Pakour but Safe: Agile Navigation with Parkour Skills

Xinyao Li*
Department of Computer Science
Shanghai Jiao Tong University
Shanghai, China
lxy4488@sjtu.edu.cn

Ziyan Li*
Department of Computer Science)
Shanghai Jiao Tong University
Shanghai, China
22_lzy@sjtu.edu.cn

***Abstract*—This paper addresses the challenges of extending parkour skills to practical navigation for legged robots, focusing on obstacle variability, the trade-off between efficiency and safety, and limited environmental perception. We propose a novel approach that integrates parkour policies with safer obstacle avoidance strategies. Our method uses depth images to inform a policy selector that decides between parkour and avoidance strategies based on obstacle characteristics. By combining simulation-based rollouts with depth-based decision-making, our framework balances agility and safety, enabling more reliable and efficient navigation in complex environments. We validate the approach through simulations, showing that it improves safety and performance compared to baseline methods.**

Keywords—legged robots, parkour skills, safe avoidance, policy selector

I. INTRODUCTION

Legged robots, such as quadrupeds, have shown remarkable progress in recent years, pushing the boundaries of extreme motion capabilities through activities like parkour. Traditionally, parkour for robots was achieved using model-based control techniques; however, with the advent of learning-based methods, robots can now perform robust parkour maneuvers across diverse terrains and obstacles. While these advancements demonstrate significant potential, they often remain confined to showcasing parkour capabilities in controlled environments, leaving a gap in practical applications. Specifically, how parkour skills can be integrated into real-world scenarios remains an open question.

We propose to address this gap by incorporating parkour skills into navigation tasks. Our objective is to enable legged robots to traverse challenging environments efficiently and gracefully, reaching hard-to-reach target locations. This ability has many real-world applications, such as search-and-rescue missions in collapsed buildings or navigating complex natural terrains. In such scenarios, robots must traverse environments by walking, running, climbing, and overcoming obstacles while minimizing collisions and avoiding task failures. Achieving these objectives requires robots to perceive their surroundings effectively, adapt to rapidly changing environments, and balance agility with safety.

However, several challenges arise when extending parkour skills to navigation in practical scenarios:

1. Obstacle Characteristics: Unlike controlled settings, obstacles in real environments vary widely in shape and size, often exceeding the robot's physical capabilities. For example, it is challenging for current quadrupeds to climb obstacles taller than their height or to jump across gaps spanning more than 1.5 to 2 times their body length. Purely relying on parkour policies can lead to failures or physical damage.

2. Trade-off Between Efficiency and Safety: Efficiently traversing obstacles through parkour skills such as climbing and jumping involves inherent risks, while safer alternatives, such as bypassing obstacles, may be significantly less efficient. Robots must strike a balance between leveraging agile parkour maneuvers and opting for safer navigation strategies.

3. Limited Perception: Environmental perception is constrained by occlusions and the limited field of view of onboard sensors, providing only partial observations of the scene. This limitation complicates the robot's ability to plan and execute reliable navigation strategies.

To address these challenges, we propose a novel approach that combines parkour policies with safer obstacle-avoidance policies trained using Proximal Policy Optimization with Lagrangian (PPO-Lagrangian). Our method integrates depth images as the primary visual input to:

1. Inform the policy selector, which determines whether to employ parkour or avoidance strategies based on the obstacle characteristics.

2. Guide the avoidance policy and also the parkour policy, enabling the robot to navigate safely and effectively among obstacles.

We leverage simulation-based rollouts to collect diverse data and train the policy selector, enabling the robot to make informed decisions based on depth information. By integrating these components, our approach aims to strike a balance between efficiency and safety, allowing legged robots to achieve their navigation goals in complex environments.

Briefly, we identify our contributions as follows:

1. We propose a novel framework that integrates parkour and obstacle-avoidance policies, striking a balance between

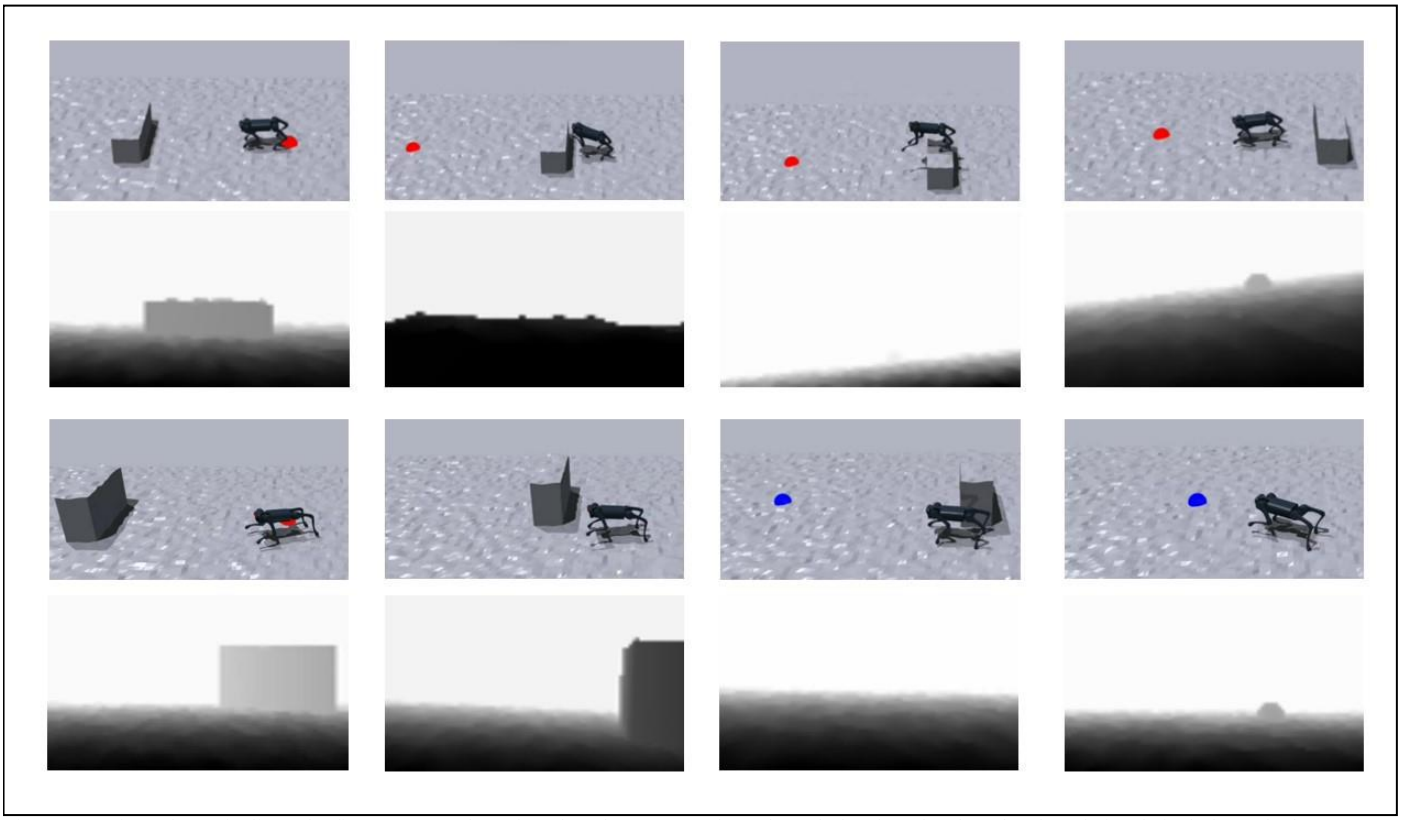

Fig. 1. Overview of our pipeline: The first two rows show the robot chooses to parkour through the obstacle when the obstacle is relatively low. The last two rows show the robot chooses to walk around when the obstacle is too high and difficult. The corresponding depth image is shown below the robot.

efficiency and safety for legged robot navigation in challenging environments.

2. We propose a data-driven method to estimate the difficulty of the environment, enabling the robot to adaptively choose the appropriate policy between parkour and avoidance strategies.

3. We validate that depth images as visual input can effectively provide information to guide both the policy selector and the obstacle-avoidance policy.

4. We demonstrate the effectiveness of our approach in simulation, showcasing the robot's ability to navigate complex terrains with improved safety and efficiency compared to baseline methods.

II. RELATED WORK

A. Robot Parkour

In recent years, reinforcement learning (RL) has achieved remarkable progress in improving the locomotion capabilities of legged robots. Particularly in the past two years, works such as [1]–[5] have successfully demonstrated agile parkour techniques on quadruped robots, including climbing, jumping, crawling, and traversing inclined terrains. Among these, [3] achieved a significant milestone by enabling the Unitree-A1 robot to cross obstacles twice its height and jump over gaps twice its body length. These approaches rely on either two-stage [1]–[3] or single-stage [4], [5] reinforcement learning frameworks to train end-to-end policies. A common strategy involves using privileged information via a teacher-student paradigm to help the robot acquire high-difficulty parkour skills effectively.

Although these methods have been validated as effective, the experimental settings are typically constrained to obstacles with heights and lengths well within the robot's physical capabilities. In contrast, real-world scenarios often present more complex and challenging obstacles that exceed these limits. Blindly attempting parkour maneuvers in such settings can lead to failures or physical damage, which highlights the need for cautious and adaptive strategies. To the best of our knowledge, there has been limited exploration of how robots might estimate the difficulty of obstacles to make safer and more informed decisions while attempting parkour actions.

B. Perception for Navigation and Locomotion

Perception plays a critical role in both navigation and locomotion for legged robots. Traditional control methods process visual information into representations such as elevation maps, traversability maps, and state estimators to guide motion planning and control. Recently, with the rise of learning-based methods, end-to-end policies have emerged, directly integrating visual and control modules. These methods utilize visual inputs from devices like depth cameras, LiDAR, and RGB cameras, representing the environment as elevation maps [6]–[11], depth images [1], [3], [12]–[14], or point cloud data[2].

Elevation maps, while effective for locomotion tasks requiring terrain height information, are often pre-processed and post-processed to reduce noise and disturbances [7], [8]. However, their limited information makes them less suitable for tasks requiring comprehensive understanding of obstacle shapes, such as high-difficulty parkour or terrain difficulty estimation. Point cloud data, on the other hand, can effectively reconstruct detailed environmental features, including occluded obstacle tops or backsides. Nevertheless, reconstructing scenes from point clouds demands significant computational resources and can introduce latency, posing challenges for real-time applications.

Depth images have become the most common visual representation in learning-based visual locomotion due to their simplicity and sufficient information density. However, their high dimensionality and noise levels introduce new challenges for visual processing modules, requiring robust learning architectures to handle these complexities effectively.

C. Collision Avoidance with Safe RL

Learning-based collision avoidance has seen significant advancements, particularly in the application of safe reinforcement learning (Safe RL) formulations to address the obstacle avoidance problem during motion. Safe RL [15] refers to the problem of ensuring certain safety or feasibility constraints are satisfied during the training or deployment of agents. A common approach in Safe RL is to transform safety constraints into optimization constraints, thus reformulating the problem as constrained reinforcement learning (Constrained RL). Specifically, in Constrained RL, the problem is modeled using a Constrained Markov Decision Process (CMDP).

In a standard Markov Decision Process (MDP), damages caused by actions are represented as negative rewards. In a CMDP, the mapping between states s_t, actions a_t, and damages is defined by a cost function C_i, which can represent different types of damage, analogous to the reward function. However, unlike the reward function, $C_i(s_t,a_t)$ is subject to constraints. The reward return is given by:

$$G_t = \sum_{l=0}^{\infty} \gamma^l R(s_{t+l}, a_{t+l}) \tag{1}$$

while the cost return is defined as:

$$G_t^{C_i} = \sum_{l=0}^{\infty} \gamma_{\cos t}^l C_i(s_{t+l}, a_{t+l}) \tag{2}$$

Constrained RL formulations commonly optimize the expected reward subject to cost constraints:

$$\pi^* = \arg\max_{\pi} \mathbb{E}_{\tau\sim\pi}[G_t] \quad \text{s.t.} \quad \mathbb{E}_{\tau\sim\pi}[G_t^{C_i}] \le d_i, \quad i = 1,\ldots,m. \tag{3}$$

The expected reward and cost can be represented as:

$$J = \mathbb{E}_{\tau\sim\pi}\left[\sum_{l=0}^{\infty} \gamma^l R(s_{t+l}, a_{t+l})\right] \tag{4}$$

and

$$J_{C_i} = \mathbb{E}_{\tau\sim\pi}\left[\sum_{l=0}^{\infty} \gamma_{\text{cost}}^l C_i(s_{t+l}, a_{t+l})\right], \tag{5}$$

respectively. Therefore, the constrained optimization problem can also take the following forms:

$$\pi^* = \arg\max_{\pi} J \quad \text{s.t.} \quad J_{C_i} \le d_i \tag{6}$$

or equivalently,

$$\pi^* = \arg\max_{\pi\in\Pi_C} J, \quad \Pi_C = \{\pi \in \Pi : \forall i, J_{C_i} \le d_i\} \tag{7}$$

Several solutions have been proposed for this type of problem. Constrained Policy Optimization (CPO) [16] is one popular method, which uses a surrogate optimization function to enforce safety constraints during reinforcement learning by applying a trusted region approach. In this method, the policy is updated within a region that ensures the constraint is not violated, thereby balancing performance improvement with safety guarantees. Another well-known approach is to solve the CMDP using Lagrangian methods [17]–[19]. In this approach, the constraints are relaxed using a Lagrange multiplier λ, and the optimization problem becomes:

$$\min_{\lambda\ge 0}\max_{\theta} L(\lambda,\theta) = \min_{\lambda\ge 0}\max_{\theta}\left[J_R^{\pi_\theta} - \lambda\cdot(J_C^{\pi_\theta} - \alpha)\right] \tag{8}$$

where the policy parameters θ and the Lagrange multiplier λ are updated simultaneously to obtain the best policy that satisfies the constraints.

III. METHOD

A. Avoid policy

We use PPO-Lagrangian to train end-to-end safe RL policies to avoid obstacles.

Lagrangian methods use adaptive penalty coefficients to enforce constraints. With $f(\theta)$ the objective and $g(\theta) \le 0$ the constraint, Lagrangian methods solve the equivalent unconstrained max-min optimization problem

$$\max_{\theta}\min_{\lambda\ge 0} \mathcal{L}(\theta,\lambda) \doteq f(\theta) - \lambda g(\theta) \tag{9}$$

by gradient ascent on θ and descent on λ. We combine the Lagrangian approach with PPO to obtain PPO-Lagrangian.

Here, we use the reward function as the $f(\theta)$ and cost function as the $g(\theta)$.

Our reward function is the summation of multiple terms:

$$r = r_{\text{task}} + r_{\text{regularization}} \tag{10}$$

where each term can be further divided into sub-terms as follows.

1) Task Rewards: The task rewards are:

$$T_{\text{task}} = 60\cdot r_{\text{possoft}} + 60\cdot r_{\text{postight}} + 30\cdot r_{\text{heading}} - 10\cdot r_{\text{stand}} + 10\cdot r_{\text{agile}} - 20\cdot r_{\text{stall}} \tag{11}$$

To be specific, our tracking terms (rpossoft, rpostight, rheading) are in the same form as shown below:

$$r_{\text{track (possoft/postight/heading)}} = \frac{1}{1+\left\|\frac{\text{error}}{\sigma}\right\|^2}\cdot\frac{\mathbf{1}(t>T-T_r)}{T_r} \tag{12}$$

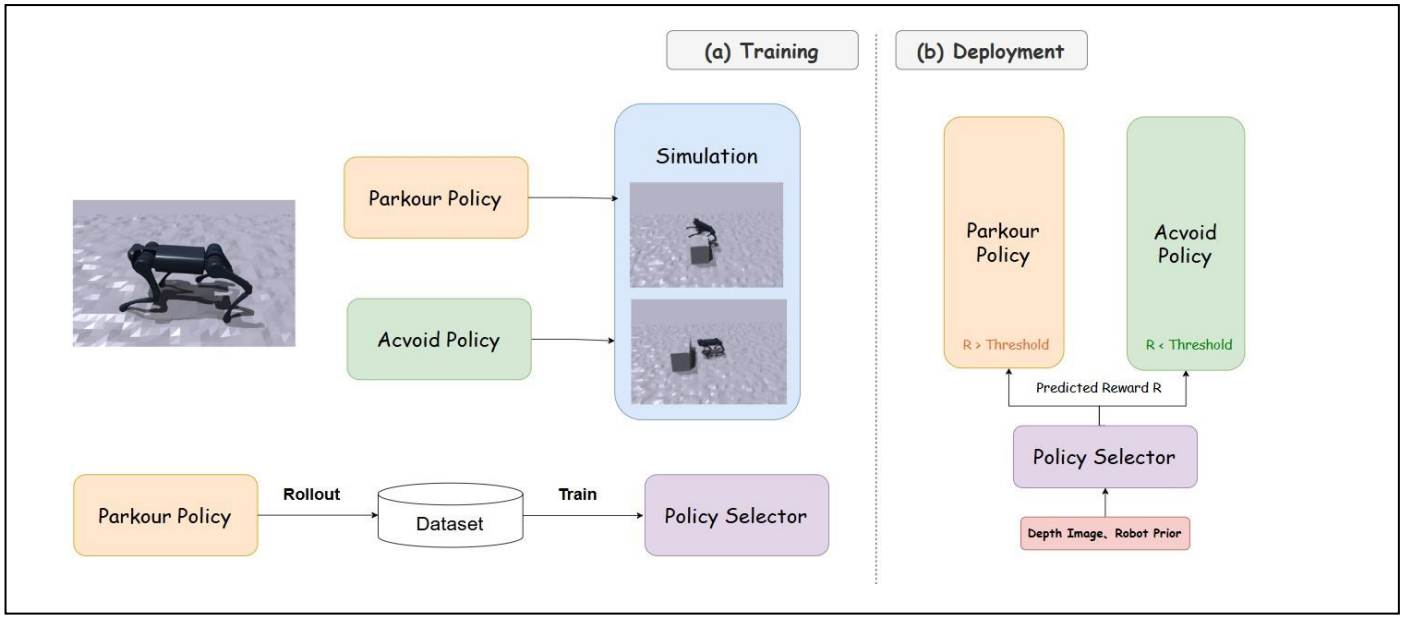

Fig. 2. Method

where σ normalizes the tracking errors, T is the episode length, and T_r is a time threshold. By doing so, the robot only needs to reach the goal before $T - T_r$ to maximize the tracking rewards, free from explicit motion constraints such as target velocities that may limit the agility.

The standing term is defined as

$$r_{\text{stand}} = |q - \bar{q}|_1 \cdot \frac{1(t>T-T_{r,\text{stand}})}{T_{r,\text{stand}}} \cdot 1\left(d_{\text{goal}} < \sigma_{\text{tight}}\right), \quad (13)$$

where q^- is the nominal joint positions for standing, $T_{r,\text{stand}} = 1s$, and d_{goal} is the distance to the goal.

The agile term is the core term that encourages the agile locomotion. It is defined as

$$r_{agile} = \max\left\{\text{ReLU}\left(\frac{v_x}{v_{max}}\right) \cdot \mathbb{1}(\text{correct direction}), \mathbb{1}\left(d_{goal} < \sigma_{tight}\right)\right\} \quad (14)$$

where v_x is the forward velocity in the robot base frame, $v_{\max} = 4.5$ m/s is an upper bound of v_x that cannot be reached (based on the hardware datasheet), and the "correct direction" means that the angle between the robot heading and the robot-goal line is smaller than 150°.

The stand term r_{stand} is 1 if the robot stays static when $d_{goal} > \sigma_{soft}$ and the robot is not in the correct direction. This term penalizes the robot for time waste.

2) Regularization Rewards: The regularization rewards are:

$$\begin{aligned} r_{regularization} = & -2 \cdot v_z^2 - 0.05 \cdot \left(\omega_x^2 + \omega_y^2\right) - 20 \\ & \cdot \left(g_x^2 + g_y^2\right) \\ & -0.0005 \cdot |\tau|_2^2 - 20 \cdot \sum_{i=1}^{12} \text{ReLU}\left(|\tau_i| - 0.85 \cdot \tau_{i,lim}\right) \\ & -0.0005 \cdot |\dot{q}|_2^2 - 20 \cdot \sum_{i=1}^{12} \text{ReLU}\left(|\dot{q}_i| - 0.9 \cdot \dot{q}_{i,lim}\right) \\ & -20 \cdot \sum_{i=1}^{12} \text{ReLU}\left(|q_i| - 0.95 \cdot q_{i,lim}\right) \\ & -2 \cdot 10^{-7} \cdot |\ddot{q}|_2^2 - 4 \cdot 10^{-6} \cdot |\ddot{a}|_2^2 - 20 \cdot I(\text{fly}) \end{aligned} \quad (15)$$

where τ is the joint torques, τ_{lim} is the hardware torque limits, $\dot{q}_{\text{lim}}$ is the hardware joint velocity limits, q_{lim} is the hardware joint position limits, and "fly" refers to when the robot has no contact with the ground. We penalize the "fly" cases as they make the robot base uncontrollable, threatening the system's safety.

Our cost function is the summation of multiple terms:

$$c = c_{\text{collision}} + c_{\text{feet\_collision}} + c_{\text{termination}} \quad (16)$$

B. Parkour policy

We use a parkour policy proposed by [3]. Specifically, we utilize Regularized Online Adaptation (ROA) for policy adaptation and employ a two-phase training approach for the vision backbone.

In Phase 1, we leverage reinforcement learning (RL) to train a locomotion policy that has access to privileged information,

such as environment parameters and scandots, in addition to the heading direction provided by waypoints. The training process integrates ROA to estimate and recover environmental information from the history of observations.

In Phase 2, we distill the knowledge from scandots into a policy that operates solely from onboard depth images. This policy autonomously determines its heading (yaw) direction, conditioned on the surrounding obstacles, enabling the robot to navigate effectively in real-world scenarios.

C. Policy Selector

To enable the robot to adaptively select between the parkour and avoidance policies, we propose a policy selector framed as a supervised learning problem. This selector leverages future rewards as a representation of terrain difficulty, enabling informed decision-making based on the robot's environment.

In this approach, the model learns to map sequences of past depth observations to predictions of future average rewards over a predefined horizon. The reward prediction function is expressed as:

$$\hat{r}_{avg}(s_t) = f(\{s_{t-n},\ldots,s_t\};\theta_f) \tag{17}$$

where:

- $f(\{s_{t-n},\ldots,s_t\};\theta_f)$ is the reward prediction function parameterized by θ_f, which outputs the predicted average reward for the future m steps based on the past n depth observations.
- $\{s_{t-n},\ldots,s_t\}$ is the sequence of latent representations (feature embeddings) of the depth images over the past n time steps.

The selector applies the parkour policy if the predicted future reward exceeds a predefined threshold, indicating that the terrain is navigable for parkour. Otherwise, the avoidance policy is employed. The decision rule is formalized as:

$$\pi_{selected}(s_t) = \begin{cases} \pi_{parkour} & \text{if } \hat{r}_{avg}(s_t) > \theta \\ \pi_{avoid} & \text{if } \hat{r}_{avg}(s_t) \le \theta \end{cases} \tag{18}$$

where:

- $\pi_{selected}(s_t)$ is the selected policy at time t based on the state s_t.
- $\hat{r}_{avg}(s_t)$ is the predicted average reward for the future m steps, based on the sequence of latent representations $\{s_{t-n},\ldots,s_t\}$.
- θ is a predefined reward threshold.

The training data is generated through continuous rollouts of the pre-trained parkour policy within a simulation environment. During each rollout, a sequence of depth images is captured, and their corresponding latent representations (i.e., feature embeddings) are extracted to form a temporal dataset. Concurrently, the reward signals are recorded for future m time steps and averaged to create the target outputs. The resulting dataset consists of sequences of depth latent representations and their associated future average rewards:

$$\mathcal{D} = \{(\{s_{t-n},\ldots,s_t\}, \bar{r}_{t+1:t+m})\}_{t=n}^{T} \tag{19}$$

where:

- D represents the dataset consisting of pairs of input sequences $\{s_{t-n},\ldots,s_t\}$ and the corresponding future average rewards $\bar{r}_{t+1:t+m}$.
- $\bar{r}_{t+1:t+m}$ is the average reward over the future m steps: t $\bar{r}_{t+1:t+m} = \frac{1}{m}\sum_{i=1}^{m} r_{t+i}$.
- T is the total number of time steps in each rollout.

To capture the temporal dynamics inherent in reinforcement learning tasks, we explore two neural network architectures:

1) Multi-Layer Perceptron (MLP): Processes concatenated latent representations from the past n steps to predict the future average reward.
2) Long Short-Term Memory (LSTM) network: Incorporates temporal dependencies by sequentially processing latent representations, enabling more nuanced modeling of temporal patterns.

By using the predicted future rewards as a proxy for terrain difficulty, the policy selector ensures adaptive and efficient navigation, dynamically choosing the appropriate policy based on environmental conditions.

D. Unified Simulation and Input Standardization

In this work, we propose two approaches to help unify the policy setup and improve the robot's performance across varying conditions.

First, we unify the simulation environments and configurations of two distinct policies by modifying key environmental factors, such as the size, shape, and arrangement of obstacles, the distance between robots, obstacles, and goals, as well as the type and movement speed of the robots. This ensures that both policies operate under consistent conditions, facilitating the system's ability to generalize across a broader range of scenarios. This unified approach allows the robot to develop more robust strategies for navigating dynamic and diverse environments, leading to a significant improvement in its overall performance.

Second, to reduce computational overhead, the Parkour policy uses Ray as the model input, where Ray represents the distance between the robot and obstacles, offering a more intuitive and effective format compared to depth images. To achieve this, we trained an additional Depth2Ray network, based on a residual neural network architecture, to convert depth images into Ray format. This standardization of input improves both efficiency and consistency.

Together, these two approaches contribute to a more unified and efficient system, enhancing the robot's adaptability and performance across a range of scenarios.

IV. Experiment

Our system aims to enable robots to traverse complex terrains safely, efficiently, and gracefully by combining parkour skills and obstacle avoidance capabilities. To evaluate our approach, we designed experiments to address the following key questions:

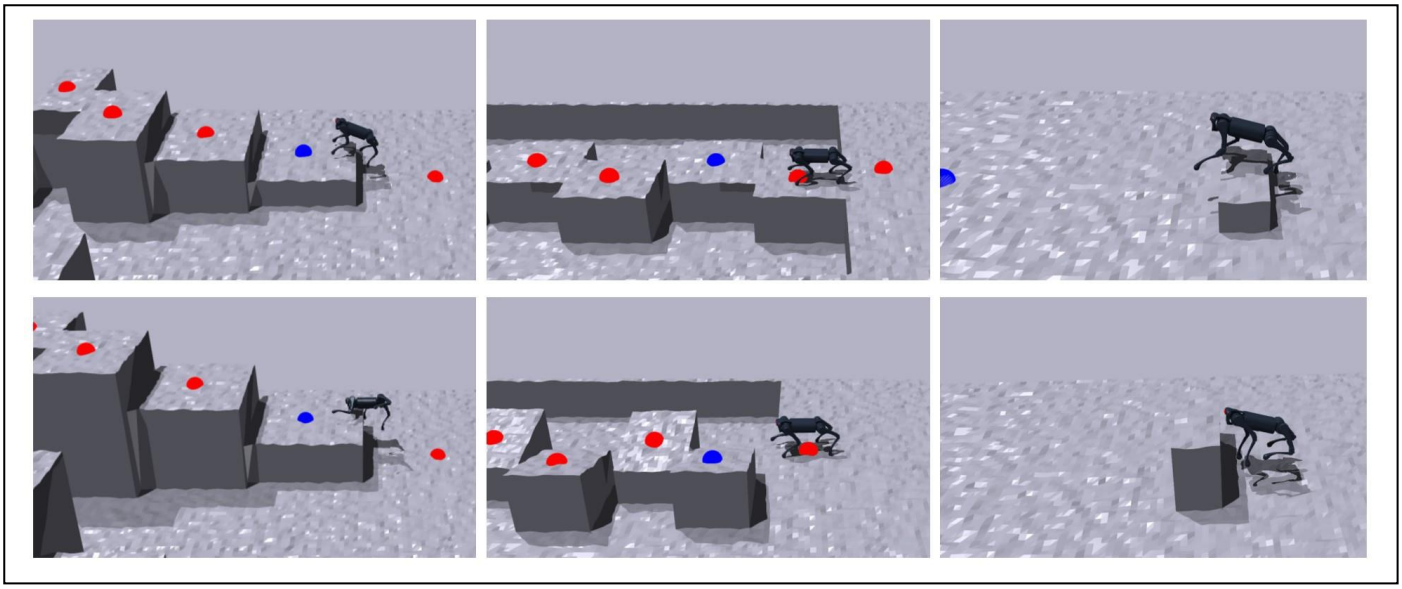

Fig. 3. Examples of our terrains and obstacles: The first row are relatively simple and safe cases. The second row are difficult cases for the parkour policy.

- Q1: Can our method enhance the robot's safety and agility when navigating complex terrains?
- Q2: How do different predictor architectures and the choice of history length influence task success rates?
- Q3: How does the predictor extract effective information from perception inputs to estimate obstacle difficulty?
- What kind of perception input best encapsulates the information required for this task?

A. Experiment Setup

Simulator: We use the GPU-based Isaac Gym simulator which supports us to train both the parkour and avoid policy in the same environment setup.

Robot: We use the Unitree A1 robot with 12 joints. Each leg is equipped with three joints: hip, knee, and ankle joints. When standing, height of the thigh joint is 26cm and body length is 40cm.

Terrains and obstacles: We introduce raised, square-shaped obstacles specifically designed to challenge the robot's jumping and navigation abilities. The obstacles have a length of 0.4 meters and heights ranging from 0.6 to 1.0 meters. They are distributed along the x-axis within the range of [3.0, 5.2] meters and vary in width between 0.4 and 0.8 meters. The terrain is flat with a pad height of 0, emphasizing the raised nature of the hurdles. Examples of our obstacles are shown in Fig 3.

Baselines: For the experimental results, we consider three settings:

- Our Policy: This includes the parkour policy, the avoid policy, and a policy selector that dynamically chooses between these two policies.
- Parkour Policy Only: This baseline uses only the parkour policy.
- Avoid Policy Only: This baseline uses only the avoid policy.

With the help of the policy selector, we expect Our Policy to outperform the other two baselines, as it should combine the strengths of both: it will be as agile as single avoid policy in safe scenarios, while maintaining the speed and efficiency of single parkour policy in more cautious situations.

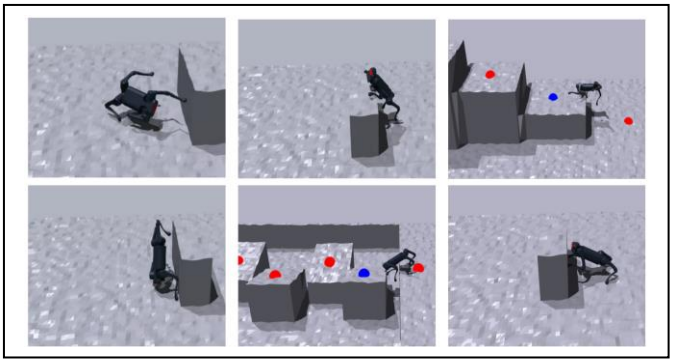

Fig. 4. Examples of parkour fail cases

B. Performance Analysis

We tested our model and the baselines on each track introduced above. Success rates (defined as the absence of falls or prolonged getting-stuck incidents leading to timeouts) and average completion times (for successful attempts) were recorded for comparison, as is shown in Table 1.

TABLE I
PERFORMANCE COMPARISON

Method	Success Rate	Avg. Completion Time (s)
Ours	92%	93.2
Parkour Policy Only	78%	87.8
Avoid Policy Only	91%	105.6

The results show that our method outperforms the baseline approaches in both success rate and average completion time. While the parkour-only policy is faster, it has a lower success rate, and the avoidance-only policy, though more reliable, takes longer to complete. By combining both policies, our approach achieves a strong balance between safety and efficiency, demonstrating its effectiveness in navigating complex environments. This hybrid strategy enables higher success rates without significantly sacrificing speed, making it a robust solution for legged robot navigation.

C. Predictor and Perception Input Evaluation

The predictor module is crucial for the overall system as it determines whether the robot can effectively select different strategies when facing various terrains and obstacles. To study the impact of predictor design on task success rates (addressing Q2 and Q3), we evaluated three different architectures, different history and future length, and different perception input types.

Model Architectures: The MLP consists of three layers with hidden sizes of 128 and 64, while the LSTM has a hidden size of 64. Both MLP and LSTM include a dropout rate of 0.2 to prevent overfitting. Training parameters include a learning rate of 1e-3 with a weight decay of 1e-4. The loss function is MSE-Loss, which computes the difference between the predicted reward and the label, and the optimizer used is Adam.

Future Reward Prediction: We use the average reward obtained over the next 100 steps to represent the difficulty of the current obstacle. Each model takes perception information from the past few steps as input and predicts the average reward for the next 100 steps. To explore the optimal history length, we tested history lengths of 50, 20, and 10 steps. Longer histories potentially provide more useful information but may also introduce noise, making the model harder to fit.

Perception Input Types: For input perception, directly using raw depth images provides abundant information but requires suitable encoders. Alternatively, intermediate representations used by the parkour and avoidance policies can be employed. Specifically:

- Depth Latent: Encoded by a depth encoder from the parkour policy to form a latent vector.
- 2D Ray: Derived from the avoidance policy, representing ray lengths from the robot's perspective in various directions (as shown in Fig 5). This is privileged information available only in simulators.
- Raw depth images: A ResNet18-based model trained to predict 2D rays from depth images to replace privileged information in real-world scenarios.

TABLE II
PERFORMANCE COMPARISON ACROSS MODELS, HISTORY LENGTHS, AND INPUT TYPES

Model	History Length	Depth Latent		2D Ray		Depth Image	
		Success (%)	Avg. Time (s)	Success (%)	Avg. Time (s)	Success (%)	Avg. Time (s)
MLP	50	90	92.4	85	96.3	80	98.1
	20	60	95.6	65	97.4	60	99.2
	10	80	85.7	75	89.0	70	91.5
LSTM	50	95	80.5	90	85.0	85	87.6
	20	75	78.4	80	79.5	75	81.3
	10	80	83.5	80	86.7	75	89.2

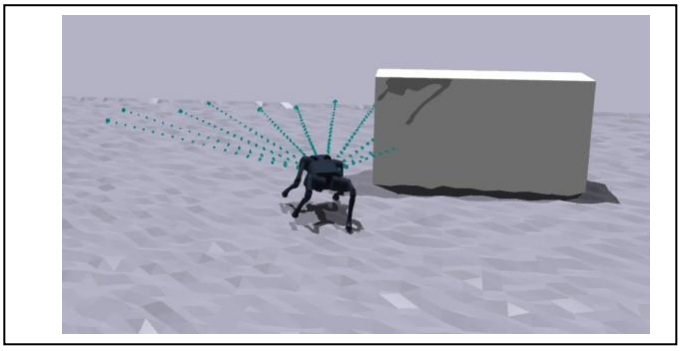

Fig. 5. Examples of 2D ray

The results across all settings are shown in Table 2. The performance analysis across models, history lengths, and input types reveals three key insights: First, LSTM consistently outperforms MLP in terms of success rate and efficiency, especially with longer history lengths and depth latent inputs. Second, longer history lengths generally improve performance, enhancing both success rates and computational efficiency for both models. Finally, Depth Latent inputs consistently yield the highest success rates across both models, while Depth Image inputs tend to be more computationally demanding, resulting in longer average times and lower success rates.

V. CONCLUSION

In conclusion, our approach introduces a novel framework that effectively integrates parkour and obstacle-avoidance policies to enhance the navigation capabilities of legged robots in complex environments. By leveraging depth images and a

data-driven method to estimate environmental difficulty, we enable the robot to adaptively switch between policies, ensuring both safety and efficiency. The results from our simulation experiments validate the effectiveness of this approach, demonstrating the robot's ability to navigate challenging terrains more effectively than traditional methods. Our work represents a significant step toward more adaptable and robust legged robotic systems, with potential applications in real-world navigation tasks where both efficiency and safety are critical.

VI. Acknowledgment

We would like to convey our warmest gratitude to teachers Yonglu Li and Cewu Lu for their guidance during our research on this topic. We also appreciate Zhengbang Zhu, Yichao Zhong and Jiahang Cao for their advice on our work.

References

[1] 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.

[2] D. Hoeller, N. Rudin, D. Sako, and M. Hutter, "Anymal parkour: Learning agile navigation for quadrupedal robots," 2023. [Online]. Available: https://arxiv.org/abs/2306.14874

[3] X. Cheng, K. Shi, A. Agarwal, and D. Pathak, "Extreme parkour with legged robots," 2023. [Online]. Available: https://arxiv.org/abs/2309.14341

[4] E. Chane-Sane, J. Amigo, T. Flayols, L. Righetti, and N. Mansard, "Soloparkour: Constrained reinforcement learning for visual locomotion from privileged experience," 2024. [Online]. Available: https://arxiv.org/abs/2409.13678

[5] S. Luo, S. Li, R. Yu, Z. Wang, J. Wu, and Q. Zhu, "Pie: Parkour with implicit-explicit learning framework for legged robots," 2024. [Online]. Available: https://arxiv.org/abs/2408.13740

[6] T. Miki, L. Wellhausen, R. Grandia, F. Jenelten, T. Homberger, and M. Hutter, "Elevation mapping for locomotion and navigation using gpu," 2022. [Online]. Available: https://arxiv.org/abs/2204.12876

[7] P. Fankhauser, M. Bloesch, and M. Hutter, "Probabilistic terrain mapping for mobile robots with uncertain localization," *IEEE Robotics and Automation Letters*, vol. 3, no. 4, pp. 3019–3026, 2018.

[8] R. O. Chavez-Garcia, J. Guzzi, L. M. Gambardella, and A. Giusti, "Learning ground traversability from simulations," *IEEE Robotics and Automation Letters*, vol. 3, no. 3, pp. 1695–1702, 2018.

[9] R. Grandia, F. Jenelten, S. Yang, F. Farshidian, and M. Hutter, "Perceptive locomotion through nonlinear model predictive control," 2022. [Online]. Available: https://arxiv.org/abs/2208.08373

[10] T. Miki, J. Lee, J. Hwangbo, L. Wellhausen, V. Koltun, and M. Hutter, "Learning robust perceptive locomotion for quadrupedal robots in the wild," *Science Robotics*, vol. 7, no. 62, Jan. 2022. [Online]. Available: http://dx.doi.org/10.1126/scirobotics.abk2822

[11] D. Jain, A. Iscen, and K. Caluwaerts, "From pixels to legs: Hierarchical learning of quadruped locomotion," 2020. [Online]. Available: https://arxiv.org/abs/2011.11722

[12] A. Agarwal, A. Kumar, J. Malik, and D. Pathak, "Legged locomotion in challenging terrains using egocentric vision," 2022. [Online]. Available: https://arxiv.org/abs/2211.07638

[13] G. B. Margolis, T. Chen, K. Paigwar, X. Fu, D. Kim, S. Kim, and P. Agrawal, "Learning to jump from pixels," 2021. [Online]. Available: https://arxiv.org/abs/2110.15344

[14] W. Yu, D. Jain, A. Escontrela, A. Iscen, P. Xu, E. Coumans, S. Ha, J. Tan, and T. Zhang, "Visual-locomotion: Learning to walk on complex terrains with vision," in *Conference on Robot Learning*, 2021. [Online]. Available: https://api.semanticscholar.org/CorpusID:237263152

[15] J. Garc´ıa and F. Fernandez, "A comprehensive survey on safe reinforce-´ ment learning," *J. Mach. Learn. Res.*, vol. 16, no. 1, p. 1437–1480, Jan. 2015.

[16] J. Achiam, D. Held, A. Tamar, and P. Abbeel, "Constrained policy optimization," 2017. [Online]. Available: https://arxiv.org/abs/1705.10528

[17] C. Tessler, D. J. Mankowitz, and S. Mannor, "Reward constrained policy optimization," 2018. [Online]. Available: https://arxiv.org/abs/1805.11074

[18] Q. Liang, F. Que, and E. H. Modiano, "Accelerated primal-dual policy optimization for safe reinforcement learning," *CoRR*, vol. abs/1802.06480, 2018. [Online]. Available: http://arxiv.org/abs/1802.06480

[19] S. Bhatnagar and K. Lakshmanan, "An online actor—critic algorithm with function approximation for constrained markov decision processes," *J. Optim. Theory Appl.*, vol. 153, no. 3, p. 688–708, Jun. 2012. [Online]. Available: https://doi.org/10.1007/s10957-012-9989-5