Rapid Locomotion via Reinforcement Learning

Gabriel B. Margolis*, Ge Yang*†, Kartik Paigwar, Tao Chen, and Pulkit Agrawal†‡

***Abstract*— Agile maneuvers such as sprinting and high-speed turning in the wild are challenging for legged robots. Reinforcement learning provides a promising framework for addressing this challenge. We present an end-to-end learned controller that achieves record agility for the MIT Mini Cheetah, sustaining speeds up to $3.9\,\mathrm{m/s}$. This system runs and turns fast on natural terrains like grass, ice, and gravel and responds robustly to disturbances. Our controller is a neural network trained in simulation via reinforcement learning and transferred to the real world. The two key components are: (i) an adaptive curriculum on velocity commands; and (ii) an online system identification strategy for sim-to-real transfer leveraged from prior work. Videos of the robot's behaviors are available at: `https://agility.csail.mit.edu/`.**

***Paper Type* – Recent Work [26].**

I. INTRODUCTION

Running fast on natural terrains is challenging. Different terrains exhibit different characteristics, ranging from variable friction and softness to sloped and uneven geometry. As a robot attempts to move at faster speeds, the impact of terrain variation on controller performance increases [6], [10]. Some additional physical considerations only begin to influence the robot's dynamics at high speeds, including the enforcement of actuator limits [7], [8], [13], the regulation of large contact forces [19], and body control during flight phases [8], [19]. One possibility is to resolve these issues by making targeted improvements to the hand-designed models used in model-based control. Impressive progress has been made in this direction [5], [6], [7], [9], [8], [10], [13], [19]. However, in model-based control, the robot's behavior and robustness are dependent on the creativity and investment of the human designer, who must invent simplified (or *reduced-order*) models that allow for reasoning about particular properties of the robot and environment under the constraint of real-time computation.

How can we perform real-time control in complex environments where efficient reduced-order models may not exist or are currently unknown? One might try to optimize the robot's actions with respect to a full physics model, but for a complex task such as running on natural terrain, this is not possible in real-time. An alternative is to amortize the cost of trajectory optimization by learning a direct mapping from sensory observations to actions (a *policy*). *Reinforcement learning* (RL) provides a way to learn such a policy. In this approach, the human designs a set of training environments and a reward function that specifies each task at hand. RL algorithms automatically discover the policy that maximizes reward across those environments and tasks. Because the RL framework does not require a human engineer to design accurate and efficient reduced-order models, it is less reliant on human effort. As a result, RL offers a scalable controller synthesis scheme for complex tasks in challenging environments. Recent works have successfully used RL for training locomotion controllers [21], [22], [25], [28], [33], [35].

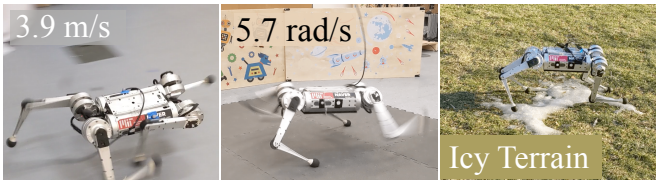


Fig. 1: Our end-to-end learned controller enables the MIT Mini Cheetah to execute fast running, spinning, and deal with natural terrains. All are realized by a single neural network trained in sim and deployed zero-shot in the real world.

Our goal is to construct a system that can traverse terrains at a large range of linear and angular velocities. This corresponds to a multi-task RL setup, where running with each combination of linear and angular velocity constitutes a task. Learning multi-task policies via RL on a broad set of tasks is known to be difficult [14]. The naive approach to train a multi-task RL policy is to provide all tasks simultaneously. However, if the majority of the tasks are challenging or infeasible, this approach is likely to fail. This is the case in the setting of high-speed locomotion, where the effect of centrifugal force constrains a robot's physically feasible combinations of linear velocity and angular speed. To provide the agent with tasks of appropriate difficulty, it makes sense to initially select easier tasks and then slowly increase their complexity using a curriculum [3]. Curriculum learning has been leveraged for training many robotic systems in the past [23], [27], [33], [38]. Manual design of curriculum is not appropriate when the difficulty or feasibility of tasks is not known apriori. We propose an automatic curriculum strategy that expands the set of tasks while respecting the physical constraints of locomotion. The proposed strategy yields performance improvements in learning omnidirectional high-speed locomotion.

When deployed in the real world, our system performs rapid locomotion in both indoors and outdoors, and successfully negotiates challenging terrains and disturbances. Our work fills in a gap in the literature: it demonstrates that for a small, agile quadruped, reinforcement learning can produce controllers that simultaneously embody diverse, high-speed behaviors and deploy successfully in the wild.

* Equal contribution. All authors are affiliated with the MIT Improbable AI Lab. Authors are also affiliated with †NSF AI Institute of AI and Fundamental Interactions (IAIFI), ‡ MIT-IBM Watson AI Lab at MIT, and the Computer Science and Artificial Laboratory (CSAIL). Correspondence to {gmargo, geyang}@csail.mit.edu

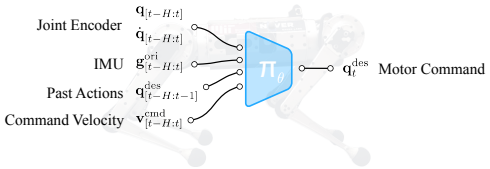


Fig. 2: Our controller is a learned mapping from sensory inputs to desired joint positions. We parameterize it as 5-layer neural network π_θ with parameters θ optimized in simulation.

II. EXPERIMENTAL SETUP

Hardware: We use the MIT Mini Cheetah [18] as our experimental platform. The robot stands $30\,\mathrm{cm}$ tall and weighs $9\,\mathrm{kg}$. It is equipped with 12 quasi-direct-drive actuators capable of maximum output torque $17\,\mathrm{N\,m}$. The robot's sensor suite consists of joint position encoders and an inertial measurement unit (IMU). Our neural network controller runs at $50\,\mathrm{Hz}$ on an onboard NVIDIA Jetson TX2 NX computer.
Simulation: We use the IsaacGym simulator [24] and code adapted from the open-source repository in [33]. We collect 400 million simulated timesteps using 4000 parallel agents for policy training. This is roughly equivalent to 92 days of data, which in simulation we can collect in under three hours of wall-clock time on a single NVIDIA RTX 3090 GPU.

III. METHOD

A. Control Architecture

Observation Space consists of: joint angles, $\mathbf{q}_t \in \mathbb{R}^{12}$, measured using motor encoders; joint velocities, $\dot{\mathbf{q}}_t \in \mathbb{R}^{12}$; and $\mathbf{g}_t^{\text{ori}} \in \mathbb{R}^3$ denoting the orientation of the gravity vector in the robot's body frame measured using the IMU. Policy $\pi_\theta(\cdot)$ takes as input a history of previous observations and actions denoted by $\mathbf{o}_{t-H:t}$ where $\mathbf{o}_t = [\mathbf{q}_t, \dot{\mathbf{q}}_t, \mathbf{g}_t^{\text{ori}}, \mathbf{a}_{t-1}]$. Because we are learning a command-conditioned policy, the input to the policy is $\mathbf{x}_{t-H:t}$ where $\mathbf{x}_t = \mathbf{o}_t \bigoplus \mathbf{v}_t^{\text{cmd}}$. During deployment, the body velocity command $\mathbf{v}_t^{\text{cmd}}$ is specified by a human operator via remote control.
Action Space The action, $\mathbf{a}_t \in \mathbb{R}^{12}$, assigns joint position commands for a PD controller. The proportional gain is 20 and the derivative gain is 0.5. We chose these low gains to promote smooth motions, and did not tune them during the course of our experiments.
Reward Function closely follows [33] with task reward terms for linear and angular velocity tracking, as well as a set of auxiliary terms for stability, smoothness, and safety.

B. Teacher-Student Training

We randomize $\mathbf{d}_t$: the body mass, center of mass, motor strength, ground friction, and ground restitution, to facilitate sim-to-real transfer and adaptation to diverse environments. A *teacher policy* that observes $\mathbf{d}_t$ is trained using reinforcement learning. Then, a deployable *student policy* is trained using supervised learning to imitate the teacher while observing only the state history.

1) Teacher Policy: We separate the teacher policy $\pi_T(\mathbf{x}_t, \mathbf{d}_t)$ into two modules g_{θ_d} and π_{θ_b}, such that $\pi_T(\mathbf{x}_t, \mathbf{d}_t) = \pi_{\theta_b}(\mathbf{x}_t, g_{\theta_d}(\mathbf{d}_t))$. The first module g_{θ_d} is the encoder, $\mathbf{z}_t = g_{\theta_d}(\mathbf{d}_t)$, which compresses $\mathbf{d}_t$ into an intermediate latent vector $\mathbf{z}_t$. The second module π_{θ_b} is the policy body, $\mathbf{a}_t = \pi_{\theta_b}(\mathbf{x}_t, \mathbf{z}_t)$, which predicts an action from the latent $\mathbf{z}_t$ and observation $\mathbf{x}_t$. Each module is parameterized as a neural network with ELU activations. We optimize the teacher's parameters θ_d, θ_b together using PPO [34] to maximize the future discounted reward.

2) Student Policy: The student policy $\pi_S(\mathbf{x}_t, \mathbf{x}_{[t-h:t-1]}) = \pi_{\theta_b}(\mathbf{x}_t, h_{\theta_a}(\mathbf{x}_{[t-h:t-1]}))$ imitates the teacher's behavior during deployment without access to $\mathbf{d}_t$. The student policy is constructed by replacing encoder $g_{\theta_d}(\mathbf{d}_t)$ with an adaptation module, $\hat{\mathbf{z}}_t = h_{\theta_a}(\mathbf{x}_{[t-h:t-1]})$, which estimates the latent $\hat{\mathbf{z}}_t$ from state history $\mathbf{x}_{[t-h:t-1]}$. We train the adaptation module so that its predictions $\hat{\mathbf{z}}_t$ match the encoder's output $\mathbf{z}_t = g_{\theta_d}(\mathbf{d}_t)$ as closely as possible. To this end, we optimize parameters θ_a using supervised learning on on-policy data, using the loss function $\mathcal{L}_{\theta_d} = (\hat{\mathbf{z}}_t - \mathbf{z}_t)^2$. This is the same teacher-student formulation from [21], using a shorter history of $h = 15$ observations for a smaller network and faster inference.

C. Curriculum Strategy

The agent learns by attempting to track different velocity commands. Longitudinal and yaw velocity commands $\mathbf{v}_x^{\text{cmd}}$, $\boldsymbol{\omega}_z^{\text{cmd}}$ during episode k are sampled from probability distribution $p^k_{\mathbf{v}_x, \boldsymbol{\omega}_z}(\cdot, \cdot)$, which changes according to a curriculum.

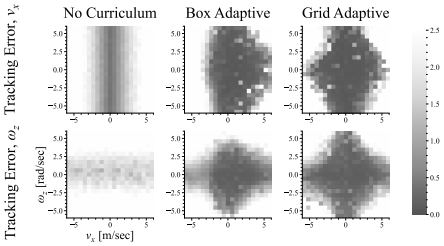
1) Box Adaptive Curriculum Update Rule: At episode k, the linear and angular velocity commands for the agent are sampled independently: $\mathbf{v}_x^{\text{cmd}} \sim p^k_{\mathbf{v}}(\cdot)$, $\boldsymbol{\omega}_z^{\text{cmd}} \sim p^k_{\boldsymbol{\omega}_z}(\cdot)$. Suppose the agent receives rewards $r_{v_x^{\text{cmd}}}, r_{\omega_z^{\text{cmd}}}$ in its attempt to follow $\mathbf{v}_x^{\text{cmd}}$, $\boldsymbol{\omega}_z^{\text{cmd}}$. Then we apply the update rule

$$p^{k+1}_{\mathbf{v}_x}(\mathbf{v}_x^{\text{n}}) \leftarrow \begin{cases} p^k_{\mathbf{v}_x}(\mathbf{v}_x^{\text{n}}) & r_{v_x^{\text{cmd}}} < \gamma\,, \\ 1 & \text{otherwise.} \end{cases} \tag{1}$$

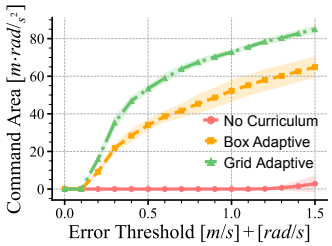
which increases the probability density on neighbors $\mathbf{v}_x^{\text{n}}$ of $\mathbf{v}_x^{\text{cmd}}$ and $\boldsymbol{\omega}_z^{\text{n}}$ of $\boldsymbol{\omega}_z^{\text{cmd}}$. Here, neighboring commands are defined as the adjacent elements in the (discretized) domain of each marginal distribution: $\mathbf{v}_x^{\text{n}} \in \{\mathbf{v}_x^{\text{cmd}} - 0.5, \mathbf{v}_x^{\text{cmd}} + 0.5\}$ and $\boldsymbol{\omega}_z^{\text{n}} \in \{\boldsymbol{\omega}_z^{\text{cmd}} - 0.5, \boldsymbol{\omega}_z^{\text{cmd}} + 0.5\}$.

2) Grid Adaptive Curriculum Update Rule: As before, if the agent succeeds in this region of command space, we would like to increase the difficulty by adding neighboring regions to the sampling distribution. However, the distributions of $\mathbf{v}_x^{\text{cmd}}$ and $\mathbf{v}_y^{\text{cmd}}$ are no longer constrained to be independent. This enables us to revise our update with a new definition of the neighboring commands. Upon termination of an episode with command $[\mathbf{v}_x^{\text{cmd}}, \boldsymbol{\omega}_z^{\text{cmd}}]$ where the agent received rewards $r_{v_x^{\text{cmd}}}, r_{\omega_z^{\text{cmd}}}$, we use the following update:

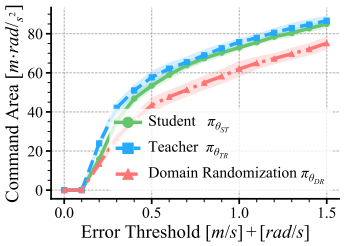
$$p^{k+1}_{\mathbf{v}_x, \boldsymbol{\omega}_z}(\mathbf{v}_x^{\text{n}}, \boldsymbol{\omega}_z^{\text{n}}) \leftarrow \begin{cases} p^k_{\mathbf{v}_x, \boldsymbol{\omega}_z}(\mathbf{v}_x^{\text{n}}, \boldsymbol{\omega}_z^{\text{n}}) & r_{v_x^{\text{cmd}}} < \gamma \text{ or } r_{\omega_z^{\text{cmd}}} < \gamma, \\ 1 & \text{otherwise.} \end{cases} \tag{2}$$



(a) Heatmap of converged tracking error for curriculum strategies. Darker is better.



(b) Training with Grid Adaptive curriculum reduces tracking error.



(c) Online system identification reduces tracking error, particularly at high speeds.

Fig. 3: Velocity tracking performance across high-speed running and spinning. Choice of curriculum strategy and use of teacher-student training each improve top-end speed.

This update adds probability density to the neighboring velocity commands $[\mathbf{v}_x^{\text{n}}, \boldsymbol{\omega}_z^{\text{n}}]$ of $[\mathbf{v}_x^{\text{cmd}}, \boldsymbol{\omega}_z^{\text{cmd}}]$. Here, neighboring commands are defined as neighbors in the 4-connected grid domain of $p^k_{\mathbf{v}_x, \boldsymbol{\omega}_z}(\cdot, \cdot)$, which is a discrete grid with resolution $[0.5\,\text{m/s}, 0.5\,\text{rad/s}]$.

IV. RESULTS

A. Curriculum for Learning High-Speed Locomotion

Figures 3b, 3a illustrate the tracking error of the policies trained with different curricular strategies as heatmaps in the $\mathbf{v}_x^{\text{cmd}}$-$\boldsymbol{\omega}_z^{\text{cmd}}$ plane. The shading corresponds to tracking error, with darker shading indicating lower error. We observe that the policy trained without any curriculum fails to learn; the reward almost always remains small, providing very little learning signal. The performance of the system is improved by implementing the Box Curriculum. The agent first learns to track well in the small initial command distribution, then gradually increases its capability as the commands become larger. Using the Grid Curriculum, the performance of the policy is best, as evidenced by the larger command area. By maintaining a joint distribution over linear and angular velocity, the grid curriculum models the interaction between them. When high linear and angular velocities are combined, a body experiences centrifugal force which must be countered by frictional force to remain on the desired path. This induces a constraint on maximum combinations of linear and angular velocity such that the two vary inversely $[\boldsymbol{\omega}_z \sim 1/\mathbf{v}_x]$ when the constraint is active. This phenomenon is in agreement with the apparent inverse shape of the command area boundary shown in Figure 3a, which indicates that the robot has reached a physical limit on its ability to turn at high speed.

B. Real-world Testing

Video of all experiments is viewable on the project website: `https://agility.csail.mit.edu/`.

Indoor Running To evaluate how fast our robot can run in the real world, we ramped the velocity command to $6.0\,\text{m/s}$. We conducted this experiment in a motion capture arena to accurately estimate the robot's running speed (Figure 1, left). We found that policies trained with a system identification module and grid curriculum were capable of sustaining an average of $3.8\,\text{m/s}$ on hardware across multiple seeds, with the best of three trials reaching a sustained speed of $3.9\,\text{m/s}$. This is higher than the previous record of $3.7\,\text{m/s}$ achieved by a model-predictive control algorithm [19]. Together with concurrent work [16], this is substantially faster than previously reported applications of reinforcement learning.

To understand the impact of online system identification in crossing the sim-to-real gap, we also evaluated policies without the system identification module. We found that these policies only sustained an average peak speed of $2.49\,\text{m/s}$. We conclude that online system identification helps increase maximum agility in the real world. It partially compensates for the sim-to-real gap and improves agent performance. It is unclear how much of the remaining performance gap between simulated and real environments arises from a shortcoming of the controller or from the physical limits of the real-world robot and terrain.

Outdoor Running Outdoor terrain presents multiple challenges not present in indoor running, among which are change in ground height, friction, and terrain deformation. Under these variations, the robot must actuate its joints differently to reach high speed than it would on a treadmill or paved road. To test if our system is able to run on outdoor terrains, we conducted an outdoor dash across an uneven grassy patch. We record an outdoor 10-meter dash time of 2.94 seconds, corresponding to an average speed of $3.4\,\text{m/s}$.

Yaw Control We evaluate our controller's yaw velocity control in the lab setting as shown in Figure 1 (center). The robot accelerates to a maximum yaw rate of $5.7\,\text{rad/s}$, then stops safely. This is 90% of the fastest yaw rate recorded on the Mini Cheetah using a model-based controller [4]. However, the model-based records were achieved using different controllers for linear [19] and angular [4] velocity, while all of our results for running and spinning inside and outside the lab are achieved by a single policy. To challenge the controller's spinning skills, we brought the robot outside after a snowstorm and piloted it onto an icy patch, illustrated in Figure 1 (right). The robot maintained stability while spinning as its feet frequently slipped on ice.

Response to Terrain Changes and Hardware Failures We tested our system in a diverse set of challenging real-world scenarios: (1) ascending a steep incline made of small pebbles. (2) maintaining balance despite a mechanical blockage to one motor. (3) tripping at high speed, flying

upside down and landing on its feet. (4) recovering via a change in gait after tripping over a small barrier. These qualitative results are presented in the accompanying video.

We also deployed the model-predictive controller from [19] in scenarios (1) and (4), which were the most convenient to replicate. We found that unlike our learned controller, the baseline did not recover from (1) slipping down the gravelly incline and (4) tripping over the barrier. We want to emphasize that while these results are encouraging, we are not claiming that is not possible to design a non-learning system that is as or more robust than our learned policy. Our claim is simply that by freeing the human from directly refining the robot's model or behavior, RL offers a scalable alternative to obtain robust responses to diverse conditions.

C. Impact of Online System Identification

We evaluate (1) the benefit of access to privileged information when learning to run at high speeds and (2) the ability of the student policy to retain the performance gains using only available sensor data. We compare $\pi_{DR}(\mathbf{x}_t)$, $\pi_T(\mathbf{x}_t, \mathbf{d}_t)$, $\pi_S(\mathbf{x}_t, \mathbf{x}_{[t-h:t-1]})$ as described in Section III-B in the high-speed regime. All policies are trained under the same randomization of the privileged state.

We find that access to privileged information expands the command area, which corresponds to increased performance across all speeds. Figure 3c plots the command area for the three policies as the threshold for error increases. The privileged teacher π_T trained with access to environment parameters attains a strictly larger command area than the policy π_{DR} trained with only the robot state. Using the online system identification module, we show that the student policy π_S is able to nearly match the performance of the teacher.

V. RELATED WORK

Model-based Control for Locomotion Seminal work in the field used simplified models and hand-specified gaits to make legged robots balance and move dynamically [12], [17], [31]. Subsequent works introduced expanded models with layered control architectures capable of operation on subsets of rough, soft, and slippery terrains [6], [10], [20], [29], [30], [32]. Recent innovations have addressed specific limitations of simple models with respect to high-speed running, including whole-body control [8] and regulation of large ground reaction forces under body motion [5], [19].

Reinforcement Learning for Locomotion [15], [36], [37] combined model-free reinforcement learning with dynamics randomization to learn locmotion policies in simulation and transfer them to the real world for the ANYmal, Cassie, and Minitaur robots. Followup works expanded ANYmal's robustness by training on diverse terrains using the teacher-student learning paradigm [22], [28], [33]. The mechanical design of the ANYmal robot is thought to limit it from running at higher speeds. [11], [21] investigated the capability of model-free controllers to efficiently traverse diverse terrains on the Unitree A1, a small robot with similar size, actuation, and cost to the Mini Cheetah. The A1's built-in MPC controller has a maximum running speed of $3.3\,\mathrm{m/s}$, but these learning-based works only demonstrated running up to maximum speed $1.8\,\mathrm{m/s}$. Concurrently with our work, Ji et al. [16] also trained agile running policies for the Mini Cheetah robot using reinforcement learning. Unlike our work, [16] used a fixed-schedule curriculum on forward linear velocity only.

Curricula for On-Policy Reinforcement Learning Prior works have shown that a curriculum on environments can enable discovery of behaviors that are challenging to learn directly using reinforcement learning [3]. [2] demonstrated an Automatic Domain Randomization strategy in which domain randomization scales are increased based on agent performance. Curricula on environments have also been demonstrated in locomotion context; [22], [28], [33] applied a curriculum on terrains to learn highly robust walking controllers on nonflat ground. [38] evaluated terrain curriculum strategies, including adaptive curricula, in the setting of stepping stone traversal with a physically simulated biped.

VI. DISCUSSION

This work has shown that a neural network controller trained fully end-to-end in simulation can push a small quadruped to the limits of its agility, achieving omindirectional mobility competitive with well-engineered model-predictive controllers in the regime of high speed. Because our controller uses minimal sensing, we are able to implement it on a low-cost robot [18] with commercially available analogues [1]. Our method can therefore be readily tested and built upon by others using relatively accessible materials.

Our ability to fully characterize the robot's outdoor performance was limited by instrumentation and repeatability. We cannot use motion capture to record the robot's state outdoors, as we do in the lab. Also, it is unsafe and impractical to record a large number of high-speed trips on the real robot. This forced our analysis of the robot's outdoor behavior to be more qualitative, while we performed our quantitative analysis in the laboratory setting.

The behaviors we demonstrate in this work are diverse, but still limited relative to the full space of possible locomotion tasks. The system we demonstrate has only been trained for the task of controlling the robot's body velocity in the ground plane. Other categories of behavior such as jumping, choreographed dance, and loco-manipulation would potentially require a very different task specification. Our system also does not use vision, so in general it cannot perform tasks that require planning ahead of time, like efficiently climbing up stairs or avoiding pitfalls.

Finally, we emphasize that while our system demonstrates high speed, its distinctive locomotion gait should not be interpreted as generally "better" than the many possible alternatives. To the contrary, many designers wish to optimize for objectives beyond speed, such as energy efficiency or minimization of wear on the robot. Body speed alone is an underspecified objective, meaning that many equally preferable motions may attain the same speed. To combine learned agile locomotion with auxiliary objectives or human preferences remains a promising direction for future work.

REFERENCES

[1] Unitree Robotics, A1, 2022, https://www.unitree.com/products/a1, [Online; accessed Apr. 2022].

[2] I. Akkaya, M. Andrychowicz, M. Chociej, M. Litwin, B. McGrew, A. Petron, A. Paino, M. Plappert, G. Powell, R. Ribas, *et al.*, "Solving Rubik's cube with a robot hand," *arXiv preprint*, 2019.

[3] Y. Bengio, J. Louradour, R. Collobert, and J. Weston, "Curriculum learning," in *Proc. Int. Conf. Mach. Learn. (ICML)*, Montreal, Canada, June 2009, pp. 41–48.

[4] G. Bledt and S. Kim, "Extracting legged locomotion heuristics with regularized predictive control," in *Proc. IEEE Int. Conf. Robot. Automat. (ICRA)*, Virtual, May 2020, pp. 406–412.

[5] G. Bledt, M. J. Powell, B. Katz, J. Di Carlo, P. M. Wensing, and S. Kim, "MIT Cheetah 3: Design and control of a robust, dynamic quadruped robot," in *Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst. (IROS)*, Madrid, Spain, Oct. 2018, pp. 2245–2252.

[6] W. Bosworth, J. Whitney, S. Kim, and N. Hogan, "Robot locomotion on hard and soft ground: Measuring stability and ground properties in-situ," in *Proc. IEEE Int. Conf. Robot. Automat. (ICRA)*, Stockholm, Sweden, May 2016, pp. 3582–3589.

[7] M. Chignoli, D. Kim, E. Stanger-Jones, and S. Kim, "The MIT humanoid robot: Design, motion planning, and control for acrobatic behaviors," in *Proc. IEEE/RAS Int. Conf. Humanoid Robot. (Humanoids)*, Munich, Germany, July 2021, pp. 1–8.

[8] H. Dai, A. Valenzuela, and R. Tedrake, "Whole-body motion planning with centroidal dynamics and full kinematics," in *Proc. IEEE/RAS Int. Conf. Humanoid Robot. (Humanoids)*, Madrid, Spain, Nov. 2014, pp. 295–302.

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

[10] S. Fahmi, M. Focchi, A. Radulescu, G. Fink, V. Barasuol, and C. Semini, "STANCE: Locomotion adaptation over soft terrain," *IEEE Trans. Robot. (T-RO)*, vol. 36, no. 2, pp. 443–457, Apr. 2020.

[11] Z. Fu, A. Kumar, J. Malik, and D. Pathak, "Minimizing energy consumption leads to the emergence of gaits in legged robots," in *Proc. Conf. Robot Learn. (CoRL)*, London, UK, Nov. 2021, pp. 928–937.

[12] A. Herdt, N. Perrin, and P.-B. Wieber, "Walking without thinking about it," in *Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst. (IROS)*, Taipei, Taiwan, Oct. 2010, pp. 190–195.

[13] A. Herzog, S. Schaal, and L. Righetti, "Structured contact force optimization for kino-dynamic motion generation," in *Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst. (IROS)*, Daejeon, Korea, Oct. 2016, pp. 2703–2710.

[14] M. Hessel, H. Soyer, L. Espeholt, W. Czarnecki, S. Schmitt, and H. van Hasselt, "Multi-task deep reinforcement learning with popart," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, no. 01, 2019, pp. 3796–3803.

[15] J. Hwangbo, J. Lee, A. Dosovitskiy, D. Bellicoso, V. Tsounis, V. Koltun, and M. Hutter, "Learning agile and dynamic motor skills for legged robots," *Sci. Robot.*, vol. 4, no. 26, p. aau5872, Jan. 2019.

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

[17] S. Kajita, F. Kanehiro, K. Kaneko, K. Fujiwara, K. Harada, K. Yokoi, and H. Hirukawa, "Biped walking pattern generation by using preview control of zero-moment point," in *Proc. IEEE Int. Conf. Robot. Automat. (ICRA)*, vol. 2, Taipei, Taiwan, Sept. 2003, pp. 1620–1626.

[18] B. Katz, J. D. Carlo, and S. Kim, "Mini Cheetah: A platform for pushing the limits of dynamic quadruped control," in *Proc. IEEE Int. Conf. Robot. Automat. (ICRA)*, Montreal, QC, Canada, May 2019, pp. 6295–6301.

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

[20] S. Kuindersma, R. Deits, M. Fallon, A. Valenzuela, H. Dai, F. Permenter, T. Koolen, P. Marion, and R. Tedrake, "Optimization-based locomotion planning, estimation, and control design for the atlas humanoid robot," *Auton. Robot.*, vol. 40, no. 3, pp. 429–455, July 2015.

[21] A. Kumar, Z. Fu, D. Pathak, and J. Malik, "RMA: Rapid motor adaptation for legged robots," in *Proc. Robot.: Sci. and Syst. (RSS)*, Virtual, July 2021.

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

[23] R. Li, A. Jabri, T. Darrell, and P. Agrawal, "Towards practical multi-object manipulation using relational reinforcement learning," in *Proc. IEEE Int. Conf. Robot. Automat. (ICRA)*, Virtual, May 2020, pp. 4051–4058.

[24] V. Makoviychuk, L. Wawrzyniak, Y. Guo, M. Lu, K. Storey, M. Macklin, D. Hoeller, N. Rudin, A. Allshire, A. Handa, *et al.*, "Isaac Gym: High performance GPU-based physics simulation for robot learning," *arXiv preprint*, 2021.

[25] G. B. Margolis, T. Chen, K. Paigwar, X. Fu, D. Kim, S. Kim, and P. Agrawal, "Learning to jump from pixels," in *Proc. Conf. Robot Learn. (CoRL)*, London, UK, Nov. 2021, pp. 1025–1034.

[26] G. B. Margolis, G. Yang, K. Paigwar, T. Chen, and P. Agrawal, "Rapid locomotion via reinforcement learning," *Proc. Robot.: Sci. and Syst. (RSS)*, June 2022.

[27] T. Matiisen, A. Oliver, T. Cohen, and J. Schulman, "Teacher–student curriculum learning," *IEEE Trans. Neural Net. Lrn. Sys.*, vol. 31, no. 9, pp. 3732 – 3740, Sept. 2019.

[28] T. Miki, J. Lee, J. Hwanbo, L. Wellhausen, V. Koltun, and M. Hutter, "Learning robust perceptive locomotion for quadrupedal robots in the wild," *Sci. Robot.*, vol. 7, no. 62, p. abk2822, Jan. 2022.

[29] H.-W. Park, P. M. Wensing, and S. Kim, "High-speed bounding with the MIT Cheetah 2: Control design and experiments," *Int. J. Robot. Res. (IJRR)*, vol. 36, no. 2, pp. 167–192, Mar. 2017.

[30] M. Raibert, K. Blankespoor, G. Nelson, and R. Playter, "Bigdog, the rough-terrain quadruped robot," *IFAC Proceedings Volumes*, vol. 41, no. 2, pp. 10 822–10 825, Mar. 2008.

[31] M. H. Raibert, *Legged Robots That Balance*. MIT press, 1986.

[32] L. Righetti and S. Schaal, "Quadratic programming for inverse dynamics with optimal distribution of contact forces," in *Proc. IEEE/RAS Int. Conf. Humanoid Robot. (Humanoids)*, Osaka, Japan, Nov. 2012, pp. 538–543.

[33] N. Rudin, D. Hoeller, P. Reist, and M. Hutter, "Learning to walk in minutes using massively parallel deep reinforcement learning," in *Proc. Conf. Robot Learn. (CoRL)*, London, UK, Nov. 2021, pp. 91–100.

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

[35] J. Siekmann, K. Green, J. Warila, A. Fern, and J. Hurst, "Blind bipedal stair traversal via sim-to-real reinforcement learning," in *Proc. Robot.: Sci. and Syst. (RSS)*, Virtual, July 2021.

[36] J. Tan, T. Zhang, E. Coumans, A. Iscen, Y. Bai, D. Hafner, S. Bohez, and V. Vanhoucke, "Sim-to-real: Learning agile locomotion for quadruped robots," in *Proc. Robot.: Sci. and Syst. (RSS)*, Pittsburgh, Pennsylvania, USA, June 2018, pp. 1–9.

[37] Z. Xie, P. Clary, J. Dao, P. Morais, J. Hurst, and M. van de Panne, "Learning locomotion skills for Cassie: Iterative design and sim-to-real," in *Proc. Conf. Robot Learn. (CoRL)*, Osaka, Japan, Nov. 2020, pp. 1–13.

[38] Z. Xie, H. Y. Ling, N. H. Kim, and M. van de Panne, "ALLSTEPS: Curriculum-driven learning of stepping stone skills," *Computer Graphics Forum*, vol. 39, no. 8, pp. 213–224, Nov. 2020.

ACKNOWLEDGEMENT

The authors thank the members of the Improbable AI Lab and the Biomimetic Robotics Laboratory for providing valuable feedback on the project direction and the manuscript. We are grateful to MIT Supercloud and the Lincoln Laboratory Supercomputing Center for providing HPC resources. The Mini Cheetah robot used in this work was donated by the MIT Biomimetic Robotics Laboratory and NAVER. This research was supported by the DARPA Machine Common Sense Program and in part by the MIT-IBM Watson AI Lab, and the National Science Foundation under Cooperative Agreement PHY-2019786 (The NSF AI Institute for Artificial Intelligence and Fundamental Interactions, http://iaifi.org/). Research was also sponsored by the United States Air Force Research Laboratory and the United States Air Force Artificial Intelligence Accelerator and was accomplished under Cooperative Agreement Number FA8750-19-2-1000. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the United States Air Force or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation herein.