Agile Continuous Jumping in Discontinuous Terrains

Yuxiang Yang[1], Guanya Shi[2], Changyi Lin[2], Xiangyun Meng[1], Rosario Scalise[1], Mateo Guaman Castro[1],
Wenhao Yu[3], Tingnan Zhang[3], Ding Zhao[2], Jie Tan[3], and Byron Boots[1]
[1]University of Washington [2]Carnegie Mellon University [3]Google Deepmind

Abstract—We focus on agile, continuous, and terrain-adaptive jumping of quadrupedal robots in discontinuous terrains such as stairs and stepping stones. Unlike single-step jumping, continuous jumping requires accurately executing highly dynamic motions over long horizons, which is challenging for existing approaches. To accomplish this task, we design a hierarchical learning and control framework, which consists of a learned heightmap predictor for robust terrain perception, a reinforcement-learning-based centroidal-level motion policy for versatile and terrain-adaptive planning, and a low-level model-based leg controller for accurate motion tracking. In addition, we minimize the sim-to-real gap by accurately modeling the hardware characteristics. Our framework enables a Unitree Go1 robot to perform agile and continuous jumps on human-sized stairs and sparse stepping stones, for the first time to the best of our knowledge. In particular, the robot can cross two stair steps in each jump and completes a 3.5m long, 2.8m high, 14-step staircase in 4.5 seconds. Moreover, the same policy outperforms baselines in various other parkour tasks, such as jumping over single horizontal or vertical discontinuities. Experiment videos can be found at **https://yxyang.github.io/jumping_cod/**.

I. INTRODUCTION

Achieving animal-level agility has long been a coveted goal in legged locomotion research. As a notable example, quadrupedal animals can traverse challenging terrains at high speeds with continuous jumping and precise foot placement. Such jumping is particularly effective on discontinuous terrains like stairs and stepping stones, which is often non-traversable using the standard walking gait. Inspired by this observation, researchers have made significant efforts to reproduce this agile jumping behavior in quadrupedal robots, including building high-performance robot platforms [1, 2, 3, 4], designing long-distance jumping controllers [5, 6, 7, 8], and learning rapid hand-eye coordination [9, 10, 11, 12]. However, achieving *continuous*, *long-distance*, and *terrain-adaptive* jumping remains a challenging task.

While continuous, terrain-aware jumping has been studied in simulation [13, 14], transferring this behavior to the real world remains difficult, and reflects many fundamental challenges in robot agility. The first challenge is *perception*. Despite recent progress, end-to-end learning-based methods [15, 9, 10, 12] still face a large sim-to-real gap, especially for high-speed motions with significant camera oscillations. The second challenge is *motion accuracy*. Unlike single-step jumping, continuous jumping requires the robot to link
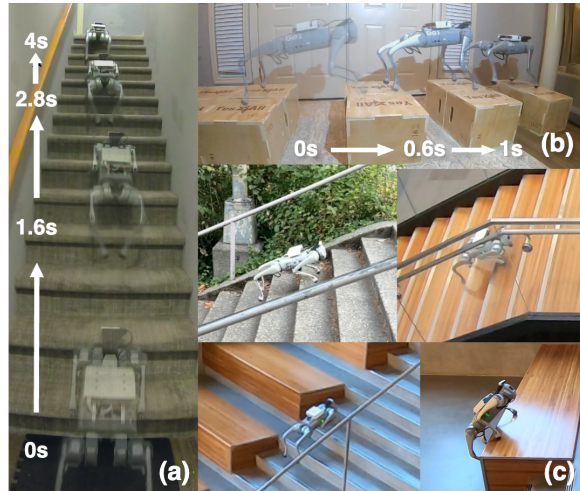
Fig. 1: Our framework enables a quadrupedal robot to jump continuously over real-world stairs and steps.

consecutive jumps with carefully planned body and foot motions, and accurately track these poses against unexpected perturbations. Despite recent results in high-performance single-step jumping [9, 10, 12], most of these frameworks usually cannot maintain an accurate landing pose, making it difficult to extend from single-step to continuous jumping. The last challenge is *simulation fidelity*. While domain randomization (DR) [16, 17] can bridge small dynamics mismatches between simulation and the real world, it often falls short when the robot operates close to the hardware limit, which is common in agile, continuous jumping. Because of these challenges, achieving high-speed jumping on real-world discontinuous terrains remains a difficult task.

In this work, we present a hierarchical framework [18] for agile, continuous, and terrain-aware jumping that addresses all these challenges. Our framework consists of a learned heightmap predictor for terrain perception, a reinforcement learning (RL)-trained motion policy for motion planning, and a model-based leg controller for motion tracking. Unlike end-to-end, pixel-to-action approaches [9, 10], we choose the sagittal-plane heightmap as the intermediate representation, which improves the robustness and interpretability of terrain perception. As the robot jumps forward, the heightmap predictor reconstructs this heightmap from onboard depth images. Based on this heightmap, the motion policy plans body and foot motions, which is tracked by the leg controller. To ensure consistent landing between jumps, we train the motion policy with an extra reward for accurate motion plan-

ning, and use a mixture of feedforward and feedback terms in the leg controller for robust motion tracking. To improve simulation fidelity, we identify and reproduce key hardware characteristics in simulation, including camera delay and motor saturations. Combining these efforts, our framework achieves animal-like agile continuous jumping on stairs and stepping stones for the first time.

We conduct extensive real-robot experiments on a Unitree Go1 robot [4] to validate the effectiveness of our framework (Fig. 1). With the continuous jumping gait, our robot jumps over a human-sized, 14-step staircase in less than 4.5 seconds, traversing 2 stair steps in each jump with an average horizontal speed of 0.8m/s and vertical speed of 0.6m/s. This speed is more than 2 times faster than the standard walking gait (Table. I). To the best of our knowledge, this is the first time any quadrupedal robot has achieved such high-speed animal-like traversal on similar staircases. In addition to stair jumping, our framework also learns to jump continuously on horizontal stepping stones, and achieves state-of-the-art performance in single-step jumps up to 80cm ($2\times$ body length) horizontally and 60cm ($2.4\times$ body height) vertically [10, 9]. We further conduct an ablation study to validate important design choices.

In summary, our contributions in this paper are as follows:

1) We present a hierarchical framework for quadrupedal jumping with direct perception input.
2) We develop a robust training pipeline that learns versatile jumping skills with minimal sim-to-real gap.
3) Our framework achieves agile continuous jumps in real-world discontinuous terrains for the first time.

II. RELATED WORK

A. Optimal Control for Dynamic Legged Locomotion

Researchers have had a long history building optimal-control based controllers for dynamic quadrupedal locomotion [19, 20, 21, 22, 23]. By optimizing for control outputs at high frequency, these controllers can accurately track the reference trajectory, even for high-speed motions like running [19], galloping [20] or jumping [20, 24, 25]. However, due to the constraint of real-time control, these frameworks are typically confined to a short plan horizon, and require additional effort in offline trajectory optimization [7, 8, 26, 6], dynamics model simplification [5, 27, 28], and optimal control relaxation [28] to plan for complex, long-horizon jumps. Another bottleneck of these frameworks is perception, which not only requires extra computation resource to process the perception signals, but also increases the complexity of the optimal control problem, making it more difficult to solve in real-time [29, 30]. Using manually-designed terrain perception and motion adaptation, Park et al. [31] achieved bounding over fixed-shape hurdles on an MIT Cheetah 2 Robot. More recently, by classifying step feasibility and formulating it into foot placement optimization, Grandia et al. [32] achieved low-speed walking on uneven stepping stones on an Anymal robot. In this work, we use an optimal controller for robust low-level motion tracking, but introduce learning-based perception and motion planning for real-time replanning of complex jumping motions.

B. Learning Perceptive Locomotion

Recently, learning-based approaches emerge as a promising alternative for dynamic, terrain-adaptive quadrupedal locomotion [33, 34, 17, 35, 10, 9, 36, 37]. The core idea is to construct an end-to-end policy that directly maps from perceptual and proprioceptive inputs to motor actions, and train the policy using reinforcement learning, usually in simulation [38]. These approaches have enabled legged robots to walk [15] and jump [10, 9, 12, 11] dynamically on challenging terrains such as mountain trails [39] and stepping stones [40]. Despite their success, these frameworks tend to be less robust on highly dynamic tasks, and face even more challenges on continuous jumping, which requires accurate motion planning and tracking. Unlike these end-to-end approaches, we decompose our pipeline into perception, motion planning, and motion tracking, with interpretable intermediate outputs and robust real-world jumping performance.

C. Hierarchical Frameworks

Hierarchical frameworks offer a promising alternative that combines the benefit of learning-based and optimal-control-based controllers. By combining a high-level learned policy with a low-level motor controller [41, 42, 43, 44, 18, 45, 46], these frameworks can learn robust and generalizable locomotion behaviors with versatile gait selection [42, 44], foot placement [43, 18, 41] and body motion planning [43, 18]. In our prior work [18], we addressed the computational bottleneck of hierarchical frameworks and achieved continuous forward jumping on flat terrains. In this work, we extend the prior framework with perception, and achieve terrain-adaptive jumping in both forward and upward directions.

III. HIERARCHICAL PERCEPTIVE JUMPING FRAMEWORK

To achieve high-speed, terrain-aware, continuous jumping on the real robot, we design a hierarchical learning-control framework with separate modules for perception, motion planning, and motion tracking (Fig. 2). At the perception level, a *heightmap predictor* receives depth images from the onboard depth camera, and estimates a heightmap in the front-back axis of the robot. Based on this heightmap, a *motion policy* plans the reference robot motions, including the gait frequency f, the swing foot residuals $\boldsymbol{p}_r$, and the desired body velocity v_x, v_z, v_θ. This reference motion is then tracked by a low-level leg controller, which runs a gait generator to determine the desired contact state of each leg, and uses separate control strategies for swing and stance legs. We run the motion policy and heightmap predictor at 100Hz for stable policy output, and run the leg controller at 500Hz for responsive torque control.

To train our framework, we first constructed a high-fidelity simulation environment with carefully identified hardware characteristics. Due to the high computation cost in depth image rendering, we train our framework in two stages [15, 9, 10]. In the first stage, we train the motion policy
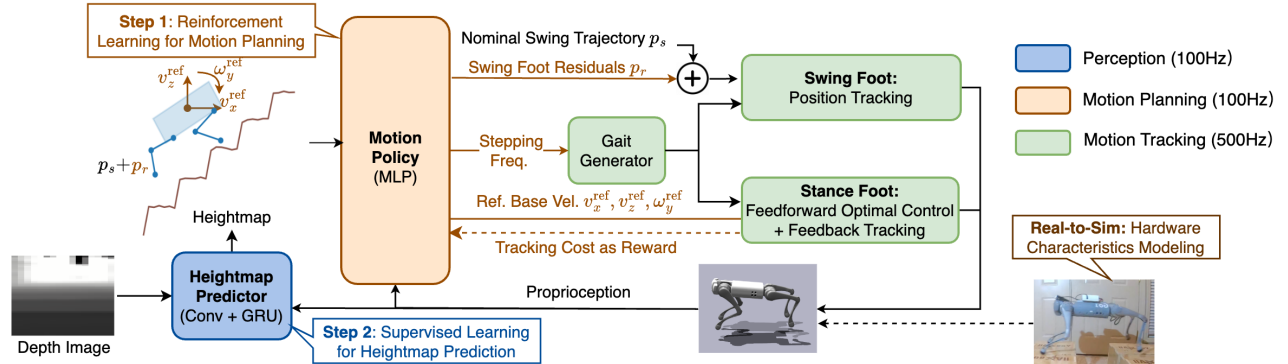
Fig. 2: Our hierarchical learning-control framework consists of a heightmap predictor, a motion policy, and a low-level leg controller. We use the heightmap as the intermediate representation for perception and motion planning (Section. IV), train high-performance motion planning with reward to encourage accurate tracking (Section. V), and combine a feedforward and a feedback controller for robust tracking of body orientations (Section. VI). In addition, we reduce the sim-to-real gap by accurately identifying key hardware characteristics and reproducing them in simulation (Section. VII).

using reinforcement learning (RL), where the motion policy receives heightmaps that are directly sampled from the environment. In the second stage, we train the heightmap predictor to estimate this heightmap from depth images. Once the framework is trained, we deploy it to the real robot without additional fine-tuning.

IV. HEIGHTMAP-CENTERED TERRAIN PERCEPTION

A robust and interpretable terrain understanding is critical for continuous jumping in noisy real-world environments. In our framework, we adopt the *heightmap* as the intermediate representation between raw sensor data and motion planning. The heightmap predictor estimates this heightmap from depth images, while the motion policy uses it to plan body and foot motions. This approach enhances the framework's interpretability and enables easy error isolation. Moreover, by randomizing the heightmap during training, we improve the motion policy's robustness against real-world uncertainties.

A. Iterative Training for Heightmap Prediction

We design the heightmap predictor as a 3-layer convolutional network followed by a 1-layer GRU, where the convolutional network extracts features from the depth images, and the GRU reconstructs the heightmap from past memories. To ensure accurate heightmap reconstruction around the trajectories that the robot is more likely to visit, we train the heightmap predictor iteratively using supervised learning, similar to DAgger [47]. The training loop alternates between rolling out the trajectories using the latest heightmap predictor and the motion policy, and training the heightmap predictor on the trajectories collected.

B. Robust Perception with Heightmap Randomization

Due to camera occlusion and real-world sensor noises, it can be difficult to accurately reconstruct the ground-truth heightmap at all time. To make the motion policy aware of this insufficiency, we randomly shift the observed heightmap

during policy training ([-8cm, 8cm] horizontally, and [-5cm, 5cm] vertically). This randomization improves the robustness of the motion policy and prevents it from overfitting to irrelevant details of the heightmap. More importantly, it creates tolerance for inevitable heightmap reconstruction errors during real-world operation.

V. LEARNING VERSATILE MOTION PLANNING

To learn versatile jumping skills in challenging terrains, we train the motion policy using reinforcement learning, where the environment consists of diverse terrains in a curriculum of increasing difficulty. In addition, to ensure the feasibility of planned motions, we include the motion tracking cost as part of the reward during policy training.

A. Environment Setup

In our environment, the state space includes robot proprioception, gait timing, and the randomized terrain heightmap. The action space includes the stepping frequency f for gait progress, the swing foot residual $\boldsymbol{p}_r$, and the reference base velocity v_x^{ref}, v_z^{ref}, ω_y^{ref}. We implement the environment and the low-level controller on GPU, and solve an approximated version of the optimal control problem [18] to speed up training. During real-world deployment, we solve the exact optimal control problem for extra robustness.

B. Motion Tracking Cost as Policy Reward

Note that not all motion references (v_x^{ref}, v_z^{ref}, ω_y^{ref}) can be accurately tracked by the leg controller. For example, when only the front leg is in contact, the robot cannot track a large v_z^{ref} and a large ω_y^{ref} at the same time, as the prior requires jumping up with extended legs, and the later requires tilting down with retracted legs. When facing such a conflicting objective, the leg controller typically computes an intermediate solution with large tracking errors. Moreover, this solution can be highly sensitive to the current robot state, leading to large sim-to-real gaps and unpredictable
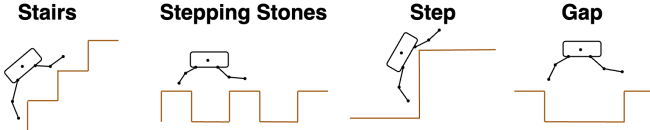
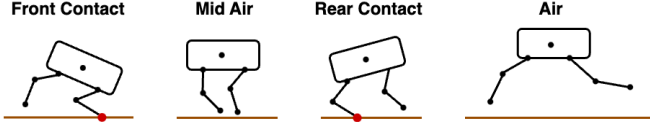Fig. 3: The environment consists of 4 terrain types.

Fig. 4: We adopt the bounding gait with 4 contact modes. Red dot indicates foot contact.

Fig. 5: The feedback controller improves the tracking accuracy of body angular velocities.

real-world failures. To alleviate this issue, we include the cost of the low-level optimal controler (Eq. 1) as part of the reward during policy training, so that the policy learns to plan feasible trajectories with small tracking errors. The rest of the reward terms were adapted from our prior work [18], which encourages the robot to follow the contact schedule and jump over long distances.

C. Terrain Curriculum for Skill Training

To learn continuous jumping over complex terrains, we design 4 terrain types in the training environment (Fig. 3) with single and multiple discontinuities. Each terrain comes with a curriculum of increasing difficulty. The robot starts randomly in the easiest terrain at the beginning of training, and advances to the next level after it learns to jump over a sufficiently long distance in the current level [48].

VI. ACCURATE MOTION TRACKING WITH FEEDBACK

The leg controller tracks the motion references from the motion policy with separate control strategies for swing and stance legs. The switch between swing and stance legs is modulated by a gait generator. For accurate tracking of the body orientation changes, we combine a feedforward and a feedback controller for stance legs.

A. Gait Generation and Swing Leg Control

To maximize the jumping distance, the gait generator adopts the bounding gait with alternating foot contacts (Fig. 4). The gait generator tracks the leg's progress in each bounding cycle with a phase variable $\phi \in [0, 2\pi)$, where different values of ϕ correspond to different contact modes. Given the stepping frequency f from the motion policy, the gait generator advances the phase ϕ by $2\pi f$ and uses the newly computed phase to determine swing and stance legs. The swing leg controller receives the reference foot positions as the sum of a nominal trajectory p_s from the Raibert heuristics [49], and a learned residual p_r, converts the foot position to joint position using inverse kinematics (IK), and tracks this joint position using PD control.

B. Feed-forward Optimal Control

Given the reference base velocities v_x^{ref}, v_z^{ref}, ω_y^{ref}, the feed-forward controller computes a reference CoM acceleration $a^{\mathrm{ref}} \in \mathbb{R}^6$ via PD rule, and solves a quadratic program (QP) to find the ground reaction forces (GRFs) $f \in \mathbb{R}^{12}$. We set
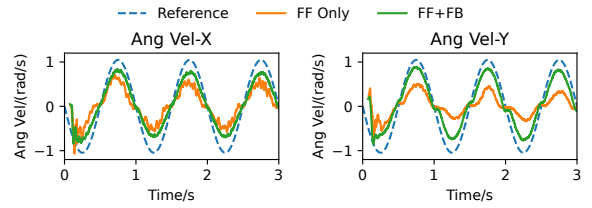
the objective to track the reference acceleration a^{ref} with an additional regularization term:

$$\min_f \|a - a^{\mathrm{ref}}\|_{\mathbf{U}} + \|f\|_{\mathbf{V}} \tag{1}$$

where we compute the acceleration a via the linearized centroidal dynamics model [19], and set $\mathbf{U}$ and $\mathbf{V}$ as diagonal weight matrices. The constraints include the contact schedule and friction cone constraints. Once the GRFs are optimized, the controller then convert them into motor torques via Jacobian transpose: $\tau = \mathbf{J}^\top f$.

C. Feedback Velocity Tracking

While the feedforward (FF) controller can accurately track trajectories with upright body poses, we find that it cannot accurately track trajectories with large body orientation changes (Fig. 5) due to unmodeled details in robot orientation dynamics. For accurate trajectory tracking under all body poses, we introduce a feedback (FB) controller [20, 22] to compensate for this tracking error. This FB controller computes the reference foot *velocities* from the reference body velocities, and tracks it using joint PD control. Note that the world frame position of a foot $p_{\mathrm{foot}}^{\mathrm{W}}$ can be expressed in terms of its body frame position $p_{\mathrm{foot}}^{\mathrm{B}}$ as:

$$p_{\mathrm{foot}}^{\mathrm{W}} = p_{\mathrm{B}}^{\mathrm{W}} + \mathbf{R}_{\mathrm{B}}^{\mathrm{W}} p_{\mathrm{foot}}^{\mathrm{B}} \tag{2}$$

where $p_{\mathrm{B}}^{\mathrm{W}}$ is the body position in the world frame, and $\mathbf{R}_{\mathrm{B}}^{\mathrm{W}}$ is the body rotation matrix. Taking the derivative on both sides, we can express the world-frame foot velocity $v_{\mathrm{foot}}^{\mathrm{W}}$ in terms of the body-frame foot velocity $v_{\mathrm{foot}}^{\mathrm{B}}$:

$$v_{\mathrm{foot}}^{\mathrm{W}} = v_{\mathrm{B}}^{\mathrm{W}} + \omega_{\mathrm{B}}^{\mathrm{W}} \times p_{\mathrm{foot}}^{\mathrm{B}} + \mathbf{R}_{\mathrm{B}}^{\mathrm{W}} v_{\mathrm{foot}}^{\mathrm{B}} \tag{3}$$

where $v_{\mathrm{B}}^{\mathrm{W}}, \omega_{\mathrm{B}}^{\mathrm{W}}$ is the body linear and angular velocity in the world frame. Assuming static foot contacts $\left(v_{\mathrm{foot}}^{\mathrm{W}} = 0\right)$ and that the body is moving at reference velocities, we can express the reference body-frame foot velocities $v_{\mathrm{foot}}^{\mathrm{B}}$ in terms of the reference body velocities.

$$v_{\mathrm{foot}}^{\mathrm{B}} = -\left(\mathbf{R}_{\mathrm{B}}^{\mathrm{W}}\right)^{-1} \left(v^{\mathrm{ref}} + \omega^{\mathrm{ref}} \times p_{\mathrm{foot}}^{\mathrm{B}}\right) \tag{4}$$

We then convert the reference foot velocities to the reference joint velocities via Jacobian inverse $\dot{q}^{\mathrm{ref}} = \mathbf{J}^{-1} v_{\mathrm{foot}}^{\mathrm{B}}$ and compute the feedback torques as $\tau_{\mathrm{fb}} = k_d^{\mathrm{fb}}(\dot{q}^{\mathrm{ref}} - \dot{q})$. We set $k_d^{\mathrm{fb}} = 1$ to provide effective feedback while not causing significant interference with the feedforward controller.

VII. HIGH-FIDELITY SIMULATION WITH REAL-TO-SIM

Sim-to-real is particularly challenging for highly dynamic jumping tasks. While domain randomization (DR) can improve the robustness of the policy, randomization of standard

	Robot	Style	Stairs (Max Speed)	Stepping Stones (Max Speed)	Step (Max Height)	Gap (Max Width)
Egocentric-Vision[15]	A1	Walking	0.46m/s	0.36m/s	0.26m (1×)	0.17m (0.43×)
Robot Parkour [10]	A1	Jumping	-	-	0.4m (1.6×)	0.6m (1.5×)
Extreme Parkour [9]	A1	Jumping	-	-	0.5m (2×)	**0.8m (2×)**
Ours	Go1	Jumping	**1m/s**	**1.4m/s**	**0.6m (2.4×)**	**0.8m (2×)**

TABLE I: Performance comparison between our framework and baselines. Multiplier (×) shows distance relative to body dimensions (length or height).

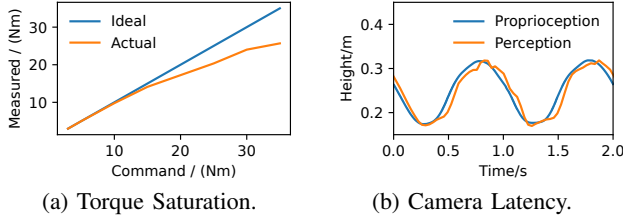(a) Torque Saturation.   (b) Camera Latency.

Fig. 6: We conduct individual studies to measure the motor torque saturations (Fig. 6a) and camera latency (Fig. 6b).

simulation parameters may not capture the unique behavior of real hardware, especially for highly dynamic tasks that operate close to hardware limit. In this work, we design individual experiments to carefully measure these characteristics, and reproduce them to improve simulation fidelity.

a) Measuring Motor Saturation: Jumping over long distances requires the robot to fully utilize its motor torque outputs to generate the necessary foot forces. Assuming a linear torque-current relationship, standard DC motors track a torque command by tracking the current output. However, at high torque commands, the torque-current relationship tend to saturate, leading to insufficient torque output at the desired current [16]. To accurately capture this torque saturation, we measure the motor torque output at different torque commands using a dynamometer (Fig. 6a). We then reproduce this relationship in simulation using a piecewise linear function, and convert all torque commands before stepping the simulation [16].

b) Measuring Camera Latency: To rapidly respond to terrain changes, its critical for the robot to understand any latency in terrain perception. To measure this camera latency, we execute sinusoidal body height motions on the real robot, and compare the body height estimated from joint angles using forward kinematics with the body height estimated from the received depth images (Fig.6b). The perception-based height estimation shows a consistent delay of about 50ms, or 5 environment steps. We subsequently implement this delay in all depth image rendering in simulation.

VIII. Results and Analysis

We design experiments to validate the performance of our framework in agile jumping over discontinuous terrains. More specifically, we aim to answer the following questions:

1) Can our framework jump over challenging discontinuous terrains at high speed in the real world?
2) How does the jumping performance of our framework compare with existing works?
3) Can our framework perform terrain-aware body and foot-step planning?

4) What are the important design choices to facilitate successful sim-to-real transfer?

A. Experimental Setup

We test our framework on the Unitree Go1 robot [4], where we mounted a Intel Realsense D435i camera to capture depth images, and stream the images to a Mac Mini computer for policy inference. We build the simulation environment in IsaacGym [38], and implemented the low-level leg controller in PyTorch [50]. We train the motion policy for 8000 gradient steps using Proximal Policy Optimization (PPO) [51], which takes about 7 hours on a Nvidia RTX 4090 GPU. We train the heightmap predictor for 30 DAgger steps where each step collects 5000 state-action pairs. Training the heightmap predictor takes about 1 hour on the same computer.

B. Continuous Jumping over Discontinuous Terrains

We deploy the trained framework in two real-world discontinuous terrains (Fig. 1 (a) (b)), including a staircase and a stepping stone environment. The robot carefully coordinates its body and foot motion and traverses through both terrains at high speeds. In the first case, the robot completes the 14-step staircase in 4.5 seconds, with an average horizontal speed of 0.8m/s and vertical speed of 0.6m/s. A closer look at the jumping behavior reveals that the robot crosses the entire 14-step stair cases in 8 jumps, with 2 stair steps per jump most of the time. In the second case, we test the robot on a stepping stone environment.The robot jumps across one gap each time, and traverses the entire terrain in less than 2 seconds. To the best of our knowledge, this is the first time a quadrupedal robot has demonstrated such agile continuous jumping on these challenging terrains.

We compare the terrain traversal performance of our framework with a few baselines in the first two columns of Table. I. Jumping significantly improves the robot's traversal speed in these terrains, resulting in more than 2 times speedup compared to the walking gait [15]. In addition, *continuous* jumping is important for these terrains with repeated discontinuities. While prior works [9, 10] have demonstrated high-performance single-step jumps, these results do not generalize to continuous jumps, which requires accurate and robust motion control over long horizons.

To test the generalizability of our framework, we deploy our framework on 4 real-world terrains of different geometries and materials (Fig. 1(c)). Our framework performs well on 3 of them with a success rate of 100%. The only exception is the staircase on the top-right of Fig. 1(c), where the robot achieved a success rate of 20%. This staircase features a unique geometry with disconnected planked steps, which is not seen during training. Therefore, both the motion policy
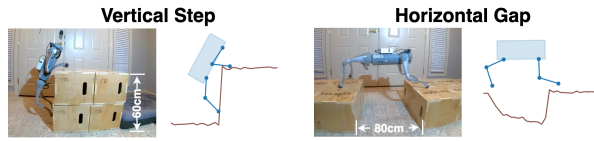
Fig. 7: Key frames and the estimated heightmap of the robot jumping over a 60cm step (left) and a 80cm gap (right).
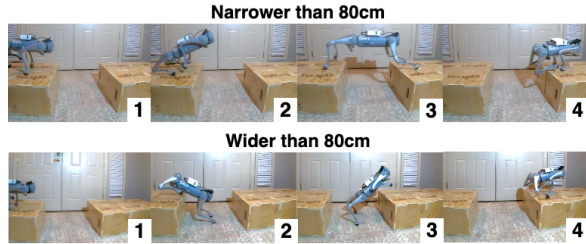

Fig. 8: The motion policy plans different foot placements for different gap widths. **Top**: For narrower gaps, the robot jumps over the gap directly. **Bottom**: for wider gaps, the robot lands on the ground and jumps over the gap again.

and the heightmap predictor had difficulty recognizing and reacting to this stair case.

C. High-performance Jumping over a Single Discontinuity

To further validate the capability of our framework, we test our framework in manually constructed terrains with a single discontinuity, including a vertical step or a horizontal gap (Fig. 7). Our framework handles these terrains well, and can cross horizontal gaps up to 80cm and climb vertical steps up to 60cm. We further visualize the estimated heightmap and the robot pose in Fig. 7. Using only proprioception and ego-centric depth image, the heightmap predictor accurately estimates the terrain shape with small contact errors. This jumping performance matches or even exceeds the performance of prior works on similar-sized robots (Table. I), even though our hardware platform, the Go1 Robot [4], has weaker motor outputs compared to the prior platform (A1).

D. Emergent Foot and Body Motion Planning

As the core of our framework, the motion policy learns versatile body and foot motion planning based on perceived terrain. For example, the motion policy plans different foot placements when jumping over gaps of different sizes (Fig. 8), from single-step jumping for narrower gaps to two-step jumping with intermediate landing for wider gaps. The motion policy also plans different body pitch trajectories when jumping on steps of different heights (Fig. 9), and prefers larger pitch angles for higher steps.

E. Ablation Study

To validate the design choices of our framework, we perform an ablation study, and test the simulation and real-world performance of our frameworks and the baselines on a 14-step stair case (Fig. 10). We run each baseline 5 times, and report the number of steps reached before failure. Most baselines perform well in simulation, and complete the entire 14-step staircase. One exception is the baseline where the motion policy is trained without heightmap randomization
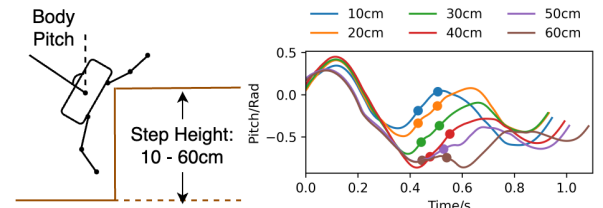

Fig. 9: The motion policy plans different body pitch trajectories when jumping over different step heights. The markers indicates the air phase of the bounding gait (Fig. 4).
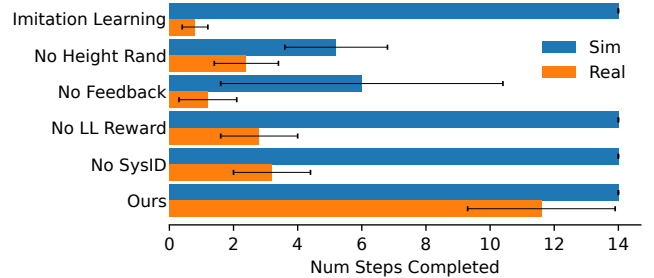

Fig. 10: The number of steps before failure (over 5 trials) in simulation and in the real world on a 14-step staircase.

(No Height Rand). While this baseline can jump well with the ground-truth heightmap, it cannot keep a similar performance even with small errors in heightmap estimation. Another exception is the baseline where the stance leg does not use the feedback controller (No Feedback). We find that the robot cannot track accurate body orientation changes, and cannot perform well on vertical jumps, which is consistent with prior observations (Fig. 5).

While the rest of the baselines completed the entire 14-step staircases in simulation, their performances drop significantly when transferred to the real world. Without the heightmap as the intermediate terrain representation, the imitation learning baseline, which directly imitates the motion policy's actions from depth images, cannot jump more than 2 staircases in the real world due to perception noise. Without the low-level motion tracking reward, the No LL Reward baseline manages to jump for a few steps in the real world, but eventually falls over. Lastly, without knowledge of the sensor and actuator characteristics, the "No SysID" baseline jumps very fast in the real world and crashes into the stair edges, likely due to its unawareness of motor saturation.

IX. CONCLUSION

In this work, we present a hierarchical framework for agile continuous jumping in discontinuous terrains. Using our framework, a quadrupedal robot achieves animal-like high-speed jumping on stairs and stepping stones for the first time, and outperforms baselines in various other parkour tasks. One limitation of our framework is that the robot motion is limited to the sagittal plane, and does not support sideways or turning motions. Another limitation is that our framework follows a fixed directional command, and does not strategically plan a path to reach the destination. In future work, we plan to address these limitations by expanding the scope of the current training pipeline, and achieve versatile, goal-oriented agile locomotion in complex terrains.

REFERENCES

[1] ANYbotics, "Anymal: Autonomous legged robot," Product Information, 2020. [Online]. Available: https://www.anybotics.com/anymal

[2] G. Bledt, M. J. Powell, B. Katz, J. Di Carlo, P. M. Wensing, and S. Kim, "Mit cheetah 3: Design and control of a robust, dynamic quadruped robot," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2018, pp. 2245–2252.

[3] "A1 Website," 2020. [Online]. Available: https://www.unitree.com/products/a1/

[4] "Go1 Website," 2021. [Online]. Available: https://www.unitree.com/products/go1/

[5] C. Nguyen, L. Bao, and Q. Nguyen, "Continuous jumping for legged robots on stepping stones via trajectory optimization and model predictive control," in *2022 IEEE 61st Conference on Decision and Control (CDC)*. IEEE, 2022, pp. 93–99.

[6] A. W. Winkler, C. D. Bellicoso, M. Hutter, and J. Buchli, "Gait and trajectory optimization for legged systems through phase-based end-effector parameterization," *IEEE Robotics and Automation Letters*, vol. 3, no. 3, pp. 1560–1567, 2018.

[7] S. Gilroy, D. Lau, L. Yang, E. Izaguirre, K. Biermayer, A. Xiao, M. Sun, A. Agrawal, J. Zeng, Z. Li *et al.*, "Autonomous navigation for quadrupedal robots with optimized jumping through constrained obstacles," in *2021 IEEE 17th International Conference on Automation Science and Engineering (CASE)*. IEEE, 2021, pp. 2132–2139.

[8] Q. Nguyen, M. J. Powell, B. Katz, J. Di Carlo, and S. Kim, "Optimized jumping on the mit cheetah 3 robot," in *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 2019, pp. 7448–7454.

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

[10] Z. Zhuang, Z. Fu, J. Wang, C. G. Atkeson, S. Schwertfeger, C. Finn, and H. Zhao, "Robot parkour learning," in *CoRL 2023*, 2023.

[11] K. Caluwaerts, A. Iscen, J. C. Kew, W. Yu, T. Zhang, D. Freeman, K.-H. Lee, L. Lee, S. Saliceti, V. Zhuang *et al.*, "Barkour: Benchmarking animal-level agility with quadruped robots," *arXiv preprint arXiv:2305.14654*, 2023.

[12] D. Hoeller, N. Rudin, D. Sako, and M. Hutter, "Anymal parkour: Learning agile navigation for quadrupedal robots," *arXiv preprint arXiv:2306.14874*, 2023.

[13] X. B. Peng, G. Berseth, and M. Van de Panne, "Dynamic terrain traversal skills using reinforcement learning," *ACM Transactions on Graphics (TOG)*, vol. 34, no. 4, pp. 1–11, 2015.

[14] C. Nguyen and Q. Nguyen, "Contact-timing and trajectory optimization for 3d jumping on quadruped robots. arxiv. 10.48550," *arXiv preprint ARXIV.2110.06764*, 2021.

[15] A. Agarwal, A. Kumar, J. Malik, and D. Pathak, "Legged locomotion in challenging terrains using egocentric vision," in *Conference on Robot Learning*. PMLR, 2023, pp. 403–415.

[16] J. Tan, T. Zhang, E. Coumans, A. Iscen, Y. Bai, D. Hafner, S. Bohez, and V. Vanhoucke, "Sim-to-real: Learning agile locomotion for quadruped robots," in *Proceedings of Robotics: Science and Systems*, Pittsburgh, Pennsylvania, June 2018.

[17] A. Kumar, Z. Fu, D. Pathak, and J. Malik, "Rma: Rapid motor adaptation for legged robots," *Robotics: Science and Systems*, 2021.

[18] Y. Yang, G. Shi, X. Meng, W. Yu, T. Zhang, J. Tan, and B. Boots, "Cajun: Continuous adaptive jumping using a learned centroidal controller," *arXiv preprint arXiv:2306.09557*, 2023.

[19] J. Di Carlo, P. M. Wensing, B. Katz, G. Bledt, and S. Kim, "Dynamic locomotion in the mit cheetah 3 through convex model-predictive control," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2018, pp. 1–9.

[20] D. Kim, J. Di Carlo, B. Katz, G. Bledt, and S. Kim, "Highly dynamic quadruped locomotion via whole-body impulse control and model predictive control," *arXiv preprint arXiv:1909.06586*, 2019.

[21] T. Horvat, K. Melo, and A. J. Ijspeert, "Model predictive control based framework for com control of a quadruped robot," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2017, pp. 3372–3378.

[22] F. Jenelten, J. Hwangbo, F. Tresoldi, C. D. Bellicoso, and M. Hutter, "Dynamic locomotion on slippery ground," *IEEE Robotics and Automation Letters*, vol. 4, no. 4, pp. 4170–4176, 2019.

[23] O. Villarreal, V. Barasuol, P. M. Wensing, D. G. Caldwell, and C. Semini, "Mpc-based controller with terrain insight for dynamic legged locomotion," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 2436–2442.

[24] Y. Ding, A. Pandala, and H.-W. Park, "Real-time model predictive control for versatile dynamic motions in quadrupedal robots," in *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 2019, pp. 8484–8490.

[25] C. Gehring, S. Coros, M. Hutter, C. D. Bellicoso, H. Heijnen, R. Diethelm, M. Bloesch, P. Fankhauser, J. Hwangbo, M. Hoepflinger *et al.*, "Practice makes perfect: An optimization-based approach to controlling agile motions for a quadruped robot," *IEEE Robotics & Automation Magazine*, vol. 23, no. 1, pp. 34–43, 2016.

[26] Z. Song, L. Yue, G. Sun, Y. Ling, H. Wei, L. Gui, and Y.-H. Liu, "An optimal motion planning framework for quadruped jumping," in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2022, pp. 11 366–11 373.

[27] A. Pandala, R. T. Fawcett, U. Rosolia, A. D. Ames,

and K. A. Hamed, "Robust predictive control for quadrupedal locomotion: Learning to close the gap between reduced-and full-order models," *IEEE Robotics and Automation Letters*, vol. 7, no. 3, pp. 6622–6629, 2022.

[28] H. Li and P. M. Wensing, "Cafe-mpc: A cascaded-fidelity model predictive control framework with tuning-free whole-body control," *arXiv preprint arXiv:2403.03995*, 2024.

[29] R. Grandia, F. Farshidian, R. Ranftl, and M. Hutter, "Feedback mpc for torque-controlled legged robots," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2019, pp. 4730–4737.

[30] S. Qi, W. Lin, Z. Hong, H. Chen, and W. Zhang, "Perceptive autonomous stair climbing for quadrupedal robots," in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2021, pp. 2313–2320.

[31] H.-W. Park, P. M. Wensing, and S. Kim, "Jumping over obstacles with mit cheetah 2," *Robotics and Autonomous Systems*, vol. 136, p. 103703, 2021.

[32] R. Grandia, F. Jenelten, S. Yang, F. Farshidian, and M. Hutter, "Perceptive locomotion through nonlinear model-predictive control," *IEEE Transactions on Robotics*, vol. 39, no. 5, pp. 3402–3421, 2023.

[33] G. B. Margolis, G. Yang, K. Paigwar, T. Chen, and P. Agrawal, "Rapid locomotion via reinforcement learning," *arXiv preprint arXiv:2205.02824*, 2022.

[34] G. B. Margolis and P. Agrawal, "Walk these ways: Tuning robot control for generalization with multiplicity of behavior," in *Conference on Robot Learning*. PMLR, 2023, pp. 22–31.

[35] A. Agarwal, A. Kumar, J. Malik, and D. Pathak, "Legged locomotion in challenging terrains using egocentric vision," *arXiv preprint arXiv:2211.07638*, 2022.

[36] L. Smith, J. C. Kew, T. Li, L. Luu, X. B. Peng, S. Ha, J. Tan, and S. Levine, "Learning and adapting agile locomotion skills by transferring experience," *arXiv preprint arXiv:2304.09834*, 2023.

[37] T. He, C. Zhang, W. Xiao, G. He, C. Liu, and G. Shi, "Agile But Safe: Learning Collision-Free High-Speed Legged Locomotion," in *Proceedings of Robotics: Science and Systems*, Delft, Netherlands, July 2024.

[38] V. Makoviychuk, L. Wawrzyniak, Y. Guo, M. Lu, K. Storey, M. Macklin, D. Hoeller, N. Rudin, A. Allshire, A. Handa, and G. State, "Isaac gym: High performance gpu-based physics simulation for robot learning," 2021.

[39] 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, p. eabk2822, 2022.

[40] Z. Xie, H. Y. Ling, N. H. Kim, and M. van de Panne, "Allsteps: curriculum-driven learning of stepping stone skills," in *Computer Graphics Forum*, vol. 39, no. 8. Wiley Online Library, 2020, pp. 213–224.

[41] 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 *5th Annual Conference on Robot Learning*, 2021.

[42] X. Da, Z. Xie, D. Hoeller, B. Boots, A. Anandkumar, Y. Zhu, B. Babich, and A. Garg, "Learning a contact-adaptive controller for robust, efficient legged locomotion," *arXiv preprint arXiv:2009.10019*, 2020.

[43] Z. Xie, X. Da, B. Babich, A. Garg, and M. van de Panne, "Glide: Generalizable quadrupedal locomotion in diverse environments with a centroidal model," *arXiv preprint arXiv:2104.09771*, 2021.

[44] Y. Yang, T. Zhang, E. Coumans, J. Tan, and B. Boots, "Fast and efficient locomotion via learned gait transitions," in *Conference on Robot Learning*. PMLR, 2022, pp. 773–783.

[45] Y. Yang, X. Meng, W. Yu, T. Zhang, J. Tan, and B. Boots, "Continuous versatile jumping using learned action residuals," *arXiv preprint arXiv:2304.08663*, 2023.

[46] G. Bellegarda and Q. Nguyen, "Robust quadruped jumping via deep reinforcement learning," *arXiv preprint arXiv:2011.07089*, 2020.

[47] S. Ross, G. Gordon, and D. Bagnell, "A reduction of imitation learning and structured prediction to no-regret online learning," in *Proceedings of the fourteenth international conference on artificial intelligence and statistics*. JMLR Workshop and Conference Proceedings, 2011, pp. 627–635.

[48] 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*. PMLR, 2022, pp. 91–100.

[49] M. H. Raibert, *Legged robots that balance*. MIT press, 1986.

[50] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga *et al.*, "Pytorch: An imperative style, high-performance deep learning library," *Advances in neural information processing systems*, vol. 32, 2019.

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