

Zero-Shot Terrain Generalization for Visual Locomotion Policies

Alejandro Escontrela^{1,2}, George Yu¹, Peng Xu¹, Atil Iscen¹, Jie Tan¹

Abstract—Legged robots have unparalleled mobility on unstructured terrains. However, it remains an open challenge to design locomotion controllers that can operate in a large variety of environments. In this paper, we address this challenge of automatically learning locomotion controllers that can generalize to a diverse collection of terrains often encountered in the real world. We frame this challenge as a multi-task reinforcement learning problem and define each task as a type of terrain that the robot needs to traverse. We propose an end-to-end learning approach that makes direct use of the raw exteroceptive inputs gathered from a simulated 3D LiDAR sensor, thus circumventing the need for ground-truth heightmaps or preprocessing of perception information. As a result, the learned controller demonstrates excellent zero-shot generalization capabilities and can navigate 13 different environments, including stairs, rugged land, cluttered offices, and indoor spaces with humans.

I. INTRODUCTION

The ability to traverse unstructured terrains make legged robots an appealing solution to a wide variety of tasks, including disaster relief, last-mile delivery, industrial inspection, and planetary exploration [1], [2]. To deploy robots in these settings successfully, we must design controllers that work well across many different terrains. Due to the diversity of environments that a legged robot can operate in, hand-engineering such a controller presents unique challenges. Deep Reinforcement Learning (DRL) has proven itself capable of automatically acquiring control policies to accomplish a large variety of challenging locomotion tasks. However, many of these approaches learn control policies that succeed in a single type of terrain with limited variations. This approach limits the robot’s ability to generalize to new or unseen environments, which is a crucial feature of a useful locomotion controller.

In this paper, we develop an end-to-end reinforcement learning system that enables legged robots to traverse a large variety of terrains. To facilitate learning generalizable policies, we make two purposeful design decisions for our learning system. First, we formulate the problem as a Multi-Task Partially Observable Markov Decision Problem and show that the robot learns a robust policy that works well across a wide variety of tasks (terrains). To this end, we develop a novel procedural terrain generation method, which can efficiently generate a large variety of terrains for training. Second, we design an end-to-end neural network architecture that can handle both perception and locomotion. We call this parameterization a *visual-locomotion* policy. While many

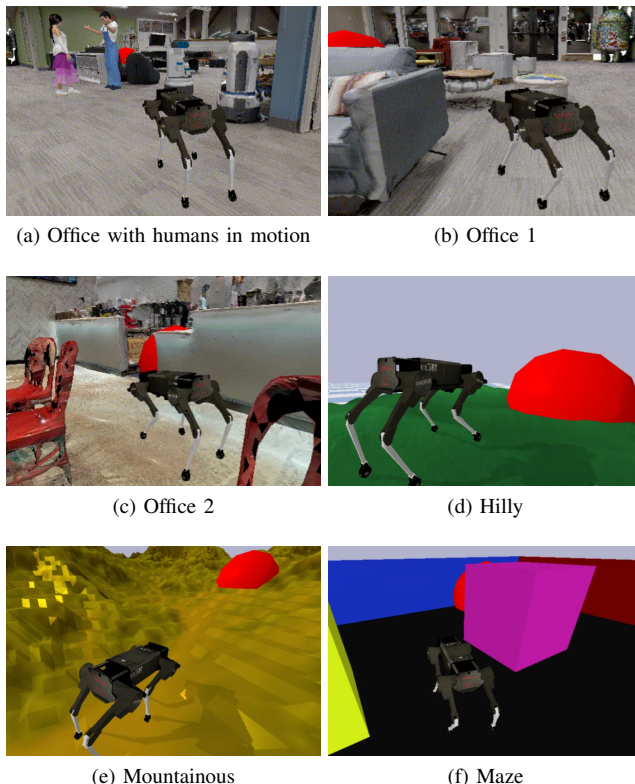


Fig. 1. A Laikago robot navigating a variety of complex terrains not encountered during training.

prior works in the legged robot literature focused on blind walking, which does not involve exteroceptive sensors (e.g., camera, LiDAR), we find that exteroceptive perception is essential for robots to navigate in diverse environments. Our end-to-end visual-locomotion policy takes both exteroceptive (a LiDAR scan) and proprioceptive information of the robot and outputs low-level motor commands. We embed the Policies Modulating Trajectory Generator (PMTG) [3] framework into our policy architecture to generate cyclic and smooth actuation patterns, and to facilitate the learning of robust locomotion policies.

We evaluate our learning system using a high-fidelity physics simulator [4] and visually-realistic indoor scans [5] (Figure 1). We test the learned policy in thirteen different and realistic simulation environments (five training and eight testing). Our system learns highly generalizable locomotion policies, which demonstrate zero-shot generalization to unseen testing environments. We also show that our visual-locomotion policy’s parameterization is key to generalization and yields far better performance than commonly-used

¹ Google Brain Robotics, {georgeyu, pengxu, atil, jietan}@google.com

² Georgia Institute of Technology, aescontrela@gatech.edu
Work performed while Alejandro was an intern at Google Brain.

reactive policies. This paper’s main contributions include an end-to-end visual-locomotion policy parameterization and a complete multi-task learning system, with which a quadruped robot learns a single locomotion policy that can traverse a diverse set of terrains.

II. RELATED WORK

A. Legged Locomotion

Locomotion controllers can be developed using trajectory optimization [6], whole-body control [7], model predictive control [8], and state-machines [9]. While the controllers developed by these techniques can generalize to a certain degree, expertise and manual tuning are often needed to adapt them to different terrains.

In contrast, Deep Reinforcement Learning [10] can automatically learn agile and robust locomotion skills [11], [12], [13], [14]. Prior work in RL has learned policies that are specific for a single environment [15], or generalize to variations of a single type of terrain [16], [17], [18]. Recently, Lee *et al.* [14] combined various techniques, such as ActuatorNet [13], PMTG [3], curriculum learning and “learning by cheating” [19], which successfully performed zero-shot transfer from simulation to many challenging terrains in the real world. While our paper’s high-level goal is similar to this prior work, our approach incorporates exteroceptive sensors that enable the robot to navigate in cluttered indoor environments where blind walking may have difficulties.

B. Multi-Task Reinforcement Learning

Multi-task reinforcement learning (MTRL) [20] is a promising approach to train generalizable policies that can accomplish a wide variety of tasks. Hessel *et al.* [21] learned a single policy that achieves state-of-the-art performance on 57 Atari games. Yu *et al.* [22] evaluated the performance of various RL algorithms on a grasping and manipulation benchmark and demonstrated that a single control policy is capable of completing a variety of complex robotic manipulation tasks. In this paper, we apply MTRL to develop a learning system for locomotion that enables legged robots to navigate in a large variety of environments.

III. METHODS

In this work, we frame legged locomotion as a multi-task reinforcement learning problem (MTRL) and define each task as a type of terrain that the legged robot (agent) must traverse. To learn generalizable locomotion policies, our learning system consists of a procedural terrain generator that can efficiently generate diverse training environments, and an end-to-end visual-locomotion policy architecture that directly maps the robot’s exteroceptive and proprioceptive observations to motor commands.

A. Multi-Task Reinforcement Learning Formulation

Given a distribution of tasks \mathcal{M} , each task $M_i \in \mathcal{M}$ is a Partially Observable Markov Decision Process (POMDP). A POMDP is tuple, $M_i = \langle \mathcal{S}, \mathcal{O}, \mathcal{A}, \mathcal{T}_i, \mathcal{R} \rangle$, where \mathcal{S} is the state space, \mathcal{O} is the observation space, \mathcal{A} is the action space,

$\mathcal{T}_i : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \rightarrow \mathbb{R}_+$ is the transition probability function, and $\mathcal{R} : \mathcal{S} \times \mathcal{A} \rightarrow \mathbb{R}$ is the reward function. During training, the agent is presented with randomly sampled tasks $M_i \in \mathcal{M}$ (Section III-B). The solution of the multi-task POMDP is a stochastic policy $\pi : \mathcal{O} \times \mathcal{A} \rightarrow \mathbb{R}_+$ that maximizes the expected accumulated reward over the episode length T .

$$\pi^* = \arg \max_{\pi} \mathbb{E}_{M_i \in \mathcal{M}} \left[\sum_{t=0}^T r(\mathbf{s}_t, \mathbf{a}_t) \right]$$

Our problem is partially observable because of the limited sensors onboard the robot¹. The robot is equipped with a LiDAR sensor to perceive the distances d to the surrounding environment. Proprioceptive information comes from a simulated IMU sensor, which includes measurement of the roll ϕ , pitch θ , and the angular velocity of the torso ${}^\beta \boldsymbol{\omega} = (\dot{\phi}, \dot{\theta}, \dot{\psi})$, and from motor encoders that measure the robot’s 12 joint angles \mathbf{q} . The complete observation at timestep t is

$$\mathbf{s}_t = [\mathbf{a}_{t-1}^T, \mathbf{o}_t, \mathbf{s}_{\text{TG}}^T, g_{d,t}, g_{h,t}],$$

where $\mathbf{o}_t = [d_t^T, {}^\beta \boldsymbol{\omega}_t^T, \mathbf{q}_t^T, \phi_t, \theta_t]$ are the sensor observations, g_d and g_h are the distance and relative heading to the target, \mathbf{a}_{t-1} is the action at the last timestep, and \mathbf{s}_{TG}^T are the parameters of the trajectory generator (Section III-C). Unlike some prior work in MTRL, where the task ID is part of the observation [22], [23], we purposefully choose not to leverage such information, because identifying tasks automatically in the real world is challenging. Instead, we would like to train a policy that can rely on its own perception input and demonstrates zero-shot generalization to new tasks, without knowing the task ID explicitly. In section IV, we demonstrate our perception is crucial in learning policies which generalize well to new tasks. The output action \mathbf{a}_t of the policy specifies the desired joint angles, which are tracked by PD controllers by the simulated robot.

We employ a simple reward function, which encourages the agent to navigate to a target location $\mathbf{g} = (x_g, y_g, z_g)$ (the red ball in Figure 1):

$$r_t = \frac{g_{d,t} - g_{d,t-1}}{\Delta t},$$

where $g_{d,t}$ is the Euclidean distance from the robot to the target location at timestep t , and Δt is the timestep duration. This reward can be interpreted as the speed that the robot is moving towards the target location. Once the robot’s center of mass is within a threshold distance to the target location, the task is complete.

B. Terrain Parameterization and Procedural Task Generation

We develop a procedural terrain generator to generate diverse and challenging terrains that provide the robot with a large quantity of rich training data. The environment is composed of $m \times n$ pillars, each pillar having cross-sectional dimensions of l, w , and height h . We denote $\mathbf{H} = \{h_{i,j}\} \in$

¹Although we use a simulated robot due to limited access to the physical robot during COVID-19, we strive to make the simulation, including the sensor measurement, as faithful as possible to the real robot.

TABLE I
TERRAIN PARAMETERIZATION AND GENERATION FOR SELECTED
EXAMPLES.

Terrain	Terrain Parameterization	
	Parameters ϕ	Terrain Generation
Flat	No parameters	$\mathbf{H} = \mathbf{0}$
Rugged	Min terrain height: h_{\min}	$\mathbf{H} \sim \mathcal{U}_{m,n}(h_{\min}, h_{\max})$
	Max terrain height: h_{\max}	Apply Gaussian smoothing with σ on \mathbf{H}
	Gaussian kernel std: σ	
Holes	Number of holes: n	$\mathbf{H} = \mathbf{0}$
	Hole depth: h	Sample n index pairs (i, j)
		$\mathbf{H}(i, j) = h$
Obstacles	Number of obstacles: n	$\mathbf{H} = \mathbf{0}$
	obstacle height: h	Sample n index pairs (i, j)
		$\mathbf{H}(i, j) = h$
Stairs	Stair step height: h	$\mathbf{H}(0, :) = \mathbf{0}$
	Stair step length: l	Set column lengths to l
		$\mathbf{H}(i+1, :) = \mathbf{H}(i, :) + h$

$\mathbb{R}_{m \times n}$ as the height field for all the pillars. During training, we select a task M_i and adjust each pillar’s heights to reflect the chosen task. Each task is a set of randomly generated terrains that belongs to the same type (e.g., flat, stairs). Each type of terrain is described by a parameter vector ϕ_i , which provides the lower and upper bounds for the random sampling. The terrain generator constructs the heightfield \mathbf{H} from the given parameter vector ϕ . For example, the parameter vector ϕ for the rugged terrain task (Fig. 3b) includes the minimum and maximum values of the heightfield; for the stairs task, the parameter vector defines the height and length of each step. Table I summarizes the parameters and terrain generator for selected terrain types. With this simple parameterization, we can generate over ten different types of terrains that a robot may encounter in the real world. Our procedural terrain generation algorithm provides a rich set of training data essential for generalizable policies to emerge.

C. Visual-Locomotion Policy Architecture

Exteroceptive perception plays a crucial role when legged robots need to navigate different terrains and environments with obstacles and humans [24], [25]. As such, we aim to incorporate perception into our policy architecture such that information from the robot’s surroundings can modulate locomotion. Additionally, the policy’s low-level actuation commands need to be smooth and realizable on the physical robot. To this end, we seek to restrict the search space of possible gaits to be cyclic and smooth while still expressive enough so that the perception can modulate locomotion sufficiently to work on different terrains.

In our visual-locomotion policy architecture (Fig. 2), we use two separate neural network encoders to process the proprioceptive and exteroceptive inputs. The upper branch of Fig. 2a processes the LiDAR input, while the lower branch takes care of proprioceptive information. The learned lower-dimensional features are concatenated with the target information before being passed to the policy’s locomotion component. We chose to use Policies Modulating Trajectory

Generators (PMTG) [3] as our locomotion component architecture (Fig. 2b). PMTG encourages the policy to learn smooth and cyclic locomotion behaviors. PMTG outputs a desired trajectory for the legs that is modulated by a learned policy $\pi_{\theta}(\cdot)$: The policy observes the state of the trajectory generator (TG), s_{tg} , and the robot’s observation s_t , then outputs parameters of the TG, p_{tg} , including gait frequency, swing height, and stride length, and a residual action term μ_{fb} . The final output action of our visual-locomotion policy is the combination of the trajectory generator and the residual action: $a_t = \mu_{tg} + \mu_{fb}$. Please refer to the original paper [3] for more details. As detailed in [16], our visual-locomotion policy architecture achieves a separation of concerns between the basic locomotion skills and terrain perception, which enables the robot to adapt its smooth locomotion behaviors according to its surrounding environments.

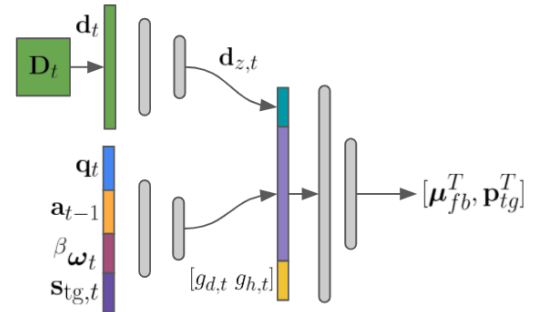
IV. EXPERIMENTAL RESULTS

We design experiments to validate the proposed system’s ability to learn a visual locomotion policy that generalizes well to terrains not encountered during training. In particular, we would like to answer the following two questions:

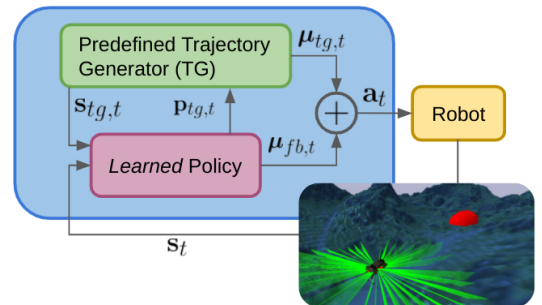
- Can our system learn visual locomotion policies that demonstrate zero-shot generalization to new terrains?
- Can our policy architecture effectively use LiDAR input and PMTG parameterization to improve the generalization performance over unseen terrains?

A. Experiment Details

To answer the above questions, we evaluate our system using a simulated Unitree Laikago quadruped robot [26],



(a) Visual-locomotion policy architecture.



(b) The locomotion component using PMTG [3] for smooth and cyclic actuation patterns.

Fig. 2. Overview of the visual-locomotion policy architecture.

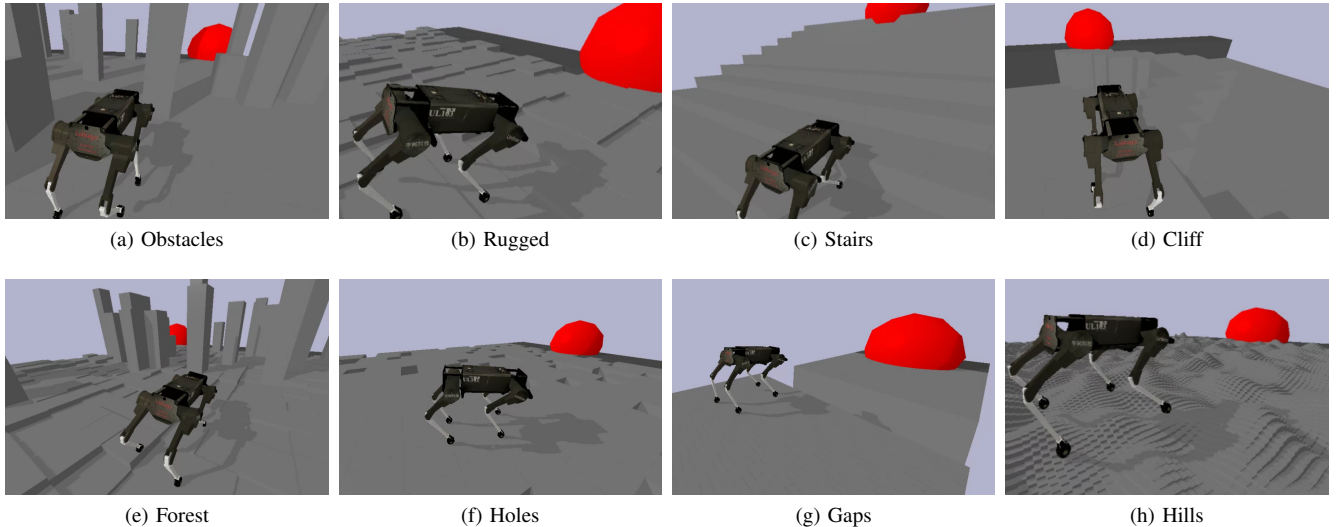


Fig. 3. A Laikago robot deployed in various procedurally generated training environments. The red sphere represents goal g and success radius r_g .

which weighs approximately 22kg and is actuated by 12 motors. We simulate the onboard Velodyne VLP-16 (Puck) LiDAR sensor, which provides the perception of the surrounding environment (See Figure 2b). The LiDAR measures the distance from the surrounding obstacles and terrain to the robot. This sensor supports 16 channels, a 360° horizontal field of view, and a 30° vertical field of view. We add Gaussian noise to the ground-truth distance readings in simulation to mimic the real-world noise model. The 3D LiDAR scan matrix \mathbf{D} is normalized to range $[0, 1]$ and flattened to a vector \mathbf{d} .

Our policy computes joint target positions (\mathbf{a}_t), which are converted to target joint torques by a PD controller running at 1kHz. Rigid body dynamics and contacts are also simulated at 1kHz. In other words, the position and velocity (provided by PyBullet [4]) and the desired torque (provided by the PD controller) are sent to the actuator model every 1ms. The actuator model then computes 10 internal $100\mu\text{s}$ steps and provides the effective output torque of the actuator, which is then used by PyBullet to compute joint accelerations. The simulation environment is configured to use an *action repeat* of 10 steps, which means that our policy computes a new action \mathbf{a}_t and receive a state \mathbf{s}_t every 10ms (100Hz).

We train the visual-locomotion policy using the MTRL formulation with simulated environments randomly generated using our procedural task generation method (Section III-B). We choose a distributed version of the Proximal Policy Optimization (PPO) [27] in TF-Agents [28] for training. We use a 2-layer fully-connected neural network of dimensions (512, 256) to parameterize the value function and another network of dimensions (256, 128) to parameterize the policy. The policy outputs the parameters of a multivariate Gaussian distribution, which we sample actions from during training. We use a greedy policy during evaluation by executing the mean of the multivariate Gaussian distribution provided by the policy network. The dimensions of the

exteroceptive and proprioceptive input encoders are both (32, 16, 4), respectively. We use the ReLU activation function for all layers in both networks [29]. The advantages are estimated using Generalized Advantage Estimation [30].

We then evaluate the trained policies on a suite of testing environments not encountered during training. Figure 1 illustrates a subset of these testing environments. These high-fidelity simulated environments are created in PyBullet physics engine [4] with Gibson scenes [5]. A policy’s ability to successfully navigate across a given terrain is measured using the *task completion rate*, tcr , which measures how close the agent gets to the target relative to its starting position:

$$tcr = 1 - \frac{g_{d,T}}{g_{d,0}},$$

where $g_{d,T}$ is the final Euclidean distance between the robot and the target when the robot falls or completes the task, and $g_{d,0}$ is the distance at the beginning of the episode. A task completion rate of 1 indicates successful navigation to the target, whereas tcr close to zero means that the robot cannot navigate across the terrain.

B. The Impact of MTRL on Generalization

Table II shows the generalization performance of our visual-locomotion policy trained on different types of terrains (rows) and tested in unseen environments (columns), including a maze (Maze), a steep and rugged mountain (Mountain), two indoor scenarios (Office 1 and Office 2), an office space with moving humans (Dynamic Env), a forest scene with rugged terrain and obstacles (Forest), a winding path with a cliff on both sides (Cliff), and a randomly-generated continuous mesh (Continuous). Policies trained on a single type of terrain achieve a low task completion rate in the testing environments due to a lack of diverse training data. In contrast, our approach achieves much higher generalization performance. For instance, our method on average achieves a task completion rate of 67% on the

TABLE II
GENERALIZATION PERFORMANCE OF OUR VISUAL-LOCOMOTION POLICY.

Train Description	Evaluation Environment Task Completion Rate							
	Maze	Mount-ain	Office 1	Office 2	Dynamic Env	Forest	Cliff	Contin-uous
Flat	0.39 ±0.17	0.22 ±0.21	0.54 ±0.12	0.51 ±0.16	0.50 ±0.12	0.43 ±0.13	0.32 ±0.18	0.57 ±0.12
Rugged	0.57 ±0.18	0.28 ±0.11	0.64 ±0.12	0.60 ±0.15	0.59 ±0.16	0.65 ±0.18	0.36 ±0.16	0.73 ±0.17
Holes	0.33 ±0.18	0.22 ±0.12	0.41 ±0.15	0.38 ±0.16	0.38 ±0.13	0.39 ±0.12	0.52 ±0.15	0.57 ±0.13
Obstacles	0.70 ±0.19	0.23 ±0.04	0.67 ±0.19	0.69 ±0.15	0.68 ±0.17	0.59 ±0.09	0.55 ±0.13	0.53 ±0.16
Stairs	0.39 ±0.18	0.28 ±0.11	0.52 ±0.14	0.49 ±0.13	0.43 ±0.16	0.60 ±0.19	0.38 ±0.18	0.68 ±0.10
Ours	0.72 ±0.13	0.67 ±0.12	0.84 ±0.11	0.83 ±0.13	0.79 ±0.14	0.78 ±0.10	0.70 ±0.17	0.72 ±0.19

TABLE III
COMPARISON OF OUR PROPOSED METHOD TO OTHER POLICIES DEPLOYED IN A MTRL TRAINING REGIME. THE PERFORMANCE DECREASES WHEN THE POLICY DOES NOT USE A PMTG PARAMETERIZATION, WHEN THE POLICY IS NOT PROVIDED EXTEROCEPTIVE INPUTS FROM THE LIDAR, AND WHEN MULTI-TASK TRAINING IS PERFORMED IN A SEQUENTIAL MANNER.

Train Description	Evaluation Environment Task Completion Rate							
	Maze	Mount-ain	Office 1	Office 2	Dynamic Env	Forest	Cliff	Contin-uous
Reactive	0.67 ±0.13	0.65 ±0.16	0.76 ±0.16	0.77 ±0.19	0.56 ±0.12	0.75 ±0.14	0.68 ±0.10	0.64 ±0.14
No Perception	0.56 ±0.17	0.44 ±0.12	0.54 ±0.16	0.51 ±0.15	0.47 ±0.17	0.57 ±0.16	0.32 ±0.08	0.54 ±0.13
Sequential	0.60 ±0.18	0.29 ±0.15	0.63 ±0.17	0.62 ±0.14	0.59 ±0.19	0.61 ±0.13	0.57 ±0.14	0.61 ±0.17
Ours	0.72 ±0.13	0.67 ±0.12	0.84 ±0.11	0.83 ±0.13	0.79 ±0.14	0.78 ±0.10	0.70 ±0.17	0.72 ±0.19

mountain task, while policies trained in a single type of terrain only achieve 28% at best (See Figure 4 for a snapshot of our policy navigating up the rugged mountain trail). These results indicate that our MTRL formulation using procedural task generation, and visual-locomotion policy architecture, results in superior generalization performance. The policy

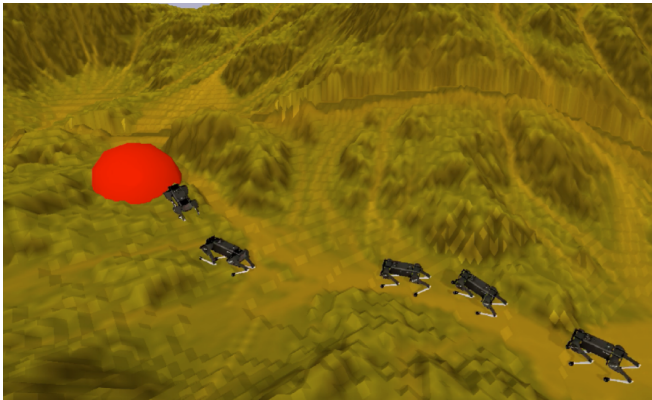


Fig. 4. Snapshot of a laikago robot navigating through mountainous terrain not encountered during training. Please refer to the supplementary video for more examples of the agent navigating challenging terrains.

learned with our system can be successfully deployed in new unseen environments.

C. Ablation Studies

We perform three ablation studies to understand the importance of each design decision in our system. Table III summarizes their impacts on the resulting generalization performance of the policy.

a) *PMTG*: We replace the locomotion component of the visual-locomotion policy with a reactive policy that does not have a trajectory generator. Our PMTG-parameterized visual-locomotion policy performs 28%-218% better than a pure reactive locomotion component. We find that PMTG produces smoother actions and leads to improved zero-shot generalization to new terrains.

b) *Exteroceptive input*: We remove the LiDAR input from the visual-locomotion policy. Observing Table III, it is clear that the exteroceptive information plays a critical role in learning generalizable locomotion policies that can adapt to a wide variety of terrains. This finding agrees with results from the field of experimental psychology, which establish the importance of exteroceptive observations in guiding foot placement when navigating over complex terrains [25], [24]. Figure 5 visualizes the trajectory produced by our visual

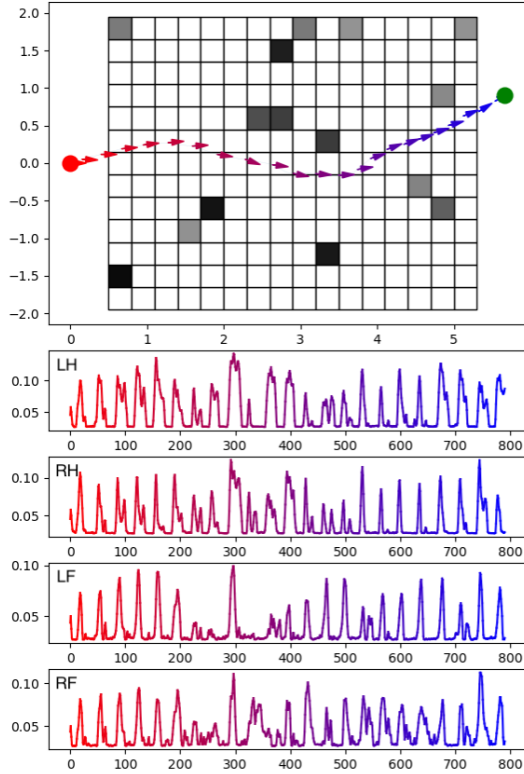


Fig. 5. Visualization of trajectory generated by our method in an environment with many obstacles. Foot Z positions for the left hind, right hind, left forward, and right forward feet are shown.

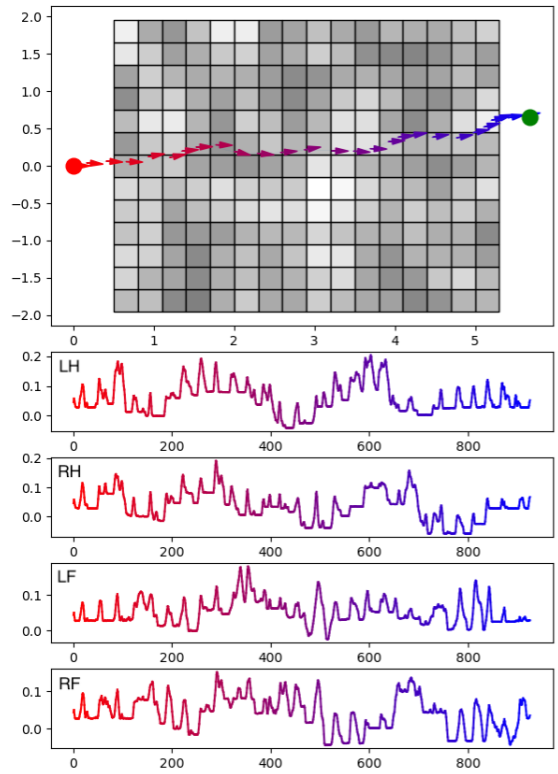
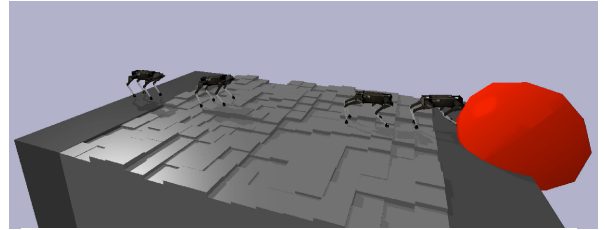


Fig. 6. Visualization of trajectory generated by our method in a rugged terrain. Foot Z positions for the left hind, right hind, left forward, and right forward feet are shown. The rugged terrain requires that the robot carefully place its feet to maintain balance.

locomotion policy in a terrain with obstacles. When walking over flat terrain, the robot’s foot height is constant and cyclic, only varying when turning to avoid obstacles. In contrast, on rugged terrain (Figure 6), the robot carefully places its feet to adapt to the geometry of the terrain to maintain balance. This careful foot placement is essential for challenging terrains and requires a visual feedback loop, which our learning system can provide.

c) MTRL training scheme: Our system generates a new random locomotion task at each episode for all the distributed workers. This ensures a steady stream of rich training data to the agent. In this study, we lower the variety of tasks supplied in a single training step by proving tasks sequentially. That is, the agent learns one task for a fixed number of training steps before switching the task. The policy trained in a sequential fashion performs poorly due to *catastrophic forgetting* [31].

These ablation studies confirm the importance of each component of our system, including the exteroceptive input and PMTG used in the visual-locomotion policy architecture, as well as our multi-task POMDP training formulation. By

combining these components, our system can learn locomotion policies that work on various terrains and demonstrate zero-shot generalization to new environments.

V. CONCLUSION

We introduce a learning system that enables legged robots to traverse various environments and demonstrates zero-shot generalization to new terrains. Our system consists of a novel multi-task reinforcement learning formulation of the locomotion problem, a visual locomotion policy architecture that encourages smooth actions and incorporates perception to modulate locomotion, and a novel procedural terrain generation algorithm that provides the agent with rich training data from a variety of simulated terrains. Our results on a suite of simulated environments show that treating legged locomotion as a multi-task POMDP leads to increased generalization performance. Additionally, we show that providing the policy with a strong prior over the space of gaits further enhances its ability to generalize to unseen terrains. In future work, we plan to evaluate our work on a real-world robot.

REFERENCES

- [1] K. Albee, A. C. Hernandez, O. Jia-Richards, and A. T. Espinoza, "Real-time motion planning in unknown environments for legged robotic planetary exploration," in *2020 IEEE Aerospace Conference*, 2020, pp. 1–9.
- [2] C. D. Bellicoso, M. Bjelonic, L. Wellhausen, K. Holtmann, F. Günther, M. Tranzatto, P. Fankhauser, and M. Hutter, "Advances in real-world applications for legged robots," *Journal of Field Robotics*, vol. 35, no. 8, pp. 1311–1326, 2018. [Online]. Available: <https://onlinelibrary.wiley.com/doi/abs/10.1002/rob.21839>
- [3] A. Iscen, K. Caluwaerts, J. Tan, T. Zhang, E. Coumans, V. Sindhwani, and V. Vanhoucke, "Policies modulating trajectory generators," in *Journal of Machine Learning Research*, ser. Proceedings of Machine Learning Research, A. Billard, A. Dragan, J. Peters, and J. Morimoto, Eds., vol. 87. PMLR, 29–31 Oct 2018, pp. 916–926. [Online]. Available: <http://proceedings.mlr.press/v87/iscen18a.html>
- [4] E. Coumans and Y. Bai, "Pybullet, a python module for physics simulation for games, robotics and machine learning," <http://pybullet.org>, 2016–2019.
- [5] F. Xia, A. R. Zamir, Z.-Y. He, A. Sax, J. Malik, and S. Savarese, "Gibson env: real-world perception for embodied agents," in *Computer Vision and Pattern Recognition (CVPR), 2018 IEEE Conference on*. IEEE, 2018.
- [6] A. W. Winkler, C. D. Bellicoso, M. Hutter, and J. Buchli, "Gait and trajectory optimization for legged systems through phase-based end-effector parameterization," *IEEE Robotics and Automation Letters*, vol. 3, no. 3, pp. 1560–1567, 2018.
- [7] D. Kim, Y. Zhao, G. Thomas, B. R. Fernandez, and L. Sentis, "Stabilizing series-elastic point-foot bipeds using whole-body operational space control," *IEEE Transactions on Robotics*, vol. 32, no. 6, pp. 1362–1379, 2016.
- [8] R. Grandia, F. Farshidian, R. Ranftl, and M. Hutter, "Feedback mpc for torque-controlled legged robots," 05 2019.
- [9] G. Bledt, M. Powell, B. Katz, J. Carlo, P. Wensing, and S. Kim, "Mit cheetah 3: Design and control of a robust, dynamic quadruped robot," 10 2018.
- [10] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press, 2018.
- [11] J. Tan, T. Zhang, E. Coumans, A. Iscen, Y. Bai, D. Hafner, S. Bohetz, and V. Vanhoucke, "Sim-to-real: Learning agile locomotion for quadruped robots," 2018.
- [12] T. Haarnoja, S. Ha, A. Zhou, J. Tan, G. Tucker, and S. Levine, "Learning to walk via deep reinforcement learning," in *Robotics: Science and Systems*, 2019.
- [13] J. Hwangbo, J. Lee, A. Dosovitskiy, D. Bellicoso, V. Tsounis, V. Koltun, and M. Hutter, "Learning agile and dynamic motor skills for legged robots," *Science Robotics*, vol. 4, no. 26, 2019.
- [14] J. Lee, J. Hwangbo, L. Wellhausen, V. Koltun, and M. Hutter, "Learning quadrupedal locomotion over challenging terrain," *Science Robotics*, vol. 5, no. 47, 2020. [Online]. Available: <https://robotics.sciencemag.org/content/5/47/eabc5986>
- [15] S. Ha, P. Xu, Z. Tan, S. Levine, and J. Tan, "Learning to walk in the real world with minimal human effort," 2020.
- [16] N. Heess, D. TB, S. Sriram, J. Lemmon, J. Merel, G. Wayne, Y. Tassa, T. Erez, Z. Wang, S. M. A. Eslami, M. Riedmiller, and D. Silver, "Emergence of locomotion behaviours in rich environments," 2017.
- [17] X. B. Peng, G. Berseth, K. Yin, and M. van de Panne, "Deeploco: Dynamic locomotion skills using hierarchical deep reinforcement learning," *ACM Transactions on Graphics (Proc. SIGGRAPH 2017)*, vol. 36, no. 4, 2017.
- [18] V. Tsounis, M. Alge, J. Lee, F. Farshidian, and M. Hutter, "Deepgait: Planning and control of quadrupedal gaits using deep reinforcement learning," 2020.
- [19] D. Chen, B. Zhou, V. Koltun, and P. Krähenbühl, "Learning by cheating," 2019.
- [20] R. Caruana, "Multitask learning: A knowledge-based source of inductive bias," in *Proceedings of the Tenth International Conference on International Conference on Machine Learning*, ser. ICML'93. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1993, p. 41–48.
- [21] M. Hessel, H. Soyer, L. Espeholt, W. Czarnecki, S. Schmitt, and H. van Hasselt, "Multi-task deep reinforcement learning with popart," 2018.
- [22] T. Yu, D. Quillen, Z. He, R. Julian, K. Hausman, C. Finn, and S. Levine, "Meta-world: A benchmark and evaluation for multi-task and meta reinforcement learning," 2019.
- [23] T. Yu, S. Jumar, A. Gupta, S. Levine, K. Hausmann, and C. Finn, "Multi-task reinforcement learning without interference," 2019.
- [24] J. S. Matthis, J. L. Yates, and M. M. Hayhoe, "Gaze and the control of foot placement when walking in natural terrain," *Current Biology*, vol. 28, no. 8, pp. 1224 – 1233.e5, 2018. [Online]. Available: <http://www.sciencedirect.com/science/article/pii/S0960982218303099>
- [25] J. S. Matthis and B. R. Fajen, "Visual control of foot placement when walking over complex terrain," *J Exp Psychol Hum Percept Perform*, vol. 40, no. 1, pp. 106–115, Feb 2014.
- [26] Unitree, "Laikago: Let's challenge new possibilities," 2018. [Online]. Available: <http://www.unitree.cc/e/action/ShowInfo.php?classid=6&id=1>
- [27] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," 2017.
- [28] S. Guadarrama, A. Korattikara, O. Ramirez, P. Castro, E. Holly, S. Fishman, K. Wang, E. Gonina, N. Wu, E. Kokioyopoulou, L. Sbaiz, J. Smith, G. Bartók, J. Berent, C. Harris, V. Vanhoucke, and E. Brevedo, "TF-Agents: A library for reinforcement learning in tensorflow," <https://github.com/tensorflow/agents>, 2018, [Online; accessed 25-June-2019]. [Online]. Available: <https://github.com/tensorflow/agents>
- [29] M. Andrychowicz, A. Raichuk, P. Stańczyk, M. Orsini, S. Girgin, R. Marinier, L. Hussenot, M. Geist, O. Pietquin, M. Michalski, S. Gelly, and O. Bachem, "What matters in on-policy reinforcement learning? a large-scale empirical study," 2020.
- [30] J. Schulman, P. Moritz, S. Levine, M. Jordan, and P. Abbeel, "High-dimensional continuous control using generalized advantage estimation," 2018.
- [31] M. McCloskey and N. Cohen, "Catastrophic interference in connectionist networks: The sequential learning problem," *Psychology of Learning and Motivation - Advances in Research and Theory*, vol. 24, no. C, pp. 109–165, Jan. 1989.