ANIMAL ROBOTS

Learning quadrupedal locomotion over challenging terrain

Joonho Lee[1]*, Jemin Hwangbo[1,2], Lorenz Wellhausen[1], Vladlen Koltun[3], Marco Hutter[1]

Legged locomotion can extend the operational domain of robots to some of the most challenging environments on Earth. However, conventional controllers for legged locomotion are based on elaborate state machines that explicitly trigger the execution of motion primitives and reflexes. These designs have increased in complexity but fallen short of the generality and robustness of animal locomotion. Here, we present a robust controller for blind quadrupedal locomotion in challenging natural environments. Our approach incorporates proprioceptive feedback in locomotion control and demonstrates zero-shot generalization from simulation to natural environments. The controller is trained by reinforcement learning in simulation. The controller is driven by a neural network policy that acts on a stream of proprioceptive signals. The controller retains its robustness under conditions that were never encountered during training: deformable terrains such as mud and snow, dynamic footholds such as rubble, and overground impediments such as thick vegetation and gushing water. The presented work indicates that robust locomotion in natural environments can be achieved by training in simple domains.

INTRODUCTION

Much of the dry landmass on Earth remains impassible to wheeled and tracked machines, the stability of which can be severely compromised on challenging terrains. Quadrupedal animals, on the other hand, can access some of the most remote parts of Earth. They can choose safe footholds within their kinematic reach and rapidly change their kinematic state in response to the environment. Legged robots have the potential to traverse any terrains that their animal counterparts are able to traverse.

Dynamic locomotion in diverse, complex natural environments as shown in Fig. 1 has been a grand challenge in legged robotics. These environments have highly irregular profiles, deformable terrains, slippery surfaces, and overground obstructions. Under such conditions, existing published controllers manifest frequent foot slippage, loss of balance, and ultimately catastrophic failure. The challenge is exacerbated by the inaccessibility of accurate information about the physical properties of the terrain. Exteroceptive sensors such as cameras and LiDAR cannot reliably measure physical characteristics such as friction and compliance; are impeded by obstructions such as vegetation, snow, and water; and may not have the coverage and temporal resolution to capture changes induced by the robot itself, such as the crumbling of loose ground under the robot's feet. Under these conditions, the robot must rely crucially on proprioception—the sensing of its own bodily configuration at high temporal resolution. In response to unforeseen events such as unexpected ground contact, terrain deformation, and foot slippage, the controller must rapidly produce whole-body trajectories subject to multiple objectives: balancing, avoiding self-collision, counteracting external disturbances, and locomotion. Although animals solve this complex control problem instinctively, it remains an open challenge in robotics.

Conventional approaches to legged locomotion on uneven terrain have yielded increasingly complex control architectures. Many rely on elaborate state machines that coordinate the execution of motion primitives and reflex controllers (*1*–*5*). To trigger transitions between states or the execution of a reflex, many systems explicitly estimate states such as ground contact and slippage (*6*–*8*). Such estimation is commonly based on empirically tuned thresholds and can become erratic in the presence of unmodeled factors such as mud, snow, or vegetation. Other systems use contact sensors at the feet, which can become unreliable under field conditions (*9*–*11*). Overall, conventional systems for legged locomotion on rough terrain escalate in complexity as more scenarios are taken into account, have become extremely laborious to develop and maintain, and remain vulnerable to situations that are beyond the design implementation of their controller (corner cases).

Model-free reinforcement learning (RL) has recently emerged as an alternative approach in the development of locomotion controller for legged robots (*12*–*14*). The idea of RL is to tune a controller to optimize a given reward function. The optimization is performed on data acquired by executing the controller itself, which improves with experience. RL has been used to simplify the design of locomotion controllers, automate parts of the design process, and learn behaviors that could not be engineered with prior approaches (*12*–*15*).

However, application of RL to legged locomotion has largely been confined to laboratory environments and conditions. Our prior work demonstrated end-to-end learning of locomotion and recovery behaviors, but only on flat ground, in the laboratory (*12*). Other work also developed RL techniques for legged locomotion but likewise focused largely on flat or moderately textured surfaces in laboratory settings (*13*, *14*, *16*–*19*).

Here, we present a robust controller for blind quadrupedal locomotion on challenging terrain. The controller uses only proprioceptive measurements from joint encoders and an inertial measurement unit, which are the most durable and reliable sensors on legged machines. The operation of the controller is shown in Fig. 1 and Movie 1. The controller was used to drive two generations of ANYmal quadrupeds (*20*) in a variety of environments that are beyond the reach of prior published work in legged robotics. The quadruped reliably trots through mud, sand, rubble, thick vegetation, snow, running water, and a variety of other off-road terrains. The same controller was also used in our entry in the Defense Advanced Research

[1]Robotic Systems Lab, ETH-Zürich, Zürich, Switzerland. [2]Robotics & Artificial Intelligence Lab, KAIST, Deajeon, Korea. [3]Intelligent Systems Lab, Intel, Santa Clara, CA, USA.
*Corresponding author. Email: jolee@ethz.ch

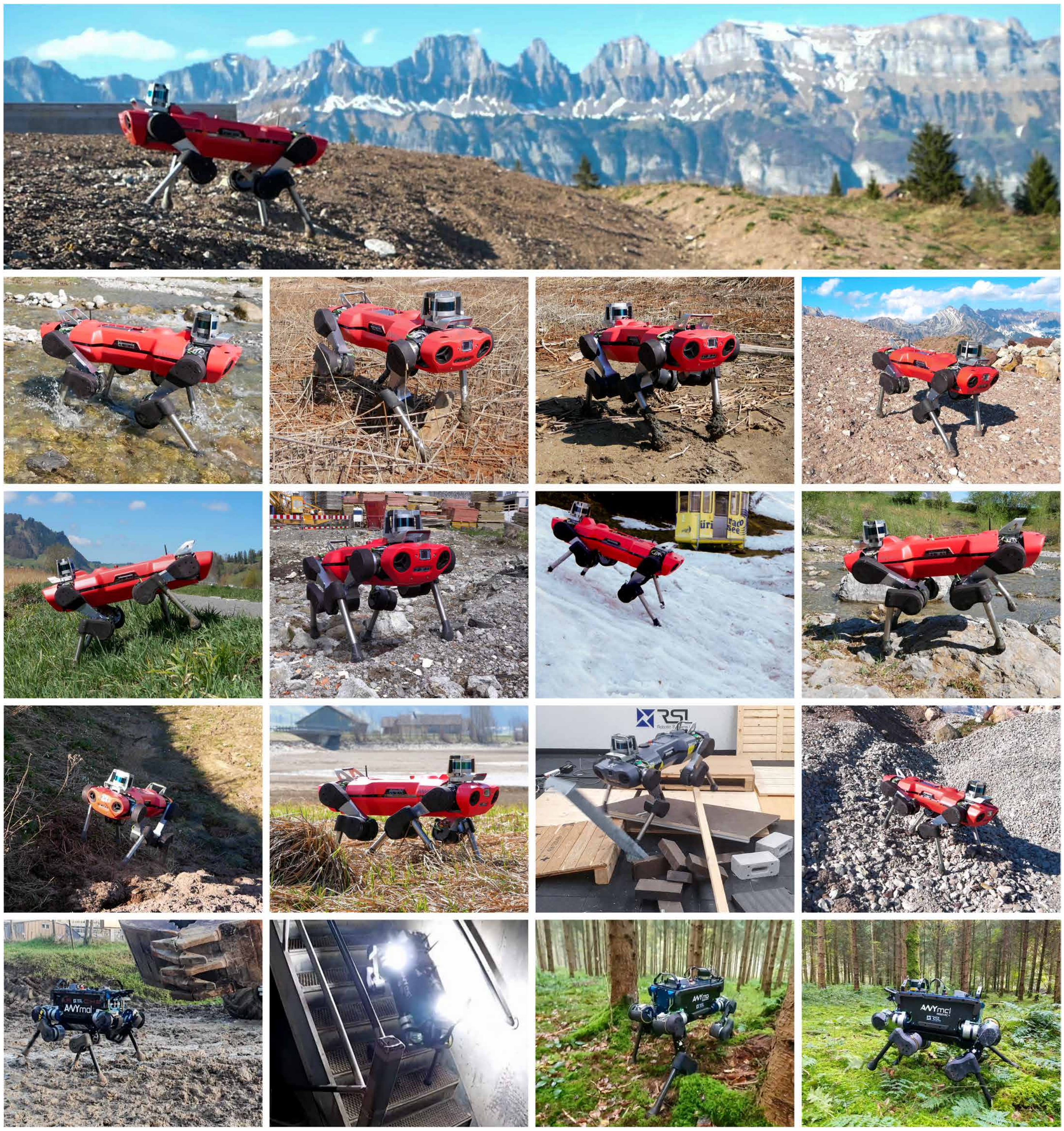

Fig. 1. Deployment of the presented locomotion controller in a variety of challenging environments.

Projects Agency (DARPA) Subterranean Challenge Urban Circuit. In all deployments, robots of the same generation were driven by exactly the same controller under all conditions. No tuning was required to adapt to different environments.

Like a number of prior applications of model-free RL to legged locomotion, we trained the controller in simulation (*12*, *14*, *16*). Prior efforts have established a number of practices for successful transfer of legged locomotion controllers from simulation to physical machines. One is realistic modeling of the physical system, including the actuators (*12*). Another is randomization of physical parameters that vary between simulation and reality, such that the controller becomes robust to a range of conditions that cover those that arise in physical

Movie 1. Robot in the wild. A learning-based locomotion controller enables a quadrupedal ANYmal robot to traverse challenging natural environments.

deployment, without the necessity to precisely model these conditions a priori (*21*).

We used these ideas as well but found that they were not sufficient to achieve robust locomotion on rough terrain. We therefore introduced and validated a number of additional approaches that are crucial to realizing the presented skills. The first is a different policy architecture. Rather than using a multilayer perceptron (MLP) that operates on a snapshot of the robot's current state, as was common in prior work, we used a sequence model, specifically a temporal convolutional network (TCN) (*22*) that produces actuation based on an extended history of proprioceptive states. We did not use explicit contact and slip estimation modules, which are known to lack robustness in challenging situations; rather, the TCN learns to implicitly reason about contact and slippage events from proprioceptive history as needed.

The second key concept that enables the demonstrated results is privileged learning (*23*). We found that training a rough-terrain locomotion policy directly via RL was not successful: The supervisory signal was sparse, and the presented network failed to learn locomotion within a reasonable time budget. Instead, we decomposed the training process into two stages. First, we trained a teacher policy that has access to privileged information, namely ground-truth knowledge of the terrain and the robot's contact with it. The privileged information enables the policy to quickly achieve high performance. We then used this privileged teacher to guide the learning of a purely proprioceptive student controller that only uses sensors that are available on the real robot. This privileged learning protocol is enabled by simulation, but the resulting proprioceptive policy is not confined to simulation and is deployed on physical machines.

The third concept that has been important in achieving the presented levels of robustness is an automated curriculum that synthesizes terrains adaptively, based on the controller's performance at different stages of the training process. In essence, terrains were synthesized such that the controller is capable of traversing them while becoming more robust. We evaluated the traversability of parameterized terrains and used particle filtering to maintain a distribution of terrain parameters of medium difficulty (*24*, *25*) that adapt as the neural network learns. The training conditions grew increasingly more challenging, yielding an omnidirectional controller that combines agility with unprecedented resilience.

The result is a legged locomotion controller that can robustly traverse complex natural terrains that are often unreachable by existing methods. The controller is consistently effective in zero-shot generalization settings. That is, it remains robust when tested under conditions that were never encountered during training. Our training in simulation only uses rigid terrains and a small set of procedurally generated terrain profiles, such as hills and steps. However, when deployed on physical quadrupeds, the controller successfully handled deformable terrains (mud, moss, and snow); dynamic footholds (stepping on a rolling board in a cluttered indoor environment or debris in the field); and overground impediments such as thick vegetation, rubble, and gushing water. Our methodology may lead to future developments for legged robotics. Moreover, our results suggest that the extraordinary complexity of the physical world can be tamed without brittle and painstaking modeling or dangerous and expensive trial and error under real-world field conditions.

RESULTS

Movie 1 summarizes the results of the presented work. We have deployed the trained locomotion controller on two generations of ANYmal robots: ANYmal-B (Fig. 2, D to G) and ANYmal-C (Figs. 2, A to C, and 3). The robots have different kinematics, inertia, and actuators.

Natural environments

The presented controller has been deployed in diverse natural environments, as shown in Fig. 1, Movie 1, and movie S1. These include steep mountain trails, creeks with running water, mud, thick vegetation, loose rubble, snow-covered hills, and a damp forest. A number of specific scenarios are further highlighted in Fig. 2 (A to F). These environments have characteristics that the policy did not experience during training. The terrains can deform and crumble, with substantial variation of material properties over the surface. The robot's legs are subjected to frequent disturbances due to vegetation, rubble, and sticky mud. Existing terrain estimation pipelines that use cameras or LiDAR (*26*) failed in environments with snow (Fig. 2A), water (Fig. 2C), or dense vegetation (Fig. 2F). Our controller does not rely on exteroception and is immune to failures related to exteroceptive sensing. The controller learns omnidirectional locomotion based on a history of proprioceptive observations and is robust in zero-shot deployment on terrains with characteristics that were never experienced during training.

We have compared the presented controller to a state-of-the-art baseline (*1*, *27*) in the forest environment. The baseline could traverse flat and unobstructed patches but failed frequently upon encountering loose branches, thick vegetation, and mud, as shown in movie S1. Our controller never failed in these experiments.

We have quantitatively evaluated the presented controller and the baseline in three conditions: moss, mud, and vegetation (Fig. 2, D to F). We have measured locomotion speed and energy efficiency. The results are reported in Table 1. The presented controller achieves higher locomotion speed under all conditions. We computed the dimensionless cost of transport (COT) to compare the efficiency of the controllers at different speed ranges. We define mechanical COT as $\sum_{12\ \text{actuators}} [\tau\dot{\theta}]^{+}/(mgv)$. τ denotes joint torque, $\dot{\theta}$ is joint speed, mg is the total weight, and v is the locomotion speed. This quantity represents positive mechanical power exerted by the actuator per unit weight and unit locomotion speed (*28*). As shown in Table 1, the presented controller is more energy efficient, with a lower COT than the baseline.

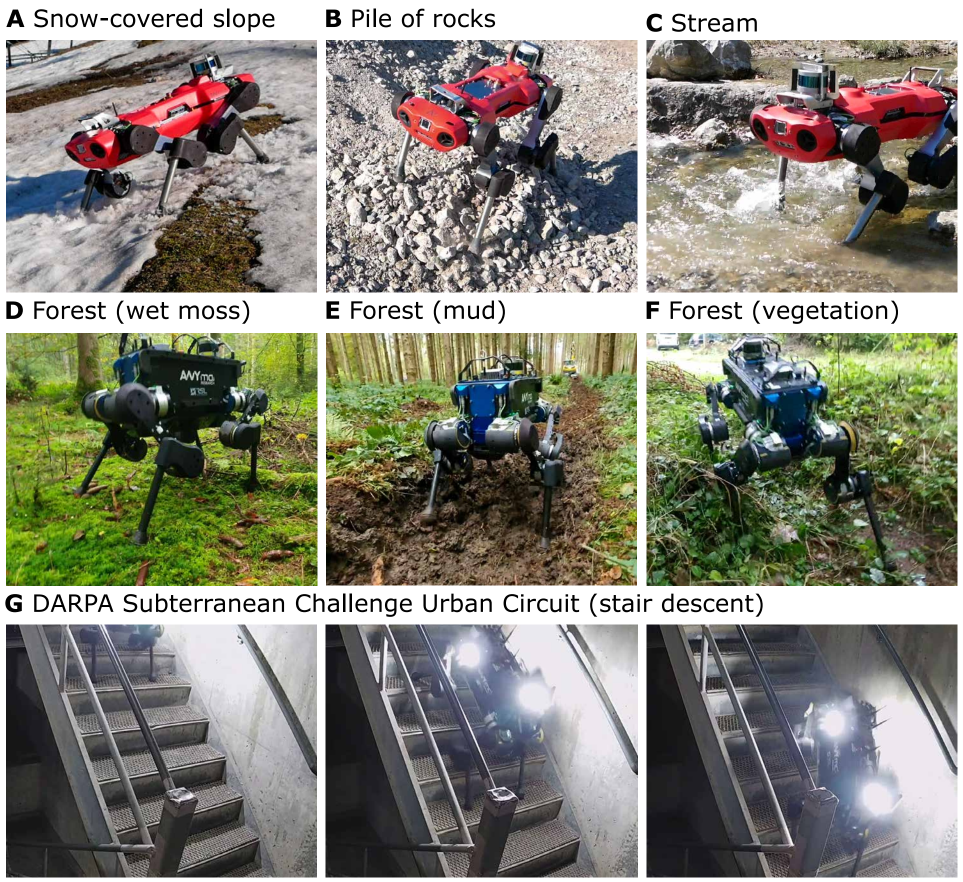

Fig. 2. A number of specific deployments. (**A** to **F**) Zero-shot generalization to slippery and deforming terrains. (**G**) Steep descent during the DARPA Subterranean Challenge. The stair rise is 18 cm, and the slope is ~45°.

The quantitative evaluation reported in Table 1 understates the difference between the two controllers because it only measured speed and energetic efficiency of the baseline during successful locomotion. The baseline's catastrophic failures were not factored into these measurements: When the baseline failed, it was reset by a human operator in a more stable configuration. Catastrophic failures of the baseline controller due to thick vegetation and other factors are shown in movie S1. Our controller exhibited no such failures.

DARPA Subterranean Challenge

Our controller was used by the Cerberus team for the DARPA Subterranean Challenge Urban Circuit (Fig. 2G). It replaced a model-based controller that had been used by the team in the past (*1*, *27*). The objective of the competition is to develop robotic systems that rapidly map, navigate, and search complex underground environments, including tunnels, urban underground, and cave networks. The human operators are not allowed to assist the robots during the competition physically; only teleoperation is allowed. Accordingly, the locomotion controller needs to perform without failure over extended mission durations.

The presented controller drove two ANYmal-B robots in four missions of 60 min. The controller exhibited a zero failure rate throughout the competition. A steep staircase that was traversed by one of the robots during the competition is shown in Fig. 2G.

Indoor experiments

We further evaluated the robustness of the presented controller in an indoor environment populated by loose debris, as shown in Fig. 3A. Support surfaces are unstable, and the robot's feet frequently slip. Such conditions can be found at disaster sites and construction zones, where legged robots are expected to operate in the future.

Results are shown in Fig. 3A and movie S2. The robot moved omnidirectionally over the area. The presented controller could stably locomote over shifting support surfaces. This level of robustness is beyond the reach of prior controllers for ANYmal robots (*1*, *27*) and is comparable to the state of the art (*2*, *29*).

The learned controller manifests a foot-trapping (FT) reflex, as shown in Fig. 3B and movie S3. The policy identifies the trapping of the foot purely from proprioceptive observations and lifts the foot over the obstacle. Such reflexes were not specified in any way during training: They developed adaptively. This distinguishes the presented approach from conventional controller design methods, which explicitly build in such reflexes and orchestrate their execution by a higher-level state machine (*1*, *3*). The step shown in Fig. 3B is 16.8-cm high, which is higher than the foot clearance of the legs during normal walking on flat terrain. The maximum foot clearance on flat terrain is 12.9 and 13.6 cm for the left-front (LF) and right-front legs, respectively, and increases up to 22.5 and 18.5 cm in the case of FT. Our controller also learns to adapt the hind leg trajectories when stepping up. The maximum foot clearance on flat terrains is 13.5 and 9.06 cm for the left-hind and right-hind legs and increases up to 16.6 and 15.9 cm when the front legs are above the step. Further analysis is provided in Materials and Methods. Note also that the reflexes learned by our controller are more general and are not tied to particular contact events. Figure 3C shows the controller responding to a mid-shin collision during the swing phase. Here, the trapping event was not signaled by foot contact, and scripted controllers that use foot contact events as triggers would not appropriately handle this situation. Our controller, on the other hand, analyzes the proprioceptive stream as a whole and is trained without making assumptions about possible contact locations. Hence, it can learn to react to any obstructions and disturbances that affect the robot's bodily configuration.

We now focus on comparing the presented approach with the baseline (*1*, *27*) in controlled settings. We first compared the robustness of the controllers in the diagnostic setting of a single step, as shown in Fig. 3D. In each trial, the robot was driven straight to a step for 10 s. A trial was a success if the robot traversed the step with both front and hind legs. We conducted 10 trials for each step height and computed the success rate. Because the baseline controller takes

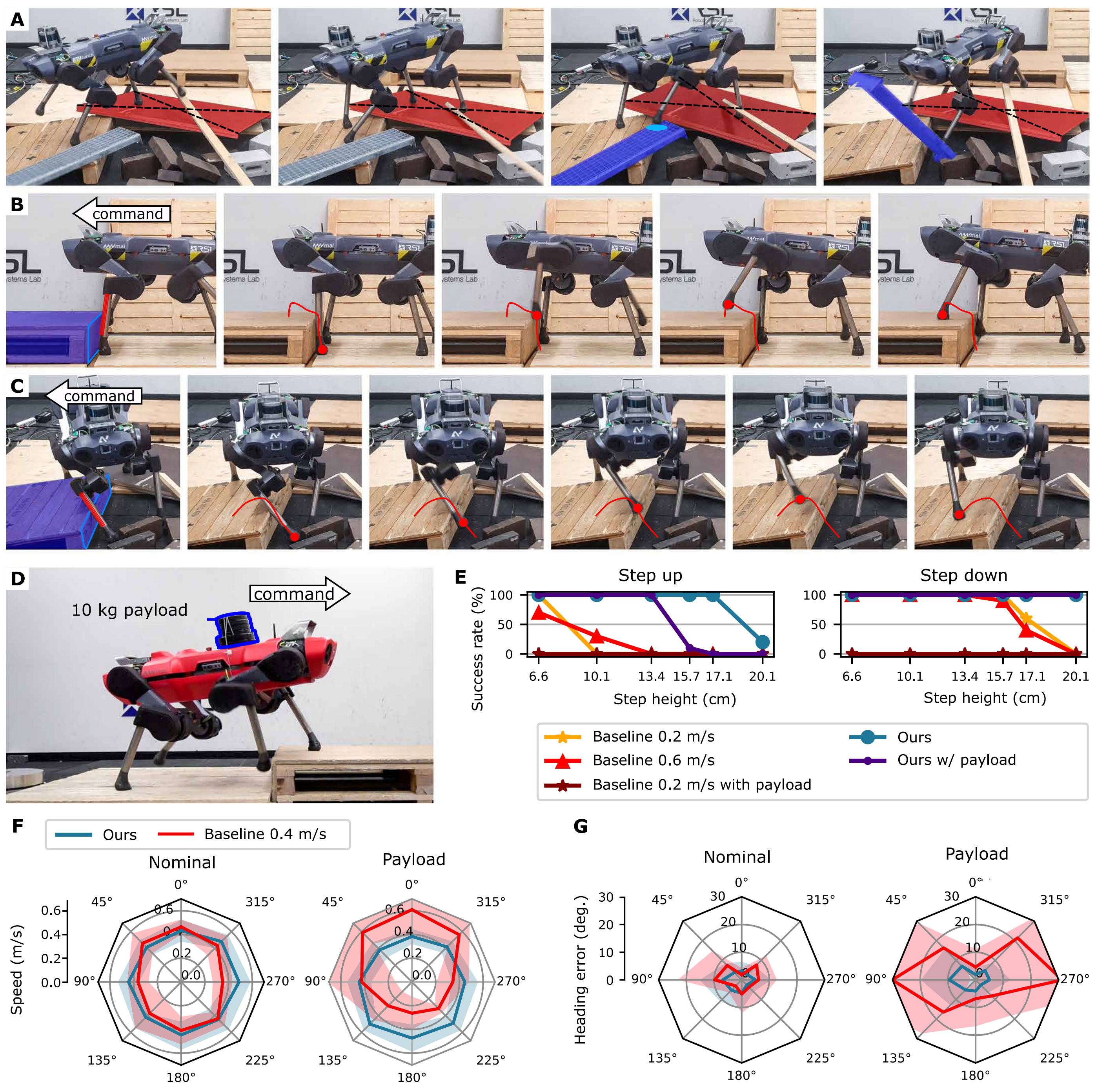

Fig. 3. Evaluation in an indoor environment. (**A**) Locomotion over unstable debris. The robot steps onto loose boards (highlighted in red and blue) that dislodge under the robot's feet. (**B**) The policy exhibits a foot-trapping reflex and overcomes a 16.8-cm step. (**C**) The policy learns to appropriately handle obstructions irrespective of the contact location. Here, it is shown reacting to an obstacle that is encountered mid-shin during the swing phase. (**D**) Controlled experiments with steps and payload. Our controller and a baseline (*1*, *27*) were commanded to walk over a step with and without the 10-kg payload. (**E**) Success rates for different step heights. The success rate was evaluated over 10 trials for each condition. (**F**) Mean linear speeds for different command directions on flat terrain. 0° refers to the front of the robot. Shaded areas denote 95% CIs. (**G**) Mean heading errors for different command directions on flat terrain. Shaded areas denote 95% CIs.

a desired linear velocity of the base as input, we commanded a forward velocity of 0.2 and 0.6 m/s. The maximum speed of the baseline is 0.6 m/s. The success rates are given in Fig. 3E. The presented controller outperformed the baseline in stepping both up and down. The baseline showed high sensitivity to FT, which often led to a fall, as shown in movie S3.

We also tested the controllers in the presence of substantial model mismatch. We attached a 10-kg payload, as shown in Fig. 3D and

Table 1. Comparison of locomotion performance in natural environments. The mechanical COT is computed using positive mechanical power exerted by the actuators.

Quantity	Controller	Terrain		
		Moss	Mud	Vegetation
Average speed (m/s)	Ours	**0.452**	**0.338**	**0.248**
	Baseline	0.199	0.197	–
Average mechanical COT	Ours	**0.423**	**0.692**	**1.23**
	Baseline	0.625	0.931	–

movie S4. This payload was 22.7% of the total weight of the robot and was never simulated during training. As shown in Fig. 3E, the presented controller could still traverse steps up to 13.4 cm despite the model mismatch. The baseline was incapable of traversing any steps under any command speed with the payload.

We then evaluated the tracking performance of the controllers on flat ground with the payload. We commanded each controller in eight directions and measured the locomotion speed and the tracking error. Target speed was fixed to 0.4 m/s for the baseline controller, which is similar to the operating speed of the presented controller. In Fig. 3F, we show the velocity profiles of the controllers. Our controller locomoted at around 0.4 m/s in all directions and performed similarly with the payload. On the other hand, the locomotion speed of the baseline varied with direction, which can be seen by the anisotropic velocity profile, and the velocity profile shifted largely off center with the payload. Figure 3G shows the heading error of the controllers in each commanded direction. The heading error is the angle between the command velocity and the base velocity of the robot. The heading error of the presented controller was consistently smaller than the baseline, both with and without the payload. The baseline's error in the lateral direction reached ~30°, and the baseline failed when a speed of 0.6 m/s was commanded, as shown in movie S4. In contrast, the average heading error of the presented controller stayed within 10° with or without the payload. We conclude that the presented controller is much more robust to model mismatch.

Next, we tested robustness to foot slippage. To introduce slippage, we used a moistened whiteboard (*1*). The results are shown in movie S5. The baseline quickly lost balance, aggressively swung the legs, and fell. In contrast, the presented controller adapted to the slippery terrain and successfully locomoted in the commanded direction.

DISCUSSION

The presented results substantially advance the published state of the art in legged robotics. Beyond the results themselves, the methodology presented in this work can have broad applications. Before our work, a hypothesis could be held that training in simulation is fundamentally constrained by the limitations of simulation environments in representing the complexity of the physical world. Existing technology is severely limited in its ability to simulate compliant contact, slippage, and deformable and crumbling terrains. As a result, phenomena such as mud, snow, thick vegetation, gushing water, and many others are beyond the capabilities of robotics simulation frameworks (*30*–*32*). The sample complexity of model-free RL algorithms, which commonly require millions of time steps for training, further exacerbates the challenge by precluding reliance on frameworks that may require seconds of computation per time step.

Our work demonstrates that simulating the abundant variety of the physical world may not be necessary. Our training environment featured only rigid terrain, with no compliance or overground obstructions such as vegetation. Nevertheless, controllers trained in this environment successfully met the diversity of field conditions encountered at deployment.

We see a number of limitations and opportunities for future work. First, the presented controller only exhibited the trot gait. This is narrower than the range of gait patterns found by quadrupeds in nature (*33*). The gait pattern is constrained in part by the kinematics and dynamics of the robot, but the ANYmal machines are physically capable of multiple gaits (*27*). We hypothesize that training protocols and objectives that emphasize diversity can elicit these.

Second, the presented controller relies solely on proprioception. This is a notable advantage in that the controller makes few assumptions on the sensor suite and is not susceptible to failure when exteroception breaks down. Existing work has argued that a blind (proprioceptive) controller should form the basis of a legged locomotion stack (*3*). Nevertheless, blind locomotion is inherently limited. If the machine is commanded to walk off a cliff, it will. Even under less extreme conditions, the robot's gait is fairly conservative because it must, by necessity, feel out the environment with its body as it locomotes. A major opportunity for future work is to use the presented methodology as a starting point in the development of a hybrid proprioceptive-exteroceptive controller that, like many animals, will be able to locomote even when vision and other external senses are disrupted but will use exteroceptive data when provided. This will enable legged machines to autonomously traverse environments that may have fatal elements, such as cliffs, and to raise speed and energetic efficiency under safer conditions. More broadly, the presented results expedite the deployment of legged machines in environments that are beyond the reach of wheeled and tracked robots and are dangerous or inaccessible to humans, whereas the presented methodology opens new frontiers for training complex robotic systems in simulation and deploying them in the full richness and complexity of the physical world.

MATERIALS AND METHODS

Overview

The main objective of the presented controller is to locomote over rough terrain following a command. The command is given either by a human operator or by a higher-level navigation controller. In our formulation, unlike many existing works (*12*, *14*, *16*) that focus on tracking the target velocity of the base (${}_{IB}^{B}v_T$), only the direction (${}_{IB}^{B}\hat{v}_T$) is given to the controller. The reason is that the feasible range of target speeds is often unclear on challenging terrain. For example, the robot can walk faster downhill than uphill.

The command vector is defined as $\langle ({}_{IB}^{B}\hat{v}_T)_{xy}, (\hat{\omega}_T)_z \rangle$. The first part is the target horizontal direction in base frame $({}_{IB}^{B}\hat{v}_T)_{xy} := \langle \cos(\psi_T), \sin(\psi_T) \rangle$, where ψ_T is the yaw angle to command direction in the base frame. The stop command is defined as $\langle 0.0, 0.0 \rangle$. The second part is the turning direction $(\hat{\omega}_T)_z \in \{-1, 0, 1\}$; 1 refers to counterclockwise rotation along the base z axis.

An overview of our method is given in Fig. 4. We use a privileged learning strategy inspired by "learning by cheating" (*23*) (Fig. 4A).

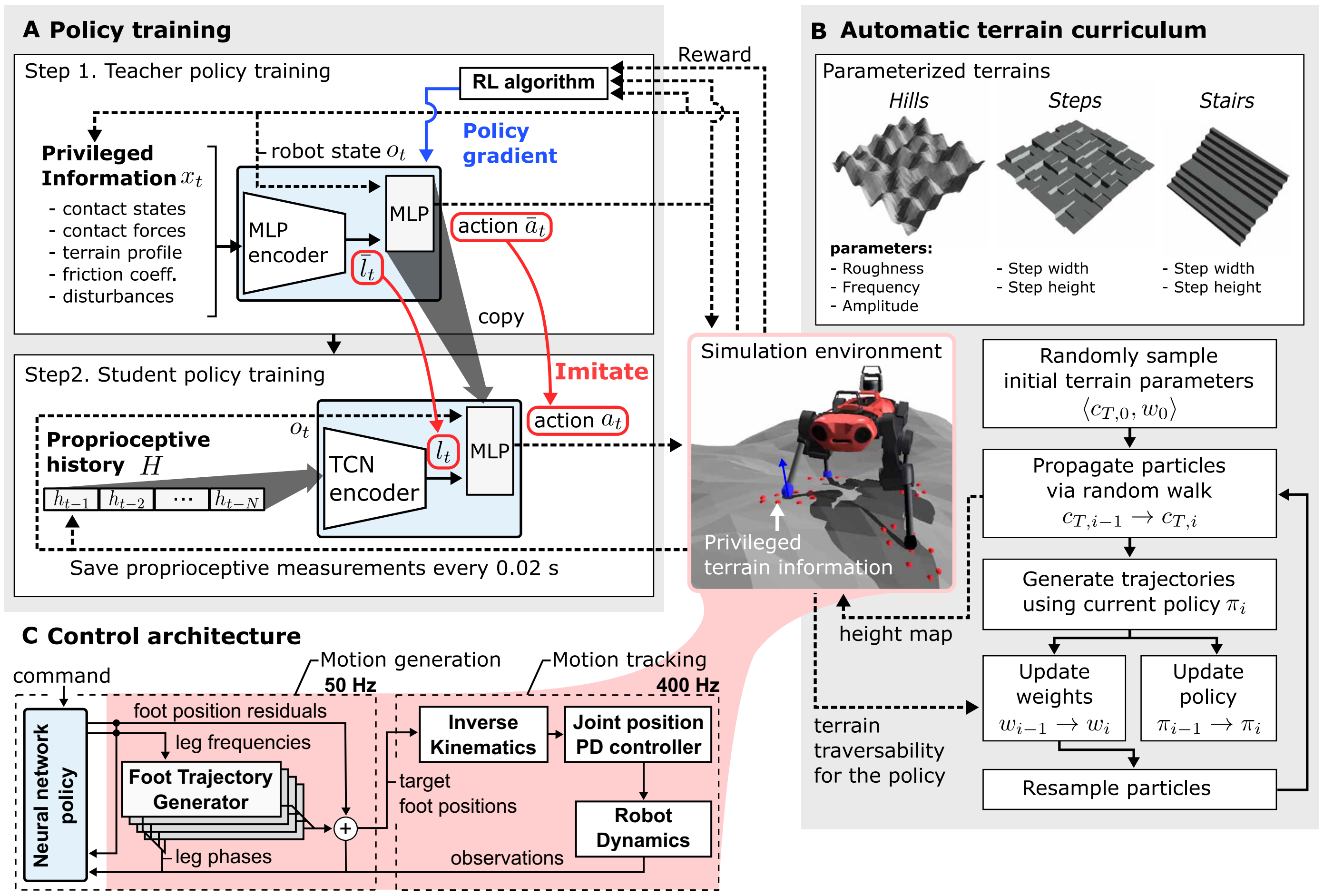

Fig. 4. Creating a proprioceptive locomotion controller. (**A**) Two-stage training process. First, a teacher policy is trained using RL in simulation. It has access to privileged information that is not available in the real world. Next, a proprioceptive student policy learns by imitating the teacher. The student policy acts on a stream of proprioceptive sensory input and does not use privileged information. (**B**) An adaptive terrain curriculum synthesizes terrains at an appropriate level of difficulty during the course of training. Particle filtering was used to maintain a distribution of terrain parameters that are challenging but traversable by the policy. (**C**) Architecture of the locomotion controller. The learned proprioceptive policy modulates motion primitives via kinematic residuals. An empirical model of the joint position PD controller facilitates deployment on physical machines.

We first train a teacher policy that has access to privileged information concerning the terrain. This teacher policy is then distilled into a proprioceptive student policy that does not rely on privileged information. The privileged teacher policy is confined to simulation, but the student policy is deployed on physical machines. One difference of our methodology from that of Chen *et al.* (*23*) is that we do not rely on expert demonstrations to train the privileged policy; rather, the teacher policy is trained via RL.

The privileged teacher model is based on MLPs that receive information about the current state of the robot, properties of the terrain, and the robot's contact with the terrain. The model computes a latent embedding $\bar{l}_t$ that represents the current state and an action $\bar{a}_t$. The training objective rewards locomotion in prescribed directions.

After the teacher policy is trained, it is used to supervise a proprioceptive student policy. The student model is a TCN (*22*) that receives a sequence of *N* proprioceptive observations as input. The student policy is trained by imitation. The vectors $\bar{l}_t$ and $\bar{a}_t$ computed by the teacher policy are used to supervise the student. This is illustrated in Fig. 4A.

Training is conducted on procedurally generated terrains in simulation. The terrains are synthesized adaptively to facilitate learning according to the skill level of the trained policies at any given time. We define a traversability measure for each terrain and develop a sampling-based method to select terrains with the appropriate difficulty during the course of training. We use particle filtering to maintain an appropriate distribution of terrain parameters. This is illustrated in Fig. 4B. The terrain curriculum is applied during both teacher and student training.

Our control architecture is shown in Fig. 4C. We use the Policies Modulating Trajectory Generators (PMTG) architecture (*34*) to provide priors on motion generation. The neural network policy modulates leg phases and motion primitives by synthesizing residual position commands.

The simulation uses a learned dynamics model of the robot's joint position proportional-derivative (PD) controller (*12*). This facilitates the transfer of policies from simulation to reality. After training in simulation, the proprioceptive controller is deployed directly on physical legged machines, with no fine-tuning.

Motion synthesis

We now elaborate on the control architecture that is illustrated in Fig. 4C. It is divided into motion generation and tracking. The input to our controller consists of the command vector and a sequence of proprioceptive measurements including base velocity, orientation, and joint states. The controller does not use any exteroceptive input (e.g., no haptic sensors, cameras, or depth sensors). The input also does not contain any handcrafted features such as foot contact states or estimated terrain geometry. The controller outputs joint position targets.

Our motion generation strategy is based on the periodic leg phase. Previous works commonly leveraged predefined foot contact schedules (*2*, *27*, *35*). We define a periodic phase variable $\phi_i \in [0.0, 2\pi)$ for each leg, which represents contact phase if $\phi \in [0.0, \pi)$ and swing phase if $\phi \in [\pi, 2\pi)$. At every time step t, $\phi_i = (\phi_{i,0} + (f_0 + f_i)t)(\bmod 2\pi)$ where $\phi_{i,0}$ is the initial phase, f_0 is a common base frequency, and f_i is the frequency offset for the ith leg. We want the legs to manifest periodic motions when $f_0 + f_i \neq 0$ and engage ground contact in contact phase. We set f_0 as 1.25 Hz, which is the value used by a previously developed conventional controller for a trot gait (*27*).

The target foot positions, which are the output of the motion generation block, are defined in the horizontal frames (*35*) of the feet (H_i, $i \in \{1,2,3,4\}$). H_i is a reference frame that is attached below the hip joint of the ith leg. The distance equals the nominal reach of the leg. The z axis of the frame (${}^{H_i}z$) is parallel to e_g, and ${}^{H_i}x$ is the projection of the base x axis (${}^{B}x$) onto the horizontal plane, i.e., the frame has the same yaw angle with the robot. The roll and pitch angles of H_i are decoupled from the base. This kinematic trick reduces the effect of base attitude on the foot motions (*35*) and consequently stabilizes training. Defining the output in H_i results in less premature termination at the beginning of the policy training, when the base motion is unstable due to random actions. Another benefit is that we can decompose the action distribution of the stochastic policy in the lateral and vertical directions during policy training. We applied larger noise in the lateral direction to promote exploration along the ground surface.

We use the PMTG architecture (*34*) to integrate a neural network to regulate the controller. Our implementation consists of four identical foot trajectory generators (FTGs) and a neural network policy. The FTG is a function $F(\phi) : [0.0,2\pi) \rightarrow \mathbb{R}^3$ that outputs foot position targets for each leg. The FTG drives vertical stepping motion when f_i is nonzero. The definition of $F(\phi)$ is given in section S3. The policy outputs f_is and target foot position residuals ($\Delta r_{fi,\,T}$), and the target foot position for the ith foot is $r_{fi,\,T} \coloneqq F(\phi_i) + \Delta r_{fi,\,T}$.

The tracking control is done using analytic inverse kinematics (IK) and joint position control. Each foot position target defined in H_i is first expressed in the robot base frame, and the joint position targets are computed using analytic IK. The joint position targets are then tracked by joint position PD controllers. The main reason for using analytic IK is to maximize computational efficiency and to reuse existing position control actuator models (*6*, *15*) for the sim-to-real transfer.

Teacher policy

We formulate the control problem as a Markov Decision Process (MDP). MDP is a mathematical framework for modeling discrete-time control processes in which the evolution of the state and the outcomes are partly stochastic. An MDP is defined by a state space $\mathcal{S}$, action space $\mathcal{A}$, a scalar reward function $\mathcal{R}(s_t, s_{t+1})$, and the transition probability $P(s_{t+1} \mid s_t, a_t)$. A learning agent selects an action a_t from its policy $\pi(a_t \mid s_t)$ and receives a reward r_t from the environment. The objective of the RL framework is to find an optimal policy π^* that maximizes the discounted sum of rewards over an infinite time horizon.

Assuming that the environment is fully observable to the teacher, we formulate locomotion control as an MDP and use an off-the-shelf RL method (*36*) to solve it. In this section, we provide the MDP for teacher training, which is defined by a tuple of state space, action space, transition probability, and reward function.

The state is defined as $s_t \coloneqq \langle o_t, x_t \rangle$, where o_t is the observation vector obtainable from the robot, and x_t is the privileged information that is usually not available in the real world. The detailed definitions are given in table S4. o_t contains command, orientation, base twist, joint positions and velocities, ϕ_i values, f_i values, and previous foot position targets. Joint position errors and velocities measured at −0.01 and −0.02 s are contained in o_t, which is the same as the input to the learned model of the joint position PD controller. This information allows the policy to exploit the actuator dynamics (*12*). To encode the leg phase, we use $\langle \cos(\phi), \sin(\phi) \rangle$ instead of ϕ, which is a smooth and unique representation for the angle. Previous foot position targets are also fed back to the policy and are used to compute the target smoothness reward that is explained in section S4. When the student controller is deployed, the quantities in o_t are replaced with readings from the proprioceptive sensors, and the base velocity and orientation are provided by a state estimator (*37*). x_t contains noiseless information that we receive directly from a physics engine. x_t mainly consists of information related to foot-ground interactions such as terrain profile, foot contact states and forces, friction coefficients, and external disturbance forces applied during training. Specifically, we represent the terrain profile with the elevation of nine scan points around each foot, which are symmetrically placed along a circle with a 10-cm radius (visualized in Fig. 4).

The action ($\overline{a}_t$) is a 16-dimensional (16D) vector consisting of leg frequencies and foot position residuals. The reward function is defined such that an RL agent receives a higher reward if it advances faster toward the goal. The reward function is specified in detail in section S4.

The policy network is constructed by two MLP blocks as shown in Fig. 4A. The MLP encoder embeds x_t into a latent vector $\bar{l}_t$. The command and robot states are not included in x_t, so $\bar{l}_t$ contains only the terrain- and contact-related features. We hypothesize that $\bar{l}_t$ drives adaptive behaviors such as changing foot clearance depending on the terrain profile. Then, $\bar{l}_t$ and o_t are provided to the subsequent MLP layers to compute action.

The Trust Region Policy Optimization (TRPO) (*36*) algorithm is used for training. The hyperparameters we used are given in table S7.

Student policy

The proprioceptive student policy only has access to o_t. A key hypothesis here is that the latent features $\bar{l}_t$ can be (partially) recovered from a time series of proprioceptive observations, h_t, which is defined as $h_t \coloneqq o_t \backslash \{f_o, \text{jointhistory}, \text{previousfootpositiontargets}\}$.

The student policy uses a TCN (*22*) encoder. The input to the TCN encoder is $H = \{h_{t-1}, \ldots, h_{t-N-1}\}$, where N is the history length. The encoder is fully convolutional and consists of three dilated causal convolutional layers, interleaved with strided convolutional layers that reduce dimensionality. The architecture is specified in tables S5 and S6.

We use the TCN architecture because it affords transparent control over the input history length, can accommodate long histories, and is known to be robust to hyperparameter settings (*22*). A comparison with a recurrent neural network architecture is provided in section S8.

The student policy is trained via supervised learning. The loss function is defined as

$$\mathcal{L} \coloneqq (\bar{a}_t(o_t, x_t) - a_t(o_t, H))^2 + (\bar{l}_t(o_t, x_t) - l_t(H))^2 \quad (1)$$

Quantities marked by a bar $(\bar{\cdot})$ denote target values generated by the teacher. We use the dataset aggregation strategy (DAgger) (*38*). Specifically, training data are generated by rolling out trajectories by the student policy. For each visited state, the teacher policy computes its embedding and action vectors $(\bar{\cdot})$. These outputs of the teacher policy are used as supervisory signals associated with the corresponding states. The hyperparameters we used are given in table S8.

Adaptive terrain curriculum

Our method is inspired by automatic curriculum learning (ACL) for RL agents (*25*, *39*). The Paired Open-Ended Trailblazer (POET) approach (*25*) generates diverse parameterized terrains for a 2D bipedal agent. The method uses minimal criteria (*24*, *40*) and aims to choose environmental parameters that are neither too challenging nor trivial for the agents: This is realized by selecting task parameters that yield midrange rewards. Florensa *et al.* (*39*) similarly choose achievable yet difficult goals for RL agents.

Our method likewise realizes a training curriculum that gradually modifies a distribution over environmental parameters such that the policy can continuously improve locomotion skills and generalize to new environments. Our work differs from POET because POET aims for open-ended search in the space of possible problems and evolves a population of specialized agents, whereas we seek to obtain a single generalist agent.

Figure 4B shows the types of terrains used in our training environment. Each terrain is generated by a parameter vector $c_T \in \mathcal{C}$. The terrains are described in detail in section S5. Our ACL method approximates a distribution of desirable c_T values using a particle filter.

We first describe how a given c_T is evaluated in simulation. Instead of directly using the reward function to evaluate the learning progress (*25*, *41*–*43*), we evaluate c_T values by the traversability of generated terrains, which is defined as the success rate of traversing a terrain. We found traversability to be more intuitive than the reward function, which consists of multiple objectives that are often unbounded. We first define a labeling function ν as

$$\nu(s_t, a_t, s_{t+1}) = \begin{cases} 1 & \text{if} \quad v_{\mathrm{pr}}(s_{t+1}) > 0.2 \\ 0 & \text{if} \quad v_{\mathrm{pr}}(s_{t+1}) < 0.2 \vee \text{termination} \end{cases} \quad (2)$$

for a state transition from s_t to s_{t+1}. $v_{\mathrm{pr}}(s_{t+1})$ stands for the inner product of the base velocity and commanded direction at time step $t + 1$. If π can locomote in the commanded direction faster than 0.2 m/s, we consider the terrain traversable in this direction. The threshold is a hyperparameter; 0.2 m/s is about one-third of the maximum speed of our robot. Traversability is defined as

$$\mathrm{Tr}(c_T, \pi) = \mathbb{E}_{\xi\sim\pi}\{\nu(s_t, a_t, s_{t+1} \mid c_T)\} \in [0.0, 1.0] \quad (3)$$

where ξ refers to trajectories generated by π. This follows a definition of empirical traversability in prior work (*44*).

The objective of our terrain generation method is to find c_T values with midrange traversability ($\mathrm{Tr}(c_T, \pi) \in [0.5, 0.9]$). The rationale is to synthesize terrains that are neither too easy nor too difficult. We define terrain desirability as follows

$$\mathrm{Td}(c_T, \pi) := \Pr(\mathrm{Tr}(c_T, \pi) \in [0.5, 0.9]) = \mathbb{E}_{\xi\sim\pi}\{\mathrm{Tr}(c_T, \pi) \in [0.5, 0.9]\} \quad (4)$$

where 0.5 and 0.9 are fixed thresholds for minimum/maximum traversability.

We use a particle filter to keep track of a distribution of high-desirability c_T values during training. We formulate a particle-filtering problem where we approximate the distribution of terrain parameters that satisfies $\mathrm{Tr}(c_T, \pi) \in [0.5, 0.9]$ with a finite set of sampling points ($c_T^k \in \mathcal{C}, k \in 1, \cdots, N_{\mathrm{particle}}$). Our algorithm is modeled on the Sequential Importance Resampling particle filter. It is based on the following assumptions.

1) Terrain parameters with similar $\mathrm{Tr}(\cdot, \pi)$ are close in Euclidean distance in parameter space.

2) A policy trained over the terrains generated by c_T values in some area of $\mathcal{C}$ will learn to interpolate to nearby terrain parameters.

3) $c_{T,0}, c_{T,1}, \ldots$ forms a Markov process, where $c_{T,j} = \{c_{T,j}^1, c_{T,j}^2, \ldots c_{T,j}^{N_{\mathrm{particle}}}\}$ at iteration j.

The first assumption comes from the insight that terrain parameters can be interpolated, e.g., the difficulty of a staircase increases as we increase the step height. The second assumption justifies the use of discrete samples from $\mathcal{C}$ to train a policy that generalizes over a certain region of $\mathcal{C}$. The last assumption is necessary for formulating a particle filter.

The importance weight w^k is defined for each c_T^k, and the set of tuples $\langle c_T^k, w^k \rangle$ approximates the target distribution (c_T values with $\mathrm{Tr}(c_T, \pi) \in [0.5, 0.9]$). We define the measurement variable y_j^k such that $y_j^k = 1$ if $\mathrm{Tr}(c_{T,j}^k, \pi) \in [0.5, 0.9]$. Then, the terrain desirability defined above becomes the measurement probability

$$\Pr(y_j^k \mid c_{T,j}^k) = \Pr(\mathrm{Tr}(c_{T,i}^k, \pi) \in [0.5, 0.9]) = \mathrm{Td}(c_{T,j}^k, \pi) \quad (5)$$

For practical implementation, the measurement probability is computed by the empirical expectation from the samples collected during policy training

$$\Pr(y_j^k \mid c_{T,j}^k) \approx \sum \frac{\mathbb{1}\left(\mathrm{Tr}\left(c_{T,j}^k, \pi\right) \in [0.5, 0.9]\right)}{N_{\mathrm{traj}}} \quad (6)$$

where N_{traj} denotes the number of trajectories generated using $c_{T,j}^k$. The trajectories are also used for policy training. Our method therefore does not require additional evaluation steps to advance the curriculum of the terrain parameters. Resampling is done such that the probability of choosing the kth sample equals the normalized importance weight $w^k / \Sigma_i^{N_{\mathrm{particle}}} w^i \in [0, 1]$.

The transition model is a random walk in $\mathcal{C}$. Each parameter of a sampling point is shifted to its adjacent value by a fixed probability $p_{\mathrm{transition}}$. It satisfies the third assumption (Markov process) because the evolution of each parameter only relies on the current value and randomly sampled noise. To improve exploration, we bounded and discretized $\mathcal{C}$ to reduce the search space. The initial samples ($c^k_{T,0}$) are either drawn uniformly from $\mathcal{C}$ or concentrated on almost flat terrains. Implementation details and an overview of the training process are provided in section S2 and algorithm S1.

Validation of the method

We present ablation studies to justify each component of our approach: (i) using a sequence model for the student policy, (ii) privileged training, and (iii) adaptive terrain curriculum.

Memory in proprioceptive control

We evaluate the importance of incorporating proprioceptive memory in the controller via the TCN architecture (*22*). Let TCN-*N* denote a TCN with a receptive field of *N* time steps. The network architectures we use are specified in detail in table S5. We test controllers in diagnostic settings designed to focus on specific capabilities. Specifically, we test omnidirectional locomotion on sloped ground, traversal of a discrete step, and robustness to external disturbances (Fig. 5A).

Figure 5 (B to D) summarizes the importance of the memory length *N*. In these experiments, *N* is varied from 1 (corresponding to 20 ms of memory) to 100 (2 s of proprioceptive memory). The latter is the default setting used in our deployed controller.

As shown in Fig. 5B, memory length does not have a strong effect in the uniform slope setting. Memory length does have a strong effect on the controller's ability to traverse a step (Fig. 5, B and C). Controllers with longer memory are able to handle higher steps. As shown in Fig. 5C, the failure rate of limited-memory controllers is particularly high when the hind legs encounter the step. Controllers with longer memory also adapt hind-leg trajectories to ensure higher foot clearances.

Figure 5D shows that controllers with longer memory are more robust to external disturbances. We applied an external 50-N force laterally to the base for 5 s during a straight walk and evaluated the resulting deviation from the intended locomotion direction. The deviation of the TCN-100 controller was 35.5% lower than that of TCN-1.

Privileged training

We now assess the importance of privileged training. As a baseline, we train a TCN-20 policy directly, without the two-stage privileged training protocol. The policy is trained by TRPO (*36*) with the same reward and hyperparameters that we use for teacher training. This baseline is compared to the same TCN-20 architecture trained via privileged learning.

The results are summarized in Fig. 5 (E to G). Figure 5E shows that the baseline fails the diagnostic tests: It is incapable of locomoting on a slope or traversing a step. Figure 5F shows that the baseline does not reach comparable reward during training as the teacher MLP architecture with privileged information or the proprioceptive TCN-20 architecture (same as the baseline, no privileged information) trained via privileged learning. Figure 5G shows the mean episode length during training, which indicates that the baseline fails to learn to balance and locomote.

Adaptive terrain curriculum

We now evaluate the effect of the adaptive terrain curriculum on teacher training. Terrains used for training (specifically, hills, steps, and stairs) are shown in Fig. 4B. As a baseline, we trained a teacher

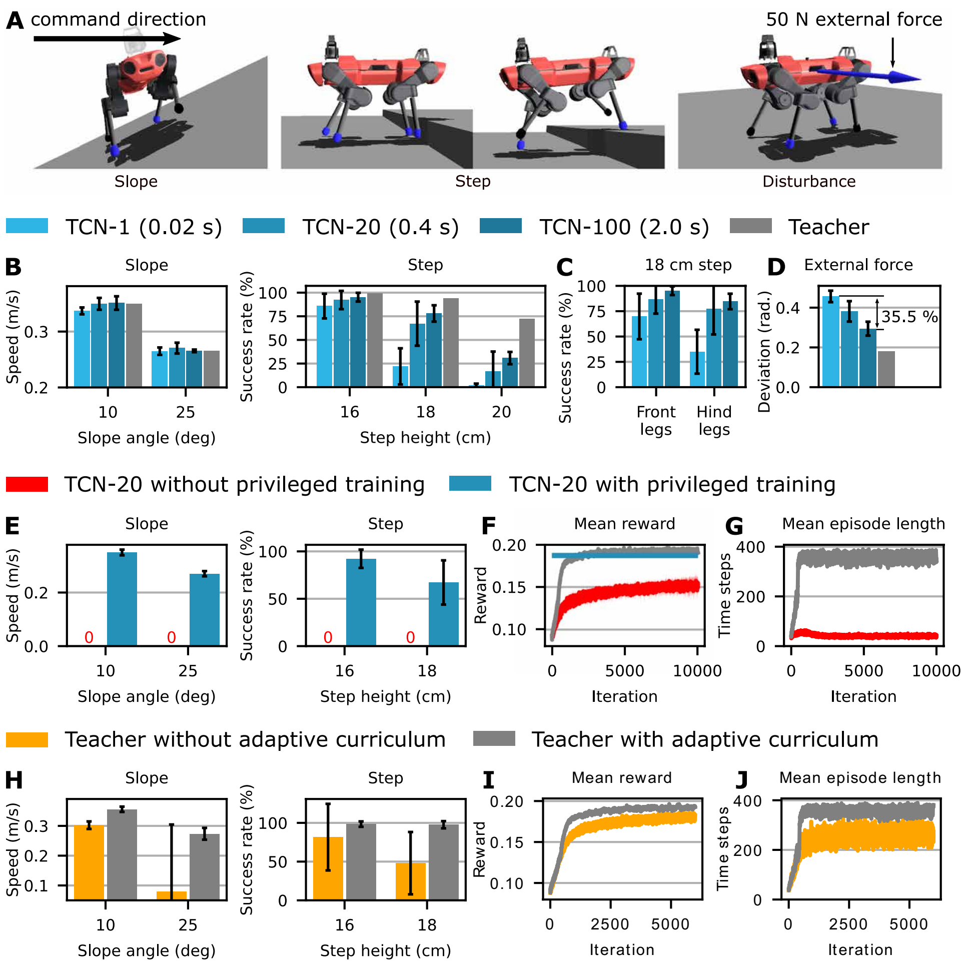

Fig. 5. Ablation studies. We trained each model five times using different random seeds. Error bars denote 95% CIs. (**A**) Test setups. The robot was commanded to advance for 10 s in the specified direction (black arrow). We conducted 100 trials for each test. On the step test, a trial was considered successful if the robot traversed the step with both front and hind legs. Robots were initialized with random joint configurations. Initial yaw angle was sampled from $U(-\pi, \pi)$ for the slope test and from $U(-\pi/6, \pi/6)$ for the other tests. The friction coefficients between the feet and the ground were sampled from $U(0.4, 1.0)$. The external force was applied for 5 s in the lateral direction. (**B** to **D**) Importance of memory length *N* in the TCN-*N* encoder. (**E** to **G**) Importance of privileged training. (F) Learning curves for the teacher (gray) and a TCN-20 student trained directly, without privileged training (red). For comparison, the blue line indicates the mean reward of a TCN-20 student trained with privileged training. The reward was computed by running each policy on uniformly sampled terrains. (**H** to **J**) Importance of the adaptive curriculum.

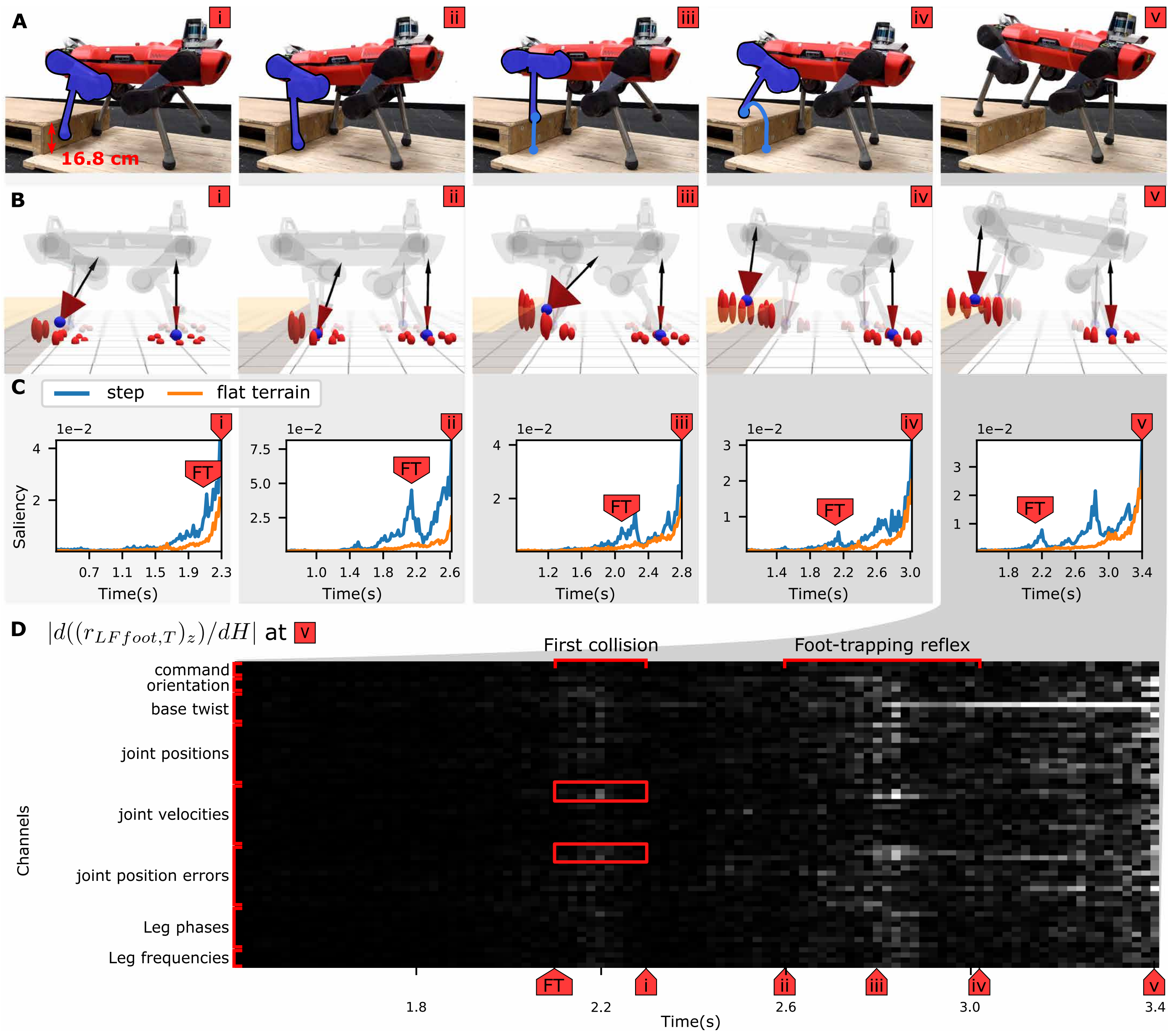

Fig. 6. Analysis of the emergent foot-trapping reflex. FT occurs when the LF foot collides with the step. (**A**) The LF foot hits the step and then manifests higher foot clearance to overcome the step (ii to iv) in the following swing phase. (**B**) Reconstructed terrain information from TCN embeddings. Red ellipsoids: Estimated terrain shape around the foot. The center of the ellipsoid refers to the estimated terrain elevation, and the vertical length represents uncertainty (SD). Black arrows: Terrain normal at the in-contact foot. Red cone: Uncertainty of normal estimation. Blue spheres: Estimated in-contact feet. (**C**) Input saliency at different moments. The peaks show that the TCN policy attends to the FT that happened around 2.1 s. The orange curve (flat terrain) shows the saliency value computed on a flat terrain at similar gait phases. (**D**) Saliency map unrolled across input channels at 3.4 s. Red boxes refer to joint measurements from the LF leg at the moment it collides with the step.

using randomly generated terrains that are uniformly sampled from $\mathcal{C}$ as specified in table S2. The success rates on the testing terrains are substantially lower when trained without the adaptive curriculum, as shown in Fig. 5H. Figure 5I shows that a teacher trained without adaptive curriculum plateaus at a lower reward level. Throughout the training process, the mean episode length is shorter for the model being trained without adaptive curriculum (Fig. 5J). This is because uniform sampling is more likely to draw terrains that cannot be successfully traversed by the policy being trained. On these terrains, the policy fails early and receives less training signal as a result. The adaptive curriculum modulates the difficulty of sampled terrains so as to maximize the didactic benefit of each episode. We provide an additional evaluation of the adaptive curriculum in section S6.

Further analysis of emergent behavior

Here, we provide further analysis on how the proprioceptive policy adapts to different situations. To investigate how the proprioceptive policy perceives the environment, we trained a decoder network that reconstructs the privileged information $x_t \in X$ from the output of an intermediate layer of a trained TCN policy. x_t consists of information that is not directly observable by the student policy such as contact states, terrain shape, and external disturbances. For classification of foot contact states, we use a standard cross-entropy loss function. For regression of other states, we predict both mean m_i and SD σ_i for each component and use a negative Gaussian log-likelihood loss to quantify the uncertainty encoded in the TCN representation (*45*)

$$\mathcal{L} = \sum_{i \in \dim(X \backslash \text{contactstates})} \frac{(m_i - m_i^{gt})^2}{2\sigma_i^2} + \log(\sigma_i) \qquad (7)$$

with added weight decay. The superscript *gt* refers to the ground truth generated in simulation. Note that the parameters of the policy network are fixed during decoder training. Therefore, the decoder network is not used for policy training. It only provides insight into the information encoded by the TCN policy after training.

In Fig. 6, we provide snapshots of the FT reflex motion (Fig. 6A) and the reconstructed privileged information. In Fig. 6B, we show the reconstructed terrain geometry and foot contact state. When the LF foot collides with the step, the estimated elevation in front of the front legs increases, and its uncertainty grows (i and ii). The estimated elevations and normal vectors adapt to the step during the FT reflex (iii and iv). After the successful step-up, the terrain uncertainty remains elevated (v), indicating an anticipation of generally rough terrain. In addition, the decoder network can detect foot contacts with horizontal and vertical surfaces while successfully identifying frontal collision as such, as indicated by the estimated terrain normal vector (i and iii). The ability to reconstruct explicit environmental information from the encoding of the proprioceptive history is a strong indicator that the TCN policy learns to build an internal representation of the environment and uses it for decision making. We provide more examples of the reconstructed privileged information in section S7.

We then analyze how the proprioceptive policy leverages past observations. We compute the saliency map of the input $H \in \mathbb{R}^{60 \times N}$ and visualize the sensitivity of the policy to each element of the input while overcoming the step (*46*). Each column of H is a proprioceptive measurement $h \in \mathbb{R}^{60}$, and we stack N measurements (history length = 0.02 s × N). We define the saliency value for the ith measurement ($i \in [0, N]$) as

$$M_i = \sum_{j \in \text{channels}} (\,|\, d((r_{f,T})_z) / dH_{i,j} \,|_H\,|\,) \in \mathbb{R} \qquad (8)$$

where $(r_{f,T})_z$ refers to the height command for the foot f. We computed the value for $(r_{f,T})_z$ because we are interested in the change in foot clearance. M_i can be interpreted as the sensitivity of the output to the ith measurement. Because we use 1D convolution over time, the output is in $\mathbb{R}^N$, i.e., each row of H is regarded as a channel.

In Fig. 6C, we can see that the saliency value at the FT is kept high while stepping up. The policy has direct access to the measurements at the moment of FT, and leverages this in the following swing phase. This is highlighted by the red boxes in Fig. 6D. The policy attends to the LF leg joint states measured at the FT.

SUPPLEMENTARY MATERIALS

robotics.sciencemag.org/cgi/content/full/5/47/eabc5986/DC1

REFERENCES AND NOTES

1. F. Jenelten, J. Hwangbo, F. Tresoldi, C. D. Bellicoso, M. Hutter, Dynamic locomotion on slippery ground. *IEEE Robot. Autom. Lett.* **4**, 4170–4176 (2019).
2. G. Bledt, P. M. Wensing, S. Ingersoll, S. Kim, Contact model fusion for event-based locomotion in unstructured terrains, in *2018 IEEE International Conference on Robotics and Automation (ICRA)* (IEEE, 2018).
3. M. Focchi, R. Orsolino, M. Camurri, V. Barasuol, C. Mastalli, D. G. Caldwell, C. Semini, Heuristic planning for rough terrain locomotion in presence of external disturbances and variable perception quality, in *Advances in Robotics Research: From Lab to Market* (Springer, 2020), pp. 165–209.
4. J. Reher, W.-L. Ma, A. D. Ames, Dynamic walking with compliance on a Cassie bipedal robot, in *European Control Conference* (IEEE, 2019), pp. 2589–2595.
5. Y. Gong, R. Hartley, X. Da, A. Hereid, O. Harib, J.-K. Huang, J. Grizzle, Feedback control of a Cassie bipedal robot: Walking, standing, and riding a Segway, in *American Control Conference* (IEEE, 2019), pp. 4559–4566 .
6. J. Hwangbo, C. D. Bellicoso, P. Fankhauser, M. Huttery, Probabilistic foot contact estimation by fusing information from dynamics and differential/forward kinematics, in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (IEEE, 2016), pp. 3872–3878 .
7. M. Camurri, M. Fallon, S. Bazeille, A. Radulescu, V. Barasuol, D. G. Caldwell, C. Semini, Probabilistic contact estimation and impact detection for state estimation of quadruped robots. *IEEE Robot. Autom. Lett.* **2**, 1023–1030 (2017).
8. M. Focchi, V. Barasuol, M. Frigerio, D. G. Caldwell, C. Semini, Slip detection and recovery for quadruped robots, in *Robotics Research* (Springer, 2018), pp. 185–199.
9. M. Bloesch, C. Gehring, P. Fankhauser, M. Hutter, M. A. Hoepflinger, R. Siegwart, State estimation for legged robots on unstable and slippery terrain, in *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems* (IEEE, 2013), pp. 6058–6064.
10. C. Gehring, C. D. Bellicoso, S. Coros, M. Bloesch, P. Fankhauser, M. Hutter, R. Siegwart, Dynamic trotting on slopes for quadrupedal robots, in *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (IEEE, 2015), pp. 5129–5135.
11. R. Hartley, J. Mangelson, L. Gan, M. G. Jadidi, J. M. Walls, R. M. Eustice, J. W. Grizzle, Legged robot state-estimation through combined forward kinematic and preintegrated contact factors, in *2018 IEEE International Conference on Robotics and Automation (ICRA)* (IEEE, 2018), pp. 1–8.
12. J. Hwangbo, J. Lee, A. Dosovitskiy, D. Bellicoso, V. Tsounis, V. Koltun, M. Hutter, Learning agile and dynamic motor skills for legged robots. *Sci. Rob.* **4**, eaau5872 (2019).
13. T. Haarnoja, S. Ha, A. Zhou, J. Tan, G. Tucker, S. Levine, Learning to walk via deep reinforcement learning (Robotics: Science and Systems, 2019).
14. Z. Xie, P. Clary, J. Dao, P. Morais, J. Hurst, M. van de Panne, Learning locomotion skills for Cassie: Iterative design and sim-to-real, in *Conference on Robot Learning* (PMLR, 2019).

15. J. Lee, J. Hwangbo, M. Hutter, Robust recovery controller for a quadrupedal robot using deep reinforcement learning. arXiv:1901.07517 [cs.RO] (22 January 2019).
16. J. Tan, T. Zhang, E. Coumans, A. Iscen, Y. Bai, D. Hafner, S. Bohez, V. Vanhoucke, Sim-to-real: Learning agile locomotion for quadruped robots (Robotics: Science and Systems, 2018).
17. Y. Yang, K. Caluwaerts, A. Iscen, T. Zhang, J. Tan, V. Sindhwani, Data efficient reinforcement learning for legged robots, in *Conference on Robot Learning* (PMLR, 2019).
18. S. Ha, P. Xu, Z. Tan, S. Levine, J. Tan, Learning to walk in the real world with minimal human effort. arXiv:2002.08550 [cs.RO] (20 February 2020).
19. X. B. Peng, E. Coumans, T. Zhang, T.-W. Lee, J. Tan, S. Levine, Learning agile robotic locomotion skills by imitating animals. arXiv:2004.00784 [cs.RO] (2 April 2020).
20. M. Hutter, C. Gehring, D. Jud, A. Lauber, C. D. Bellicoso, V. Tsounis, J. Hwangbo, K. Bodie, P. Fankhauser, M. Bloesch, R. Diethelm, S. Bachmann, A. Melzer, M. A. Höpflinger, ANYmal - a highly mobile and dynamic quadrupedal robot, in *IEEE/RSJ International Conference on Intelligent Robots and Systems* (IEEE, 2016), pp. 38–44.
21. X. B. Peng, M. Andrychowicz, W. Zaremba, P. Abbeel, Sim-to-real transfer of robotic control with dynamics randomization, in *IEEE International Conference on Robotics and Automation (ICRA)* (IEEE, 2018).
22. S. Bai, J. Z. Kolter, V. Koltun, An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. arXiv:1803.01271 [cs.LG] (4 March 2018).
23. D. Chen, B. Zhou, V. Koltun, P. Krähenbühl, Learning by cheating, in *Conference on Robot Learning* (2019).
24. J. C. Brant, K. O. Stanley, Minimal criterion coevolution: A new approach to open-ended search, in *Genetic and Evolutionary Computation Conference* (GECCO, 2017), pp. 67–74.
25. R. Wang, J. Lehman, J. Clune, K. O. Stanley, Paired open-ended trailblazer (POET): Endlessly generating increasingly complex and diverse learning environments and their solutions. arXiv:1901.01753 [cs.NE] (7 January 2019).
26. P. Fankhauser, M. Bloesch, C. Gehring, M. Hutter, R. Siegwart, Robot-centric elevation mapping with uncertainty estimates, in *Mobile Service Robotics* (World Scientific, 2014), pp. 433–440.
27. C. D. Bellicoso, F. Jenelten, C. Gehring, M. Hutter, Dynamic locomotion through online nonlinear motion optimization for quadrupedal robots. *IEEE Robot. Autom. Lett.* **3**, 2261–2268 (2018).
28. S. Collins, A. Ruina, R. Tedrake, M. Wisse, Efficient bipedal robots based on passive-dynamic walkers. *Science* **307**, 1082–1085 (2005).
29. Ghost Robotics, Vision 60: Latest blind-mode stress testing of V60 legged robot (2019); www.youtube.com/watch?v=tQsLauQWp8M.
30. J. Hwangbo, J. Lee, M. Hutter, Per-contact iteration method for solving contact dynamics. *IEEE Robot. Autom. Lett.* **3**, 895–902 (2018).
31. E. Coumans, Bullet physics library (2013); pybullet.org.
32. R. Smith, Open dynamics engine (2005); ode.org.
33. R. M. Alexander, *Principles of Animal Locomotion* (Princeton Univ. Press, 2003).
34. A. Iscen, K. Caluwaerts, J. Tan, T. Zhang, E. Coumans, V. Sindhwani, V. Vanhoucke, Policies modulating trajectory generators, in *Conference on Robot Learning* (PMLR, 2018), pp. 916–926.
35. V. Barasuol, J. Buchli, C. Semini, M. Frigerio, E. R. De Pieri, D. G. Caldwell, A reactive controller framework for quadrupedal locomotion on challenging terrain, in *2013 IEEE International Conference on Robotics and Automation* (IEEE, 2013), pp. 2554–2561.
36. J. Schulman, S. Levine, P. Abbeel, M. Jordan, P. Moritz, Trust region policy optimization, in *International Conference on Machine Learning* (PMLR, 2015), pp. 1889–1897.
37. M. Bloesch, M. Hutter, M. A. Hoepflinger, S. Leutenegger, C. Gehring, C. D. Remy, R. Siegwart, State estimation for legged robots-consistent fusion of leg kinematics and imu. *Robotics* **17**, 17–24 (2013).
38. S. Ross, G. Gordon, D. Bagnell, A reduction of imitation learning and structured prediction to no-regret online learning, in *International Conference on Artificial Intelligence and Statistics* (AISTATS, 2011), pp. 627–635.
39. C. Florensa, D. Held, X. Geng, P. Abbeel, Automatic goal generation for reinforcement learning agents, in *International Conference on Machine Learning* (PMLR, 2018), pp. 1514–1523.
40. J. Lehman, K. O. Stanley, Revising the evolutionary computation abstraction: Minimal criteria novelty search, in *Genetic and Evolutionary Computation Conference* (GECCO, 2010), pp. 103–110.
41. T. Matiisen, A. Oliver, T. Cohen, J. Schulman, Teacher-student curriculum learning, in *IEEE transactions on neural networks and learning systems* (IEEE, 2019).
42. W. Yu, G. Turk, C. K. Liu, Learning symmetric and low-energy locomotion, *ACM Transactions on Graphics (TOG)* (2018), p. 144.
43. I. Akkaya, M. Andrychowicz, M. Chociej, M. Litwin, B. McGrew, A. Petron, A. Paino, M. Plappert, G. Powell, R. Ribas, J. Schneider, N. Tezak, J. Tworek, P. Welinder, L. Weng, Q. Yuan, W. Zaremba, L. Zhang, Solving rubik's cube with a robot hand. arXiv:1910.07113 [cs.LG] (16 October 2019).
44. R. O. Chavez-Garcia, J. Guzzi, L. M. Gambardella, A. Giusti, Learning ground traversability from simulations. *IEEE Robot. Autom. Lett.* **3**, 1695–1702 (2018).
45. A. Kendall, Y. Gal, What uncertainties do we need in bayesian deep learning for computer vision?, in *Advances in Neural Information Processing Systems 30* (Neural Information Processing Systems Foundation, 2017), pp. 5574–5584.
46. K. Simonyan, A. Vedaldi, A. Zisserman, Deep inside convolutional networks: Visualising image classification models and saliency maps. arXiv:1312.6034 [cs.CV] (20 December 2013).

Acknowledgments: A substantial part of the work was carried out during J.H.'s stay at the Robotic Systems Lab, ETH Zürich. **Funding:** The project was funded, in part, by the Intel Network on Intelligent Systems, the Swiss National Science Foundation (SNF) through the National Centre of Competence in Research Robotics, the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme grant agreement no 852044 and no 780883. The work has been conducted as part of ANYmal Research, a community to advance legged robotics. **Author contributions:** J.L. formulated the main idea of the training and control methods, implemented the controller, set up the simulation, and trained control policies. J.L. performed the indoor experiments. J.H. contributed in setting up the simulation. J.L. and L.W. performed outdoor experiments together. All authors refined ideas, contributed in the experiment design, and analyzed the data. **Competing interests:** The authors declare that they have no competing interests. **Data and materials availability:** All data needed to evaluate the conclusions in the paper are present in the paper or the Supplementary Materials. Other materials can be found at https://github.com/leggedrobotics/learning_locomotion_over_challening_terrain_supplementary.

Submitted 10 May 2020
Accepted 22 September 2020
Published 21 October 2020
10.1126/scirobotics.abc5986

Citation: J. Lee, J. Hwangbo, L. Wellhausen, V. Koltun, M. Hutter, Learning quadrupedal locomotion over challenging terrain. *Sci. Robot.* **5**, eabc5986 (2020).