A. Appendix

A.1. Algorithmic Description of the Improved Transfer Strategy in Section 3.2

Algorithm 1 Improved Transfer

```
1: Input: candidate agents denoted by their policy parameter vectors $\theta^1, \theta^2, \ldots, \theta^M$, target environment $E$ with a score function $Score(\cdot)$, and threshold (i.e. max of 5 most recent scores of the incumbent agent)
2: Initialize: Set list P_Candidates empty
3: for $m = 1$ to $M$ do
4:   Compute direct transfer $D$
5:   if $Score(D) >$ threshold then
6:     Compute fine-tuning transfer $P$
7:     if $Score(P) >$ threshold then
8:       add $P$ to P_Candidates
9:     end if
10:  end if
11: end for
12: Return: $\arg\max_{\theta \in \texttt{P\_Candidates}} Score(\theta)$
```

A.2. Training CPPNs with NEAT

CPPNs are trained with the NEAT algorithm (Stanley & Miikkulainen, 2002), a neuroevolution (Stanley et al., 2019) algorithm that learns both the architecture (i.e. the topology) and the weights of CPPNs. Specifically, in this work, the NEAT-Python library (McIntyre et al.) is used to initialize and evolve the CPPNs that encode environments. The setup and choices of hyperparameter are listed in Table 1. Because POET has its own diversity preservation mechanism, NEAT is run in POET without its conventional speciation mechanism. In this work, crossover between CPPNs is not performed.

A.3. Additional Information about the Domain and Experiment Setup

A.3.1. Additional Details about the Domain

The agent, illustrated in Figure 3, has four degrees of freedom (i.e. the dimensions of its action space) as the hips and knees of each leg are controlled by two motors. It has a total of 24 inputs: readings from 10 LIDAR rangefinders and 14 positional and movement variables from the agent's body parts (Klimov, 2016).

Reward is accumulated at each step when the agent attempts to move from the left end to the right end of an environment. If the agent falls at any step, the reward for that step is -100. As long as it does not fall, the step-wise reward is $130 \times \Delta x - 5 \times \Delta \texttt{Hull\_Angle} - 3.5e-4 \times \texttt{Applied\_Torque}$, which encourages the agent to move forward while keeping the hull straight and minimizing motor torque applied at joints.

An episode terminates when 2,000 time steps (frames) have elapsed, when the agent's head touches any obstacle or ground, or when it arrives at the finish line (the right end of the obstacle course).

Following Wang et al. (2019a;b), an environment is considered *solved* when the agent both reaches the finish line and obtains a score of 230 or above. The controller (with 24 inputs and 4 outputs all bounded between -1 and 1) is a fully-connected neural network with with 2 hidden layers of 40 units each, with $tanh$ activation functions.

A.3.2. POET Experiment Setup

The hyperparameters for ES used in POET, and later in controls when relevant, are listed in Table 2. POET attempts to generate new environments every 150 iterations (M in step (1) in Section 3.1), and conducts transfer evaluation experiments every 25 iterations (N in step (3) in Section 3.1). When any environment-agent pair accepts a transfer or when a child environment-agent pair is first created, the state of its Adam optimizer, the learning rate, standard deviation for noise are reset to their initial values, respectively.

A.4. Additional Details and Results on Evaluation of Algorithmic Innovations

A.4.1. Definition of Challenging Levels

For the purpose of analyzing the quality of results, this work adopts the definitions of challenge levels for the simple, hand-designed EE (i.e. a vector of values that consists of the surface roughness, the range of stump height, and the range of gap width) from the original POET paper (Wang et al., 2019a;b), where environments are classified as either *challenging*, *very challenging*, or *extremely challenging*, based on how many conditions they satisfy out of the three listed in Table 3. Satisfying one of the three conditions makes an environment *challenging*; satisfying two of the three conditions makes an environment *very challenging*; and an *extremely challenging* environment satisfies all three conditions. Shown in Table 3, each of these conditions is much more demanding than the corresponding values used in the original Bipedal Walker Hardcore environment (Klimov, 2016) in OpenAI Gym (Brockman et al., 2016).

A.4.2. Performance Comparison between PATA-EC and the Oracle EC

Recall that PATA-EC is general and can be applied to nearly any environment and with any encoding. The idea in this experiment is to apply the new PATA-EC to the environment in the original POET (which still uses the original, hand-crafted EE), and then to compare the result to POET's performance with the original hand-designed EC on the

Hyperparameter	Setting
initial connection	full
activation default	random
activation options	identity sin sigmoid square tanh
aggregation default	sum
bias init stdev	0.1
bias init type	gaussian
bias max value	10.0
bias min value	-10.0
bias mutate power	0.1
bias mutate rate	0.75
compatibility disjoint coefficient	1.0
compatibility weight coefficient	0.5
enabled default	True
feed forward	True
node add prob	0.1
node delete prob	0.075
num inputs	1
num outputs	1
response init mean	1.0
response init type	gaussian
response max value	10.0
response min value	-10.0
response mutate power	0.2
single structural mutation	True
structural mutation surer	default
weight init stdev	0.25
weight init type	gaussian
weight max value	10.0
weight min value	-10.0
weight mutate power	0.1
weight mutate rate	0.75

Table 1. The setup and hyperparameter values for instantiating and evolving CPPNs.

same hand-crafted EE. The question is whether the more generic PATA-EC can perform reasonably close to an EC explicitly hand-crafted for this domain. With this setup, holding everything the same as in the original POET paper except for the EC, we find that the general PATA-EC can indeed produce the same diversity and levels of challenge environments as the original hand-designed EC, although it is less efficient at doing so. It requires $82.4 \pm 7.31\%$ more computation, measured in ES steps, to produce the same level of complexity (Figure 6).

A.4.3. Performance Comparison between Different Transfer Strategies

As shown in Wang et al. (2019a;b), periodic transfer attempts are essential to obtaining solutions in extremely challenging environments, despite being computationally expensive (Section 3.2). Here the different transfer strategies are compared in the same setup as in original POET (Wang et al., 2019a;b). POET with the improved transfer strategy creates (and solves) the same fraction of extremely challenging environments and achieves a similar diversity and challeng levels as the original POET, but with only $79.7 \pm 1.67\%$ of the computation (measured in number of ES steps) of the original POET (Figure 7). Furthermore, the chance of an existing agent paired with an environment being replaced by a transferred agent from another environment dropped from $50.44 \pm 3.39\%$ with the original POET (this high number suggests many false positives) to $22.31 \pm 2.42\%$. For comparison, the corresponding number from Innovation Engines (Nguyen et al., 2016) (which exhibited healthy goal-switching and optimization dynamics) was a similar 17.9%. A simple alternative, which is removing all fine tuning (i.e. not running ES at all as part of a transfer attempt), performs poorly (Figure 7). These results justify that some fine tuning is indeed necessary for finding

Hyperparameter	Setting
number of sample points for each ES step	512
weight update method	Adam (Kingma & Ba, 2014)
initial learning rate	0.01
lower bound of learning rate	0.001
decay factor of learning rate per ES step	0.9999
initial noise standard deviation for ES	0.1
lower bound of noise standard deviation	0.01
decay factor of noise standard deviation per ES step	0.999

Table 2. Hyperparameters for ES in POET experiments and controls.

	Maximum Stump Height	Maximum Gap Width	Roughness
POET	≥ 2.4	≥ 6.0	≥ 4.5
Original Bipedal Walker Hardcore environment	2.0	3.0	1.0

Table 3. **The challenge level of an environment is based on how many conditions it satisfies out of the three listed here.** Satisfying one, two or all three conditions makes an environment challenging, very challenging, or extremely challenging, respectively (Wang et al., 2019a;b). The values used in the experiments of POET to determine the challenge level of an environment are 1.2, 2.0, and 4.5 times the corresponding values used in the original Bipedal Walker Hardcore environment in OpenAI Gym (Klimov, 2016).

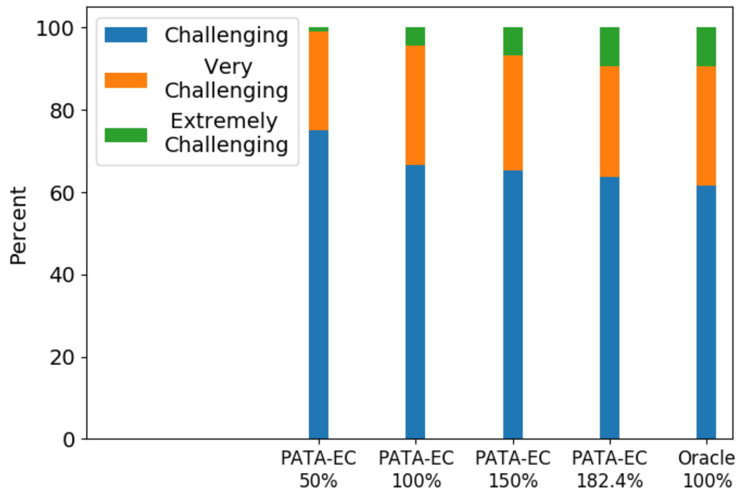


Figure 6. **Performance comparison between PATA-EC and the oracle EC when applied to the environment in the original POET (with the hand-crafted simple EE).** The domain-general PATA-EC can match the ability of the hand-designed, domain-specific Oracle EC to generate diverse environments of different challenge levels, although it requires more computation to do so. The number below each treatment reports the fraction of computation that treatment received relative to that with the Oracle EC (100%). As compute increases, POET with the domain-general PATA-EC can generate increasingly diverse, challenging environments, eventually matching the hand-designed, domain-specific Oracle EC in that regard. Definitions of "*challenging*", "*very challenging*", and "*extremely challenging*" environments are the same as those in the original POET (Wang et al., 2019a;b) and are also explained in Appendix A.4.1).

promising stepping stones. They also suggest that there are efficiency gains when only paying the computational cost of such fine tuning once the direct transfer test is satisfied.

A.5. Sample Environments

Figure 8 illustrates 12 sample environments that were created and solved by Enhanced POET with the CPPN-based EE in a single run. Videos of agents can be found at `https://eng.uber.com/enhanced-poet-machine-learning`. These environments are selected based on the following procedure: Let set $\mathcal{A}$ contain all the environments that POET created and solved in a run, and let $\mathcal{S}$ be initialized as a single-element set that contains only the perfectly flat environment, i.e., the

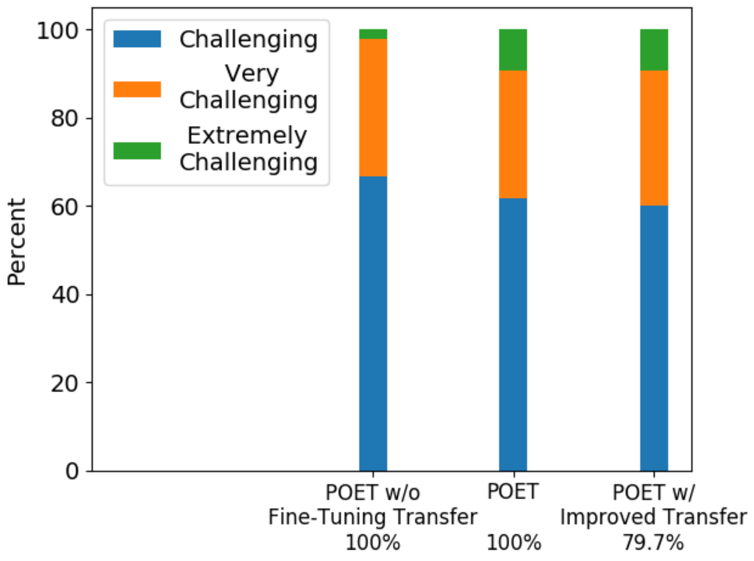


Figure 7. **Percentage of environments with different challenge levels created and solved by POET with different transfer strategies.** With the improved transfer strategy, POET is able to create and solve environments with the same diversity and challenge levels as the original POET with less computation. In contrast, removing all fine-tuning transfer performs poorly. The number below each treatment reports the fraction of computation that treatment received relative to that for original POET (100%).

very first environment that any POET run starts with. At each iteration of this procedure, we add to $\mathcal{S}$ environment E that satisfies $\arg\max_{E\in\mathcal{A},E\notin\mathcal{S}}\min_{e\in S}\mathcal{D}(E,e)$, where $\mathcal{D}(\cdot,\cdot)$ measures the distance between any given two environment. Here we adopt the same distance measure based on PATA-EC as that used in novelty calculation as stated in Section 3.2. We repeat the iterative process until number of environments other than the perfect flat environment in $\mathcal{S}$ reaches a preset value.

The intuition behind this selection process is that we continually add the environment that is furthest away from those in $\mathcal{S}$ using the distance measure proposed in this work. It is evident that those environments in Figure 8 exhibit a broad variety of obstacles that are vastly diverse not only in overall shapes and heights but also in fine details and local variations. This collection is a validation of the diversity-promoting nature of POET, and also a validation that the proposed distance metric based on PATA-EC does help to capture how different the environments are from each other.

For comparison with the CPPN-encoded environments illustrated in Figure 8, Figure 9 illustrates 12 sample environments that were created and solved in original POET with the simple, hand-designed encoding that only supports surface roughness and two regularly-shaped obstacles, i.e., stumps and gaps. Each column illustrates six sample environments from one run of original POET, where the upper, middle, bottom two rows illustrate the environments categorized as "challenging", "very challenging", and "extremely challenging" environments, respectively as defined in Appendix A.4.1.

A.6. Phylogenetic Tree

One way to visualize the diversity POET produces is by viewing a phylogenetic tree (i.e. a family tree) of the environments it has created at any given point. Natural phylogenetic trees have numerous, deep, nested branches. For example, nature has many phlya (e.g. mammals, plants, fungi, bacteria, etc.), each of which has many different, branches within it that are long (in that they have persisted over long periods of evolutionary time). Historic attempts at creating open-ended explosions of complexity in computer simulations of evolving systems in the fields of artificial life and computational evolutionary biology rarely, if ever, produce such phylogenetic trees (Lenski et al., 2003). Instead, usually one type of agent becomes more fit than everything else and replaces all the other types of agents, eliminating diversity (i.e. pruning all branches of the tree save one). These trees thus have one long trunk and a few shallow branches at the end that capture the not-yet-wiped out diversity in the current population.

Phylogenetic trees produced by POET, in sharp contrast, more resemble those from nature. Figure 10 shows the phylogenetic tree of the first 100 environments of a POET run. Each node corresponds a unique environment that POET created with an inserted picture illustrating its landscape. An edge connects an environment (on the upper end) to its child environment (on the lower end). The shape of nodes distinguishes whether environments are still in the active population (circular) or already in the archive (square) at the iteration when the 100th environment is added to the population, while the color of the border of each node suggests its time of creation: darker color means being created later in the process. Note the hierarchical organization, with major families, families within those, etc. The signature of open-ended algorithms is there: complex phylogenetic trees, meaning those that have multiple, deep, hierarchically nested branches. Of course, this tree is much smaller than those in nature, but that could be a function of (1) limited computational resources, and (2) that the environment search space in these first experiments with Enhanced POET are limited to obstacle courses only. Both are subjects in the discussion in Section 7.

The red arrows indicate when successful transfers happened (i.e. existing paired agents being replaced by transferred agents after fine-tuning) during one of transfer iterations (after the 100th environment is added, but before the 101th environment is added). Although most successful transfers happen between neighboring (more similar) environments, some agents manage to transfer to environments that are far from their paired environments (long red arrows in Figure 10).

A.7. Illustration of the Generalization Ability of Agents

Figure 11 illustrates the vectors of scores for how all agents perform across all the first 80 environments that a POET run creates and solves. More specifically, the ith column from left illustrates the vector of scores of all agents evaluated in the ith environment numbered in the order of being created, while the ith row from the top indicate the performance of the agent paired with the ith environment when tested across all the environments, respectively.

For the purpose of illustration, the raw score is normalized by 230, the minimum score POET has to achieve to solve the respective target environment (Wang et al., 2019a;b), and clipped between 0.0 and 1.0. That way, a normalized score of 1.0 is equivalent to an agent *solving* the environment, a normalized score of 0.0 means the agent achieved a zero or negative score in the environment, and a normalized score between 0.0 and 1.0 indicates that the agent makes some progress, but ultimately fails to solve the environment. The color intensity of matrix entries linearly scales with the normalized scores with white for 0.0 and black for 1.0.

In Figure 11, all diagonal entries are 1.0 because all environments shown are solved by their paired agents, while any given row reflects the ability of the corresponding agent to generalize across different environment. The upper-right triangular portion indicates how agents perform in environments created later than their paired environments. As environments created later are often more challenging, it makes sense that agents would perform poorly in the environments created much later than their paired ones (indicated by areas towards upper right). As a result, that area is mostly white with some light gray squares near the diagonal. It is also interesting to see that, based on the lower-left triangular portion, agents have limited success in environments that are created earlier than their paired environment as well. This phenomenon indicates that there are no universal "generalists" created by POET that are capable of solving all or most of the environments. Instead, over time POET creates "specialists" that are mostly specialized to their paired environments.

A.8. Additional Details and Results about Direct Optimization Control and the Ground-Interpolation Curriculum Control

A.8.1. PPO Experiment Setup

We adopt the PPO2 implementation from OpenAI Baselines (Dhariwal et al., 2017). The controller consists of a policy network and a value network. The policy network has the same architecture and activation functions as those used in POET (see A.3.1). The value network shares the input and the hidden layers with the policy network and it has a separate fully-connected layer that connects to the value output. Hyperparameters listed in Table 4 are chosen based on a grid search that yields the highest average scores across three environments randomly sampled from all target environments. We then hold this set of hyperparameters for all the PPO runs for all the target environments. Note that, as illustrated in Figures 12 and 13, PPO with these hyperparameters has effectively solved early-stage and some middle-stage target environments either by direct optimization or through the ground integration curriculum.

A.8.2. Equivalent Computational Budget

For both direct optimization control and the ground interpolation curriculum control, each run is given the same computational budget as POET spent to solve the target environment, measured in total number of time steps in simulation. It also includes all the simulation rollouts taken in order for POET to solve all the ancestor environments on the direct line leading to the target environment and all the computations related to transfer attempts into those environments.

A.8.3. Other Details and Results

As described in Section 6, the 15 target environments were sampled from the three different stages of POET runs. That is, for each target environment, we attempted 5 independent runs from different random seeds using direct optimization with ES, direct optimization with PPO, ground-integration

Hyperparameter	Setting
batch size	65536
number of training minibatches per update	4
number of training epochs per update	4
λ	0.95
γ	0.99
value function loss coefficient	0.5
gradient norm clipping coefficient	0.5
learning rate	0.0003
learning rate schedule	Anneal linearly to 0

Table 4. Hyperparameters for PPO experiments.

curriculum control with ES, and ground-integration curriculum with PPO, respectively (for a total of 300 runs for all target enviroments).

Figure 12 reports the normalized scores (following the same normalization method in Appendix A.7) and percentage solved by direct optimization for target environments at different stages. As with environments discovered by original POET, direct optimization can only solve target environments selected at the earlier stages of a POET run (when the produced environments are often less challenging), but neither ES nor PPO could solve more challenging target environments selected at later stages of POET runs. The normalized scores of direct optimization on middle and late stage target environments are significantly lower than 1.0, and the percentage of middle and late stage target environments solved by direct optimization are significantly below 100% ($p < 0.01$; Wilcoxon signed rank test).

The ground interpolation curriculum control follows a setup similar to that of the direct-path curriculum-building control in the original POET (Wang et al., 2019a;b). For each run, the agent starts in a perfectly flat environment. When in one environment the agent achieves a score above the reproduction eligibility threshold of POET (i.e. the condition for when an environment-agent pair is eligible to reproduce in POET), it moves to the next environment (whose scaling factor is increased by 0.02 from the current one). The run stops when the agent reaches and solves the target environment, or when the computational budget (Appendix A.8.2) is used up.

Figure 13 illustrates the scaling factor of the last-solved environment that is closest to the target environment along the path, and the percentage of target environments that are solved by the ground-interpolation curriculum. Statistical tests demonstrate that the ground-interpolation curriculum controls significantly underperform POET in solving late-stage target environment ($p < 0.01$; Wilcoxon signed rank test).

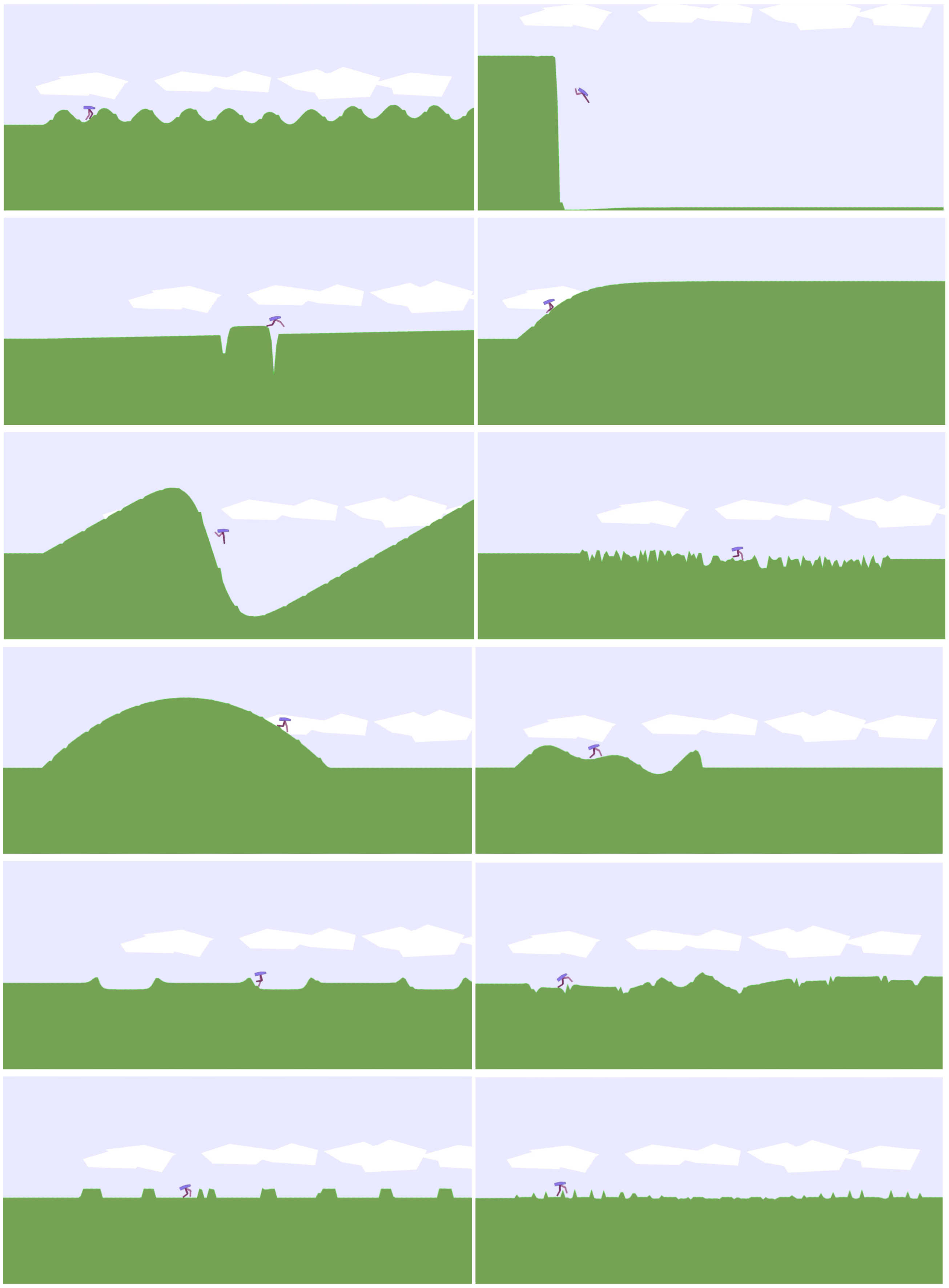


Figure 8. **Sample environments created and solved in a single run by Enhanced POET with the CPPN-based EE.** These environments exhibit a wide diversity of macro and micro environmental features.

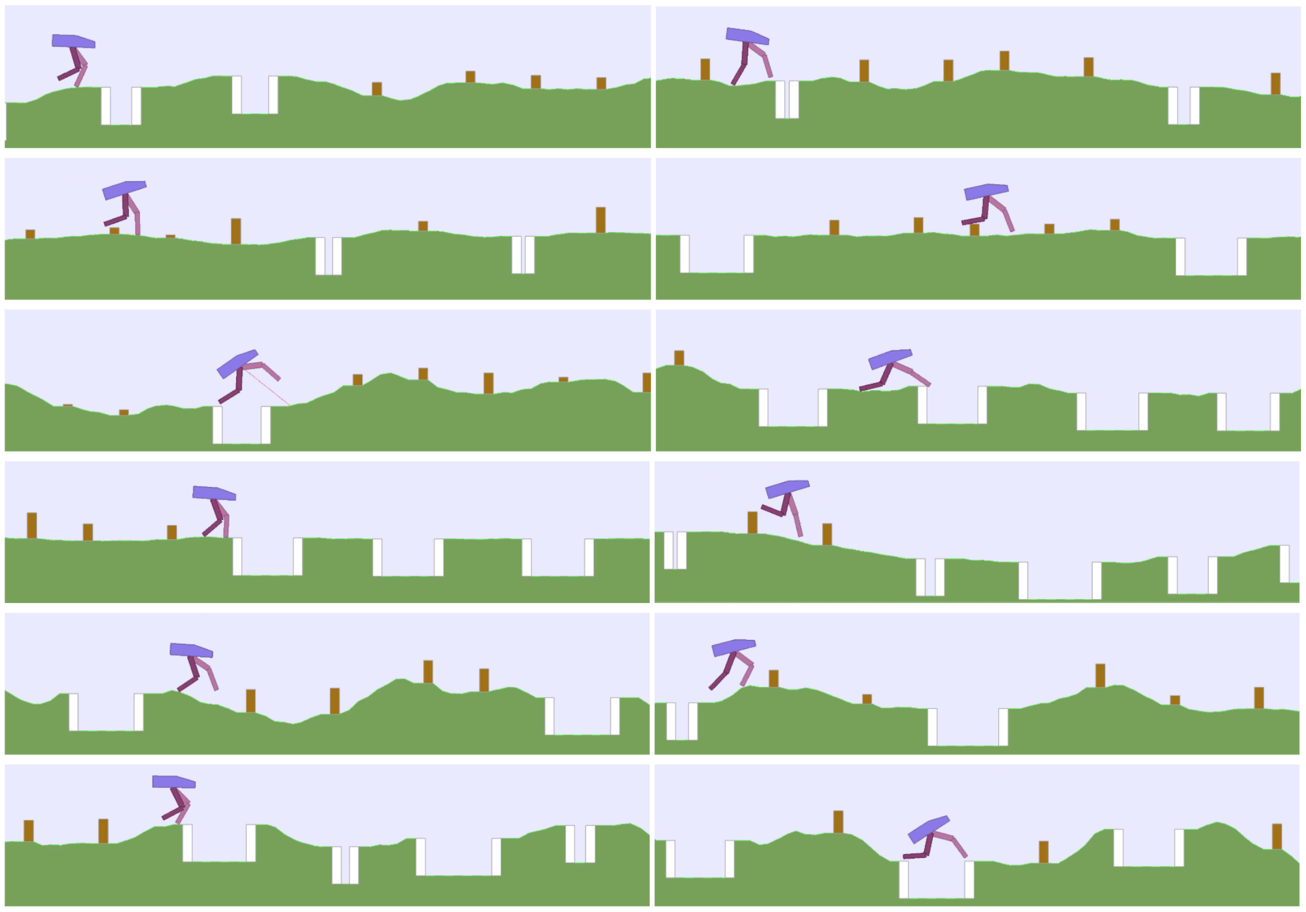


Figure 9. **Sample environments in original POET.** The simple, hand-designed EE can only support a finite set of types of obstacles, i.e., rough surfaces, stumps with fixed width and variable heights, and gaps with variable widths. This search space sustains some, but limited, diversity. Each column shows six sample environments from one run, where the upper, middle, and bottom two rows illustrate two “challenging”, “very challenging”, and “extremely challenging” environments, respectively, according to the challenge levels defined in Appendix A.4.1.

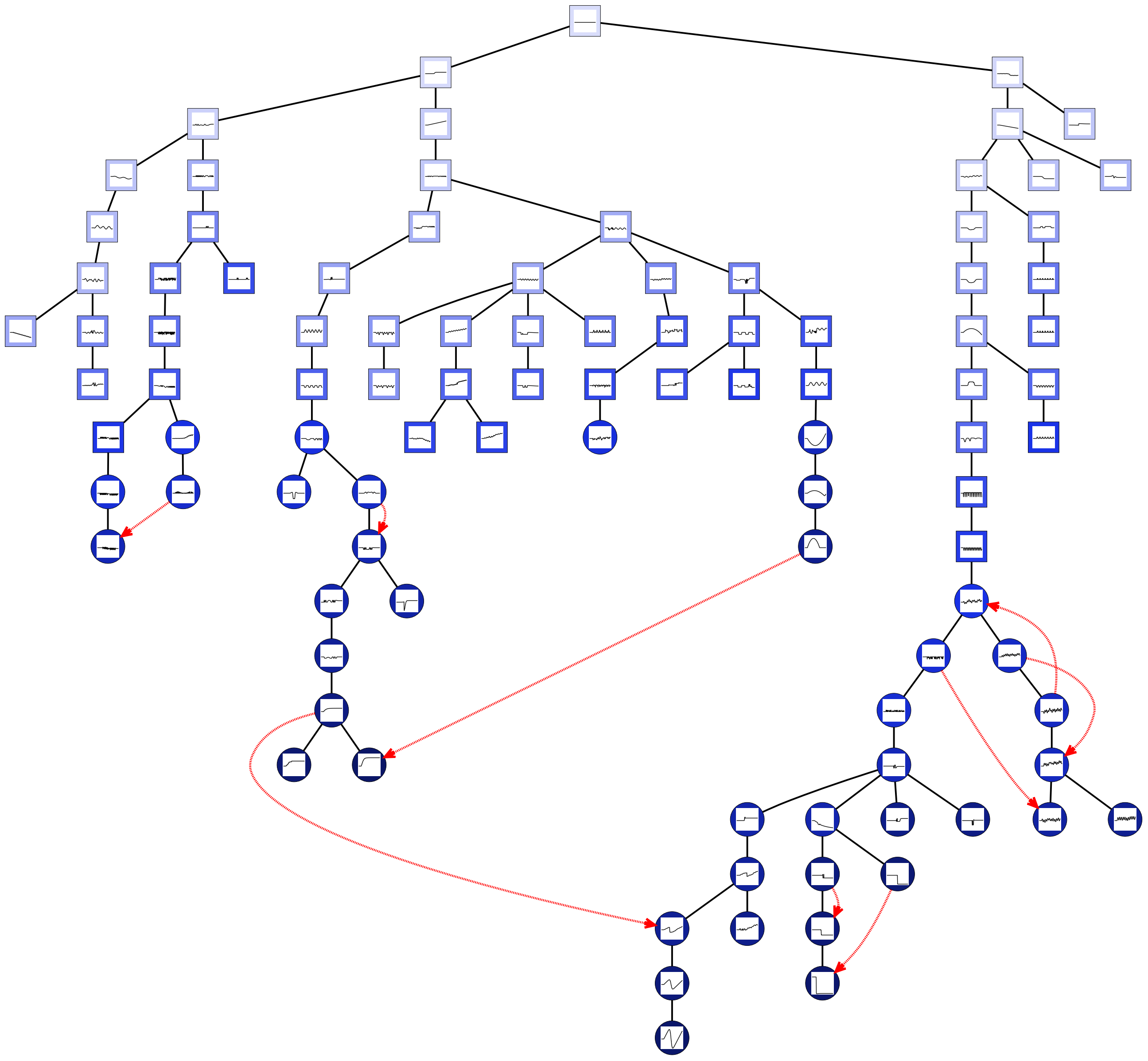


Figure 10. **Phylogenetic tree of the first 100 environments of a POET run.** Each node contains a landscape picture (zooming in the digital version enables seeing more detail) depicting a unique environment, with outgoing edges on its bottom connecting to its children. The circular or square shape of a node indicates that the environment is in the active population or the archive, respectively, while the color of the border of each node suggests its time of creation: darker color means being created later in the process. The red arrows label successful transfers during a single transfer iteration, specifically between the addition of the 100th and 101st environment.

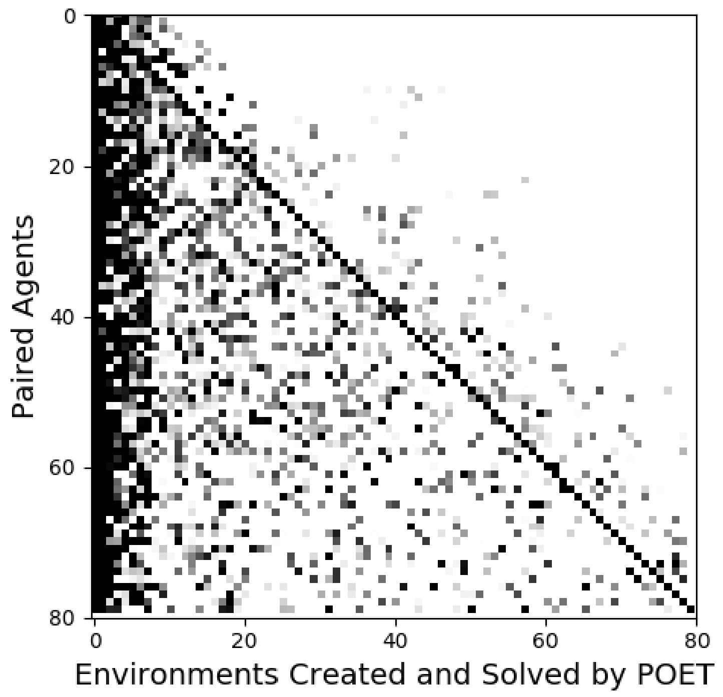
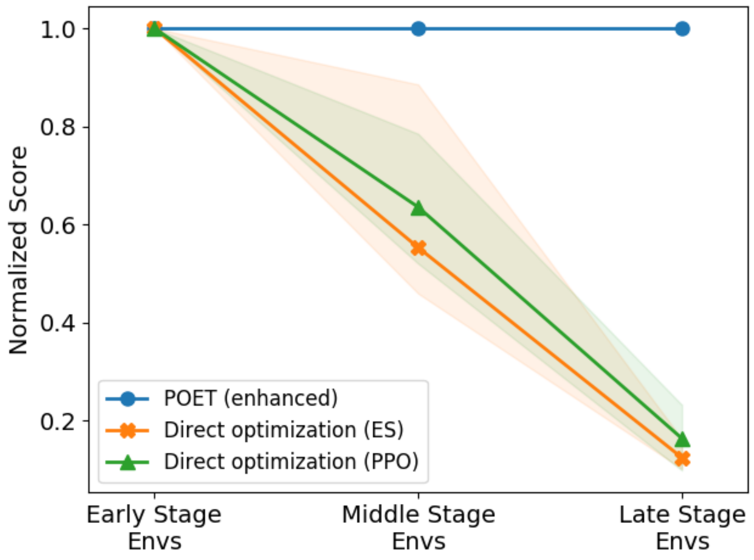
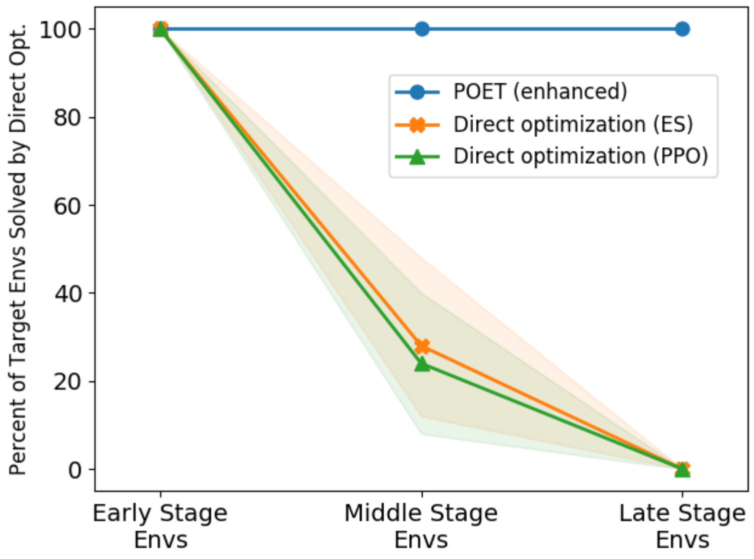


Figure 11. **Illustration of the vectors of scores of how all agents perform across all the first 80 environments that a POET run creates and solves.** Environments are ordered from left to right by the time of creation, while rows corresponds to their respective paired agents. Each square in the plot indicates the normalized score that the agent in that row performs in the environment of its respective column. The normalized score is between 0.0 and 1.0, and color-coded linearly as a grayscale between white and black. Normalization of raw scores is described in Appendix A.7.

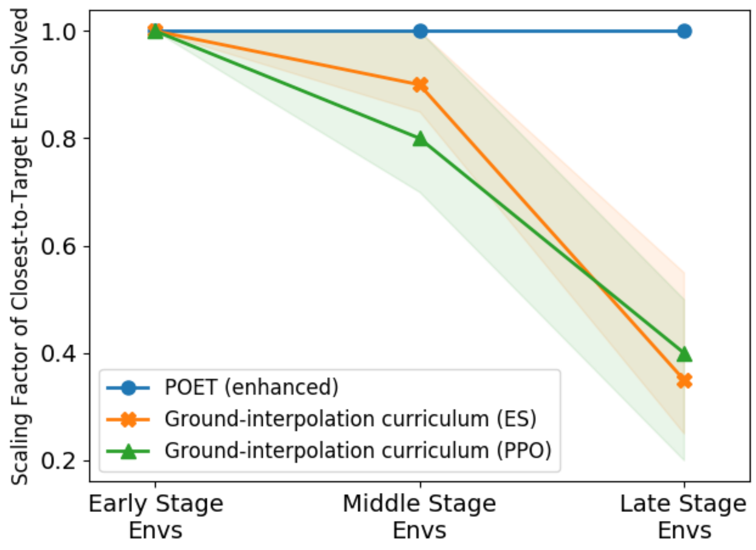


(a) Normalized scores of direct optimization on target environments. Symbols: median. Shaded regions: 95% bootstrapped confidence interval.

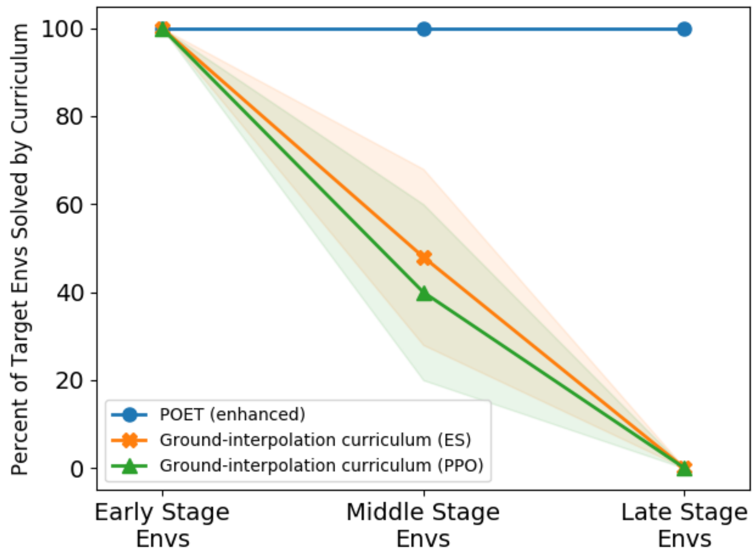


(b) Percentage of target environments solved by direct optimization. Symbols: mean. Shaded regions: 95% bootstrapped confidence interval.

Figure 12. **POET generates and solves challenges that direct optimization cannot solve.** Target environments were selected from different stages of POET runs (see text) and are all solved by POET (blue). Direct optimization with ES (orange) and PPO (green) respectively, can only solve those selected at the earlier stages of a POET run (which are therefore often less challenging), but could not solve more challenging target environments selected at later stages of POET runs.



(a) Scaling factors of last solved environments on the ground-interpolation curriculum towards target environments. Symbols: median. Shaded regions: 95% bootstrapped confidence intervals.



(b) Percentage of target environments solved by the ground-interpolation curriculum. Symbols: mean. Shaded regions: 95% bootstrapped confidence intervals.

Figure 13. **Performance comparisons between Enhanced POET and controls with the ground-interpolation curriculum.** Even with curricula customized to each challenging environment (linearly interpolating the ground), both ES (orange) and PPO (green) are unable to match the performance of the Enhanced POET (blue) with its implicit, self-generated curricula.