \tag{7}$$

Furthermore, we enhance the quadruped robot's overall mobility through an auxiliary reward mechanism refer to [38].

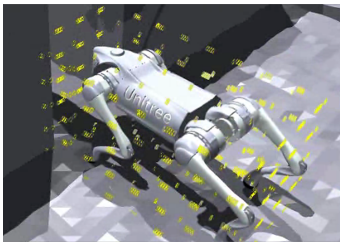

Fig. 3: Schematic of the collision domain

C. Collision Estimator

In real-world applications, since robots cannot directly recognize collisions and their specific locations, we propose

using a collision estimator φ to estimate collision information online. In this study, collision information is simplified into a Boolean vector c_t, indicating whether each link has encountered a collision. In the simulation environment, we cannot directly obtain c_t , so we determine that a collision has occurred if the force on each link exceeds a certain threshold. This threshold needs to be set according to different robots and the actual environment they are in. The aforementioned discussion indicates that most of the existing work on collision detection relies on dynamic information from a single timestep, which is effective for detecting collisions involving legs or arms. This is because collisions with legs and arms usually immediately reflect in the dynamics parameters, such as sudden changes in torque and acceleration, which can be captured through data within a single timestep. However, collisions involving the torso and joint parts are difficult to accurately capture due to the dispersed effects on dynamics parameters and response delays, and require analysis using long time series information. The collision estimator uses the robot's historical observation data to generate $\hat{c}_t$, which is the estimated value of the actual collision vector c_t.

$$\hat{c}_t = \varphi(o_{t-k}, o_{t-k+1} : o_t) \tag{8}$$

Using sequence information can better estimate the source of collisions. In this paper, we used historical observation data from 10 timesteps. The length of the historical record should not be too long, as it may lead to overfitting. The selected length of the historical record should be sufficient to represent the sum of the occurrence of a complete collision event and the response time generated. The impact of history sequence on collision estimation is detailed in the experimental section.

Given that collision information is only divided into two states: collision occurred and collision did not occur, and the total probability of these two states sums to 1, we do not need to concern ourselves with a specific value vector, but only need to output a probability value. Therefore, the collision estimation problem can be simplified into a binary classification problem. Our method only requires the use of the current moment's collision state c_t and its observation history, all of which can be obtained through simulation. The architecture of the collision estimator includes a linear layer, three convolutional neural network (CNN) layers, and another linear layer. The CNN layers apply convolution over the time dimension in the observation history to capture the temporal correlations in the input data. The flattened output processed by the CNN layers goes through another linear layer and applies a Sigmoid activation function, thereby producing an estimate of the collision state $\hat{c}_t$. This collision estimator is trained through supervised learning, with the goal of minimizing the following BCE loss function:

$$\mathcal{L}_{\varphi}^{Est} = -(c_t log\hat{c}_t + (1 - c_t)log(1 - \hat{c}_t)) \tag{9}$$

D. Implicit Collision Domain imagination

To better perceive and respond to obstacles in the complex 3D environments, this study introduces the concept of a collision domain, which is an area around the robot's body containing information about obstacles, as shown in Fig. 3. Specifically, we have set a cuboid sampling domain centered on the robot, see Table 1 for specific values. A boolean value indicates whether each point of the grid is inside an obstacle to detect imminent collisions. In complex environments such as dense forests and low visibility areas where external sensors like lidar and cameras often fail, proprioception becomes the sole means of understanding the collision domain. Therefore, this study proposes a method that uses only proprioception to deeply estimate the characteristics of the collision domain.

Previous research has shown that terrain characteristics can be estimated using only proprioception[21], [22]. In this study, we extend this idea to understanding features in 3D space, that is, the implicit imagination of the collision domain. Here, we propose a two-stage training method that combines implicit collision domain imagination to develop a policy effective in traversing complex environments. We combine proprioception with the aforementioned collision estimator to comprehensively estimate the potential attributes of the collision domain. This approach is better suited to adapt to the cognitive uncertainties present in real-world scenarios than methods that directly input proprioceptive sensing.

The specific training framework of our method is shown in Fig. 2. In the first phase, we use an encoder to extract latent features from the collision domain, inputting the resulting latent variable p_t directly into the teacher policy to train the robot. In the second phase, we supervise the training of the student policy using the teacher policy. Since the student policy cannot directly access p_t, we propose a hybrid imagination model that combines proprioceptive observations o_t with estimates $\hat{c}_t$ from the collision estimator, feeding them into a GRU pipeline to estimate latent features of the collision domain. The advantage of the hybrid imagination model is that, after embedding explicit collision estimates, the GRU module's temporal feature extraction mechanism can accurately represent the process of the robot interacting with obstacles. This allows for a dynamic estimation of environmental obstacle information, further estimating the latent features of the collision domain, thereby improving the robot's response capability after encountering obstacles.

III. EXPERIMENTS

A. Training set up

The simulator utilized for training the policy is Isaac Gym[37], supplemented by the prior open-source libraries Legged Gym and RSL-RL[38] for. We conduct parallel training across 4,096 domain-randomized environments on a single NVIDIA RTX 4090 GPU, teacher policy engaging in approximately 12000 training iterations and student policy engaging in not more than 10000 training iterations, each comprising 24 steps, with the policy operating at a control frequency of 50Hz.

To accurately reflect the diversity of obstacles encountered in the real world, we categorize four types of obstacles,

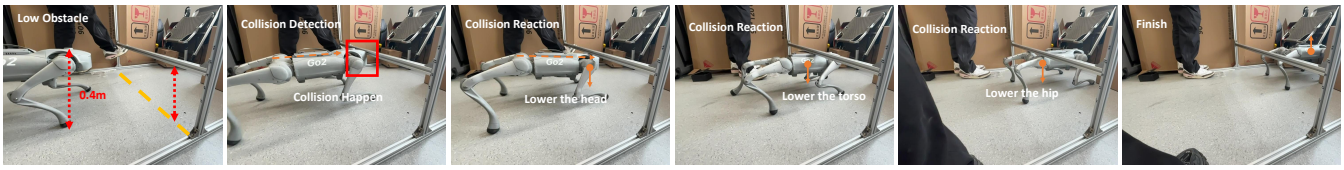

Fig. 4: Locomotion through a tunnel: Adaptation strategies for collision avoidance.

TABLE I: Curriculum Learning: Obstacle Configuration and Robot Parameter Settings.

Properties	Obstacle Train Ranges (m) ($[l_{easy}, l_{hard}]$)	Test Ranges (m) ($[l_{easy}, l_{hard}]$)
Highland	[0.05, 0.55]	[0.25, 0.55]
Barrier	[0.31, 0.00]	[0.16, 0.00]
Tunnel	[0.40, 0.25]	[0.38, 0.25]
Crack	[0.38, 0.28]	[0.32, 0.28]

Properties	Parameters Go2 Body (m)	Collision Domain (m)
length	0.71	0.90
width	0.32	0.40
height	0.40	0.50

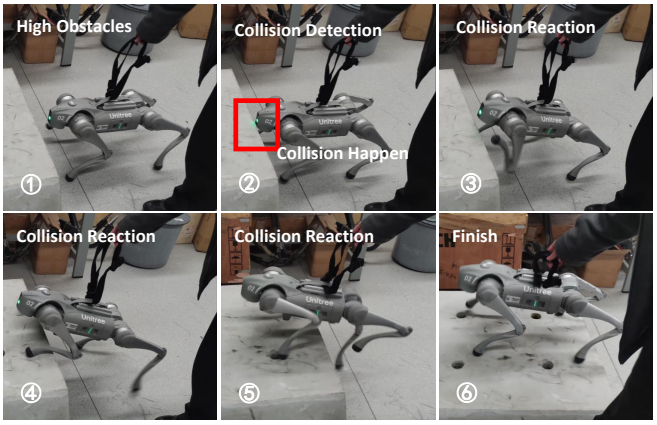

Fig. 5: Collision response strategies when facing a highland.

which included: Highlands, which are climbable terrains directly ahead, as shown in Fig. 5.; Barriers in Fig. 7, obstacles located on the left or right side of the robot; Tunnels, low tunnel-like obstacles that the robot can pass through by lowering its body height like Fig. 4; and Cracks, narrow passages shown in Fig. 7. Additionally, we implement a strategic obstacles curriculum designed to enhance the policy's generalization capabilities and convergence rate.

B. Hardware and Depoly

In our research, we employ the Unitree Go2 robot to assess the efficacy of the collision detection estimator and the policy's response to collisions. The body parameters of Go2 are shown in Table 1, which serves as an important reference for setting up our training environment. For the deployment of policies on real robots, these policies are executed on the onboard NVIDIA Jetson Orin Nano of the Go2. Furthermore, [36] provides an exemplary framework that facilitates the straightforward transplantation of these policies to the Go2 robot.

C. Compared Method

We compare our collision estimation and response policy with several baseline and ablations as follows.

- **Ours w/o R.V:** We use the ordinary linear velocity tracking reward [38] instead of the linear velocity tracking reward with heading constraints.
- **Ours w/o Col:** Training without collision estimator.
- **Ours w/o H.O:** Training collision estimator without history observation.
- **Baseline:** Training directly with only proprioception.
- **RMA[21]:** Employing an Adaptation Module to Estimate All Privileged Observations Including Terrain Height from Historical Data, Harnessing Potential Information.
- **WTW(Walk-These-Ways)[36]:** Leverages expert knowledge realize Multiplicity of Behavior.
- **Go2-default:** Go2 default controller based on NMPC, which only compared in real experiment.
- **Go2-special:** Go2 special controller based on NMPC, which only compared in real experiment.

D. Simulation Experiments

Baselines and Ablations. We conducted a comparative analysis of our method against baseline approaches such as RMA and WTW, which are two typical methods in this domain. The RMA approach extracts information from 2D elevation maps and privileged data to adapt to the environment, whereas WTW relies on expert knowledge to set rewards and manually adjust the robot's posture for obstacle traversal. In addition to RMA's framework, we implemented our reward function. As shown in TABLE II, the 2D elevation maps used in RMA struggle to capture the complete features of 3D obstacles in the environment, resulting in significantly lower performance across multiple tasks compared to our method. WTW can control the robot's height, which allows the robot to crawl to traverse Tunnel obstacles but lacks the ability to climb or sidestep, failing to navigate complex obstacles like Highland, Crack, and Barrier.

Comparing our method with Ours w/o R.V, we found that the velocity reward without heading direction constraint failed to traverse Highland and Crack obstacles, and also

TABLE II: We test our method against several baselines and ablations in the simulation with obstacle of varying difficulty (see in Table I). We initialized a total of 4096 robots and randomly distributed them across the map . We collected data on the success rate (the ratio of the number of robots that failed to traverse obstacles to the total number of robots) and the average movement distance which was normalized to [0, 1] after a duration of 10 seconds.

	Success Rate ↑				Average Displacement↑			
	Highland	Barrier	Tunnel	Crack	Highland	Barrier	Tunnel	Crack
Baseline	0.00	0.12	0.00	0.00	0.004	0.132	0.005	0.006
RMA	0.27	0.61	0.67	0.53	0.256	0.550	0.614	0.497
WTW	0.00	0.11	1.00	0.00	0.004	0.103	1.000	0.005
Ours w/o R.V	0.00	0.31	0.30	0.00	0.003	0.197	0.218	0.005
Ours w/o Col	0.89	0.81	0.89	0.76	0.769	0.771	0.898	0.625
Ours w/o H.O	0.93	0.87	0.94	0.83	0.796	0.793	0.959	0.711
Ours	0.94	0.92	0.96	0.89	0.821	0.853	0.979	0.798
Teacher	0.99	1.00	1.00	1.00	1.000	1.000	1.000	1.000

performed poorly with Barrier and Tunnel. This is because the previous method's velocity reward was solely dependent on the fixed-base linear velocity, leading robots to quickly circumvent rather than traverse obstacles, a common policy during training due to the lower exploration difficulty of avoiding obstacles.

The comparison between Ours and Ours w/o H.O highlights the importance of historical information for collision estimation, particularly when traversing Barrier and Crack obstacles. These obstacles often involve more collisions at the hip joint, and estimating collisions from a single timepoint is challenging. Furthermore, the comparison suggests that the inclusion of collision information significantly improves the robot's performance in navigating various complex obstacles. Collision information aids the robot in inferring the characteristics of the obstacle from the contact, enabling further appropriate responses.

Effects of Implicit Collision Domain imagination. To thoroughly investigate the impact of Implicit Collision Domain imagination on the task and to quantify the efficacy of the Hybrid Imagination Model, we employed the t-distributed stochastic neighbor embedding (t-SNE) to reduce the dimensions of the output from the Hybrid Imagination Model. As illustrated in the Fig. 6., we discovered that the latent representation of the Collision Domain estimated by the Hybrid Imagination Model shows a clear distribution across different obstacle categories. This indicates two points: 1. The Collision Domain can effectively extract features of obstacles in complex three-dimensional environments. 2. The Hybrid Imagination Model can efficiently estimate the latent representation of the Collision Domain, thereby possessing sufficient environmental information to aid the robot in responsive decision-making in complex settings. Additionally, the t-SNE plot shows that some overlap between the Barrier and Crack obstacles, suggesting that the body locations where the robot collides with these two types of obstacles are often similar, which also indirectly shows that the latent representation of the Collision Domain accurately reflects the actual situation.

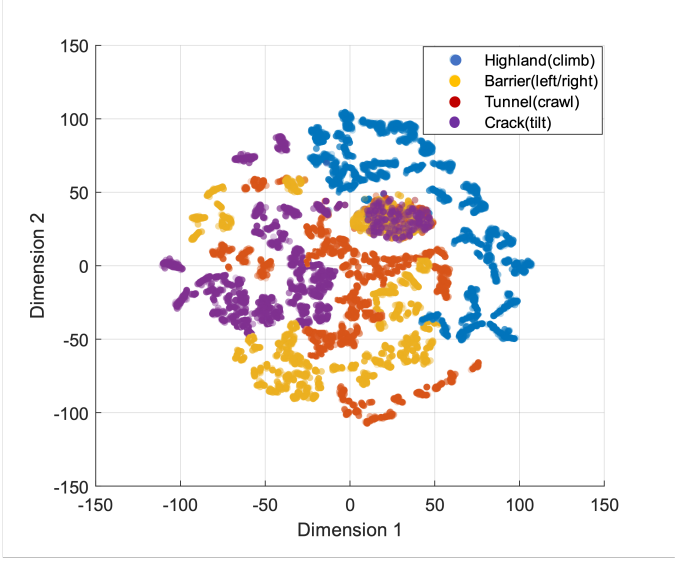

Fig. 6: The t-SNE visualization for Collision Domain in the latent space.

E. Real-World Experiments

Qualitative Experiments. In an indoor environment, we conducted a series of experiments aimed at exploring different obstacles through collision detection including highlands, barriers, tunnels, cracks to qualitatively analyze the impact of collisions on a robot's ability to traverse unknown obstacles.In exploring the Highlands,as shown in Fig. 5., the robot first used its head collision detection to perceive potential highland obstacles. Subsequently, the robot extended its right front foot for further exploration. Upon detecting a collision with the right front foot, the robot determined the presence of a highland obstacle and chose to overcome it by climbing. When facing a Barrier, the robot detected the obstacle through a collision on the left side of its head. Then, by probing with its front feet, it confirmed the obstacle was

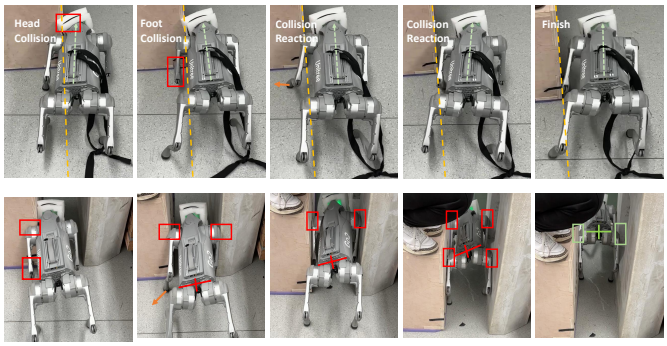

Fig. 7: Collision response strategies when facing a barrier (first row) and crack obstacles (second row).

on the left while the right side was clear, thus, the robot chose to turn right to bypass the obstacle, as shown in Fig. 7.. In the Tunnel obstacles, after an initial head collision detection, the robot did not detect any horizontal collisions with its hands, leading to the determination that the obstacles was a tunnel. The robot then successfully passed through by lowering the height of its torso and hips, as shown in Fig. 4. In the exploration of Crack obstacles, the robot's left front hip and left front foot first detected a collision. Attempting to move right, the robot encountered another collision at the right front hip, thereby determining the obstacles to be a crack. The robot chose to adjust the roll axis angle of its body to navigate through the obstacle, as shown in Fig. 7.

Additionally, we tested our approach in complex wild scenarios, as depicted in Fig. 1., where the robot traverse unstructured rocky obstacles. This presents a significant challenge as both collision points and potential footholds are unpredictable, making it difficult for the robot to identify obstacle characteristics. To our knowledge, there have been almost no other works that have achieved similar results. But our team managed to traverse wild scenarios which has various obstacles without external sensors, achieving a success rate of over 50%. More details on these wild experiments are available in the accompanying video.

Quantitative Experiment. As depicted in the Fig. 8., we have quantitatively compared our method with three other strategies: Go2 Default, Go2 Special, and WTW, in real-world scenarios. The data shown represents the average success rate of the robot when confronting various types of obstacles at different levels of difficulty, across 10 trials, aiming to showcase the obstacle negotiation capability of our approach in actual environments. It is clear from the figure that experiments utilizing our policy exhibited superior performance in almost all obstacle types. Notably, in the specific scenario of navigating through Tunnels, the WTW policy performed slightly better, primarily due to its use of human knowledge priors and manual adjustment of the robot's height for successful passage. In contrast, our method is adaptive, which may lead to certain errors in estimating environmental obstacles, resulting in a degree of performance degradation. However, this performance degradation is within an acceptable range.

IV. CONCLUSION

We present a novel quadruped robot collision response motion controller based on collision domains and explicit collision estimator, capable of precise collision detection, localization, and agile response in unknown and complex 3D environments, solely relying on proprioception. Comprehensive evaluations in both simulated and real-world environments have demonstrated the precision of our collision detection technique, the robustness of our collision response, and the effective traversability in complex environments.However, a limitation of this work is that during the simulation training process, our approach necessitates a relatively accurate robot collision model, as it significantly dictates the robot's capability to perceive environmental obstacles. However, crafting such an accurate collision model in the simulation phase demands substantial computational resources, and the calculation of collision contact forces in complex models introduces certain errors. We look forward to finding a solution to this challenge in the future. Additionally, in our forthcoming research, we aim to integrate a recognition network to enhance the accuracy of estimating both the direction and magnitude of forces. This will enable us to precisely mitigate potential disturbances and implement adaptive control across environments characterized by diverse stiffness and contact elasticity.

REFERENCES

[1] S. Haddadin, A. De Luca and A. Albu-Schäffer, "Robot Collisions: A Survey on Detection, Isolation, and Identification", in *IEEE Transactions on Robotics*, vol. 33, no. 6, pp. 1292-1312, Dec. 2017.

[2] A. De Luca, A. Albu-Schaffer, S. Haddadin and G. Hirzinger, "Collision Detection and Safe Reaction with the DLR-III Lightweight Manipulator Arm", 2006 *IEEE/RSJ International Conference on Intelligent Robots and Systems(IROS)*, Beijing, China, 2006, pp. 1623-1630.

[3] A. De Luca and L. Ferrajoli, "Exploiting Robot Redundancy in Collision Detection and Reaction", 2008 *IEEE/RSJ International Conference on Intelligent Robots and Systems(IROS)*, Nice, France, 2008, pp. 3299-3305.

[4] M. Iskandar, O. Eiberger, A. Albu-Schäffer, A. De Luca and A. Dietrich, "Collision Detection, Identification, and Localization on the DLR SARA Robot with Sensing Redundancy", 2021 *IEEE International Conference on Robotics and Automation (ICRA)*, Xi'an, China, 2021, pp. 3111-3117.

[5] Ren, T.Y., Dong, Y.F., Wu, D., & Chen, K., "Collision detection and identification for robot manipulators based on extended state observer", *Control Engineering Practice*, 79, 144–153, 2008.

[6] X. Fan et al., "Acoustic Collision Detection and Localization for Robot Manipulators", 2020 *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Las Vegas, NV, USA, 2020, pp. 9529-9536.

[7] D. Popov, A. Klimchik and N. Mavridis, "Collision detection, localization & classification for industrial robots with joint torque sensors", 2017 26th *IEEE International Symposium on Robot and Human Interactive Communication (RO-MAN)*, Lisbon, Portugal, 2017, pp. 838-843.

[8] Nahrendra, I & Yu, Byeongho & Myung, Hyun. "DreamWaQ: Learning Robust Quadrupedal Locomotion With Implicit Terrain Imagination via Deep Reinforcement Learning", *arXiv preprint arXiv:2301.10602*, 2023.

[9] Wu, J.Z., Xue, Y.F., Qi, C.K. "Learning Multiple Gaits within Latent Space for Quadruped Robots", *arXiv preprint arXiv:2308.03014*, 2023

[10] Fink, G., Focchi, M., & Caldwell, D.G. , "On the detection and localization of shin collisions and reactive actions in quadruped robots", *Synergy of Automation, IoT & AI*, 2019.

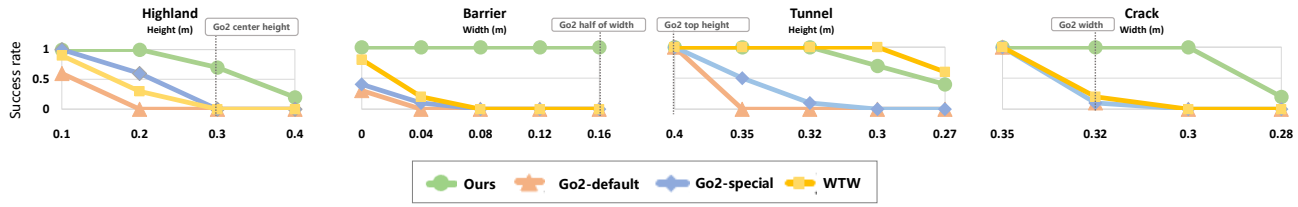

Fig. 8: Real-world quantitative experiments.

[11] S. Faraji and A. J. Ijspeert, "Designing a virtual whole body tactile sensor suit for a simulated humanoid robot using inverse dynamics", 2016 *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Daejeon, Korea (South), 2016, pp. 5564-5571.

[12] F. Flacco, A. Paolillo and A. Kheddar, "Residual-based contacts estimation for humanoid robots", 2016 *IEEE-RAS 16th International Conference on Humanoid Robots (Humanoids)*, Cancun, Mexico, 2016, pp. 409-415.

[13] J. Vorndamme, M. Schappler and S. Haddadin, "Collision detection, isolation and identification for humanoids", 2017 *IEEE International Conference on Robotics and Automation (ICRA)*, Singapore, 2017, pp. 4754-4761.

[14] van Dam, J., Tulbure, A., Minniti, M.V., Abi-Farraj, F., Hutter, M., "Collision detection and identification for a legged manipulator", *arXiv preprint arXiv:2207.14745*, 2022.

[15] J. Hwangbo, C. D. Bellicoso, P. Fankhauser and M. Hutter, "Probabilistic foot contact estimation by fusing information from dynamics and differential/forward kinematics", 2016 *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Daejeon, Korea (South), 2016, pp. 3872-3878.

[16] M. Camurri et al., "Probabilistic Contact Estimation and Impact Detection for State Estimation of Quadruped Robots", in *IEEE Robotics and Automation Letters*, vol. 2, no. 2, pp. 1023-1030, April 2017.

[17] Maravgakis, Michael & Argiropoulos, Despina-Ekaterini & Piperakis, Stylianos & Trahanias, Panos., "Probabilistic Contact State Estimation for Legged Robots using Inertial Information", *arXiv preprint arXiv:2303.00538*, 2023.

[18] Huynh, Ba-Phuc & Bae, Joonbum., "Impact Intensity Estimation of a Quadruped Robot without Using a Force Sensor", *arXiv preprint arXiv:2204.01003*, 2022.

[19] Meriçli, T., Meriçli, Ç., Akın, H.L., "A Robust Statistical Collision Detection Framework for Quadruped Robots", *RoboCup 2008: Robot Soccer World Cup XII*, 2008.

[20] Lin T Y, Zhang R, Yu J, et al, "Legged robot state estimation using invariant Kalman filtering and learned contact events", *arXiv preprint arXiv preprint arXiv:2106.15713*, 2021.

[21] A. Kumar, Z. Fu, D. Pathak, and J. Malik, "RMA: Rapid Motor Adaptation for Legged Robots", in *Robotics: Science and Systems XVII*, Jun. 2021.

[22] J. Lee, J. Hwangbo, L. Wellhausen, V. Koltun, and M. Hutter, "Learning Quadrupedal Locomotion over Challenging Terrain", *Science Robotics*, Oct. 2020.

[23] Jemin Hwangbo et al, "Learning agile and dynamic motor skills for legged robots", *Science Robotics*, Jan. 2019.

[24] Takahiro Miki et al, "Learning robust perceptive locomotion for quadrupedal robots in the wild", *Science Robotics*, Jan. 2022.

[25] David Hoeller et al, "ANYmal parkour: Learning agile navigation for quadrupedal robots", *Science Robotics*, Mar. 2024.

[26] Joonho Lee et al, "Learning robust autonomous navigation and locomotion for wheeled-legged robots", *Science Robotics*, Apr. 2024.

[27] Fabian Jenelten et al, "DTC: Deep Tracking Control", *Science Robotics*, Jan. 2024.

[28] T. He, C. Zhang, W. Xiao, G. He, C. Liu, and G. Shi, "Agile But Safe: Learning Collision-Free High-Speed Legged Locomotion." *arXiv preprint arXiv:2401.17583*, 2024.

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

[30] J. Long, Z. Wang, Q. Li, J. Gao, L. Cao, and J. Pang, "Hybrid Internal Model: Learning Agile Legged Locomotion with Simulated Robot Response." *arXiv preprint arXiv:2312.11460*, 2023.

[31] M. Liu, Z. Chen, X. Cheng, Y. Ji, R. Yang, and X. Wang, "Visual Whole-Body Control for Legged Loco-Manipulation." *arXiv preprint arXiv:2403.16967*, 2024.

[32] T. Miki, J. Lee, L. Wellhausen, and M. Hutter, "Learning to walk in confined spaces using 3D representation." *arXiv preprint arXiv:2403.00187*, 2024.

[33] L. Smith et al., "Learning and Adapting Agile Locomotion Skills by Transferring Experience." *arXiv preprint arXiv:2304.09834*, 2023.

[34] T. Li, H. Jung, M. Gombolay, Y. K. Cho, and S. Ha, "CrossLoco: Human Motion Driven Control of Legged Robots via Guided Unsupervised Reinforcement Learning." *arXiv preprint arXiv:2309.17046*, 2023.

[35] T. Li, J. Won, S. Ha, and A. Rai, "FastMimic: Model-based Motion Imitation for Agile, Diverse and Generalizable Quadrupedal Locomotion." *arXiv preprint arXiv:2109.13362*, 2021.

[36] G. B. Margolis and P. Agrawal, "Walk These Ways: Tuning Robot Control for Generalization with Multiplicity of Behavior", *6th Annual Conference on Robot Learning*, 2022.

[37] 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", *Advances in Neural Information Processing Systems*, Track on Datasets and Benchmarks , 2021.

[38] N. Rudin, D. Hoeller, P. Reist, and M. Hutter, "Learning to walk in minutes using massively parallel deep reinforcement learning", in Proc. *Conference on Robot Learning (CoRL)* , 2022.

[39] Laura Smith, J Chase Kew, Xue Bin Peng, Sehoon Ha, Jie Tan, and Sergey Levine, "Legged Robots that Keep on Learning: Fine-Tuning Locomotion Policies in the Real World", *arXiv preprint arXiv:2110.05457*, 2021.

[40] Ziwen Zhuang, Zipeng Fu, Jianren Wang, Christopher Atkeson, Sören Schwertfeger, Chelsea Finn, and Hang Zhao, "Robot Parkour Learning", In *Conference on Robot Learning (CoRL)*, 2023.

[41] Xuxin Cheng, Kexin Shi, Ananye Agarwal, Deepak Pathak, "Extreme Parkour with Legged Robots", *arXiv preprint arXiv:2309.14341*, 2023.

[42] Zipeng Fu, Xuxin Cheng, and Deepak Pathak, "Deep whole-body control: learning a unified policy for manipulation and locomotion", In *Conference on Robot Learning (CoRL)* , 2022.

[43] Jinze Wu, Guiyang Xin, Chenkun Qi, and Yufei Xue. "Learning robust and agile legged locomotion using adversarial motion priors", IEEE Robotics and Automation Letters, 2023.

[44] A. Escontrela et al., "Adversarial Motion Priors Make Good Substitutes for Complex Reward Functions." *arXiv preprint arXiv:2203.15103*, 2022.

[45] L. Ye, J. Li, Y. Cheng, X. Wang, B. Liang, and Y. Peng, "From Knowing to Doing: Learning Diverse Motor Skills through Instruction Learning." *arXiv preprint arXiv:2309.09167*, 2023.

[46] Nikita Rudin, David Hoeller, Philipp Reist, and Marco Hutter. Learning to walk in minutes using massively parallel deep reinforcement learning. In Conference on Robot Learning (CoRL), 2021.

Appendix

A. Reward Function

Reward items are listed in TABLE III. The task reward helps the robot adapt to the specific task scenarios, and the details of the design (4,6,7) is shown in II.B above. Where we set the Goal Velocity's reward coefficient $K_{positive}$ to 1 and the penalty coefficient $K_{negative}$ to -3. This set of parameters was optimized based on our comprehensive simulation experiments. Additionally, we designed the regularization reward referring to [46], adjusted it according to the type of robots and task scenarios.

TABLE III: Reward Function

Term	Equation	Weight
Task reward		
Goal Velocity	$r_{vel}(4)$	1.5
Yaw Angular Velocity	$exp(-4\lvert\omega_{yaw}^{cmd} - \omega yaw\rvert)$	0.5
Hip Position	$r_{Pos}(6)$	-0.5(W_1=W_2=0.5)
Collision	$r_{collision}(7)$	-10.0
Regularization reward		
Z Velocity	v_z^2	-0.5
X&Y Velocity	$\omega_x^2 + \omega_y^2$	-0.01
Dof Acceleration	$\sum_{i=1}^{12} \ddot{q}_i^2$	$-2.5 * 10^{-7}$
Action Rate	$\sqrt{\sum_{i=1}^{12}(a_t - a_{t-1})^2}$	-0.1
Delta Torques	$\sum_{i=1}^{12}(\tau_t - \tau_{t-1})^2$	$-1.0 * 10^{-7}$
Torques	$\sum_{i=1}^{12}(\tau_t)^2$	$-1.0 * 10^{-5}$
Dof Error	$\sum_{i=1}^{12}(q - q_{default})^2$	-0.04
Feet Stumble	$\lvert F_{feet}^{hor}\rvert > 4 * \lvert F_{feet}^{ver}\rvert$	-1
Dof Position Limits	$\sum_{i=1}^{12}(q_i^{out}{}_{q_i > q_{\max} \parallel q_i < q_{\min}})$	-10.0

B. Additional Training Details

Network Architecture Our learning framework's overall network architecture includes not only the components shown in Figure 2 but also some additional modules not depicted in the figure. The teacher policy consists of six Multilayer Perceptron (MLP) parts: collision domain encoder, collision estimator, privileged information encoder, privileged information estimator, velocity estimator, and teacher policy network. Where $\boldsymbol{h}_t$ denotes collision domain information and $\boldsymbol{g}_t$ denotes privilege information. The privileged information encoder supervises the privileged information estimator and optimizes learning using the ROA method. Privileged Estimator's network type is also a CNN with a similar structure to the Collision Estimator. The student policy includes five parts: the collision estimator, velocity estimator, and privileged information estimator from the teacher policy, along with the hybrid imagination model comprised of a Gated Recurrent Unit (GRU) network and an MLP-based student policy network. Table I provides more details on each layer.

Training course Course learning is crucial for robots to effectively travel obstacles in complex environments. Without this capability, robots would struggle to learn effectively. We have implemented the velocity-travel course training method[43], [46], where the robot's linear velocity is randomly sampled within the range of [0, 1]. If the robot's travel distance in one iteration exceeds half of the preset heading speed integral, the terrain difficulty is increased; otherwise, it is decreased. The levels of terrain difficulty are detailed in TABLE I.

TABLE IV: Network architectures

Module	Inputs	Hidden Layers	Outputs
Teacher policy			
Collision Domain Enc	$\boldsymbol{h}_t$	[256, 128, 64]	$\boldsymbol{p}_t$
Collision Estimator	$\boldsymbol{o}_{t-10}, \ldots, \boldsymbol{o}_{t-1}, \boldsymbol{o}_t$	/	$\hat{\boldsymbol{c}}_t$
Privileged Encoder	$\boldsymbol{g}_t$	[64, 32]	$\boldsymbol{e}_t$
Privileged Estimator	$\boldsymbol{o}_{t-5}, \ldots, \boldsymbol{o}_{t-1}, \boldsymbol{o}_t$	/	$\hat{\boldsymbol{e}}_t$
Velocity Estimator	$\boldsymbol{o}_t$	[128, 64]	$\hat{\boldsymbol{v}}_t$
Teacher Network	$\boldsymbol{o}_t, \boldsymbol{p}_t, \hat{\boldsymbol{c}}_t, \boldsymbol{e}_t, \hat{\boldsymbol{v}}_t$	[512, 256, 128]	$\boldsymbol{a}_t$
Student policy			
Hybrid Imagination Model	$\hat{\boldsymbol{c}}_t, \boldsymbol{o}_t$	[128, 64]	$\hat{\boldsymbol{p}}_t$
Student Network	$\boldsymbol{o}_t, \hat{\boldsymbol{p}}_t, \hat{\boldsymbol{c}}_t, \hat{\boldsymbol{e}}_t, \hat{\boldsymbol{v}}_t$	[512, 256, 128]	$\hat{\boldsymbol{a}}_t$

Hyperparameter	Value
Discount Factor	0.99
GAE Parameter	0.95
Timesteps per Rollout	21
Epochs per Rollout	5
Minibatches per Epoch	4
Entropy Bonus (α_2)	0.01
Value Loss Coefficient (α_1)	1.0
Clip Range	0.2
Reward Normalization	yes
Learning Rate	2e$-$4
# Environments	4096
Optimizer	Adam

TABLE V: PPO hyperparameters.

Policy training and Imitation training We use PPO with hyperparameters listed in TABLE V to train the teacher policy. We regard the process of teacher policy supervising student policy learning as imitation learning process, a_{t} and $\hat{a}_{\mathrm{t}}$ are the action vectors from the teacher and student respectively in an actor network. The gradient of the loss $\mathcal{L}$ can be defined as the sum of the squared norms of the difference between a_{t} and $\hat{a}_{\mathrm{t}}$ over all actions:

$$\mathcal{L}_{Imitation} = \|\pi_{\text{teacher}}(\cdot|s) - \pi_{\text{student}}(\cdot|s)\|_2^2$$

Here, $\|\cdot\|_2$ denotes the L^2 norm. Where $\pi_{\text{teacher}}(\cdot|s)$ is the action from the Teacher policy and $\pi_{\text{student}}(\cdot|s)$ is the action from the Student policy .