Learning Robust and Agile Legged Locomotion Using Adversarial Motion Priors

Jinze Wu, Guiyang Xin, *Member, IEEE*, Chenkun Qi, *Member, IEEE*, and Yufei Xue

***Abstract*—Developing both robust and agile locomotion skills for legged robots is non-trivial. In this work, we present the first blind locomotion system capable of traversing challenging terrains robustly while moving rapidly over natural terrains. Our approach incorporates the Adversarial Motion Priors (AMP) in locomotion policy training and demonstrates zero-shot generalization from the motion dataset on flat terrains to challenging terrains in the real world. We show this result on a quadruped robot Go1 using only proprioceptive sensors consisting of the IMU and joint encoders. Experiments on the Go1 demonstrate the robust and natural motion generated by the proposed method for traversing challenging terrains while moving rapidly over natural terrains.**

***Index Terms*—Legged Robots, Reinforcement Learning, Machine Learning for Robot Control.**

I. Introduction

ANIMALS are capable of robust locomotion over challenging terrains and agile locomotion over natural terrains. Reproducing such robust and agile behaviors has been a hot topic in the legged robotic community, with a large number of works devoted to designing control strategies for robust and agile locomotion skills [1], [2], [3], [4], [5], [6], [7], [8], [9], [10], [11], [12], [13]. However, developing either robust or agile controllers for legged robots is non-trivial. To obtain robust locomotion skills over challenging terrains, most of the existing works rely on computationally intensive methods for environment sensing, using cameras and LIDARs [4], [5], [6]. These front-end perception parts are highly susceptible to lighting conditions and weather conditions, which may affect the back-end calculations such as the foothold planning. To obtain agile locomotion over natural terrains, most of the previous approaches use model predictive control with hand-designed models [12], [13], which face the problem of how to balance model accuracy and computational complexity. These issues above present a huge challenge to combine the two capabilities of robust and agile locomotion. In contrast, proprioceptive sensors such as joint encoders and IMU are more robust compared to exteroceptive sensors and are fundamental for the controller to achieve both robust and agile locomotion skills.

Due to the uncertainties from the unforeseen events, it is difficult to design blind locomotion controllers with traditionally optimization-based approaches [10], [11], [12], [13]. Reinforcement Learning (RL), a uniform framework that learns a control policy from the agent's interactions with the environment, is convenient for designing blind locomotion controllers. Recently, there has been a surge of works that use RL to learn robust blind locomotion for legged robots [1], [2], [3]. Moreover, RL provides a direct mapping from sensory observations to actions, enabling real-time control of agile locomotion in complex environments [7], [8]. Previous approaches using RL have pursued either extreme robust locomotion over challenging terrains [2], [3], [4], [5] or extreme agile locomotion over natural terrains [7], [8], whether it can achieve both capabilities with a single policy network remains unknown.

A. Related RL Methods for Locomotion

Recent works have successfully employed RL to learn locomotion policies. [1] used an actuator network to model complex actuator dynamics to aid sim-to-real transfer of RL polices for ANYmal robot. Follow-up works [2], [3] extended the robustness of the locomotion by training robots on diverse terrains using the privileged learning paradigm [14]. Recently, [4], [5] integrated proprioceptive and exteroceptive states and improved the locomotion efficiency of quadrupedal robots in the wild. However, these works only showed locomotion of low or moderate speed over challenging terrains, while high-speed locomotion over natural terrains was not tested. To obtain agile locomotion skills, [7], [8] used RL to train Mini Cheetah to learn high-speed movements over natural terrains. Although they obtained remarkable results in high-speed sprinting and spinning, their locomotion gaits are unnatural. Moreover, it is unknown if their controllers can traverse challenging terrains like stairs, curbs and vegetation.

Although RL has achieved substantial improvements for legged robots over model-based methods, most of the legged movements in the experiments of the papers employing RL methods are unnatural and jerky [3], [5], [7], [8]. Adding predefined gait priors into training procedure may accelerate

Manuscript received 27 February 2023; accepted 22 June 2023. Date of publication 28 June 2023; date of current version 7 July 2023. This letter was recommended for publication by Associate Editor J. Carpentier and Editor A. Kheddar upon evaluation of the reviewers' comments. This work was supported in part by the National Key Research and Development Plan under Garnt 2021YFF0307900, in part by the Fundamental Research Funds for the Central Universities under Grant 82232025, and in part by the Liaoning and Guangdong Basic and Applied Basic Research Foundation under Grants 2023JH2/101600033 and 2022A1515140156. *(Corresponding author: Chenkun Qi.)*

Jinze Wu, Chenkun Qi, and Yufei Xue are with the School of Mechanical Engineering, Shanghai Jiao Tong University, Shanghai 200240, China (e-mail: wjz_sjtu@sjtu.edu.cn; chenkqi@sjtu.edu.cn; xue_yeiii@sjtu.edu.cn).

Guiyang Xin is with the School of Biomedical Engineering, Dalian University of Technology, Dalian 116024, China (e-mail: guiyang.xin@dlut.edu.cn).

This letter has supplementary downloadable material available at https://doi.org/10.1109/LRA.2023.3290509, provided by the authors.

Digital Object Identifier 10.1109/LRA.2023.3290509

the convergence to a normal gait for legged robots [15], [16]. These methods often utilize some motion tracking techniques, where an agent mimics a desired motion by explicitly using a phase variable to track a sequence of target poses generated by an expert. However, it is difficult for motion tracking techniques to imitate multiple reference motions according to a single phase variable. To mitigate this limitation, hierarchical policies and latent space models are used to learn reusable skills from large motion datasets by explicitly dividing the RL problem into pre-training and transfer stages [17], [18], [19]. However, most of these methods are verified in simulation with moderate challenging terrains, which is difficult to guarantee the effectiveness of the proposed methods in the real world. To aid sim-to-real transfer of RL policies, trajectory optimization (TO) techniques provide a uniform framework to generate stable demonstrations [20], [21], without the need for motion capture data or artificially designed animation data [15], [16]. However, these imitation-based RL methods still rely on a pre-training stage to track reference motion, which is often irrelevant to high-level tasks.

Generative adversarial imitation learning (GAIL) is a more general approach for imitating behaviors from unstructured demonstration datasets without motion tracking [22]. This method integrates a generative adversarial network (GAN) [23] into RL to construct a discriminator that measures the similarity between the policy and the demonstrations, addressing some of the limitations of behavioral cloning such as distribution drift. The training procedure of GAIL can be interpreted as training a policy to generate the states and actions that are indistinguishable from the demonstrator in the view of the discriminator. However, GAIL is not directly applicable when the demonstrator's actions are unobservable. Recently, [24] proposed the Adversarial Motion Priors (AMP), which uses GAIL framework to predict whether a state transition $(\boldsymbol{s}_t, \boldsymbol{s}_{t+1})$ is a real sample from the dataset or a fake sample produced by the agent. This approach enables simulated agents to perform high-level tasks with motion styles learned from large unstructured motion datasets. The follow-up work [25] extended AMP to the latent space to learn large-scale reusable adversarial skill embeddings for physically simulated characters.

Although the AMP has shown promising results in the animation domain, only a few works have deployed the AMP on real robots. [26] used the AMP to learn quadrupedal locomotion on flats without requiring complex reward functions. [27] extended the AMP to multiple style control for wheeled-legged locomotion. The follow-up work [28] showed the AMP can enable quadruped robot to learn agile skills by using a Wasserstein GAN formulation and transitions from rough partial demonstrations. Since most of the motion priors were collected on even terrains in the above methods, the effectiveness of the AMP for learning robust locomotion over complex terrains remains unknown.

B. Contributions

To address the aforementioned shortcomings of the existing RL methods for legged robots, we present the first blind locomotion system capable of traversing challenging terrains robustly while moving rapidly over natural terrains. Our approach incorporates AMP in locomotion policy training and demonstrates zero-shot generalization from the motion dataset on flat terrains to challenging terrains in the real world. As shown in Fig. 1 and the accompanying video, the emergent behaviors for robust locomotion enable our robot (a $28\,\mathrm{cm}$ tall quadruped robot) to traverse the obstacles of height up to $25\,\mathrm{cm}$ while the emergent behaviors for agile locomotion enable the robot to sprint up to $3.5\,\mathrm{m/s}$ and spin $5.8\,\mathrm{rad/s}$ in outdoors. The main contributions are listed as follows:

1) We generate an AMP dataset based on TO techniques for our robot Go1. The AMP introduces soft gait priors into our training procedure, which replaces complex auxiliary rewards with a style reward to learn robust and agile locomotion skills with natural motion styles.
2) We propose a teacher-student training framework that concurrently learn robust locomotion over challenging terrains and agile locomotion over even terrains, using only proprioception. We demonstrate that a single RL policy can enable our robot to achieve robust and agile locomotion both indoors and outdoors and successfully negotiate challenging terrains.

Fig. 1. Our robot can achieve robust locomotion over challenging terrains and agile locomotion over natural terrains. The robot can successfully traverse curbs, stairs, rocky and vegetation while running and spinning at high speed over natural terrains. Whether on even or complex terrains, at high or low speeds, the emergent behaviors learned by our robot exhibit a natural gait and smooth motion.

II. Method

A. Reinforcement Learning Problem Formulation

We formulate our control problem with discrete-time dynamics, where the environment is completely defined by the state $\boldsymbol{x}_t$ at time step t. The policy performs an action $\boldsymbol{a}_t$, and then the environment moves to the next state $\boldsymbol{x}_{t+1}$ with transition probability $P(\boldsymbol{x}_{t+1} \mid \boldsymbol{x}_t, \boldsymbol{a}_t)$ and returns a reward r_t. The goal of the RL is to find the optimal parameters θ of a policy π_θ to maximize the expected discounted return over the future trajectory:

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

where γ^t is a discount factor.

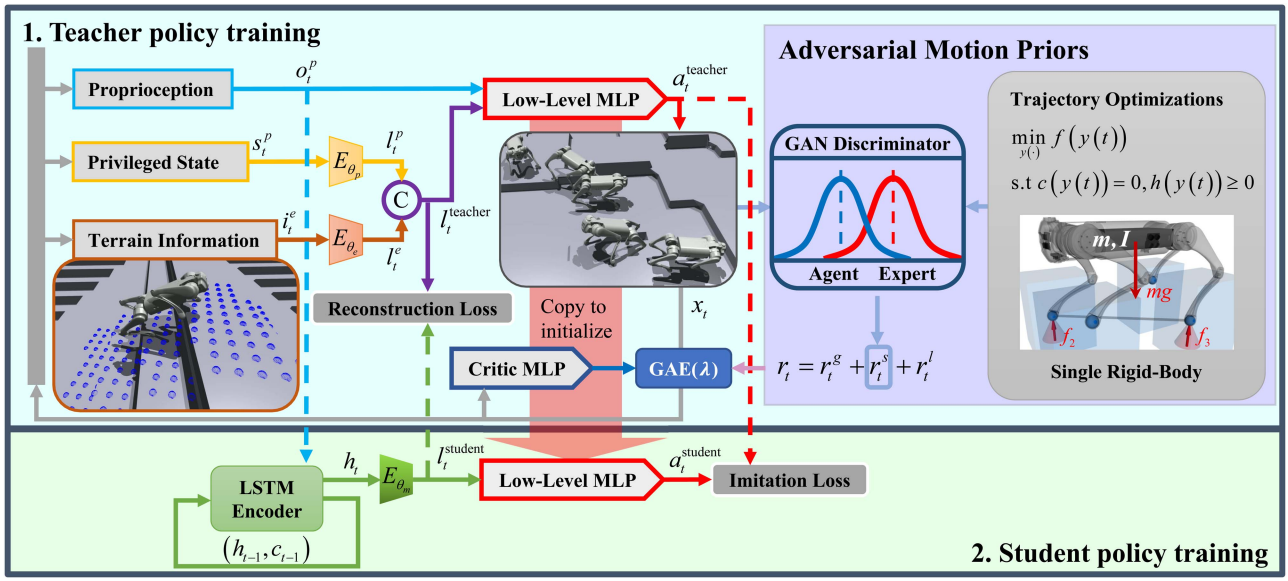

Fig. 2. Overview of the training methods. We first train a teacher policy with access to proprioceptive observation $\boldsymbol{o}_t^p$, privileged state s_t^p and terrain information i_t^e using RL. The total reward r_t obtained by the teacher consists of a task reward r_t^g, a style reward r_t^s based on AMP, and a regularization reward r_t^l which imposes constraints on the motion. A student policy is then trained by a supervised fashion, where the student imitates the teacher's action a_t^{teacher} and reconstructs the latent representation l_t^{teacher} using only proprioceptive observation o_t^p. We reuse the learned weights of the teacher's low-level net to initialize the student's low-level net to speed up training. The motion priors are generated from the TO, using a single rigid-body model.

In this work, we are interested in learning a blind locomotion controller that is agile and robust. However, the blind locomotion case is modeled as a partially observable Markov decision process (POMDP), since the terrains are not fully observable without exteroceptive sensors. Privileged learning [14] can reduce the training difficulties of this challenging case. In this paradigm, a teacher policy is trained with privileged states that are only available in the simulation. A student policy then learns to imitate the teacher's encoding of privileged states and the teacher's actions accordingly. We first train a teacher policy that can observe the dynamic properties of the robot and the environment. The teacher policy is then distilled into a student policy that learns to infer these properties from a history of observations via supervised learning.

State Space: For teacher policy, the state $\boldsymbol{x}_t^{\text{teacher}}$ consists of proprioceptive observation $\boldsymbol{o}_t^p$, privileged state $\boldsymbol{s}_t^p$ and terrain information $\boldsymbol{i}_t^e$, as shown in the top of Fig. 2. The $\boldsymbol{o}_t^p$ contains the orientation of the gravity vector and base angular velocity in the robot's base frame, the joint positions and velocities, the previous action $\boldsymbol{a}_{t-1}$ selected by the current policy, and the desired base velocity command vector. Since the base linear velocity computed from state estimation algorithms is noisy, we consider the base linear velocity as part of the privileged state $\boldsymbol{s}_t^p$, as well as ground friction coefficients, ground restitution coefficients, contact forces, external forces and their positions on the robot, and collision states (trunk, thigh and shank). The $\boldsymbol{i}_t^e$ contains 187 measurements of terrain points sampled from a grid around the robot base, representing the vertical distance from the sampled point to the robot base. By contrast, the proprioceptive observation $\boldsymbol{o}_t^p$ is the only accessible vector for the student policy.

Action Space: The policy action $\boldsymbol{a}_t$ is an 12-D vector interpreted as a target joint position offset, which is added to the time-invariant nominal joint position to specify the target motor position for each joint. The joint PD controllers then compute the torque commands using these target motor positions with fixed gains ($K_p = 20$, $K_d = 0.5$).

B. Reward Terms Design

Crafting suitable reward functions requires degrees of expert knowledge and often yields modest improvements during tedious tuning processes. Another approach is to replace complex auxiliary rewards with the style reward [24], [26]. Our goal is to design concise reward functions that can induce agents to learn robust and agile legged locomotion with a natural gait. In this work, the reward function consists of a task component r_t^g, a style component r_t^s, and a regularization component r_t^l, such that

$$r_t = r_t^g + r_t^s + r_t^l. \tag{2}$$

The task reward term consists of linear and angular velocity tracking functionalities. The style reward is used to evaluate the similarity between the demonstrator's behavior and the agent's behavior, meaning that the more similar they are, the more style rewards the agent can obtain. Since we expect our robot to obtain natural and smooth trot gait when traversing even or uneven terrains, we use a style reward function based on the AMP to encourage our robot to produce behaviors with the same gait style as those from a reference dataset $\mathcal{D}$.

Following [24], we define a discriminator D_φ presented by a neural network with parameters φ to predict whether a state

TABLE I
REWARD TERMS FOR VELOCITY COMMANDS TRACKING TASK, MOTION STYLE, AND REGULARIZATION (SMOOTHNESS, SAFETY)

Term	Equation	Weight
Task r^g	$\exp\left(-\lVert\mathbf{v}_{t,xy}^{\text{des}}-\mathbf{v}_{t,xy}\rVert_2/0.15\right)$	1.0
	$\exp\left(-\lVert\omega_{t,z}^{\text{des}}-\omega_{t,z}\rVert_2/0.15\right)$	0.5
Smoothness r^l	$-\lVert\boldsymbol{\tau}\rVert_2$	1×10^{-4}
	$-\lVert\ddot{\mathbf{q}}\rVert_2$	2.5×10^{-7}
	$-\lVert\mathbf{q}_{t-1}-\mathbf{q}_t\rVert_2$	0.1
	$\sum_{i=0}^{5}\min\left(\mathbf{t}_{air,i}-0.5,0\right)$	1.0
Safety r^l	$-n_{collision}$	0.1
Style r^s	$\max\left[0,1-0.25\left(d_t^{\text{score}}-1\right)^2\right]$	0.5

transition $(\boldsymbol{s}_t,\boldsymbol{s}_{t+1})$ is a real sample from the dataset $\mathcal{D}$ or a fake sample produced by the agent $\mathcal{A}$. Each state $\boldsymbol{s}_t^{AMP}\in\mathbb{R}^{31}$ contains the joint positions, joint velocities, base linear velocity, base angular velocity, and the base height with respect to the terrains. To mitigate the mode collapse caused by GAN-based techniques, the reference dataset $\mathcal{D}$ only contains trot gait motion clips.

The training objective for the discriminator is defined as:

$$\begin{aligned}\arg\min_{\varphi}\ &\mathbb{E}_{(\boldsymbol{s}_t,\boldsymbol{s}_{t+1})\sim\mathcal{D}}\left[\left(D_\varphi\left(\boldsymbol{s}_t,\boldsymbol{s}_{t+1}\right)-1\right)^2\right]\\&+\mathbb{E}_{(\boldsymbol{s}_t,\boldsymbol{s}_{t+1})\sim\mathcal{A}}\left[\left(D_\varphi\left(\boldsymbol{s}_t,\boldsymbol{s}_{t+1}\right)+1\right)^2\right]\\&+\frac{\alpha^{gp}}{2}\mathbb{E}_{(\boldsymbol{s}_t,\boldsymbol{s}_{t+1})\sim\mathcal{D}}\left[\left\lVert\nabla_\varphi D_\varphi\left(\boldsymbol{s}_t,\boldsymbol{s}_{t+1}\right)\right\rVert_2\right],\end{aligned}\tag{3}$$

where the first two terms are least square GAN formulation, which encourage the discriminator to distinguish whether a given input state transition is from the agent $\mathcal{A}$ or the reference dataset $\mathcal{D}$. The final term in (3) is a gradient penalty, which mitigates the discriminator's tendency of assigning non-zero gradients on the manifold of real data samples [24]. The α^{gp} is a manually-specified coefficient (we use $\alpha^{gp}=10$). The style reward is then defined by:

$$r_t^s\left[\left(\boldsymbol{s}_t,\boldsymbol{s}_{t+1}\right)\sim\mathcal{A}\right]=\max\left[0,1-0.25\left(d_t^{\text{score}}-1\right)^2\right],\tag{4}$$

where $d_t^{\text{score}}=D_\varphi(\boldsymbol{s}_t,\boldsymbol{s}_{t+1})$ and the style reward is scaled to the range $[0,1]$.

However, using the task reward and style reward to train agent requires more training data to converge to natural behaviors [24], [26]. Furthermore, it is uncertain for our robot to learn natural gait motion over challenging terrains, since the dataset $\mathcal{D}$ only contains locomotion trajectories on flat terrains. To aid sim-to-real transfer over challenging terrains, we add the regularization reward term to the total reward, which imposes constraints on the motion, taking into account smoothness and safety. The smoothness is induced by the joint-level penalty and the stride duration bonus [9]. The safety is ensured by penalties for collisions with itself or environments. The details of the reward functions are shown in Table I.

C. Motion Dataset Generation

Since we only need state transitions to construct the motion dataset $\mathcal{D}$, we adopt the single TO formulation from the previous work [11] to generate quadrupedal locomotion on the flat terrain with a trotting gait. Our robot is first represented using a simplified centroidal dynamics model to reduce computation complexity and then transcribed into a nonlinear programming problem with friction cone constraints and kinematic constraints explicitly enforced. To speed up the optimization and avoid tuning procedures, the TO problem is solved using *TOWR* [29], which does not require a cost function. The $\mathcal{D}$ consists of forward, backward, lateral left, lateral right, left steering, right steering, and combined locomotion trajectories, with a total duration of 30 seconds. The benefit of generating motion datasets using the TO is that they can fully match the state space of the simulated agent and demonstrator, avoiding the use of other motion retargeting techniques like [15].

III. TRAINING

Simulation: We trained 4096 parallel agents on different types of terrains using the IsaacGym simulator [9]. The teacher policy and the student policy were trained with 400 and 200 million simulated time steps, respectively. The overall training time for the two stages was seven hours of wall-clock time. Each RL episode lasts for a maximum of 1000 steps, equivalent to 20 s, and terminates early if it reaches the termination criteria. The control frequency of the policy is 50 Hz in the simulation. All trainings were performed on a single NVIDIA RTX 3090Ti GPU.

Termination: We terminate an episode and start the next one when the robot reaches the termination criteria, which include trunk collisions with the ground, large body inclinations, and being trapped for a long period of time.

Dynamics Randomization: To improve the robustness of our policy and facilitate transfer from simulation to the real world, we randomize the mass of the trunk and legs, the mass and position of payload applied to the body of the robot, the ground friction and restitution coefficients, the motor strength, the joint-level PD gains, and the initial joint positions in each episode. Some of these dynamic parameters are considered as privileged state $\boldsymbol{s}_t^p$ to aid the teacher policy training. In addition, the same observation noise as in [9] is added during the training phase in the simulation. The randomization ranges for each parameter are detailed in Table II.

A. Training Curriculum

For legged robots, blind locomotion over complex terrains is a challenging task due to its uncertainties in the interactions with the environments. Previous work [30] has shown that training agents on diverse terrains can indeed lead to the development of non-trivial locomotion skills. In this work, we create five types of procedurally generated terrains similar to [9], including rough flats, slopes, waves, stairs, and discrete steps.

Due to the instability of RL in the early stage, it is difficult to train robots directly on very complex terrains. We adopt and

TABLE II
DYNAMIC PARAMETERS AND THE RANGE OF THEIR RANDOMIZATION VALUES USED DURING TRAINING

Parameters	Range[Min, Max]	Unit
Link Mass	[0.8, 1.2]×nominal value	Kg
Payload Mass	[0, 3]	Kg
Payload Position	[-0.1, 0.1] relative to base origin	m
Ground Friction	[0.05, 2.75]	-
Ground Restitution	[0.0, 1.0]	-
Motor Strength	[0.8, 1.2]×motor torque	Nm
Joint K_p	[0.8, 1.2]×20	-
Joint K_d	[0.8, 1.2]×0.5	-
Initial Joint Positions	[0.5, 1.5]×nominal value	rad

TABLE III
NETWORK ARCHITECTURE FOR TEACHER POLICY AND STUDENT POLICY

Module	Inputs	Hidden Layers	Outputs
Low-Level (MLP)	l_t, o_t^p	[256, 128, 64]	a_t
Critic (MLP)	x_t	[512, 256, 128]	V_t
Memory (LSTM)	o_t^p, h_{t-1}, c_{t-1}	[256, 256, 256]	h_t
E_{θ_p} (MLP)	s_t^p	[64, 32]	l_t^p
E_{θ_e} (MLP)	i_t^e	[256, 128]	l_t^e
E_{θ_m} (MLP)	h_t	[256, 128]	l_t^{student}
D_φ (MLP)	s_t^{AMP}, s_{t+1}^{AMP}	[1024, 512]	d_t^{score}

All networks use ELU activations for hidden layers.

improve the automated terrain curriculum in [9]. We create a height-field map with 100 terrains arranged in a 20×10 grid. Each row has the same type of terrain arranged in increasing difficulty, while each terrain has a length and width of $8\,\text{m}$. The rough flats are constructed by adding noise increased from $\pm 1\,\text{cm}$ to $\pm 8\,\text{cm}$. The inclination of the slopes increases from $0\,\text{deg}$ to $30\,\text{deg}$. The waves are constructed by three sine waves across the terrain length. The amplitude of these waves increases from $20\,\text{cm}$ to $50\,\text{cm}$. The stairs have a fixed width of $30\,\text{cm}$ and a step height increased from $5\,\text{cm}$ to $23\,\text{cm}$. The discrete obstacles only have two height levels increased from $\pm 5\,\text{cm}$ to $\pm 15\,\text{cm}$. At the beginning of training, all robots are equally assigned to all terrain types with the lowest difficulty. The robot is only moved to a more difficult terrain once it has adapted to its current terrain difficulty. This adaptation is obtained when the robot can step out of the current terrains with more than 85% of the average linear velocity tracking reward. In contrast, they are reset to easier terrains if they fail to travel at least half of the distance required by their command linear velocity at the end of an episode. To avoid skills forgetting, robots solving the hardest terrains are looped back to a randomly selected difficulty of current terrain type.

Our robot is attempting to learn a command-conditioned policy by tracking different velocity commands during training. The robot is given a desired command to follow at the beginning of an episode, which is a randomly generated vector $\mathbf{v}_t^{\text{des}} = (v_x, v_y, \omega_z) \in \mathbb{R}^3$ representing the longitudinal velocity, lateral velocity and yaw velocity in the base frame. In the terrain curriculum phase, the yaw velocity command is computed from the errors between the current heading direction and the target heading direction, which enables our robot to efficiently step out of the terrains. The target heading direction is sampled uniformly from $[-180\,^\circ, 180\,^\circ]$. We separately sample the longitudinal velocity command and the lateral velocity command from a small range $[-1\,\text{m/s}, 1\,\text{m/s}]$ during terrain curriculum phase.

Due to the difficulty of training, a small range of velocity commands could be tracked over challenging terrains. However, tracking large velocity commands on even terrains is more efficient and promising for legged robots. We propose a concurrent high-speed training procedure after terrain curriculum. For agents on rough flat terrains, once they step out of the roughest flats, their (v_x, ω_z) velocity command sampling schedules are changed to the grid adaptive curriculum strategy [7], in order to learn high-speed sprinting and spinning. Only the terrain curriculum of the agents during high-speed training will be terminated, the rest of the agents will not be affected.

B. Teacher Policy Training and Architecture

In the first training phase, we train a teacher policy using proximal policy optimization (PPO) [31]. The training process of the teacher policy and the discriminator are synchronized. The teacher performs a rollout in the environment to generate a state transition $(\boldsymbol{s}_t^{AMP}, \boldsymbol{s}_{t+1}^{AMP})$. This state transition is then fed to the discriminator D_φ to obtain d_t^{score} that is used to compute the trot gait style reward r_t^s according to (4), which is added to the other rewards obtained by the teacher. After collecting the rollout data, we optimize the parameter θ of the teacher policy $\pi_\theta^{\text{teacher}}$ to maximize the total discounted return of (1), and the parameter φ of the discriminator D_φ to minimize the objective presented in (3) in every training step.

The teacher policy $\pi_\theta^{\text{teacher}}$ consists of three Multilayer Perceptron (MLP) components: a terrain encoder E_{θ_e}, a privileged encoder E_{θ_p}, and a low-level network, as shown in Fig. 2. The E_{θ_e} and E_{θ_p} compress the terrain information $\boldsymbol{i}_t^e \in \mathbb{R}^{187}$ and the privileged state $\boldsymbol{s}_t^p \in \mathbb{R}^{30}$ into low-dimensional latent representations $\boldsymbol{l}_t^e \in \mathbb{R}^{16}$ and $\boldsymbol{l}_t^p \in \mathbb{R}^8$, respectively. Although this compression loses some information, it preserves the most needed information, which facilitates the student to reconstruct the latent representations. The $\boldsymbol{l}_t^e$ and $\boldsymbol{l}_t^p$ are first concatenated into a full latent representation $\boldsymbol{l}_t^{\text{teacher}} \in \mathbb{R}^{24}$. The $\boldsymbol{l}_t^{\text{teacher}}$ and the proprioceptive observation $\boldsymbol{o}_t^p \in \mathbb{R}^{45}$ are then fed to the low-level network with a $tanh$ output layer to output the mean $\boldsymbol{\mu}_t \in \mathbb{R}^{12}$ of a Gaussian distribution $\boldsymbol{a}_t^{\text{teacher}} \sim \mathcal{N}(\boldsymbol{\mu}_t, \boldsymbol{\sigma})$, where $\boldsymbol{\sigma} \in \mathbb{R}^{12}$ denotes the variance of the action determined by the PPO. The teacher policy also has a critic network presented by the MLP with three hidden layers to provide the target values V_t for generalized advantage estimator. The discriminator D_φ is a single MLP with two hidden layers and a linear unit output layer. More details on each layer are shown in Table III.

C. Student Policy Training and Architecture

The student policy is trained to reproduce the teacher policy actions without using privileged state $\boldsymbol{s}_t^p$ and terrain information $\boldsymbol{i}_t^e$. Thus, the dynamics of the student is considered as a POMDP, and the student needs to consider the history of the observation $\boldsymbol{o}_t^p$ to estimate unobservable states. We use a memory encoder

for the student to encode the sequential correlations between histories.

During student training, we leverage a supervised fashion by minimizing two losses, including an imitation loss and a reconstruction loss as shown in Fig. 2. The imitation loss enables the student to mimic the teacher's actions $\boldsymbol{a}_t^{\text{teacher}}$. The reconstruction loss encourages the student's memory encoder to reproduce the teacher's latent representation $\boldsymbol{l}_t^{\text{teacher}}$. We use the dataset aggregation strategy (DAgger) to generate samples by rolling out the student policy to increase robustness [32]. The same training curriculum as the teacher's is applied to the student training, while no discriminator is trained.

The student policy consists of a memory encoder and a low-level MLP that remains the same structure as the teacher's low-level net. Memory encoders are frequently implemented by stacking a sequence of historical observations into the input of an MLP [7], or by using architectures that can capture past information, such as recurrent neural networks (RNN) [4] or temporal convolutional neural networks (TCN) [2]. However, for architectures such as the MLP and the TCN, a portion of the memory space needs to be set aside to store historical observations, which puts a lot of pressure on the onboard resource usage. So instead of storing total historical observations, it is more effective to use RNNs which can embed past history information within hidden states. We use the Long Short Term Memory (LSTM) as our RNN architecture [33]. First, the proprioceptive observation o_t^p along with the past hidden state h_{t-1} and cell state c_{t-1} of the LSTM are encoded into the current hidden state $h_t \in \mathbb{R}^{256}$ by the LSTM module, which is then passed to E_{θ_m} to output student's latent representation l_t^{student}. We reuse the learned weights of the teacher's low-level net to initialize the student's low-level net to speed up training. To enable our controller to learn robust blind locomotion [2], we trained the student policy with a sequence of 50 o_t^p (corresponding to 1 s of memory). More details on each layer are shown in Table III.

IV. Results and Discussion

Hardware: Our controller is deployed on the Unitree Go1 Edu robot, which stands $28\,\text{cm}$ tall and weighs $13\,\text{kg}$. The sensors used on the robot consist of joint position encoders and an IMU. The trained student policy is optimized by using the MNN framework [34] to improve inference speed and runs on the onboard computer Jetson TX2 NX. For deployment, the control frequency is $50\,\text{Hz}$.

A. Ablation Study for the Design of the Reward Terms

To evaluate the impact of different reward terms on the gait motion, we additionally trained two ablated policies considering combinations of two types of reward ($r_t^g + r_t^s$ and $r_t^g + r_t^l$). All three policies have the same training time and random seed. Fig. 3 shows the positions and velocities of the two joints (thigh and shank) of the front-right leg, given a moving-forward command of $0.6\,\text{m/s}$ on the flat in simulation. The joint states (purple lines) of the policy trained with the total reward terms ($r_t^g + r_t^l + r_t^s$) are closer to the joint states (red lines) generated from the TO than the other two. The policies trained with only

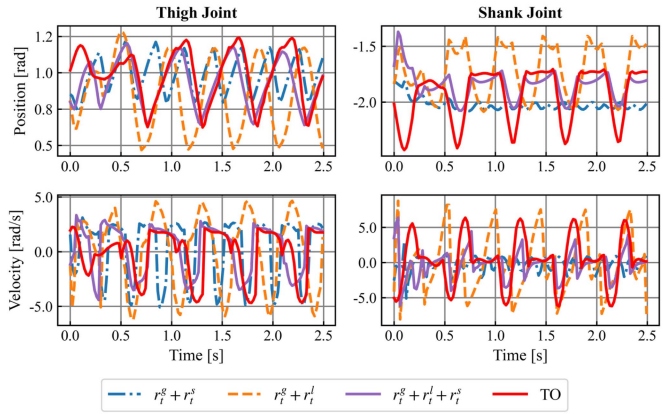

Fig. 3. Measured positions and velocities of the two joints (thigh and shank) of the front-right leg during forward walking (around $0.6\,\text{m/s}$) in simulation.

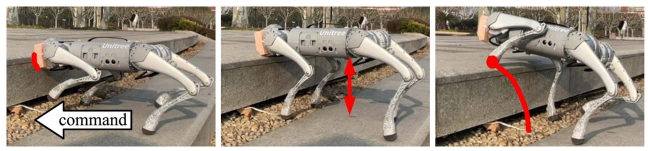

Fig. 4. Locomotion over challenging steps. Our robot learns to detect huge steps of $25\,\text{cm}$ by head collision and lifts base height and swing height to overcome obstructions.

two types of reward ($r_t^g + r_t^s$ and $r_t^g + r_t^l$) generate jerky velocities and positions (orange and blue lines). More importantly, the policy trained with the total reward terms ($r_t^g + r_t^l + r_t^s$) shows a natural and smooth gait style, as shown in the accompanying video. This indicates that using predefined gait priors based on the AMP can indeed allow the robot to learn the natural gait without limiting the ability of overcoming challenging terrains.

B. Evaluation of the Robust Locomotion

To test the robustness of our controller, we first evaluated the traversability for challenging obstacles. The obstacle shown in Fig. 4 is $25\,\text{cm}$ high, which is significantly challenging for our robot (The standing height of our robot is $28\,\text{cm}$). This height is much larger than the $8\,\text{cm}$ foot clearance in our AMP dataset which only contains locomotion trajectories during normal walking on flat terrains. To overcome this huge obstacle, our controller demonstrates a foot-trapping reflex [2], as shown in Fig. 4 and the accompanying video. The policy first identifies the head collision purely from proprioceptive observations. Then the policy raises the body height to ensure that the trunk is above the obstacle. Next, the policy lifts the foot as high as possible to step over the obstacle. Note that our controller does not use any exteroceptive sensors. The student policy analyzes the history of proprioceptive stream to be aware of the obstacle in front of the face. Hence, it can learn how to adapt itself robustly to any obstacles and disturbances that affect the robot's body configuration.

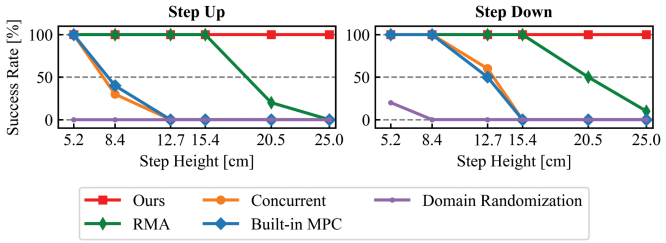

Fig. 5. Success rates of different methods for different step heights. Success rate was evaluated over 10 trials for each step height.

We next compared the presented method with the following baselines using only proprioception, in a series of single-step tests:

1) *RMA [3]:* A similar teacher-student training framework where the student policy consists of a 1-D CNN adaptation module and the low-level network copied from the teacher policy.
2) *Concurrent [8]:* The policy was concurrently trained with a state estimation network that explicitly estimates the body state. Considering the final convergence of the policy, no terrain information was provided during the training.
3) *Domain Randomization:* The policy was directly trained without any privileged state or terrain information.
4) *Built-in MPC:* The MPC controller was built in Unitree Go1 Edu.

All the RL methods above were trained using the same curriculum strategy detailed in Section III-A, the same reward functions detailed in Section II-B and the same random seed. For a fair comparison, we used the same low-level network architecture detailed in Table III for all RL methods. The CNN input sequence length for the RMA baseline was 50, with access to the same memory length as our LSTM encoder.

In each test, the robot was given a forward command of $0.4\,\mathrm{m/s}$ for $10\,\mathrm{s}$. A test was successful and passed if the robot traversed the step with both front and hind legs. We conducted 10 tests for each step height and computed the success rate. As shown in Fig. 5, our method outperformed all the other methods in stepping up and down, which was able to successfully traverse all steps up to $25\,\mathrm{cm}$. The teacher-student training framework (ours, RMA) is effective for traversing large steps due to its ability to implicitly estimate the terrain information, whereas controllers without access to terrain information (Concurrent, Domain Randomization, Built-in MPC) often result in falls when the step height is larger than $13\,\mathrm{cm}$. However, when the step height is larger than $15\,\mathrm{cm}$, the performance of the RMA decreases rapidly. This performance difference is likely due to the different treatment to the low-level network of the student policy during training. Our student policy reused the learned weights of the teacher's low-level network and continued training it using the teacher's action, while the RMA frozen the student's low-level network. As the terrain becomes more difficult, the reconstruction error of the $\boldsymbol{l}_t^e$ will be larger and the output of the student's low-level network with fixed weights copied from the teacher will be unstable.

TABLE IV
QUANTIFYING THE AVERAGED SPEED UNDER THE MAXIMUM LONGITUDINAL VELOCITY COMMAND OF $4\,\mathrm{m/s}$

Terrains	**Time Elapsed** (s)	**Speed** (m/s)
Plastic Track (Real)	2.89 ± 0.04	$\mathbf{3.46} \pm 0.04$
Rocky Road (Real)	3.19 ± 0.06	$\mathbf{3.14} \pm 0.06$
Grassland (Real)	3.65 ± 0.10	$\mathbf{2.74} \pm 0.07$

We further evaluated the robustness of the presented controller in diverse environments, as shown in Fig. 1 and the accompanying video. These environments include large curbs, dense vegetation, moderately rocky, loose rubble, and grassland. Some of these terrains can deform and crumble, with substantial variation in material properties across the surface. However, the policy learns robust locomotion based on a history of proprioceptive observations and demonstrates zero-shot generalization from simulation to the terrains with characteristics that were never experienced during training.

C. Evaluation of the Agile Locomotion

To test the agility of our controller, we first evaluated outdoor sprintings across three different terrains in the real world, including plastic track, rocky road and grassland. We recorded the time elapsed in three sprinting tests of $10\,\mathrm{m}$ to compute the average running speed. The maximum longitudinal velocity command is $4\,\mathrm{m/s}$ both in simulation and in the real world, since we do not pursue extreme high-speed locomotion, considering the safety of our robot in this work. As shown in Table IV, our robot sustained the maximum average speed of $3.46\,\mathrm{m/s}$ across the plastic track, as this terrain is more even than the others. For the rocky road and the grassland, these terrains have more variations in friction, restitution, and deformation than flats. Our policy shows strong generalization to these two terrains which were never seen in simulation, with an average speed of $3.14\,\mathrm{m/s}$ and $2.74\,\mathrm{m/s}$, respectively. We also evaluated our robot's high-speed spinning ability in the lab flat as shown in the accompanying video. The robot accelerated to a maximum yaw velocity of $5.8\,\mathrm{rad/s}$ and stopped safely. However, tracking errors still remain in the real world as reported in Table IV. This may be caused by some remaining sim-to-real gaps from robot's motor or unreal dynamics of the simulator [7].

We further evaluated the agility of the presented controller in diverse uneven environments, as shown in the accompanying video. Our robot was able to run on vegetation terrains and small stairs robustly. Since our robot does not use exteroceptive sensors like cameras and LIDARs, the tangle of branches and leaves in vegetation and small step in stairs may cause a lot of disruptions to the high-speed movements. Our robot showed impressive recovery and adaptation behaviors and continued to run over uneven terrains. These results demonstrate that one single policy can obtain both robust locomotion over challenging terrains and agile locomotion over natural terrains.

To compare our method with the previous RL methods in an overall view, Table V compares locomotion abilities across different RL controllers on different quadrupeds. Our robot can

TABLE V
OVERALL COMPARISON WITH PREVIOUS RL METHODS. MC = MINI CHEETAH

Robot	Blind	Complex Terrains	Froude [35]	Speed (m/s)
A1 [36]	Yes	Unknown	1.1	1.7
MC [7]	Yes	Unknown	5.1	3.9
ANYmal [4]	No	Yes	0.3	1.2
Ours Go1	Yes	Yes	4.2	3.5

traverse complex terrains blindly while moving fast over natural terrains, which is highly competitive compared to previous approaches.

V. CONCLUSION

Our work demonstrates that a single policy trained by RL can obtain both robust and agile locomotion, using concise reward functions and concurrent training procedure. The gait style learned from the AMP demonstrates zero-shot generalization from the motion dataset on flat terrains to challenging terrains in the real world. To the best of our knowledge, our robot is the first system capable of robust locomotion over challenging terrains and agile locomotion over natural terrains, using only proprioception.

However, blind locomotion is more of a trial and error process. Since our system only uses proprioception, it cannot perform tasks that require planning ahead of time, like efficiently climbing stairs or avoiding holes. In the future, our system could be extended with vision sensors, using exteroception to improve locomotion efficiency.

REFERENCES

[1] J. Hwangbo et al., "Learning agile and dynamic motor skills for legged robots," *Sci. Robot.*, vol. 4, no. 26, 2019, Art. no. eaau5872.

[2] J. Lee, J. Hwangbo, L. Wellhausen, V. Koltun, and M. Hutter, "Learning quadrupedal locomotion over challenging terrain," *Sci. Robot.*, vol. 5, no. 47, 2020, Art. no. eabc5986.

[3] A. Kumar, Z. Fu, D. Pathak, and J. Malik, "RMA: Rapid motor adaptation for legged robots," in *Proc. Robot.: Sci. Syst.*, 2021, Art. no. 011.

[4] T. Miki, J. Lee, J. Hwangbo, L. Wellhausen, V. Koltun, and M. Hutter, "Learning robust perceptive locomotion for quadrupedal robots in the wild," *Sci. Robot.*, vol. 7, no. 62, 2022, Art. no. eabk2822.

[5] A. Agarwal, A. Kumar, J. Malik, and D. Pathak, "Legged locomotion in challenging terrains using egocentric vision," in *Proc. 6th Annu. Conf. Robot Learn.*, 2022, pp. 403–415.

[6] W. Yu et al., "Visual-locomotion: Learning to walk on complex terrains with vision," in *Proc. Conf. Robot Learn.*, 2022, pp. 1291–1302.

[7] G. Margolis, G. Yang, K. Paigwar, T. Chen, and P. Agrawal, "Rapid locomotion via reinforcement learning," in *Proc. Robot.: Sci. Syst.*, 2022, Art. no. 022.

[8] G. Ji, J. Mun, H. Kim, and J. Hwangbo, "Concurrent training of a control policy and a state estimator for dynamic and robust legged locomotion," *IEEE Robot. Automat. Lett.*, vol. 7, no. 2, pp. 4630–4637, Apr. 2022.

[9] N. Rudin, D. Hoeller, P. Reist, and M. Hutter, "Learning to walk in minutes using massively parallel deep reinforcement learning," in *Proc. 5th Annu. Conf. Robot Learn.*, 2021, pp. 91–100.

[10] C. D. Bellicoso, F. Jenelten, C. Gehring, and M. Hutter, "Dynamic locomotion through online nonlinear motion optimization for quadrupedal robots," *IEEE Robot. Automat. Lett.*, vol. 3, no. 3, pp. 2261–2268, Jul. 2018.

[11] 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 Robot. Automat. Lett.*, vol. 3, no. 3, pp. 1560–1567, Jul. 2018.

[12] Y. Ding, A. Pandala, and H.-W. Park, "Real-time model predictive control for versatile dynamic motions in quadrupedal robots," in *Proc. Int. Conf. Robot. Automat.*, 2019, pp. 8484–8490.

[13] 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 *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2018, pp. 2245–2252.

[14] D. Chen, B. Zhou, V. Koltun, and P. Krähenbühl, "Learning by cheating," in *Proc. Conf. Robot Learn.*, 2020, pp. 66–75.

[15] X. B. Peng, P. Abbeel, S. Levine, and M. Van de Panne, "Deepmimic: Example-guided deep reinforcement learning of physics-based character skills," *ACM Trans. Graph.*, vol. 37, no. 4, pp. 1–14, 2018.

[16] X. B. Peng, E. Coumans, T. Zhang, T.-W. E. Lee, J. Tan, and S. Levine, "Learning agile robotic locomotion skills by imitating animals," in *Proc. Robot.: Sci. Syst.*, 2020, Art. no. 064.

[17] C. Florensa, Y. Duan, and P. Abbeel, "Stochastic neural networks for hierarchical reinforcement learning," in *Proc. Int. Conf. Learn. Representations*, 2017, pp. 1–17.

[18] X. B. Peng, M. Chang, G. Zhang, P. Abbeel, and S. Levine, "MCP: learning composable hierarchical control with multiplicative compositional policies," in *Proc. Adv. Neural Inf. Process. Syst.*, 2019, pp. 3686–3697.

[19] C. Yang, K. Yuan, Q. Zhu, W. Yu, and Z. Li, "Multi-expert learning of adaptive legged locomotion," *Sci. Robot.*, vol. 5, no. 49, 2020, Art. no. eabb2174.

[20] M. Bogdanovic, M. Khadiv, and L. Righetti, "Model-free reinforcement learning for robust locomotion using demonstrations from trajectory optimization," *Front. Robot. AI*, vol. 9, 2022, Art. no. 854212.

[21] Y. Fuchioka, Z. Xie, and M. van de Panne, "Opt-mimic: Imitation of optimized trajectories for dynamic quadruped behaviors," 2022, *arXiv:2210.01247*.

[22] J. Ho and S. Ermon, "Generative adversarial imitation learning," in *Proc. Adv. Neural Inf. Process. Syst.*, 2016, pp. 4572–4580.

[23] I. Goodfellow et al., "Generative adversarial networks," *Commun. ACM*, vol. 63, no. 11, pp. 139–144, 2020.

[24] X. B. Peng, Z. Ma, P. Abbeel, S. Levine, and A. Kanazawa, "AMP: Adversarial motion priors for stylized physics-based character control," *ACM Trans. Graph.*, vol. 40, no. 4, pp. 1–20, 2021.

[25] X. B. Peng, Y. Guo, L. Halper, S. Levine, and S. Fidler, "ASE: Large-scale reusable adversarial skill embeddings for physically simulated characters," *ACM Trans. Graph.*, vol. 41, no. 4, pp. 1–17, 2022.

[26] A. Escontrela et al., "Adversarial motion priors make good substitutes for complex reward functions," in *Proc. IEEE Int. Conf. Intell. Robots Syst.*, 2022, pp. 25–32.

[27] E. Vollenweider, M. Bjelonic, V. Klemm, N. Rudin, J. Lee, and M. Hutter, "Advanced skills through multiple adversarial motion priors in reinforcement learning," 2022, *arXiv:2203.14912*.

[28] C. Li, M. Vlastelica, S. Blaes, J. Frey, F. Grimminger, and G. Martius, "Learning agile skills via adversarial imitation of rough partial demonstrations," in *Proc. 6th Annu. Conf. Robot Learn.*, 2022, pp. 342–352.

[29] A. W. Winkler, "Towr–An open-source trajecetory optimizer for legged robots in C," 2018. [Online]. Available: https://github.com/ethz-adrl/towr

[30] N. Heess et al., "Emergence of locomotion behaviours in rich environments," 2017, *arXiv:1707.02286*.

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

[32] S. Ross, G. Gordon, and D. Bagnell, "A reduction of imitation learning and structured prediction to no-regret online learning," in *Proc. 14th Int. Conf. Artif. Intell. Statist.*, 2011, pp. 627–635.

[33] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[34] X. Jiang et al., "MNN: A universal and efficient inference engine," *Proc. Mach. Learn. Syst.*, vol. 2, pp. 1–13, 2020.

[35] R. M. Alexander, "The gaits of bipedal and quadrupedal animals," *Int. J. Robot. Res.*, vol. 3, no. 2, pp. 49–59, 1984.

[36] Z. Fu, A. Kumar, J. Malik, and D. Pathak, "Minimizing energy consumption leads to the emergence of gaits in legged robots," in *Proc. 5th Annu. Conf. Robot Learn.*, 2021, pp. 928–937.