TERRAIN RL SIM

Glen Berseth[1] Xue Bin Peng[2] Michiel van de Panne[1]
[1]University of British Columbia, [2]UC Berkeley
{gberseth,van}@cs.ubc.ca

ABSTRACT

We provide 88 challenging simulation environments that range in difficulty. The difficulty in these *environments* is linked not only to the number of dimensions in the action space but also to the task complexity. Using more complex and accurate simulations will help push the field closer to creating human-level intelligence. Therefore, we are releasing a number of simulation *environments* that include local egocentric visual perception. These *environments* include randomly generated terrain which the agent needs to learn to interpret via visual features. The library also provides simple mechanisms to create new environments with different agent morphologies and the option to modify the distribution of generated terrain.

1 INTRODUCTION

Research in Deep Reinforcement Learning (DRL) has grown significantly in recent years, and so to has the demand for simulated environments that can be used to evaluate DRL methods. These environments are meant to be used as a means to fairly compare the progress of DRL methods by ensuring the simulation and reward function are the same across papers. Although many *environments* have been created, little is truly known about the difficulty of the *environments*. Many control problems appear challenging due to a large number of dimensions in the control space. For example, getting a simulated biped to walk and be robust to perturbations can be challenging, however, simple control structures were created to facilitate this control years ago (Yin et al., 2007; Kajita et al., 2003; Yamaguchi et al., 1999; Kajita et al., 2001). The *environments* included in openAIGym have similar and simpler control problems that have recently been solved using methods much simpler than DRL. These methods include using Radial Basis Function (RBF) (Rajeswaran et al., 2017)[1] and random search in the network parameter space (Salimans et al., 2017; Mania et al., 2018). These papers note that the improvements in DRL methods in the recent years could be focusing on the challenges related to optimization, not exploration and discovery of good actions. Although, this might be possible the authors view the prospects of finding solutions to these problems using less complex methods a sign that the environments used are too simple.

In DRL we not only want to push the boundaries of how fast we can solve environments but to also make strides in solving challenging tasks never before seen. This is increasingly important because we want to be using DRL on problems DRL will perform the best or in some cases can only be solved with DRL. What makes a problem challenging is not only related to the control capabilities but also the affordances available in the environment (Gibson, 1979). Therefore, we need to shape the affordances available to the agent as well to affect the difficulty of a task. We proved a number of environments that can be used to challenge DRL methods.

2 RELATED WORK

There are a number of similar libraries for evaluating reinforcement learning methods. The Arcade Learning Environment is one of the first sets of *environments* that was used to show the effectiveness of DRL on tasks with high dimensional input spaces (Bellemare et al., 2012). The OpenAIGym contains a collection of discrete action as well as continuous action tasks (Brockman et al., 2016). OpenAI Roboschool is a version of OpenAIGym where a number of the *environments* have been

[1]Simple Nearest Neighbor Policy Method for Continuous Control Tasks

recreated using Bullet instead of Mujoco [2].DeepMind recently released a new character motion control library (DeepMind Control Suite) that includes control problems similar to openAIGym with additional *environments* for mocap imitation (Tassa et al., 2018). The OpenAI Universe is a different, large set of *environments* created with the goal it being used to create a general agent that can play a large number of games competitively [3]. The DeepMind-lab is another set of *environments* that focuses on using visual inputs as observations, the visual input provides the agent with partial information of the environment state (Beattie et al., 2016). Expanding upon the partially observable *environments* is ELF that includes a novel RTS game. (Tian et al., 2017) (Fast and introduces novel RTS game)

We provide a set of *environments* that include tasks similar to openAIGym and the DeepMind Control Suite and new more challenging control problems. The simulation *environments* use Bullet (Bullet, 2015) an open source free simulator where many continuous control libraries use Mujoco (Todorov et al., 2012) a non-free, closed source piece of software. Many *environments* include terrain features in the observation. In the *environments* with terrain state features the agent navigates over is randomly generated. As a result not only does the agent need to learn to locomote but it also needs to learn how to perceive its environment and avoid obstacles. Some *environments* have been so challenging they could only be solved with Hierarchical Reinforcement Learning (HRL) techniques. We provide these in hopes more will continue work in the area of HRL. Additional extra difficult*environments* are included that have never been solved. Many of these difficult tasks were created while working on other projects but we were not able to produce controllers to solve these problems, or the controllers were not of sufficient quality. Last, there are different actuation models to choose from. Most libraries only offer torques as a means to actuate and control the agent's movement. control via torques, desired velocities, desired position and muscle-based control are included.

3 TerrainRLSim

The API closely follows the openAIGym style. This reduces the assumptions put on the agent structure. We include a mechanism to set the random seed for the simulation, allowing for reproducible simulations. Many of the *environments* include features for the local terrain around the agent. The observation produced by the simulation always puts these terrain features first, for example, $(< terrain - features > || < agent - features >)$, all as a single vector. The observation can be sliced into multiple parts, allowing only the terrain features to be passed through convolution layers. The software uses the Bullet Physics library (Bullet, 2015) an open source physics simulator. The simulation performance depends primarily on the efficiency of Bullet which is highly optimized. Overall, the simulation is fast and supports different kinds of action spaces (torque, velocity, pd and muscle-based).

4 Environments

Here we describe the types of simulation *environments* included in *TerrainRLSim*. In total there are almost 100 environments.

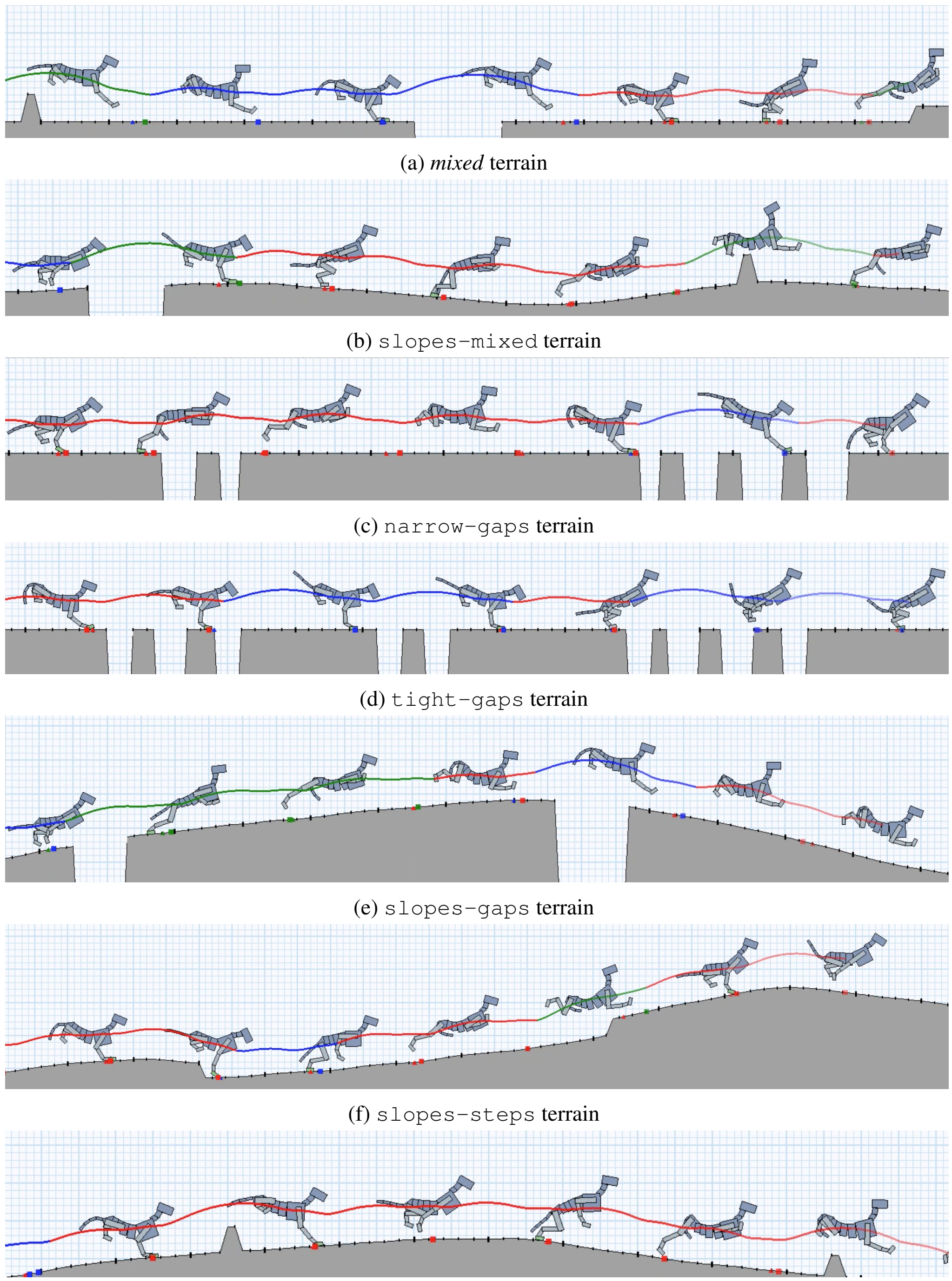
4.1 TerrainRL

The *TerrainRL environments* are based on the work in (Peng et al., 2016). In this work physics-based character with Finite State Machine (FSM) controllers are parameterized and trained to traverse complex dynamically generated terrains. Examples of different terrain types and characters are shown in Figure 1 and Figure 2.

On top of what was created for the Terrain Adaptive Locomotion (terrainRL) project we include additional character and terrain types. These include a Simple Biped Controller (SIMBICON) based biped controller and a hopper controller. The new terrain types include *cliffs* and other more challenging versions of the ones used in the paper (Peng et al., 2015).

[2]https://github.com/openai/roboschool

[3]https://blog.openai.com/universe/



(a) *mixed* terrain

(b) `slopes-mixed` terrain

(c) `narrow-gaps` terrain

(d) `tight-gaps` terrain

(e) `slopes-gaps` terrain

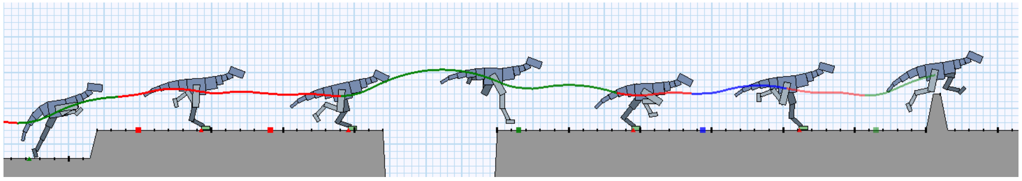
(f) `slopes-steps` terrain

(g) `slopes-walls` terrain

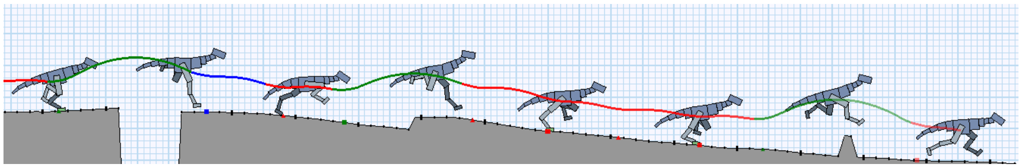
Figure 1: Dog on different terrain types.

4.2 Imitation Learning

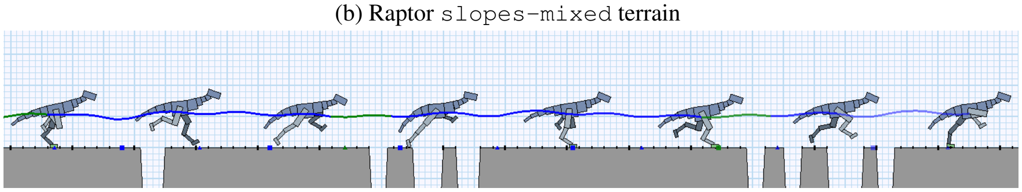
The goal in these *environments* is to train an agent to imitate particular behaviours described by a motion capture clip, and is based on the work in (Peng & van de Panne, 2017). The provided clip includes sequential character poses that are used in the reward function to instruct the character to best match the motion capture pose. For these environments there are three types of characters that are used, a *biped*, *dog* and *dog* (Figure 3). For each of these characters there are 4 different action



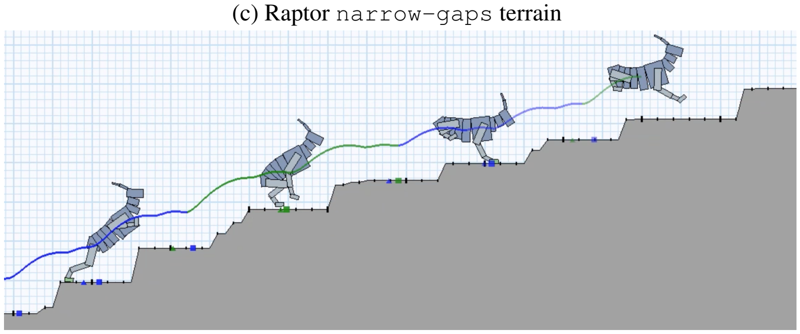
(a) Raptor *mixed* terrain



(b) Raptor `slopes-mixed` terrain



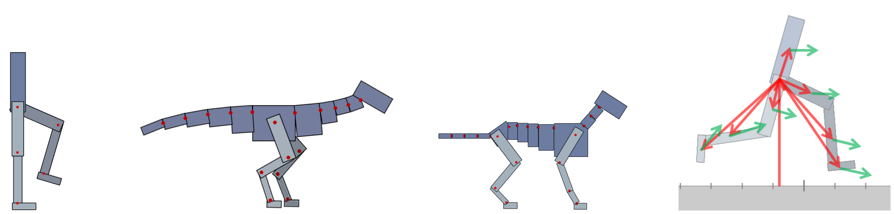
(c) Raptor `narrow-gaps` terrain



(d) Goat on `variable-steps` terrain

Figure 2: Other Control Policies

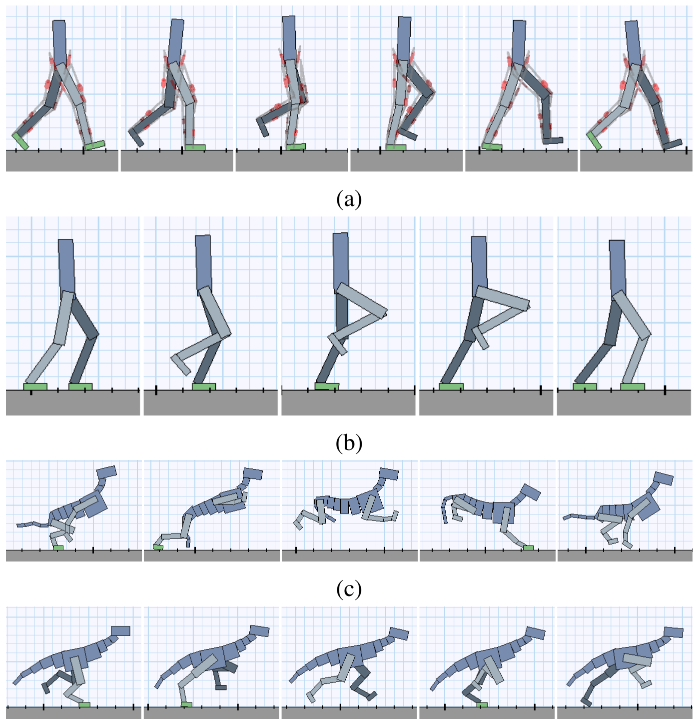
models available to actuate the joints: torques, desired velocity, desired position and muscle-based control. Example motions learned on these models are shown in Figure 4.



(a) (b) (c) (d)

Figure 3: Simulated articulated figures and their state representation. Revolute joints connect all links. From left to right: 7-link *biped*; 19-link *dog*; 21-link *dog*; State features: root height, relative position (red) of each link with respect to the root and their respective linear velocity (green).

We include additional *environments* for learning walking and running motions for 3D bipeds. There are also a number of terrain types, including *rough* and *steps*, that can be used to add randomly generated terrain into the simulation.



(a) (b) (c) (d)

Figure 4: Simulated Motions Using the PD Action Representation. The top row uses an MTU action space while the remainder are driven by a PD action space.

4.3 DeepLoco

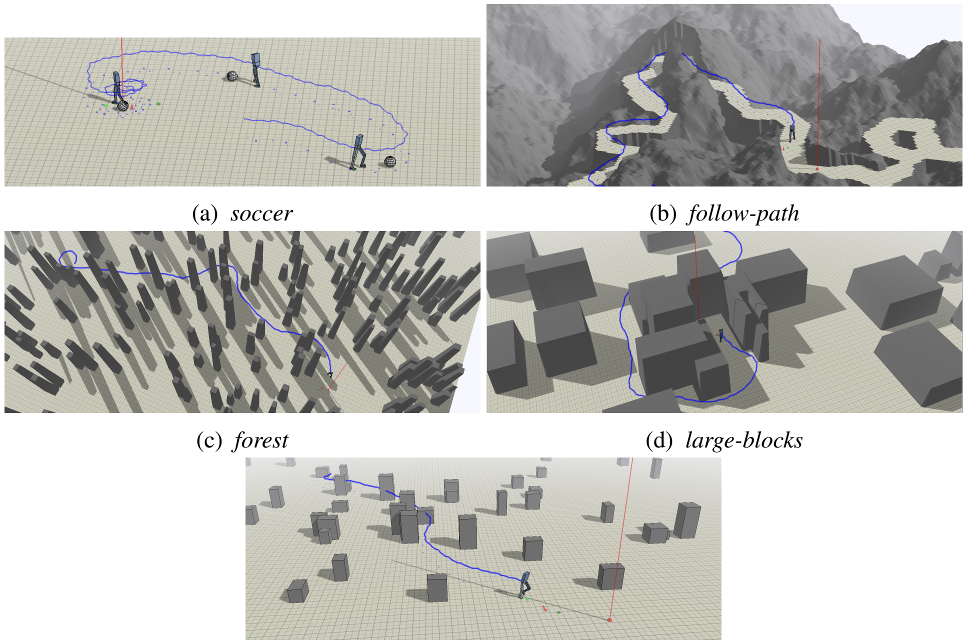
The *DeepLoco environments* are similar to the ones used in (Peng et al., 2017). They include a number of 3D simulations where the goal is to train a biped to walk in complex environments with randomly generated terrain Figure 5.

We include additional environments and configuration that were used for testing and evaluation in the process of completing this project. These include the more challenging environments and a version of the controller that does not use hierarchical control, such as a controller that includes the terrain input and operates at 30 fps. The code also include processed versions of mocap clips.

4.4 PLAiD

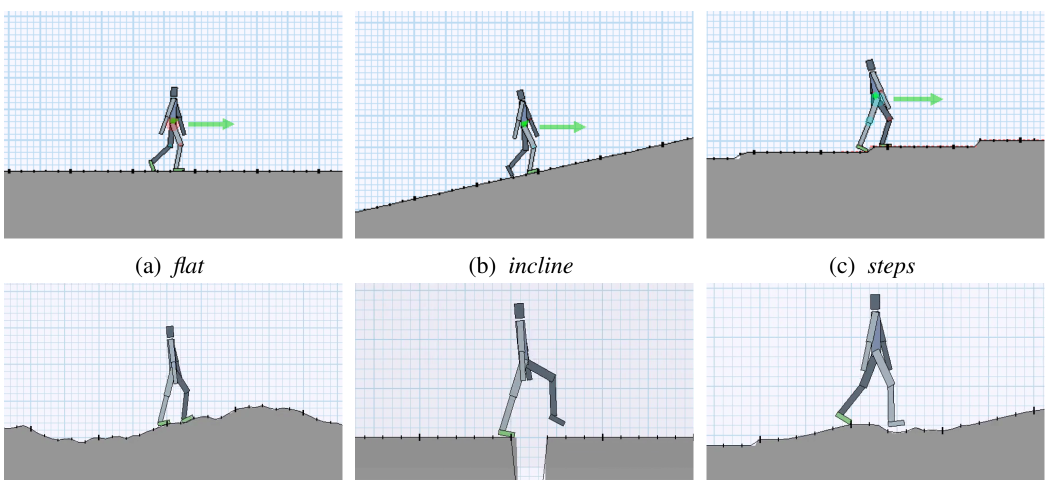
These environments are an extension of the *environments* in Section: 4.2. Here the agent has been modified to have arms and the terrain is randomly generated. With the addition of randomly generated terrain additional state features are added to provide visual perception of the terrain. Part of these environments were used in (Berseth et al., 2018). These are the only available environments that can be used for multi-task and continual learning in the continuous action space domain for Rienforcement Learning (RL). Examples of the *environments* are shown in Figure 6.

On top of the environments used in **PLAiD!** we add additional ones for both the *dog* and *dog* characters. These additional *environments* are for each of the terrain types in Figure 6 and two additional terrain types, *walls* and *slopes-mixed*. New *environments* were also created for a 3D biped with different 3D terrain types are created for this *biped*.



(a) *soccer* (b) *follow-path* (c) *forest* (d) *large-blocks* (e) *dynamic obstacles*

Figure 5: Snapshots of DeepLoco tasks. The red marker represents the target location and the blue line traces the trajectory of the character's centre of mass. **in order:** soccer dribbling, path following, pillar obstacles, block obstacles, dynamic obstacles.



(a) *flat* (b) *incline* (c) *steps* (d) *slopes* (e) *gaps* (f) *mixed*

Figure 6: The environments used to evaluate **PLAiD!**.

5 Discussion

Many of these *environments* have been used to create robust controllers that produce high quality motion. Even with the great progress in this area there is still much work to be done. These *environments* use more realistic joint torque limits that true biological version of the character might be capable of. Having realistic torque limits is a start but only captures a small portion of the complexities of generating torques. For biological creatures the maximal and minimal possible joint torques can be different depending on the direction of rotation, it can even depend on the joint pose. There are also inaccuracies in the joint dynamics and links. In many robotics applications you have to cope with the issue of *backlash* that involves the amount of free space between the engineered parts causing the system to move in unintended directions. There is also the complexity of flex in the system which is sometimes intended as springs are used to absorb forces. It is possible to model most of these phenomenon in a physics simulation already.

Apart from the physical phenomenon that we can and should modelled better in simulations used for RL there appear to be a number of simulation parameters simulation that could be given values more in-line with the real world. Examples of these include: linear dampening, gravity, angular dampening, static and kinetic friction values, proper masses and densities of objects, etc. It would increase the benefit to the community to evaluate RL methods on environments that have more purpose and can be used in games an on robots.

As we pursue RL research we should also be pushing the simulation accuracy and task difficulty to help us converge on solutions that will work in the real world. There is a great deal of work left to be done before we can create human level intelligence on both more accurate simulation models and better learning techniques.

REFERENCES

Charles Beattie, Joel Z. Leibo, Denis Teplyashin, Tom Ward, Marcus Wainwright, Heinrich Küttler, Andrew Lefrancq, Simon Green, Víctor Valdés, Amir Sadik, Julian Schrittwieser, Keith Anderson, Sarah York, Max Cant, Adam Cain, Adrian Bolton, Stephen Gaffney, Helen King, Demis Hassabis, Shane Legg, and Stig Petersen. Deepmind lab. *CoRR*, abs/1612.03801, 2016. URL `http://arxiv.org/abs/1612.03801`.

Marc G. Bellemare, Yavar Naddaf, Joel Veness, and Michael Bowling. The arcade learning environment: An evaluation platform for general agents. *CoRR*, abs/1207.4708, 2012. URL `http://arxiv.org/abs/1207.4708`.

G. Berseth, C. Xie, P. Cernek, and M. Van de Panne. Progressive Reinforcement Learning with Distillation for Multi-Skilled Motion Control. *ArXiv e-prints*, February 2018.

Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. Openai gym. *CoRR*, abs/1606.01540, 2016. URL `http://arxiv.org/abs/1606.01540`.

Bullet. Bullet physics library, 2015. http://bulletphysics.org.

James Jerome Gibson. The ecological approach to visual perception. 1979.

S. Kajita, F. Kanehiro, K. Kaneko, K. Yokoi, and H. Hirukawa. The 3d linear inverted pendulum mode: a simple modeling for a biped walking pattern generation. In *Proceedings 2001 IEEE/RSJ International Conference on Intelligent Robots and Systems. Expanding the Societal Role of Robotics in the the Next Millennium (Cat. No.01CH37180)*, volume 1, pp. 239–246 vol.1, 2001. doi: 10.1109/IROS.2001.973365.

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 *2003 IEEE International Conference on Robotics and Automation (Cat. No.03CH37422)*, volume 2, pp. 1620–1626 vol.2, Sept 2003. doi: 10.1109/ROBOT.2003.1241826.

H. Mania, A. Guy, and B. Recht. Simple random search provides a competitive approach to reinforcement learning. *ArXiv e-prints*, March 2018.

Xue Bin Peng and Michiel van de Panne. Learning locomotion skills using deeprl: Does the choice of action space matter? In *Proceedings of the ACM SIGGRAPH / Eurographics Symposium on Computer Animation*, SCA '17, pp. 12:1–12:13, New York, NY, USA, 2017. ACM. ISBN 978-1-4503-5091-4. doi: 10.1145/3099564.3099567. URL `http://doi.acm.org/10.1145/3099564.3099567`.

Xue Bin Peng, Glen Berseth, and Michiel van de Panne. Dynamic terrain traversal skills using reinforcement learning. *ACM Transactions on Graphics*, 34(4):Article 80, 2015.

Xue Bin Peng, Glen Berseth, and Michiel van de Panne. Terrain-adaptive locomotion skills using deep reinforcement learning. *ACM Transactions on Graphics*, 35(4):Article 81, 2016.

Xue Bin Peng, Glen Berseth, Kangkang Yin, and Michiel Van De Panne. Deeploco: Dynamic locomotion skills using hierarchical deep reinforcement learning. *ACM Trans. Graph.*, 36(4): 41:1–41:13, July 2017. ISSN 0730-0301. doi: 10.1145/3072959.3073602. URL `http://doi.acm.org/10.1145/3072959.3073602`.

Aravind Rajeswaran, Kendall Lowrey, Emanuel Todorov, and Sham Kakade. Towards generalization and simplicity in continuous control. *CoRR*, abs/1703.02660, 2017. URL `http://arxiv.org/abs/1703.02660`.

T. Salimans, J. Ho, X. Chen, S. Sidor, and I. Sutskever. Evolution Strategies as a Scalable Alternative to Reinforcement Learning. *ArXiv e-prints*, March 2017.

Y. Tassa, Y. Doron, A. Muldal, T. Erez, Y. Li, D. de Las Casas, D. Budden, A. Abdolmaleki, J. Merel, A. Lefrancq, T. Lillicrap, and M. Riedmiller. DeepMind Control Suite. *ArXiv e-prints*, January 2018.

Yuandong Tian, Qucheng Gong, Wenling Shang, Yuxin Wu, and C. Lawrence Zitnick. Elf: An extensive, lightweight and flexible research platform for real-time strategy games. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (eds.), *Advances in Neural Information Processing Systems 30*, pp. 2659–2669. Curran Associates, Inc., 2017. URL `http://papers.nips.cc/paper/6859-elf-an-extensive-lightweight-and-flexible-research-platform-for-real-time-str` `pdf`.

E. Todorov, T. Erez, and Y. Tassa. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 5026–5033, Oct 2012. doi: 10.1109/IROS.2012.6386109.

J. Yamaguchi, E. Soga, S. Inoue, and A. Takanishi. Development of a bipedal humanoid robot-control method of whole body cooperative dynamic biped walking. In *Proceedings 1999 IEEE International Conference on Robotics and Automation (Cat. No.99CH36288C)*, volume 1, pp. 368–374 vol.1, 1999. doi: 10.1109/ROBOT.1999.770006.

KangKang Yin, Kevin Loken, and Michiel van de Panne. Simbicon: Simple biped locomotion control. *ACM Transctions on Graphics*, 26(3):Article 105, 2007.