Modular mobile robot design selection with deep reinforcement learning

Julian Whitman, Matt Travers, and Howie Choset. (Carnegie Mellon University)

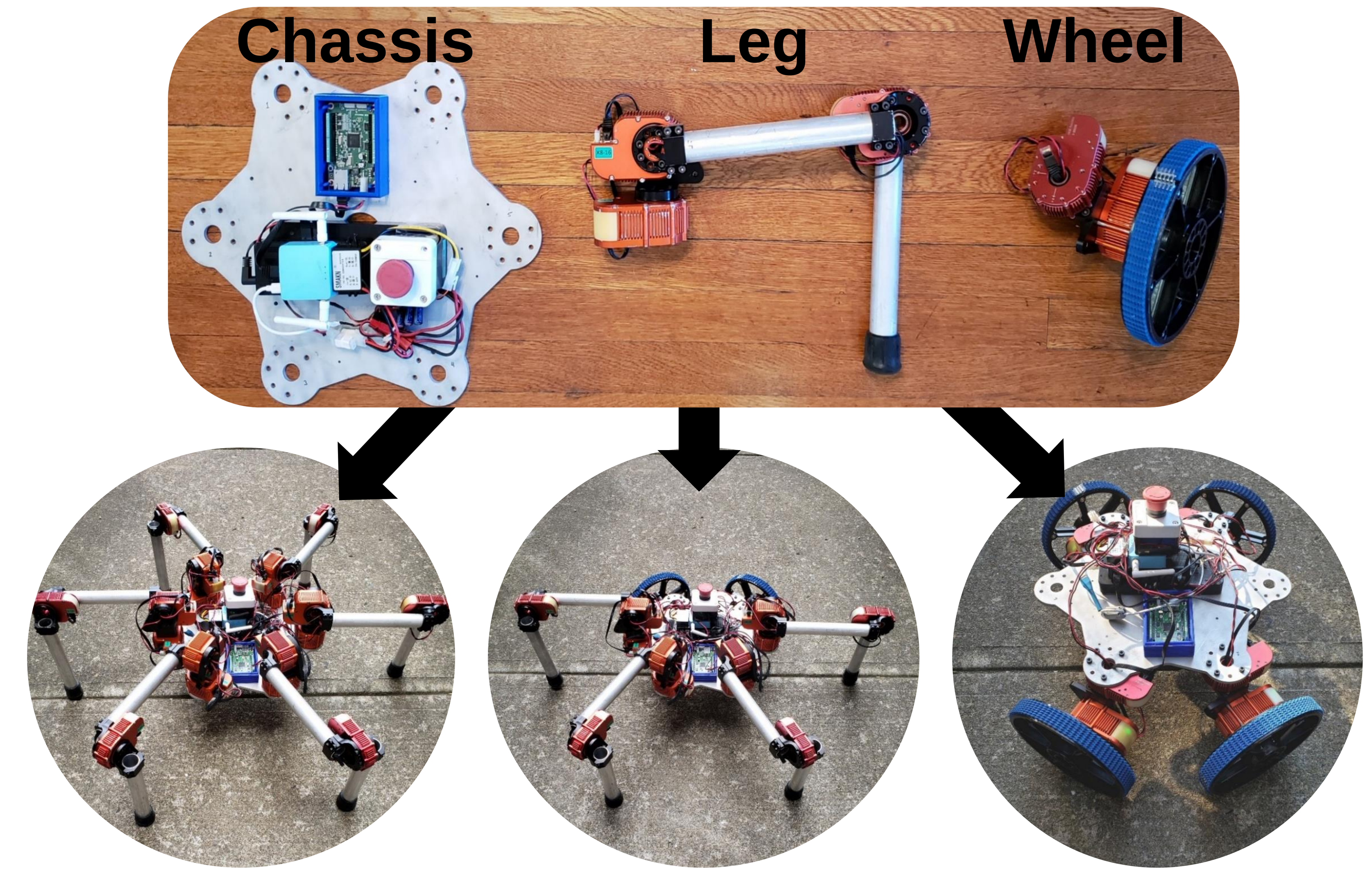
Machine Learning for Engineering Modeling, Simulation, and Design Workshop, NeurIPS 2020

Modular robots can be used to make many designs from a small set of components.

How to select the best design for a given job?

Even with a small number of modules, the number of combinations suffers from the curse of dimensionality.

How to conduct the design search quickly?



Our robots are comprised of legs and wheels, modules that can be easily interchanged, lending the robot different capabilities on different terrains.

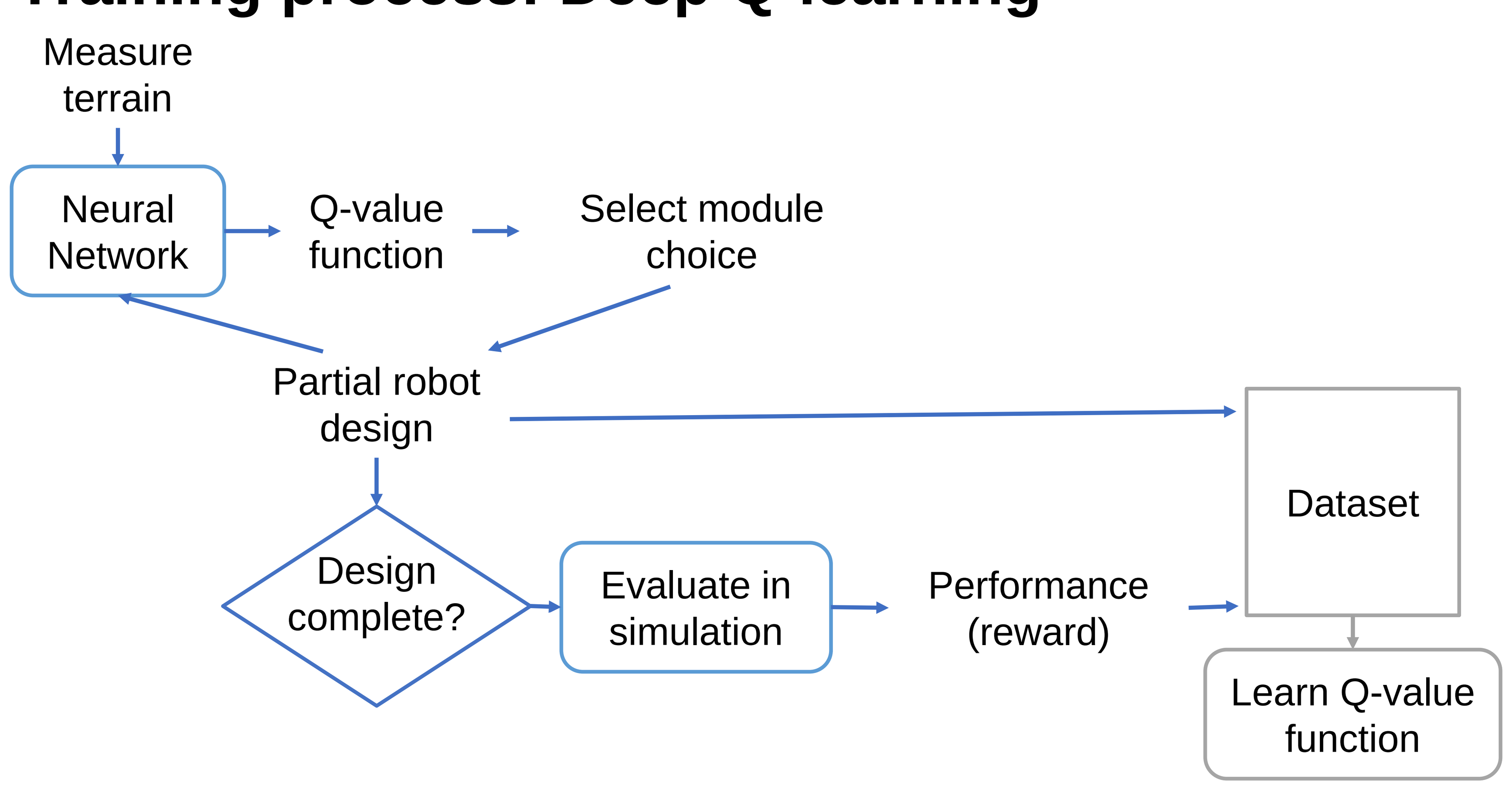
Abstract

The widespread adoption of robots will require a flexible and automated approach to robot design. Exploring the full space of all possible designs when creating a custom robot can prove to be computationally intractable, leading us to consider modular robots, composed of a common set of repeated components that can be reconfigured for each new task. But, conducting a combinatorial optimization process to create a specialized design for each new task and setting is computationally expensive, especially if the task changes frequently. In this work, our goal is to select mobile robot designs that will perform highest in each environment under a known control policy, with the assumption that the selection process must be conducted for new environments frequently. We use deep reinforcement learning to create a neural network that, given a terrain map as an input, outputs the mobile robot designs deemed most likely to locomote successfully in that environment.

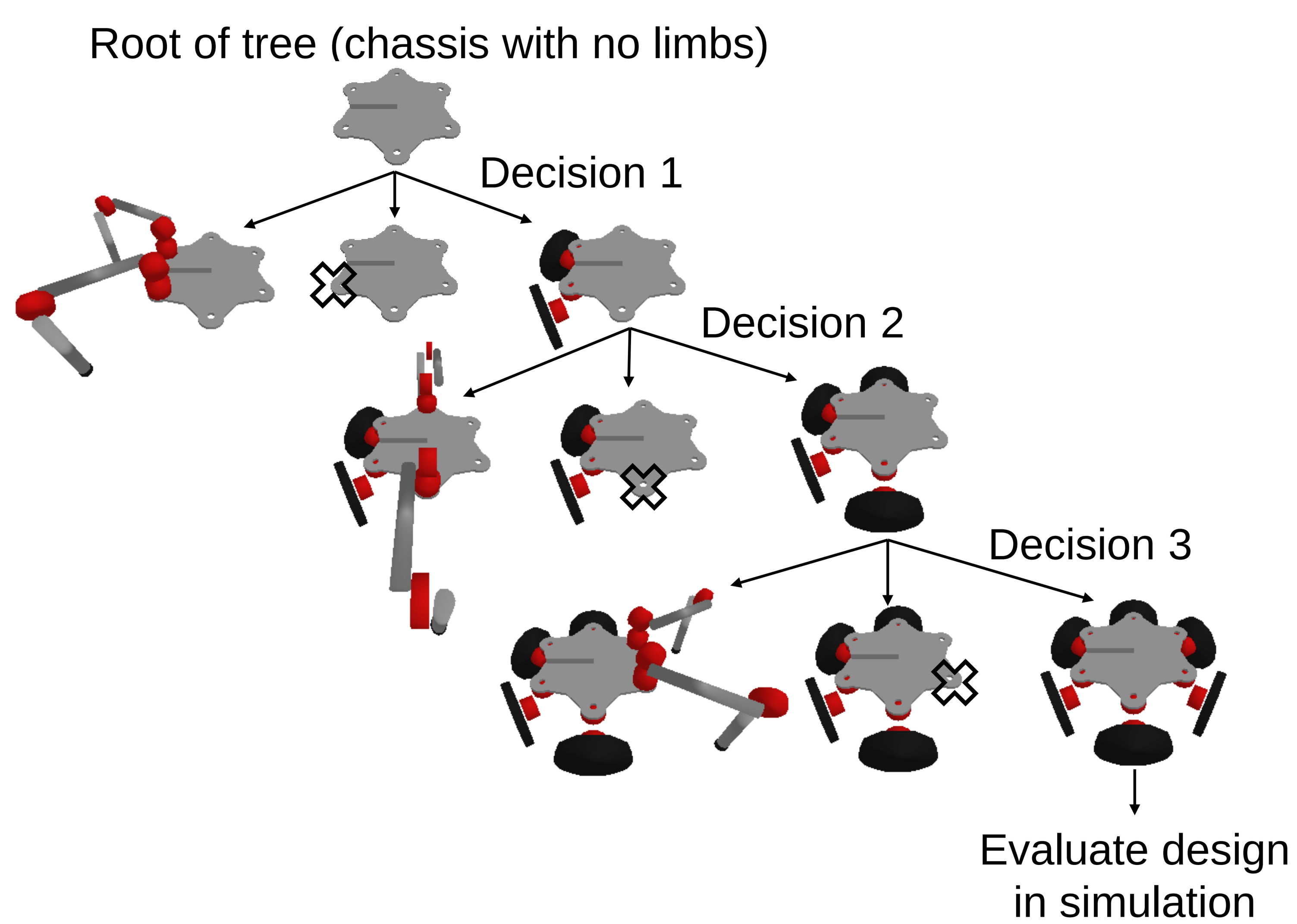
Methods

- Prior modular design synthesis methods [1, 2] introduced the notion of incrementally constructing and searching a tree of modular arrangements for manipulators.
- We extend this idea to mobile robots, where each node added represents adding a module to the robot.
- On this tree, each state (node) represents a partially complete design, and each action (edge) represents adding a module.
- We learn a state-action value function [4] approximating the benefit of adding each module type given the terrain, thereby learning the capabilities of each design in each environment.

Training process: Deep Q-learning

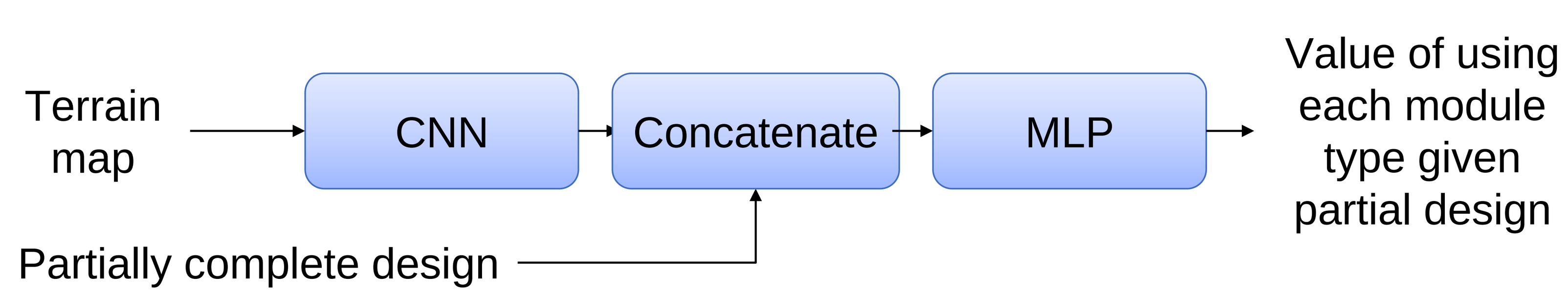


Robot design as a tree search



We view the design space as a tree, in which modular limbs are sequentially added. At the root lies a chassis with no modules, and each step adds a module.

Network architecture

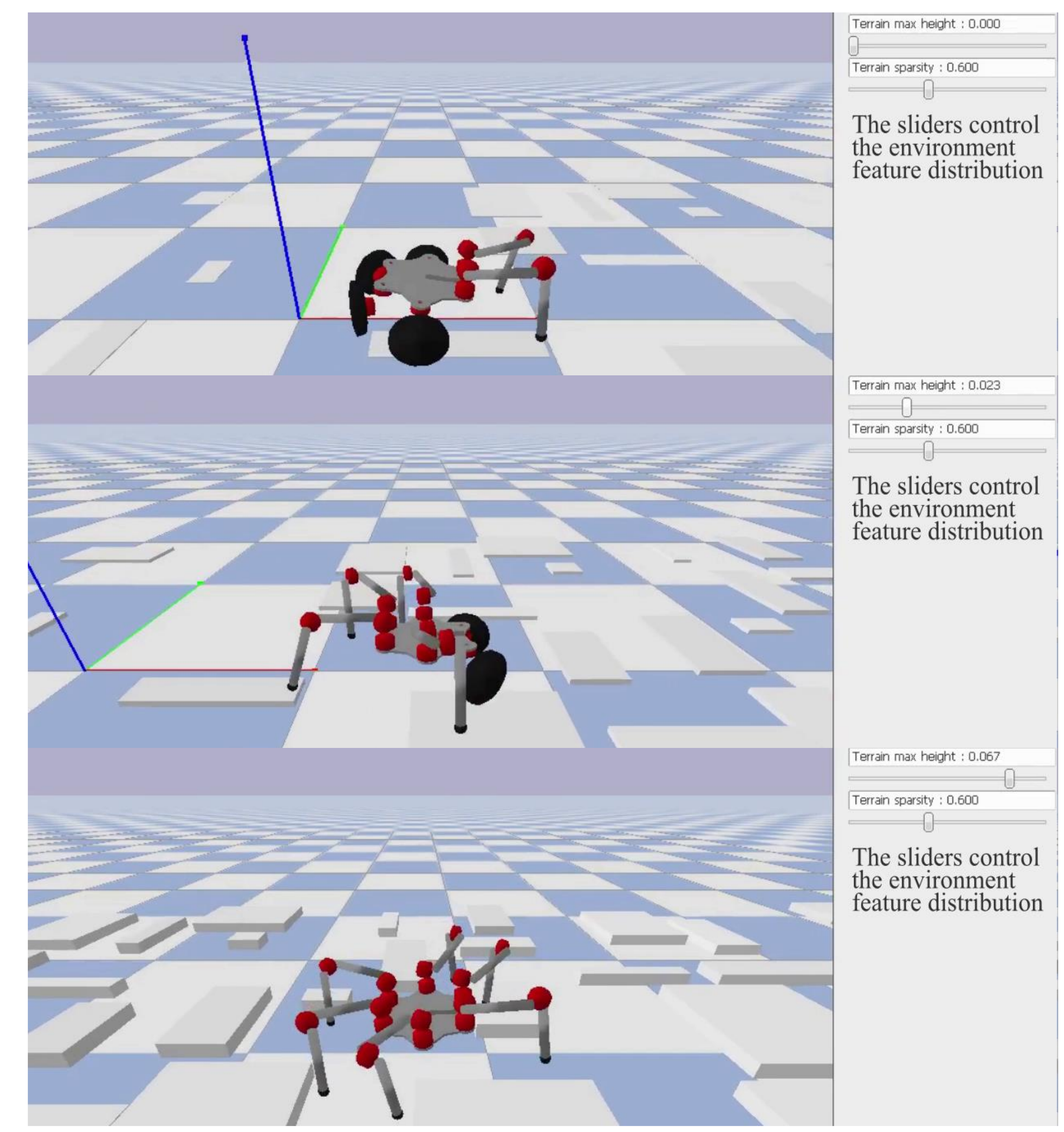


The design generator is conditioned on a terrain map. The output of the network is interpreted as the state-action value of each module type that could be added to the partially complete robot design.

Quantitative results: Prediction accuracy

We compared the output designs of the highest performance estimates from the network against the performance in simulation. The top designs overlap well between the estimated and simulated performance, indicating the network can predict the best-performing designs for a terrain.

Qualitative results: Interactive GUI



After training, the network can be in an interactive user interface, a slider changes the height of the terrain. The network select the best robot and simulates it in real-time. This allows us to quickly investigate how different terrain feature distributions effect the optimal design—without additional training or intensive computation

References

1. J. Whitman, R. Bhirangi, M. J. Travers, and H. Choset, “Modular robot design synthesis with deep reinforcement learning.,” in AAAI, 2020.
2. S. Ha, S. Coros, A. Alspach, J. M. Bern, J. Kim, and K. Yamane, “Computational design of robotic devices from high-level motion specifications,” IEEE Transactions on Robotics, 2018.
3. V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, et al., “Human-level control through deep reinforcement learning,” Nature, 2015.