Visually Guided Agile Quadruped Locomotion

Gabriel B Margolis, Tao Chen, Xiang Fu, Sangbae Kim, Pulkit Agrawal
Massachusetts Institute of Technology
{gmargo, taochen, xiangfu, sangbae, pulkitag}@mit.edu

Kartik Paigwar
Arizona State University
kpaigwar@asu.edu

Donghyun Kim
University of Massachusetts Amherst
donghyunkim@cs.umass.edu

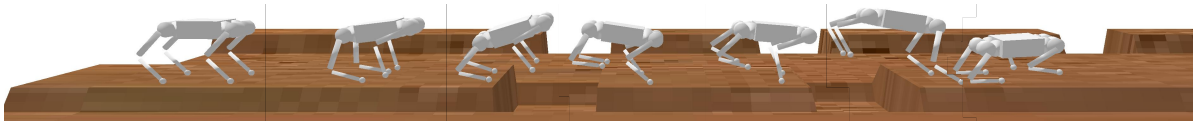

Figure 1: We present a method for vision-guided agile motion that enables the Mini Cheetah robot to cross gaps up to 1.3× its body length. Video is available at https://sites.google.com/view/visual-loco.

Abstract

Today's robotic quadruped systems can walk over a diverse set of natural and complex terrains. They are robust to perturbations and adverse conditions such as snow, rain, slip, rubble, etc. However, combining visual information with adaptive and agile behaviors, such as jumping across gaps or obstacles, is beyond the scope of such previous works. We present a system that can, in real-time, process visual data to command a quadruped robot to jump over wide gaps on diverse terrains. These behaviors are made possible using a novel combination of low-level reactive controllers and a learned high-level model-free planner. Overall, our method synthesizes the best of classical control and deep reinforcement learning to achieve dynamic and agile locomotion.

1. Introduction

One of the grand challenges in robotics is to construct systems that can successfully navigate novel and complex landscapes. In unpublished work, the *Spot* robot from Boston Dynamics, and in published work, the *ANYmal* and *MIT Cheetah* robots have achieved impressive performance in traversing a wide diversity of natural and man-made terrains [9, 13]. Although such prior works address robust blind locomotion both theoretically and practically, they leave open the problem of adapting a locomotion controller in response to visual information about upcoming terrain.

We present a general framework for incorporating visual inputs into model-based control. We demonstrate results in simulated and real-world experiments using the MIT Mini Cheetah robot. We achieve the *best-of-both-worlds* of classical control and learning by combining a low-level model predictive controller (MPC) and whole-body impulse controller (WBIC) that commands joint torques, positions, and velocities with a high-level trajectory planner trained using deep reinforcement learning (RL). Due to the high sample complexity of training RL policies, we first perform learning in simulation and then transfer to the real world. We find that planning in an action space of target trajectories, rather than motor commands, mitigates some challenges associated with generalization and sim-to-real transfer in visuomotor control.

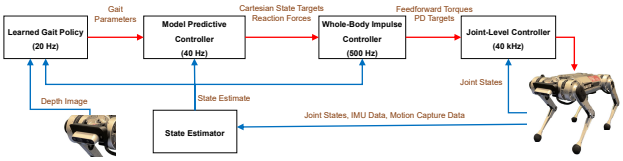

Figure 2: Our hierarchical architecture for visual locomotion.

2. Experimental Setup

2.1. Materials

Hardware: We use the MIT Mini-Cheetah as our experimental platform [9]. This 9kg electrically-actuated robot quadruped stands 28cm tall with a body length of 38cm. A front-mounted Intel RealSense D435 camera provides real-time stereo depth data. The robot is also equipped with an onboard computer [10] that supports a hierarchical trajectory-tracking controller described in Section 3. Data

from the depth camera is processed by an offboard computer that communicates the output of the high-level policy to the onboard tracking controller via an Ethernet cable.

Simulator: We trained our vision-conditioned policy using the PyBullet [3] simulator. In addition to simulating the robot dynamics, PyBullet simulates the frames of our mounted depth camera, calibrated from an accurate CAD model of our robot and from the sensor's known intrinsic parameters.

2.2. Environment

Gap Crossing Task: In order to evaluate the ability of our system to dynamically traverse discontinuous terrains, we define a test environment consisting of variable-width gaps and flat regions. Prior work has shown that narrow gaps can be crossed using relatively conservative gaits [18]. However, the analysis below supports that more dynamic gaits are necessary to succeed on terrains with more challenging gaps. The difficulty of traversing gap worlds depends on the proximity of gaps as well as gap width, with closer and wider gaps presenting a greater challenge to the controller.

Theoretical Limit on Fixed-Gait Gap Crossing: A quadruped stepping at a fixed cycle frequency f and moving at velocity v will locomote a distance of $\frac{v}{f}$ each gait cycle, and so the nominal foot placement for any given foot under the Raibert heuristic will advance by a distance of $\frac{v}{f}$. In the pronking gait, wherein all legs contact the ground simultaneously, this places the upper limit on gap crossing at $\frac{v}{f}$. For a trotting gait, wherein pairs of diagonal legs meet the ground in alternating timing, this limit is reduced by half to $\frac{v}{2f}$. The Mini Cheetah nominally trots at a frequency of $f = 3$Hz, theoretically limiting its maximal gap crossing at a velocity of $v = 1$m/s to 33cm in the pronking case, or 17cm in the trotting case. In practice, the popular ANYmal C by ANYbiotics, over twice as long and 5 times as massive as the Mini Cheetah, is rated by its manufacturers to cross gaps of up to 25cm [1].

3. Method

3.1. Trajectory Representation

We define a quadruped's *cartesian trajectory* $\mathcal{T}_{t:t+H}$ beginning at time t and extending over horizon H as

$$\mathcal{T}_{t:t+H} = \begin{bmatrix} \mathcal{T}_t \\ \mathcal{T}_{t+1} \\ \dots \\ \mathcal{T}_{t+H} \end{bmatrix}$$

$$\mathcal{T}_t = [\mathbf{p}_\text{b}, \dot{\mathbf{p}}_\text{b}, \ddot{\mathbf{p}}_\text{b}, \mathbf{p}_\text{f}, \dot{\mathbf{p}}_\text{f}, \ddot{\mathbf{p}}_\text{f}, \mathbf{C}]_t \in \mathbb{R}^{54} \times [0,1]^4$$

where $\mathbf{p}_\text{b} \in \mathbb{R}^6$ is the six-dimensional robot body pose, $\mathbf{p}_\text{f} \in \mathbb{R}^{12}$ encodes the position of each foot, and $\mathbf{C} = [1_C^\text{LF}, 1_C^\text{RF}, 1_C^\text{LR}, 1_C^\text{RR}] \in [0,1]^4$ is the binary contact state of each foot, with 1_C^f taking a value of 1 if foot f is in contact with the ground and a value of zero otherwise.

We choose to constrain our space of trajectories to those satisfying the *Raibert heuristic* for foot placements: $\mathbf{p}_\text{f} = \mathbf{p}^\text{raibert} = \frac{\Delta t_d^{l_i}}{2} v + k(v - v^\text{cmd}) + \mathbf{p}_\text{hip}$, where $\Delta t_d^{l_i}$ is duration of time foot i will remain in contact, v is the estimated robot body velocity, v^cmd is the commanded robot velocity, and k is a tunable gain term. Assuming the robot achieves the planned body trajectory, the Raibert Heuristic selects the foot placements such that each leg's lever angle of incidence on the ground is equal to its angle of departure. The foot trajectory between a foot's departure and landing location is represented as a three-point Bezier curve with a fixed maximum height. Under this set of assumptions, each assignment of $[\mathbf{p}_\text{b}, \mathbf{C}]_{t:t+H}$ completely specifies a unique sequence of foot positions, $\mathbf{p}_\text{f} = \mathbf{p}^\text{raibert}$. Therefore, the space of horizon-H trajectories satisfying the Raibert Heuristic is parameterized by the sequence of assignments to the body state and contact state, $[\mathbf{p}_\text{b}, \mathbf{C}]_{t:t+H} \in \mathbb{R}^{6\times H} \times [0,1]^{4\times H}$ (with $\dot{\mathbf{p}}_\text{b}, \ddot{\mathbf{p}}_\text{b}$ selected for dynamic consistency).

3.2. Trajectory-Tracking Controller

We use the hybrid control scheme developed in [11] to track planned trajectories. It is composed of a model predictive controller (MPC) and a whole-body impulse controller (WBIC) that perform online optimization using only the low-dimensional robot joint state, observed by joint encoders, and robot body state, observed by an inertial measurement unit (IMU) or a motion capture system. Given the fully defined robot trajectory, the MPC uses a simplified centroidal robot model and computes the optimal target ground reaction forces at the feet at 40Hz. From this result, the WBIC computes feedforward joint torques as well as target position and velocity commands at 500Hz given the reactive force profile. A PD controller at the lowest level runs at 40kHz to track target joint positions and velocities computed by the WBIC. This scheme is illustrated in Fig. 2 and has been shown to robustly actuate a variety of blind dynamic gaits. For more details, we refer readers to [11].

3.3. Model-Free Trajectory Selection

We apply model-free reinforcement learning to select a suitable h-step extension of the cartesian trajectory: $\mathcal{T}_{t+H:t+H+h}$, given the robot state $\mathbf{s}_t$ and terrain observation $\mathbf{o}_t$. In this section, we provide details of the network architecture, training environment, and optimization procedure for our reinforcement learning approach.

3.3.1 Network Architecture

The policy $\pi_\theta(\mathbf{a}_t|\mathbf{s}_t, \mathbf{o}_t, \mathbf{a}_{t-1})$ is modeled using a deep recurrent neural network that includes a convolutional neural network (CNN) to process the high-dimensional terrain observation $\mathbf{o}_t$. The output features of the perception module are concatenated with proprioceptive inputs $\mathbf{s}_t$ and previous

action $\mathbf{a}_{t-1}$ and passed through a sequence of fully connected layers to output a probability distribution over the next action $\mathbf{a}_t$. The architecture details of the neural network are provided in Figure 3.

3.3.2 Rollout Procedure

For each training episode, the robot is initialized in a standing pose on flat ground. The locations of gaps and their widths are randomized. The initial cartesian trajectory $\mathcal{T}_{0:H}$ is initialized to a stationary standing state with all four feet in contact and the body 0.30m above the origin. Then, an on-policy rollout proceeds as in Algorithm 1.

Algorithm 1 Policy Rollout Procedure

$t \leftarrow 0$; $\mathbf{a}_{-1} \leftarrow \mathbf{0}$
observe $\mathbf{s}_0$, $\mathbf{o}_0$
while not IS-TERMINAL($\mathbf{s}_t$) **do**
  sample $\mathbf{a}_t \sim \pi_\theta(\mathbf{a}_t|\mathbf{s}_t, \mathbf{o}_t, \mathbf{a}_{t-1})$
  $\mathcal{T}_{t+H:t+H+h} \leftarrow \mathcal{T}(\mathbf{a}_t)$
  $t_{\text{end}} \leftarrow t + h$
  while $t < t_{\text{end}}$ **do**
    TRACK-TRAJECTORY($\mathbf{s}_t, \mathcal{T}_{t:t+H}$)
    $t = t + 1$
    observe $\mathbf{s}_t$, $\mathbf{o}_t$
  end while
end while

3.3.3 Reward Function

The reward r_t at time t is defined as: $r_t = c_1(p^b_{t,x} - p^b_{t-1,x}) - c_2 \max(0, ||v^b_t||_2 - V_{thresh}) - c_3|\alpha^b_t| - c_4|\beta^b_t| - c_5|\gamma^b_t| - c_6|\dot{q}|$ where $p^b_{t,x}$ is the projection of the body frame position onto the x-axis in the world frame, v^b_t is the body velocity, V_{thresh} is the maximum body velocity magnitude used as a soft safety constraint. $\alpha^b_t, \beta^b_t, \gamma^b_t$ are the yaw, pitch, roll angles of the body frame. We found that penalizing the joint velocity $\dot{q}$ was critical to encourage the policy to explore lower-frequency gait patterns without explicitly constraining the gait space.

3.3.4 Training Procedure

The parameters of the neural network (θ) are optimized using Proximal Policy Optimization (PPO) [16]. We use Adam optimizer with a learning rate 0.0003 and a batch size of 256. During training, the maximum episode length is 500, and 32 environments are simulated in parallel. An episode terminates if any of three terminal conditions are met: (1) the body height is less than 20 centimeters; (2) body roll or pitch exceeds 0.7 radians; or (3) a foot is placed in a gap. We find that our policies converge within 6000 training episodes, equivalent to to 60 hours of simulated locomotion.

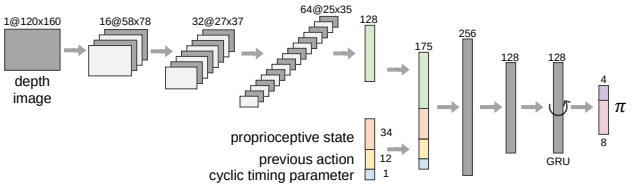

Figure 3: High-level gait prediction network

Table 1: Comparison between the success rate of visual and blind controller at crossing gaps; 20 trials in simulation.

Gap Width	*4cm*	*16cm*	*26cm*	*50cm*
Blind Trot	95%	0%	0%	0%
Blind Pronk	100%	30%	0%	0%
Visual Trot	100%	100%	0%	0%
Visual Pronk	100%	95%	95%	0%
Visual Gait-Free	90%	80%	70%	50%

4. Experiments

4.1. Evaluation with Fixed Contact Schedule

We train our framework to cross gaps using a space of cartesian trajectories constrained to trotting and pronking gaits. When training with constrained gait, the policy selects $\mathbf{a}_t \in \mathbb{R}^4$ and computes the cartesian trajectory extension via the mapping

$$\mathcal{T}_{t+H:t+H+h}(\mathbf{a}_t) = [(\mathbf{p}_b, \dot{\mathbf{p}}_b, \ddot{\mathbf{p}}_b)(\mathbf{a}_t), (\mathbf{p}_f, \dot{\mathbf{p}}_f, \ddot{\mathbf{p}}_f)^{\text{raibert}}, \mathbf{C}_{fixed}]$$

where the key quantity adapted by the policy is the body velocity $\dot{\mathbf{p}}_b(\mathbf{a}_t) = [\dot{x} = \mathbf{a}_t[0], \dot{y} = \mathbf{a}_t[1], \dot{z} = \mathbf{a}_t[2], \dot{\alpha} = 0, \dot{\beta} = 0, \dot{\gamma} = \mathbf{a}_t[3]]$, from which $\mathbf{p}_b(\mathbf{a}_t)$ and $\ddot{\mathbf{p}}_b(\mathbf{a}_t)$ are fixed for dynamic consistency. $\mathbf{C}_{fixed}$ corresponds to the contact schedule of the fixed gait.

In our experiments with fixed gaits, we fix the gait frequency at 3Hz and cap the body velocity command at 1m/s. $\mathbf{p}_f^{\text{raibert}}, \dot{\mathbf{p}}_f^{\text{raibert}}, \ddot{\mathbf{p}}_f^{\text{raibert}}$ are foot targets satisfying the Raibert Heuristic as described in Section 3.1.

Table 1 reports the performance of our method for adaptive fixed-gait gap crossing in simulation. While the baseline fixed gaits without vision are capable of crossing gaps by chance, our visually-guided approach succeeds at over 90% of theoretically feasible gap crossing tasks, as described in Section 2.2.

4.2. Evaluation with Adaptive Contact Schedule

We relax all constraints on contact schedule and train a controller with a *vision-adaptive contact schedule* to cross wide gaps. When training with adaptive gait, the policy selects $\mathbf{a}_t \in \mathbb{R}^4 \times [0,1]^{4\times h}$ and computes the cartesian trajectory extension via the mapping

$$\mathcal{T}_{t+H:t+H+h}(\mathbf{a}_t) = [(\mathbf{p}_b, \dot{\mathbf{p}}_b, \ddot{\mathbf{p}}_b)(\mathbf{a}_t), (\mathbf{p}_f, \dot{\mathbf{p}}_f, \ddot{\mathbf{p}}_f)^{\text{raibert}}, \mathbf{C}(\mathbf{a}_t)]$$

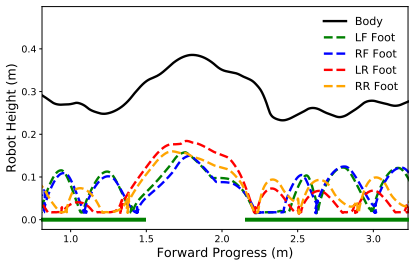

Figure 4: A sample trajectory generated by our controller during a successful crossing of a single 60-centimeter gap in simulation. The bold green line represents the terrain.

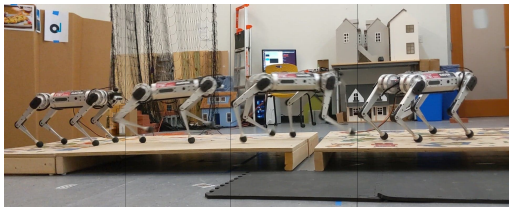

Figure 5: Successful policy deployment on the real robot.

where $\dot{\mathbf{p}}_b(\mathbf{a}_t) = [\dot{x} = \mathbf{a}_t[0], \dot{y} = \mathbf{a}_t[1], \dot{z} = \mathbf{a}_t[2], \dot{\alpha} = 0, \dot{\beta} = 0, \dot{\gamma} = \mathbf{a}_t[3]]$, from which $\mathbf{p}_b(\mathbf{a}_t)$ and $\ddot{\mathbf{p}}_b(\mathbf{a}_t)$ are fixed for dynamic consistency. $\mathbf{C}(\mathbf{a}_t)$ is the contact state selected by the policy through a discrete output head.

Figure 4 illustrates a sample trajectory generated during locomotion across a wide gap. The high-level policy demonstrates that it has learned to schedule a flight phase of appropriate duration when a wide gap is perceived. This result demonstrates that the visual locomotion behaviors produced by our method go beyond simple foothold adaptation.

4.3. Sim-to-Real Transfer

We demonstrate the successful deployment of our trot-constrained controller on the Mini-Cheetah robot. Our test environment consists of five wooden platforms from which we construct gapped terrains similar to those in simulation. Visual information is processed directly from an onboard forward-facing depth camera in real-time (20Hz), while a motion capture system provides precise robot body state readings at 100Hz. The robustness of the low-level controller enables the transfer of learned locomotion policies to the real robot without any randomization or detailed modeling of dynamics at training time. An example of successful deployment is illustrated in Figure 5.

4.4. Impact of Hierarchical Architecture

We implemented as a baseline a fully model-free learning approach to quadruped locomotion, Policies Modulating Trajectory Generators (PMTG), which has been shown to produce behaviors feasible for sim-to-real transfer in previous work [7, 8]. Although this method is capable of learning visual gap-crossing behaviors, we note several drawbacks of model-free approaches relative to our method. (1) *Necessity of ad-hoc reward shaping*: Our approach uses a simple reward which is a linear combination of standard state variables. (2) *Necessity of curriculum to drive exploration* [8]: Our method explores efficiently in the space of trajectories and requires no curriculum to cross large gaps. (3) *Dependence on simulator accuracy*: Pure learning-based methods must use dynamics randomization, careful simulator design, or reward design to avoid "cheating" behaviors.

5. Related Work

Previous works on motion planning tackle legged locomotion over complex terrain using hierarchical motion planning [2, 17], planning with visual segmentation of terrain [6, 15], planning with heuristic foot placement adaptation and learning-based visual recognition [14], and planning over a learned visual representation [12]. These methods rely on an explicit, conservative model for robot dynamic stability. DeepGait [18] learns a gait planner that predicts the target gait and a gait controller that tracks the target gait with reinforcement learning. This method's dependence on the CROC model limits the high-level controller from selecting more agile and dynamic gaits. [4] learns a non-fixed contact schedule, as we do, and produces interesting behaviors, but does not incorporate visual input. [19, 5] learn terrain-aware policies for model-based controllers, but focus on less-dynamic tasks than our work. Recent works on joint-space reinforcement learning [8, 20] have also shown quadruped jumping skill using simulated LiDAR sensor in simulation with a carefully designed curriculum. In contrast, our method does not require any curriculum and can still achieve large gap jumping.

6. Discussion

We identify several limitations remaining in our approach: (1) **State estimation**: Trajectory tracking performance quality varies with state estimate quality, and high-precision foot placement is necessary in the gap-crossing domain; (2) **Computational expense**: Running our trajectory-tracking controller during training limits simulation speed.

In future work, we intend to perform detailed evaluation of the robustness of our sim-to-real approach for a broader range of dynamic behaviors; this will require additional safety considerations to be made in our lab environment. We also plan to apply our method to additional locomotion tasks in which vision holds utility, such as stairs, stepping stones, and terrains with varied contact properties.

References

[1] Anymal website. https://www.anybotics.com/anymal-legged-robot/. Accessed: 2021-2-26. 2

[2] Dominik Belter, Przemysław Łabęcki, and Piotr Skrzypczyński. Adaptive motion planning for autonomous rough terrain traversal with a walking robot. *Journal of Field Robotics*, 33(3):337–370, 2016. 4

[3] Erwin Coumans and Yunfei Bai. Pybullet, a python module for physics simulation in robotics, games and machine learning, 2017. 2

[4] Xingye Da, Zhaoming Xie, David Hoeller, Byron Boots, Animashree Anandkumar, Yuke Zhu, Buck Babich, and Animesh Garg. Learning a contact-adaptive controller for robust, efficient legged locomotion. *arXiv preprint arXiv:2009.10019*, 2020. 4

[5] Siddhant Gangapurwala, Mathieu Geisert, Romeo Orsolino, Maurice Fallon, and Ioannis Havoutis. Rloc: Terrain-aware legged locomotion using reinforcement learning and optimal control. *arXiv preprint arXiv:2012.03094*, 2020. 4

[6] Robert J Griffin, Georg Wiedebach, Stephen McCrory, Sylvain Bertrand, Inho Lee, and Jerry Pratt. Footstep planning for autonomous walking over rough terrain. In *2019 IEEE-RAS 19th International Conference on Humanoid Robots (Humanoids)*, pages 9–16. IEEE, 2019. 4

[7] Atil Iscen, Ken Caluwaerts, Jie Tan, Tingnan Zhang, Erwin Coumans, Vikas Sindhwani, and Vincent Vanhoucke. Policies modulating trajectory generators. In *Conference on Robot Learning*, pages 916–926. PMLR, 2018. 4

[8] Atil Iscen, George Yu, Alejandro Escontrela, Deepali Jain, Jie Tan, and Ken Caluwaerts. Learning agile locomotion skills with a mentor. In *2021 International Conference on Robotics and Automation (ICRA)*, 2021. 4

[9] Benjamin Katz, Jared Di Carlo, and Sangbae Kim. Mini cheetah: A platform for pushing the limits of dynamic quadruped control. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 6295–6301. IEEE, 2019. 1

[10] Donghyun Kim, Daniel Carballo, Jared Di Carlo, Benjamine Katz, Gerardo Bledt, Bryan Lim, and Sangbae Kim. Vision aided dynamic exploration of unstructured terrain with a small-scale quadruped robot. In *IEEE International Conference on Robotics and Automation*, 2020. 1

[11] Donghyun Kim, Jared Di Carlo, Benjamin Katz, Gerardo Bledt, and Sangbae Kim. Highly dynamic quadruped locomotion via whole-body impulse control and model predictive control. *ArXiv*, abs/1909.06586, 2019. 2

[12] Tobias Klamt and Sven Behnke. Towards learning abstract representations for locomotion planning in high-dimensional state spaces. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 922–928. IEEE, 2019. 4

[13] Joonho Lee, Jemin Hwangbo, Lorenz Wellhausen, Vladlen Koltun, and Marco Hutter. Learning quadrupedal locomotion over challenging terrain. *Science Robotics*, 5(47), 2020. 1

[14] Octavio Antonio Villarreal Magana, Victor Barasuol, Marco Camurri, Luca Franceschi, Michele Focchi, Massimiliano Pontil, Darwin G Caldwell, and Claudio Semini. Fast and continuous foothold adaptation for dynamic locomotion through cnns. *IEEE Robotics and Automation Letters*, 4(2):2140–2147, 2019. 4

[15] H. Park, P. Wensing, and Sangbae Kim. Online planning for autonomous running jumps over obstacles in high-speed quadrupeds. In *Robotics: Science and Systems*, 2015. 4

[16] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017. 3

[17] Steve Tonneau, Nicolas Mansard, Chonhyon Park, Dinesh Manocha, Franck Multon, and Julien Pettré. A reachability-based planner for sequences of acyclic contacts in cluttered environments. In *Robotics Research*, pages 287–303. Springer, 2018. 4

[18] Vassilios Tsounis, Mitja Alge, Joonho Lee, Farbod Farshidian, and Marco Hutter. Deepgait: Planning and control of quadrupedal gaits using deep reinforcement learning. *IEEE Robotics and Automation Letters*, 5(2):3699–3706, 2020. 2, 4

[19] Zhaoming Xie, Xingye Da, Buck Babich, Animesh Garg, and Michiel van de Panne. Glide: Generalizable quadrupedal locomotion in diverse environments with a centroidal model, 2021. 4

[20] Zhaoming Xie, Hung Yu Ling, Nam Hee Kim, and Michiel van de Panne. Allsteps: Curriculum-driven learning of stepping stone skills. *Computer Graphics Forum*, 39(8):213–224, 2020. 4