The Paired Open-Ended Trailblazer: POET

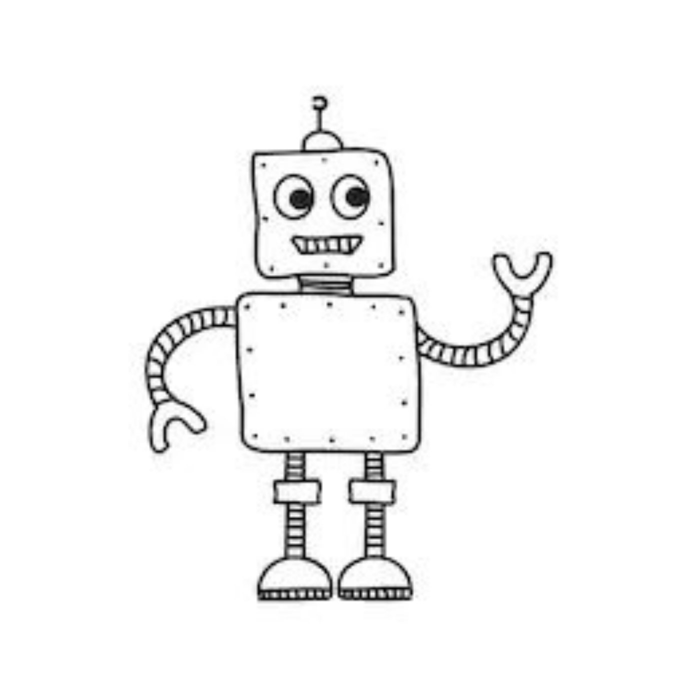
Wang, Lehman, Clune, & Stanley @ Uber AI Labs

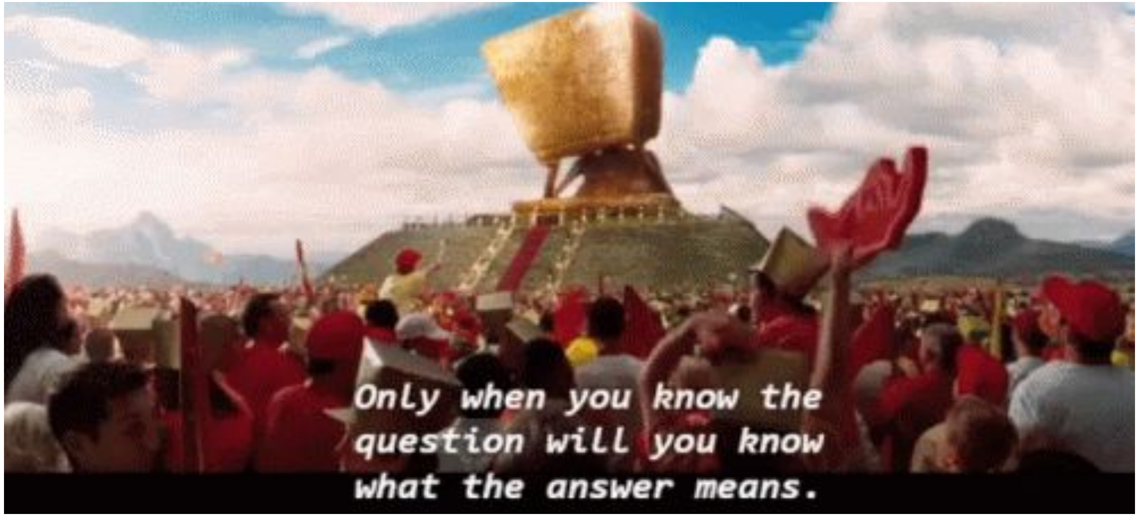
Jan 2019

Psych 239 **Sacha Uritis** **8 March 2019**

Humans create problems and build Machines to Solve them.

Why don't we have Machines Create the Problems and solve them, too?





“An intriguing question is whether it is possible to conceive an algorithm whose results would be worth waiting a billion years to see.”

Build-Up

Machine Learning Algorithms solve difficult problems

- ImageNet (2009)

Modern Deep Neural Networks begin to beat humans (RL and AI)

- ResNet (2015)
- Atari games
- Go & AlphaGo

Now What?

"Exotic Alternative" Let Machines Find their own Challenges + Solutions

Simultaneously...

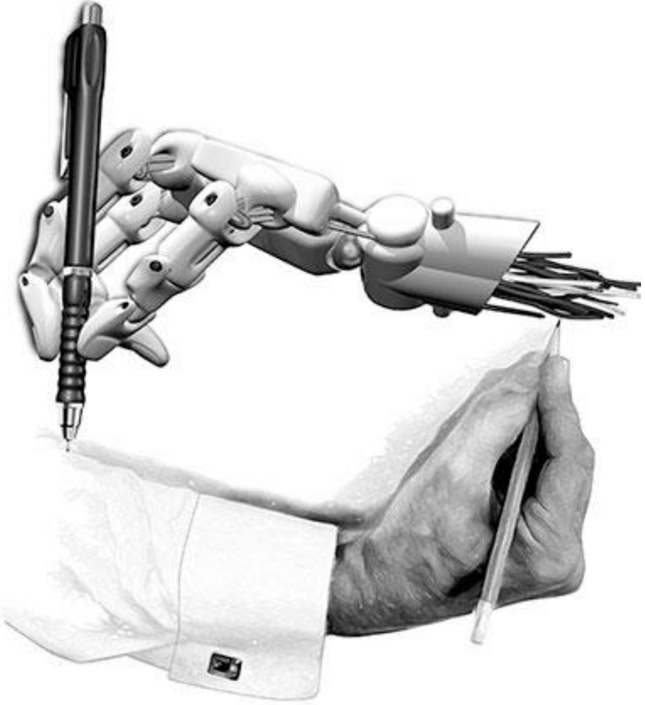
Autonomously...

Parallel & Asynchronously...

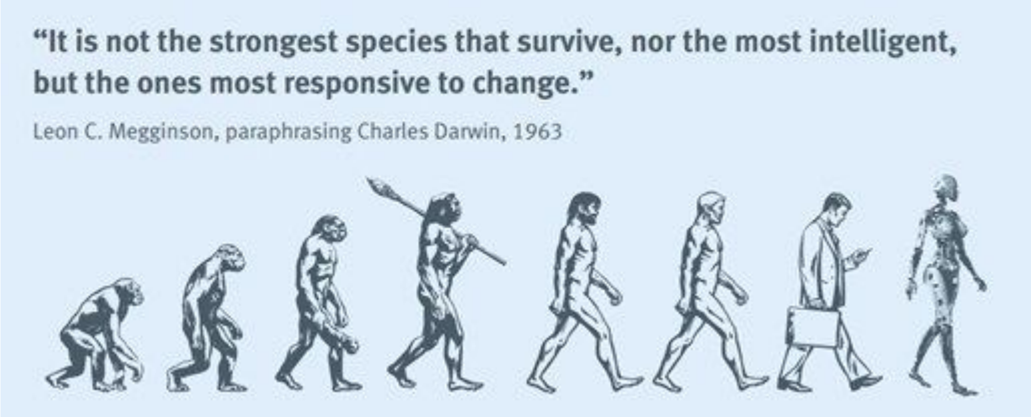
Limitless...

Forever...

Coevolution or Machine Takeover?

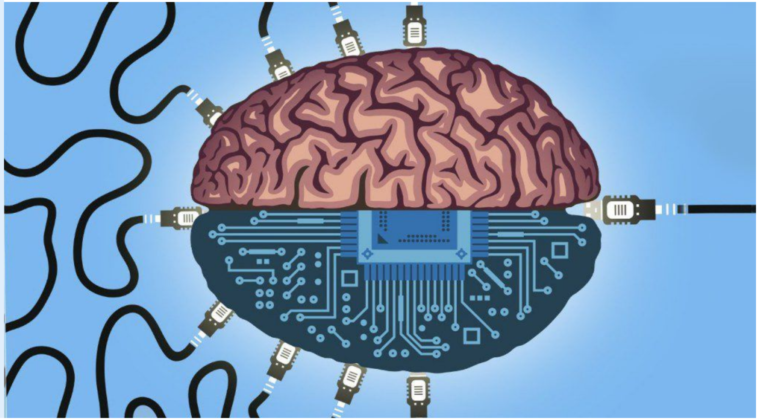


Self-contained & Open-Ended Curriculum-Generating



"It is not the strongest species that survive, nor the most intelligent, but the ones most responsive to change."

Leon C. Megginson, paraphrasing Charles Darwin, 1963



Human Environment : Nature, our World (never ceases to come up with new challenges...doesn’t converge & doesn’t stagnant)

Humans have evolved in this environment, reacting to changes, inventing challenges, and coming up with solutions.

Vocabulary Review

Paired : creates environmental challenges + optimizes agents to find solutions

Open-ended : continue running without bound as long as environment and computational power allows

Trailblazing : creating novel and interesting challenges and solutions

Population-based Algorithms : as opposed to single-based algorithms (single player games). You have a population of individuals trying to solve problems.

Background - Behavioral Diversity + Stepping Stone Solutions

Current Problem...

Algorithms usually become trapped in local optima due to decrease in complexity in domain and solutions

Solutions that minimize this...

Novelty Search, Behavioral Diversity, Reward Divergence, Quality Diversity, Goal-Switching

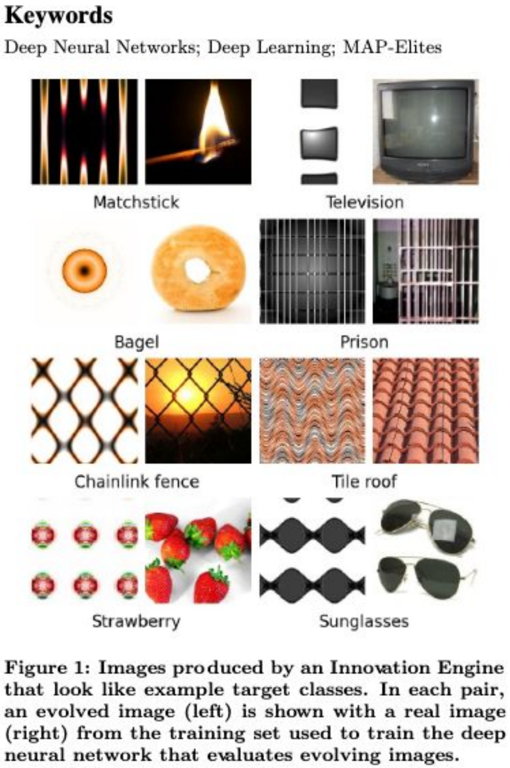
Example : Promoting diversity + Preserving Stepping Stones

Innovation Engines -

Transfers solutions from one objective to many others.

Keeps an archive of interestingly different stepping stones (e.g. states of a game).

Repetitive process until high-quality solution is found.



Keywords

Deep Neural Networks; Deep Learning; MAP-Elites

Matchstick
Television
Bagel
Prison
Chainlink fence
Tile roof
Strawberry
Sunglasses

Figure 1: Images produced by an Innovation Engine that look like example target classes. In each pair, an evolved image (left) is shown with a real image (right) from the training set used to train the deep neural network that evaluates evolving images.

Background - Open-Ended Search Via MCC

Another problem...

Diversity promoting algorithms are not enough for open-ended search. Static environments are the issue.

Solution for this...

Mutations/Creation of New Envs with set Minimal Criterion Coevolution (MCC) - Members earn the right to reproduce by satisfying a minimal criterion.

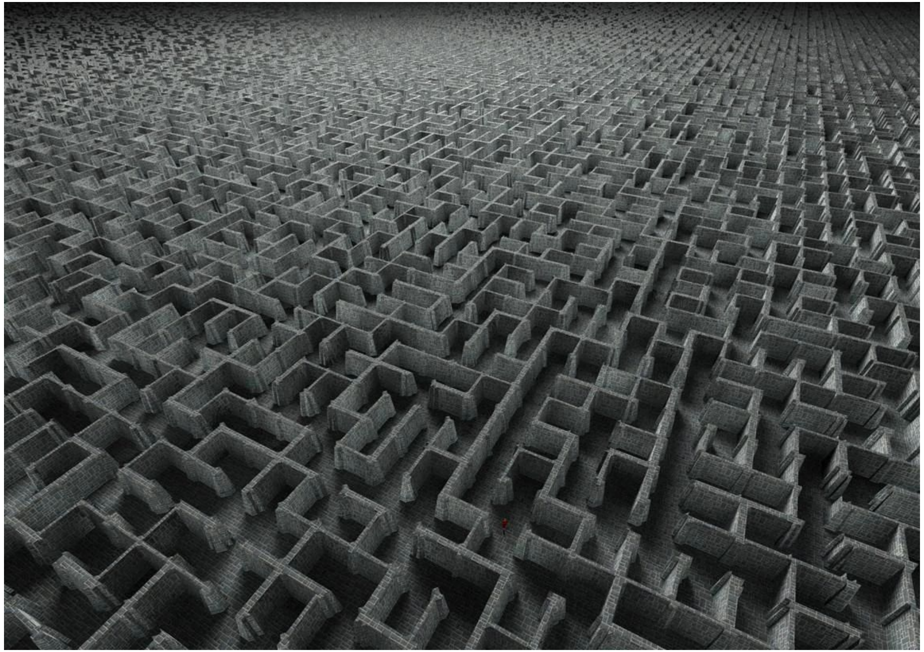
Example : Heading Towards True Open-Endedness

Minimal Criterion Coevolution : A New Approach to Open-Ended Search (Brant and Stanley)

Mazes : problems

Maze Solvers : solutions

Minimal Criterion : solvers must solve at least one of the mazes and mazes must be solved by at least one solver.



Background - Evolution Strategies (ES)

Another problem...

MCC does not force optimization of solutions; aims for completion, not mastery.

What can we do?

ES has shown similar performance levels as those from conventional simple gradient-based RL algorithms on complex domains like those in Atari.

Main Traits

Of POET

Behavioral Diversity + Stepping Stones

+

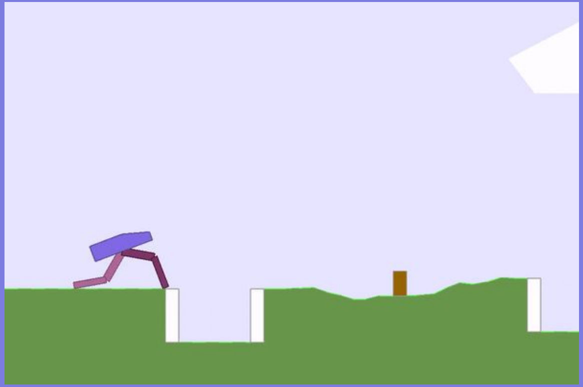
Open-Ended Search via MCC

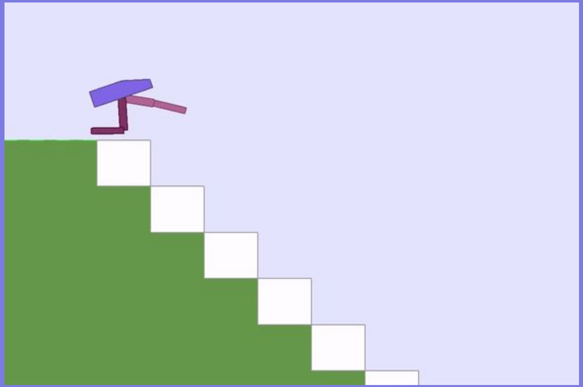
+

Evolution Strategies (ES)

POET

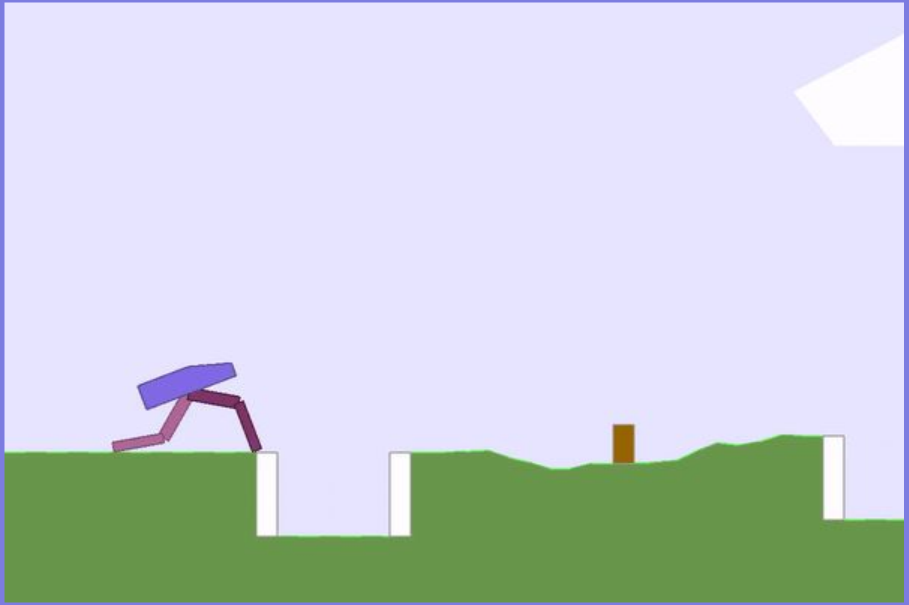
- Tested using modified version of "BiPedal Walker Hardcore" by OpenAI Gym
- Easy to Observe & Assess Qualitatively
- Easily modifiable environments
- Fast to simulate





Environment & Agent

- Agent has Two legs
- Hips and knees controlled by 2 motor joints
- Four Dim Action Space
- 10 LIDAR rangefinders + Internal sensors to assess the terrain
- 14 state variables (hull angle, hull angular velocity, horiz and vert speeds, positions of joints and joint angular velocities, whether legs touch the ground)

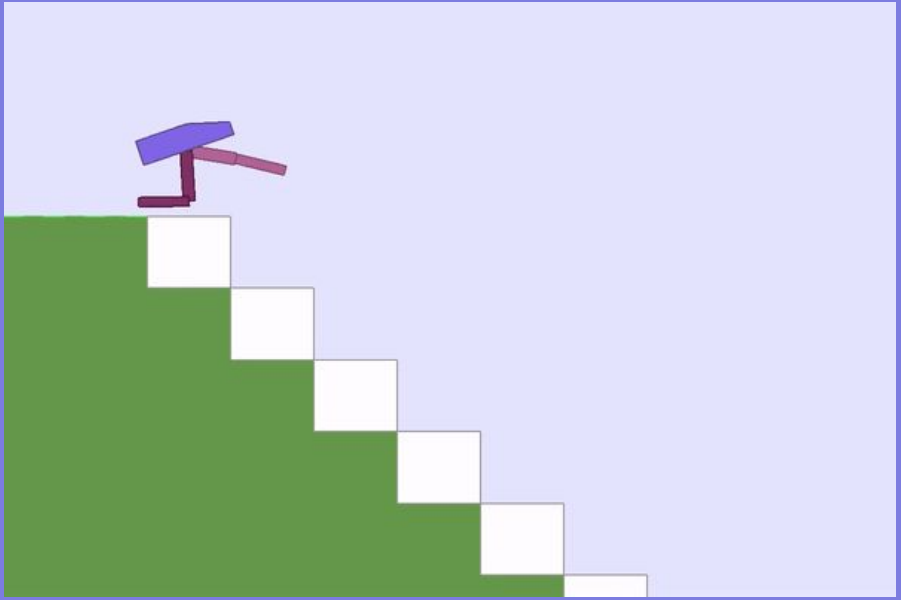


Goal

- Agent must navigate without falling
- Time limit

Obstacles

- Stumps, Gaps, & Stairs with varying roughness (height, width, frequency, etc.)



Reward

$$\text{Reward per step} = \begin{cases} -100, & \text{if robot falls} \\ 130 \times \Delta x - 5 \times \Delta\text{hull\_angle} - 0.00035 \times \text{applied\_torque}, & \text{otherwise.} \end{cases}$$

- Moving Forward ❤
- keep their hulls (main body) straight + minimize motor torque ❤
- Falling 🚫

Episode/Step Ends…

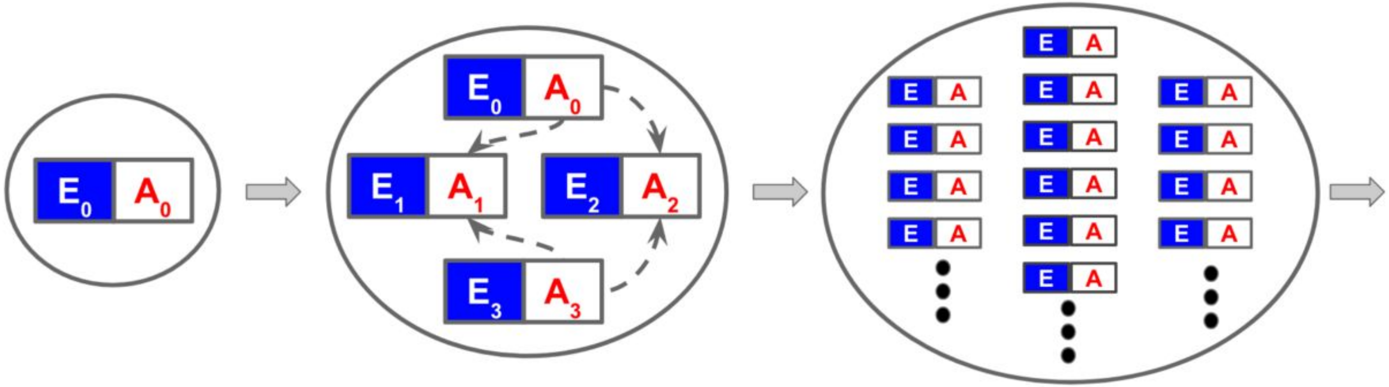
- Time Limit reached
- Agent Falls
- Course is complete

Episode Solved… (Part of Minimal Criterion Coevolution)

- Agent Reaches the far end of the environment
- AND Agent earns a score of 230 or greater

Population of EA-Pairs

- Population of Environments
- Population of Agents (NN controllers)



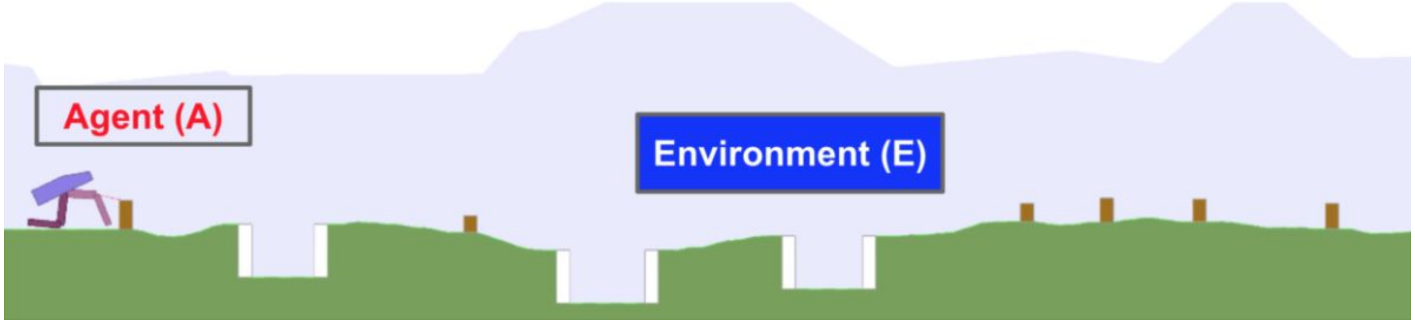
Idea + Algorithm

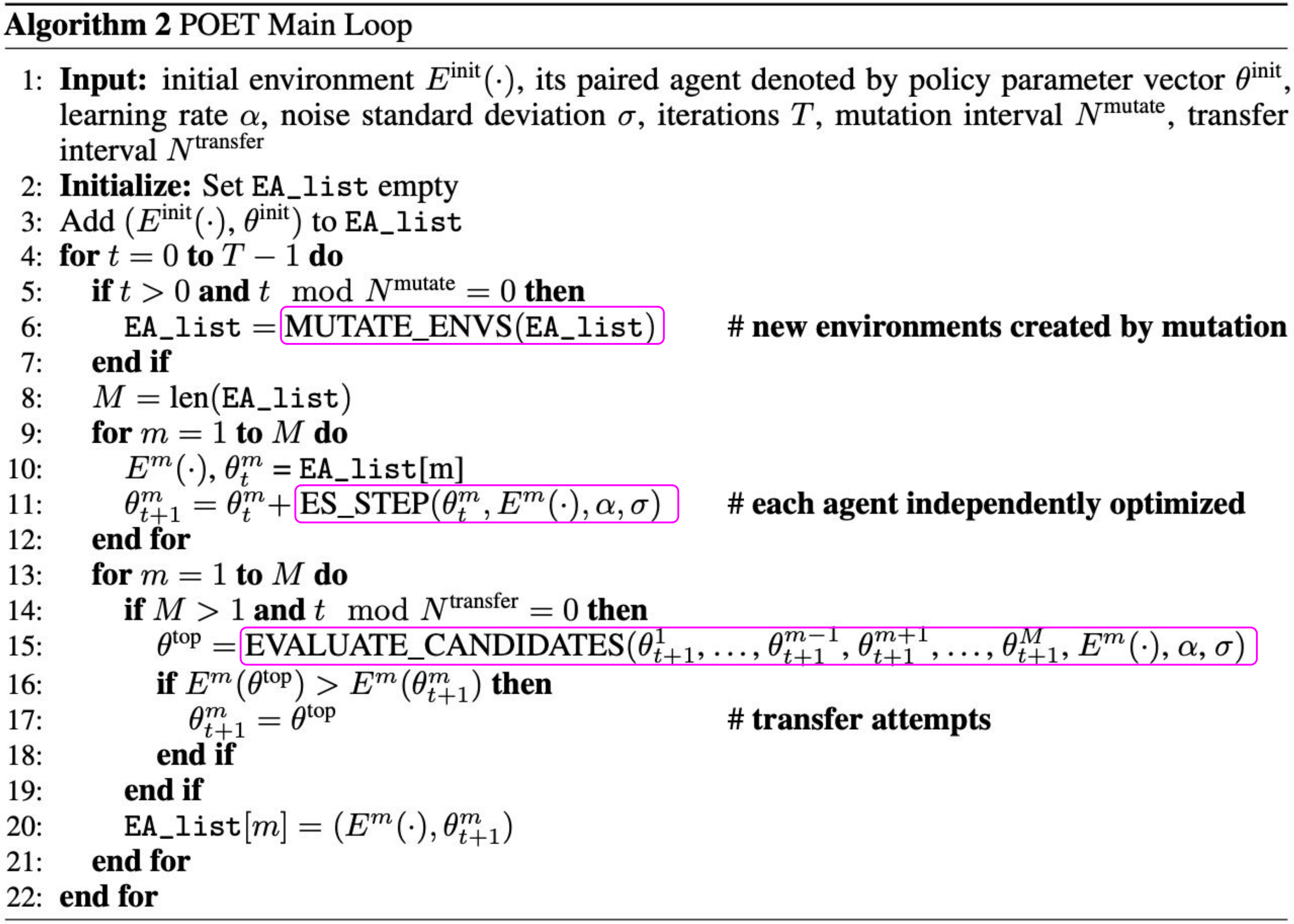
* Open-ended process in a single run.

* MCC - mutated envs kept if they are not too hard and not too easy for current population of agents to solve (score between 50 and 300)

* Optimizing behavior of each agent within its environment (CMOEA)

* Objective : increase challenges and skills within a single run.





Algorithm 2 POET Main Loop

```
Input: initial environment $E^{\text{init}}(\cdot)$, its paired agent denoted by policy parameter vector $\theta^{\text{init}}$, learning rate $\alpha$, noise standard deviation $\sigma$, iterations $T$, mutation interval $N^{\text{mutate}}$, transfer interval $N^{\text{transfer}}$
Initialize: Set EA_list empty
Add $(E^{\text{init}}(\cdot), \theta^{\text{init}})$ to EA_list
for $t = 0$ to $T - 1$ do
    if $t > 0$ and $t \mod N^{\text{mutate}} = 0$ then
        EA_list = $\boxed{\text{MUTATE\_ENVS(EA\_list)}}$        # new environments created by mutation
    end if
    $M = \text{len}(\text{EA\_list})$
    for $m = 1$ to $M$ do
        $E^m(\cdot), \theta^m_t$ = EA_list[m]
        $\theta^m_{t+1} = \theta^m_t + \boxed{\text{ES\_STEP}(\theta^m_t, E^m(\cdot), \alpha, \sigma)}$        # each agent independently optimized
    end for
    for $m = 1$ to $M$ do
        if $M > 1$ and $t \mod N^{\text{transfer}} = 0$ then
            $\theta^{\text{top}} = \boxed{\text{EVALUATE\_CANDIDATES}(\theta^1_{t+1}, \ldots, \theta^{m-1}_{t+1}, \theta^{m+1}_{t+1}, \ldots, \theta^M_{t+1}, E^m(\cdot), \alpha, \sigma)}$
            if $E^m(\theta^{\text{top}}) > E^m(\theta^m_{t+1})$ then
                $\theta^m_{t+1} = \theta^{\text{top}}$        # transfer attempts
            end if
        end if
        EA_list$[m] = (E^m(\cdot), \theta^m_{t+1})$
    end for
end for
```

List of active env-agent pairs: EA_List.

Initialize...

One Loop:

1. Generate new environments from those currently active
2. Optimize paired agents within their environments
3. Attempt transfer current agents from one env to another.

How ES Generally Works

	Typical RL Context
$E(\cdot)$	← Environment
w	← Parameter vector under parameterized policy
$E(w)$	← Reward we want to maximize with respect to w

ES seeks to maximize **expected fitness** of an agent over many policies sampled from probability distribution parameterized by θ

Stochastic Reward $E(w)$

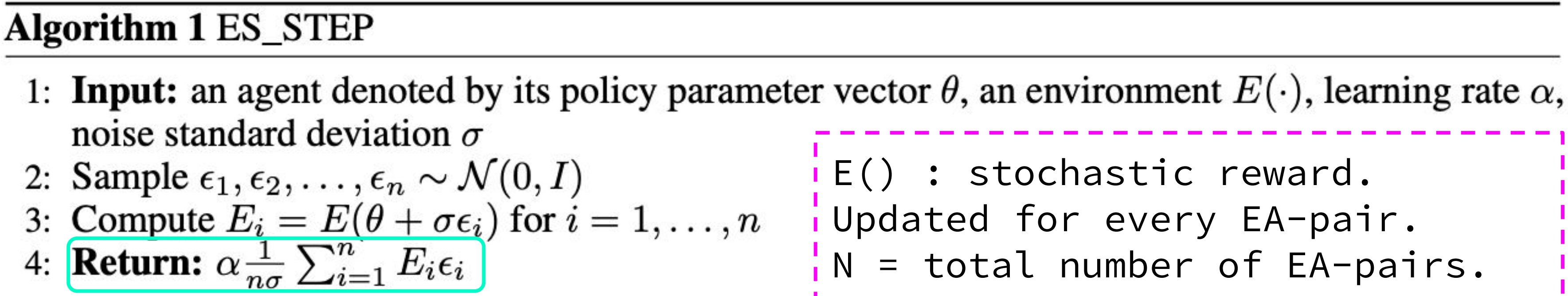
Expected Fitness → $J(\theta) = \mathbb{E}_{w \sim p_\theta(w)}[E(w)]$

Gradient of expected fitness can be estimated by → using a sample of size n.

$$\nabla_\theta J(\theta) \approx \frac{1}{n} \sum_{i=1}^{n} E(\theta_i) \boxed{\nabla_\theta \log p_\theta(\theta_i)}$$

$$\nabla_\theta J(\theta) \approx \frac{1}{n\sigma} \sum_{i=1}^{n} E(\theta + \sigma\epsilon_i)\epsilon_i.$$

Evolution Strategies Step



Algorithm 1 ES_STEP

```
**Input:** an agent denoted by its policy parameter vector $\theta$, an environment $E(\cdot)$, learning rate $\alpha$, noise standard deviation $\sigma$
Sample $\epsilon_1, \epsilon_2, \ldots, \epsilon_n \sim \mathcal{N}(0, I)$
Compute $E_i = E(\theta + \sigma\epsilon_i)$ for $i = 1, \ldots, n$
**Return:** $\alpha \frac{1}{n\sigma} \sum_{i=1}^{n} E_i \epsilon_i$
```

E() : stochastic reward.
Updated for every EA-pair.
N = total number of EA-pairs.

Returns Estimate for gradient of Expected Fitness to update the Policy.

- Policy parameter is randomly initialized weight vector θ
- Learning rate init 0.01 → 0.001 by factor 0.9999/step
- Noise standard deviation init 0.1 → 0.01 by factor 0.999/step

Transfer accepted or child EA pair is created, reset Adam, learning rate, and noise.

Mutating Environments

Active environments are mutated when these requirements are met:

1. EA-pairs proven enough progress to earn reproducibility.
2. Cannot be too hard or too easy for current population.
3. Priority given to the most divergent!
4. Maximum size for population of active environments (oldest environments are removed to make room)

Analogous to evolution of human population and our world.

Technical Details

All controllers implemented with neural networks

- 3 fully-connected layers
- *tanh* activation functions
- 24 inputs and 4 outputs
- 2 hidden layers (40 units each)
- Weight updates via Adam optimizer

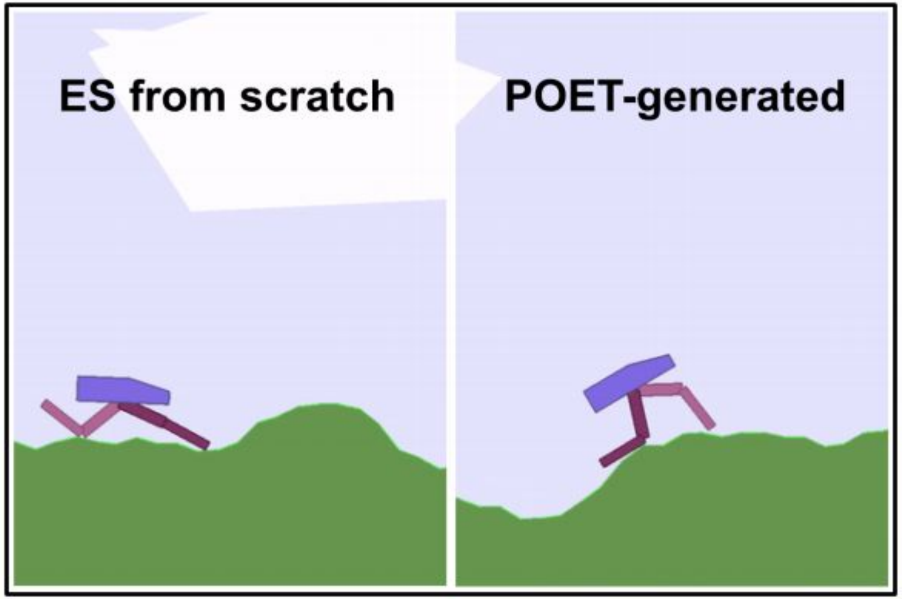
Population Size maintained at 512.

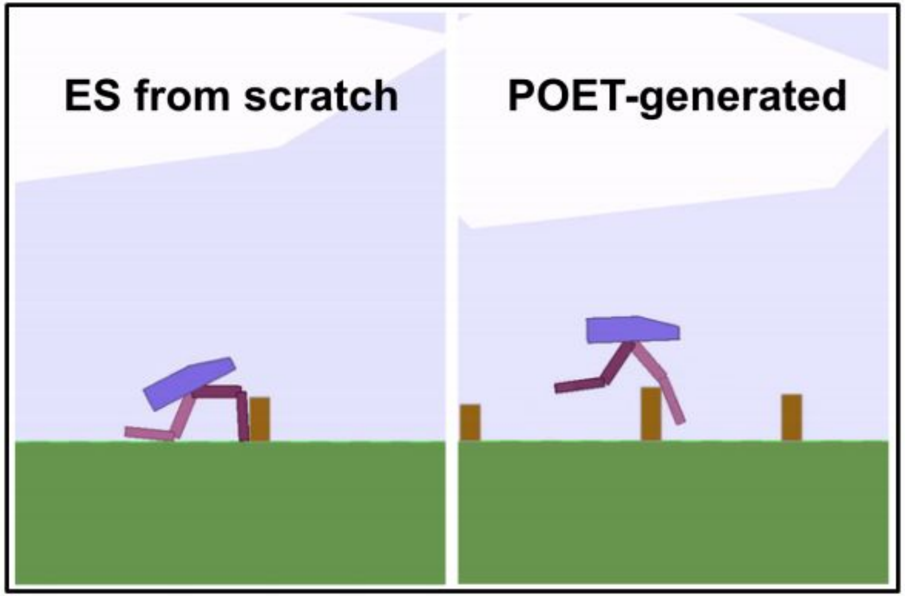
Features + Power

- Optimization and Transfer steps can happen independently and therefore parallelized easily.
- Most promising stepping stones for the best outcome may not come from the current best agent.
- Parallelization feature can utilize power of multiple parallel processors
- 256 Parallel CPU Cores
- Workers managed via Ipyparallel

Experiment Set 1 :

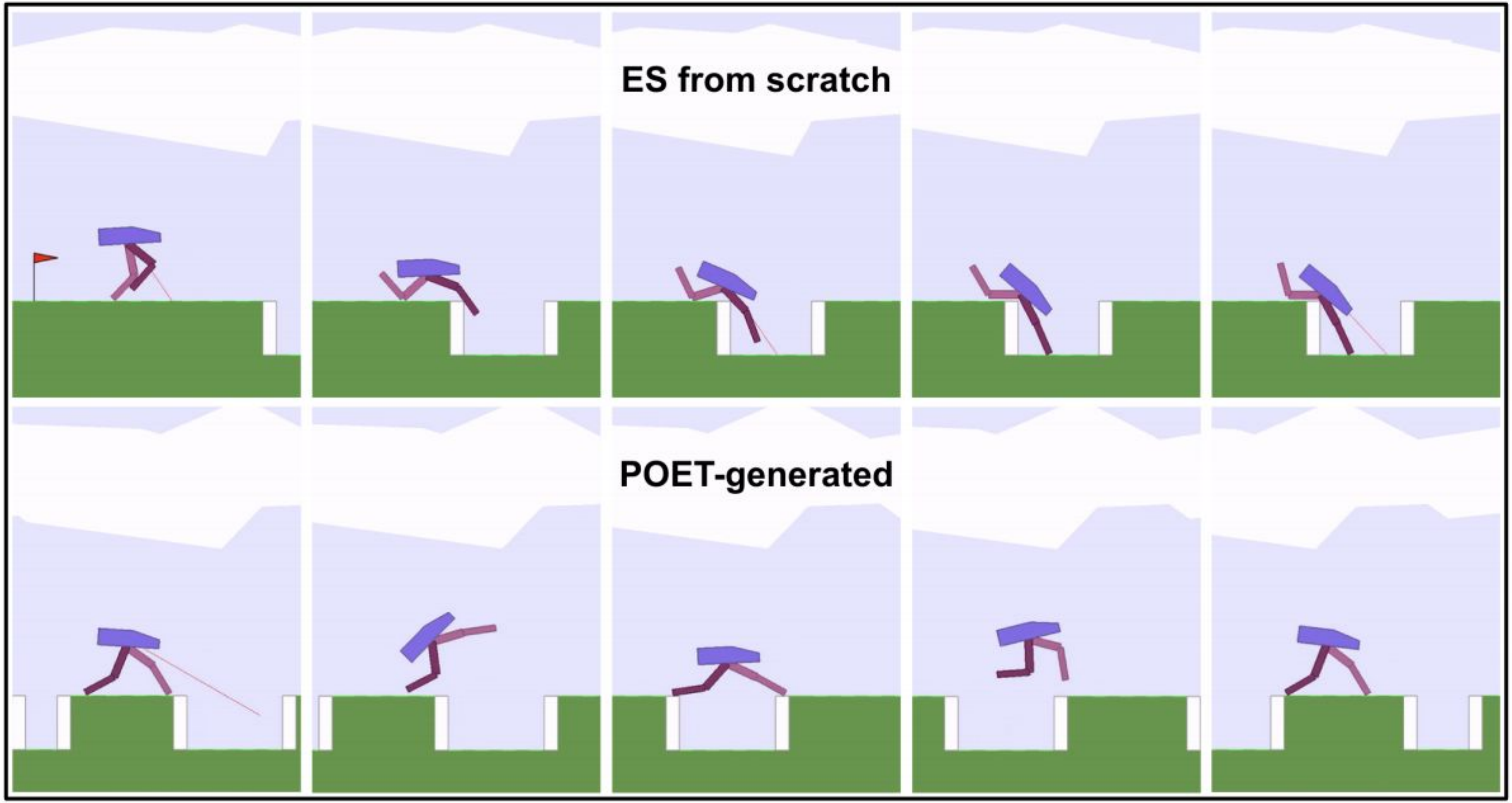
ES Alone from Scratch vs. POET

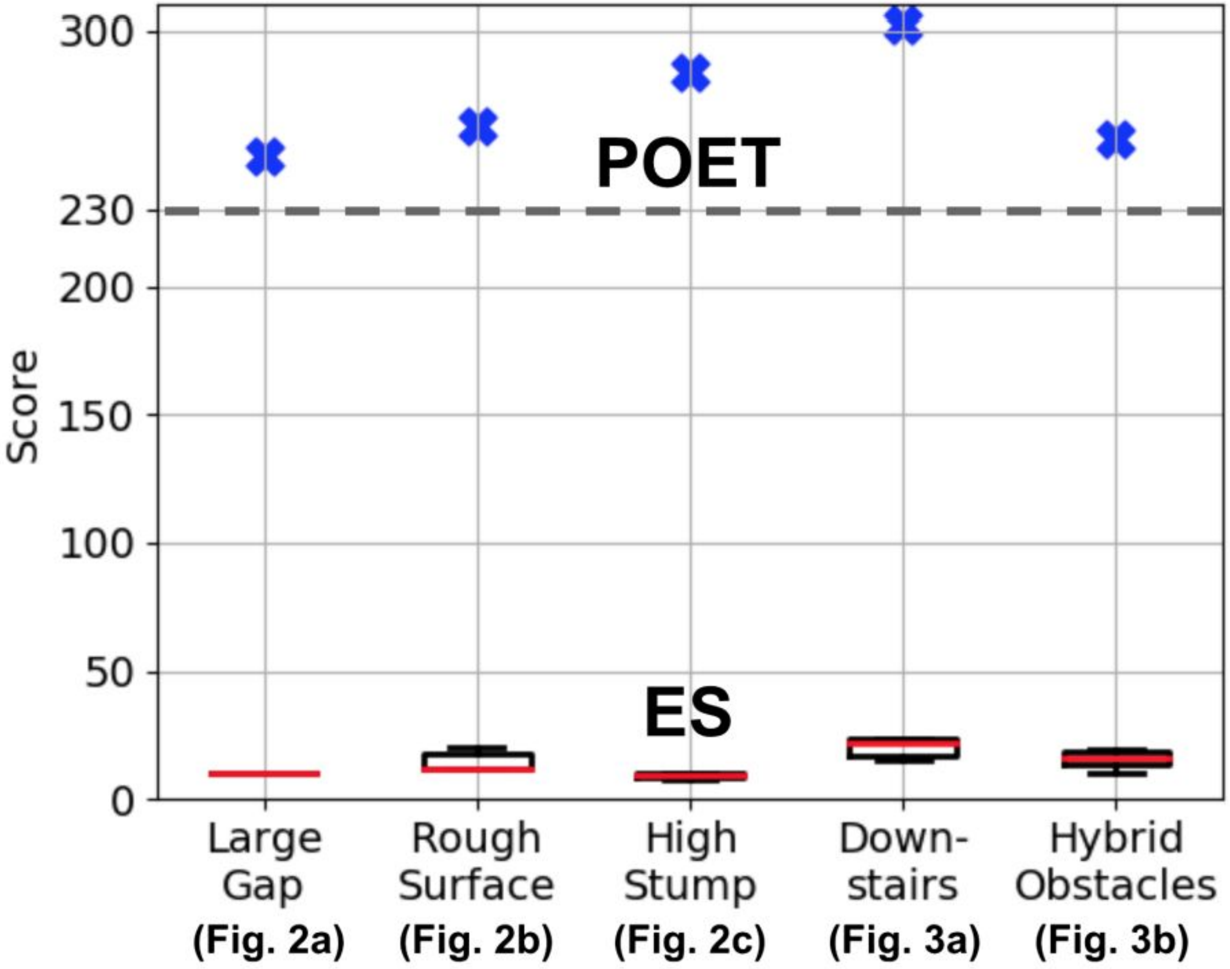




Agents directly optimized by ES converge to degenerate behaviors.

POET agents are more daring, adventurous, and risky.
They ultimately become and graceful, agile, and efficient.


ES from scratch
POET-generated



Bottom Boxplots show distribution of reward scores for ES-only algorithms across various challenges.

Recall 230 is POET’s threshold score for success.

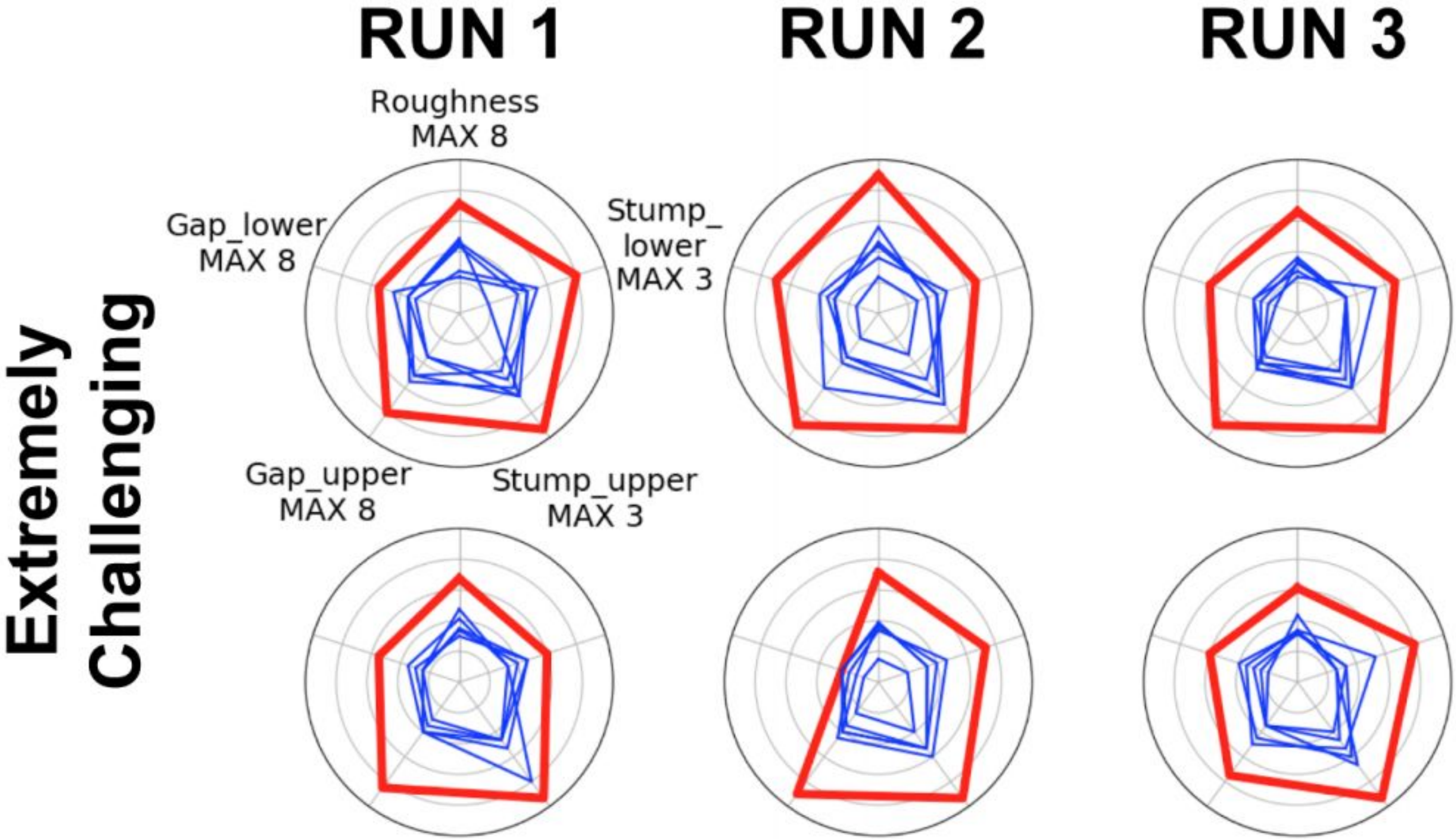
Conclusion:
Premature Convergence to Degenerate Behavior

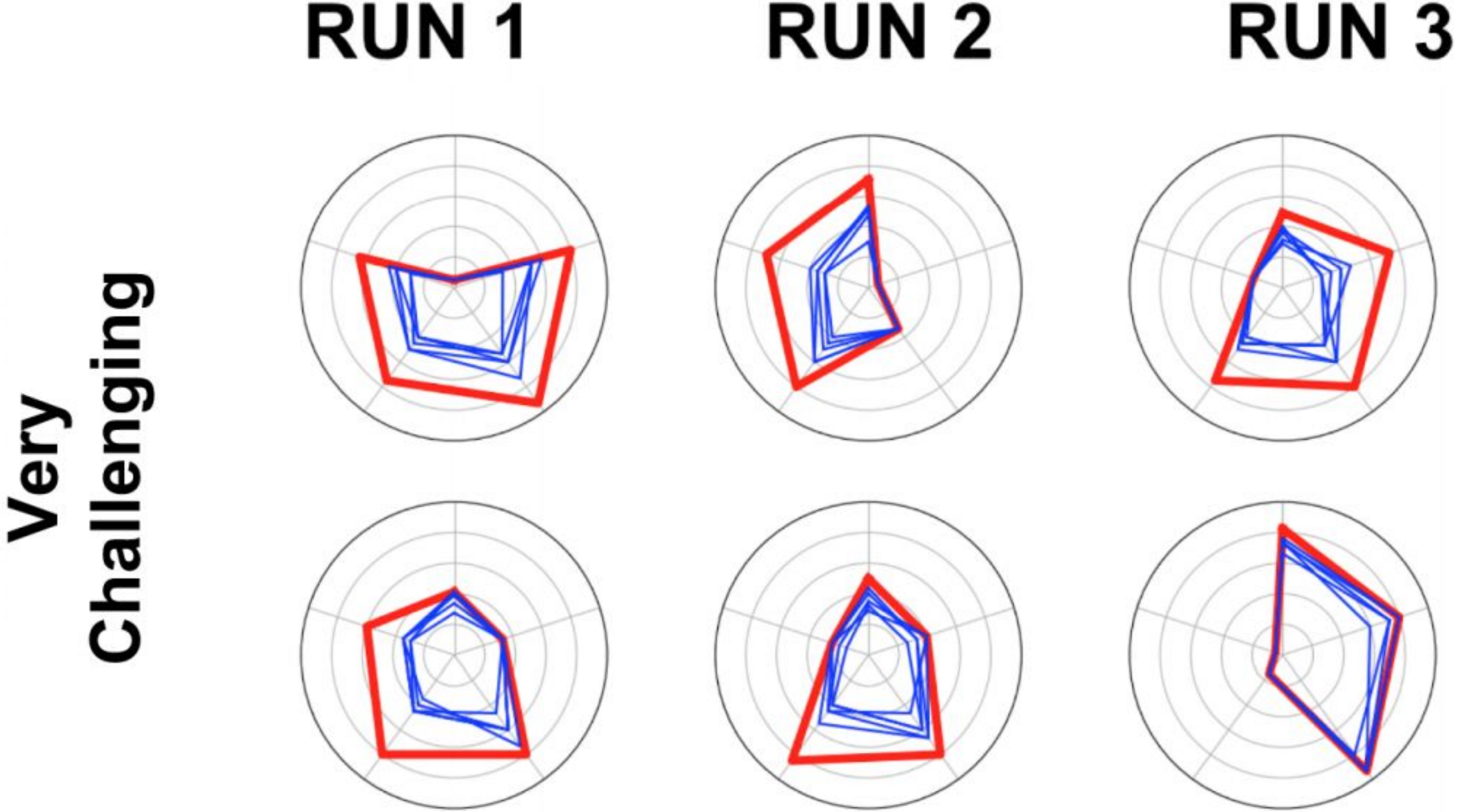
Experiment Set 2 :

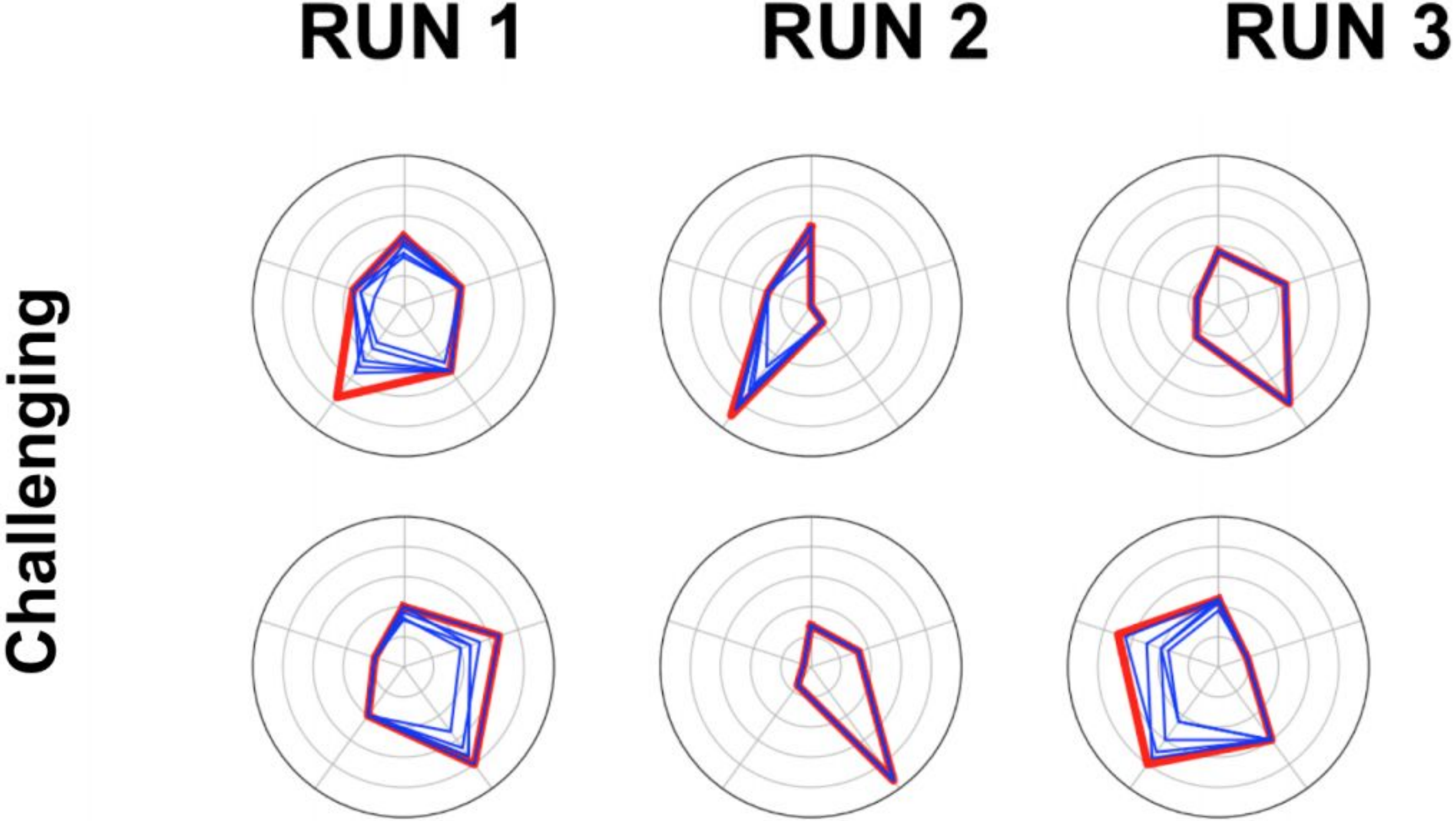
Can direct-path curriculum-building Control Alg. solve a series of POET-generated environments?

Experiment Setup

- Sample of sequences of envs created and solved by POET (challenging, very challenging, extremely challenging).
- Apply direct-path control to each one separately to see if it can reach same capabilities on its own.
- Each sequence starts with flat ground.
- Then, mutation/new envs only happen when the agent has earned a score eligible for reproducibility.
- **Can the control alg. produce complex envs that POET can AND can it solve them?**


RUN 1
RUN 2
RUN 3
Extremely Challenging
Roughness MAX 8
Stump_ lower MAX 3
Stump_upper MAX 3
Gap_upper MAX 8
Gap_lower MAX 8


RUN 1
RUN 2
RUN 3
Very Challenging


RUN 1
RUN 2
RUN 3
Challenging

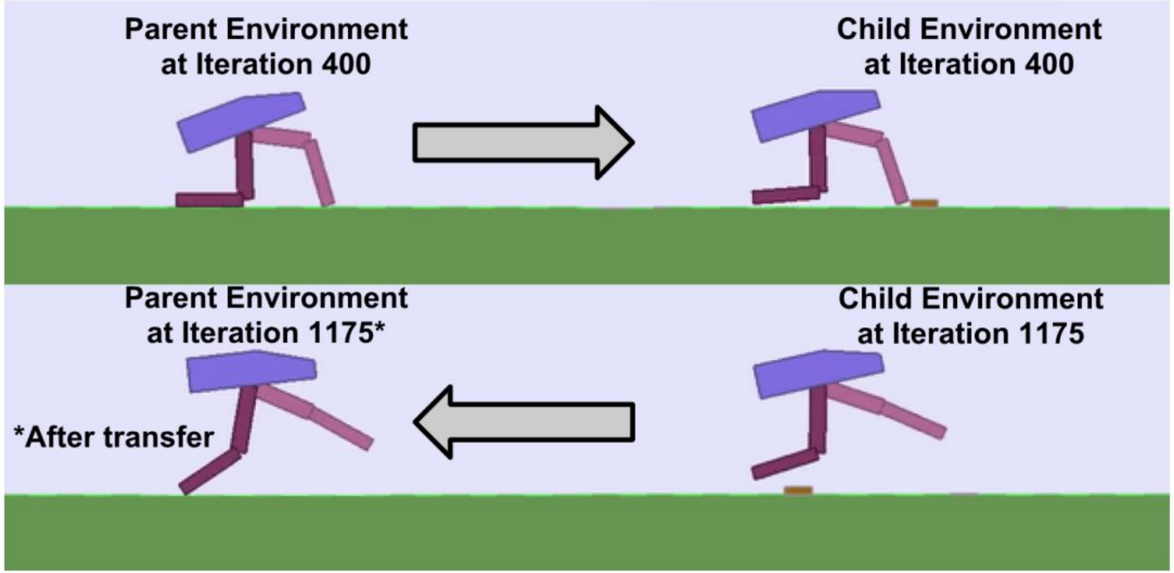
Conclusion:

Not bad with challenging envs, but Not so good with more challenging envs.

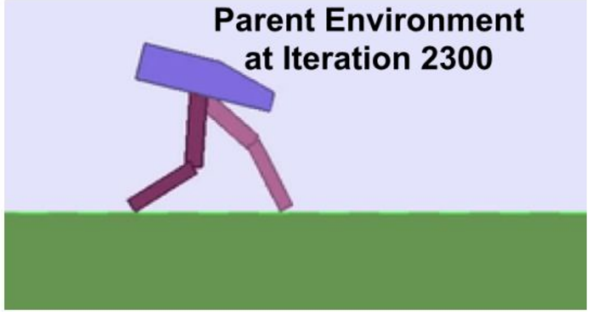
Lesson:

Skills learned in one environment can be critical for learning in another environment…transfer is important!

Synergistic two-way transfers!


Parent Environment at Iteration 400
Child Environment at Iteration 400
Parent Environment at Iteration 1175*
Child Environment at Iteration 1175
*After transfer

(a) Transfer from agent in parent environment to child environment and vice versa

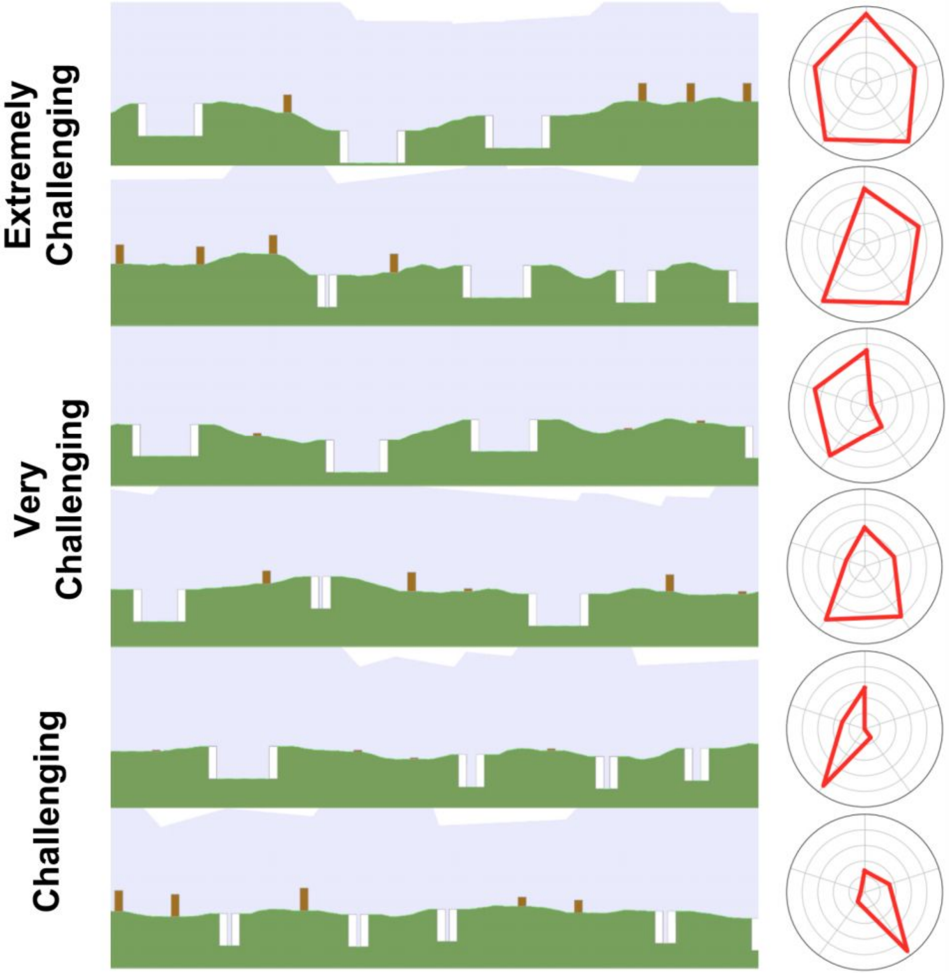


(b) The walking gait of agent in parent environment at Iteration 2,300

Analogy : Parents can learn a thing or two from their children.

Analysis :

Broad Diversity of various challenges w/ functional solutions in a single run



Extremely Challenging
Very Challenging
Challenging

Results Summary

1. Solutions found by POET for challenging environments cannot be found directly on those same environmental challenges by optimizing on them from scratch.
2. They cannot be found through curriculum-based process aimed at gradually building up to the same environments POET invented and solved.
3. Periodic transfer attempts of solutions from some environments to others (goal-switching) is important for POET's success.
4. Diversity are invented and solved in the same single run.

Shortcomings

Of POET

- 2-D walking course - space is limited by maximal ranges (max gap, max stump ht, etc.)
- Body of agent is fixed. No morphology.

Future Work

Opportunities to Extend POET

- Play around with other variants of ES (more open-endedness!)
- Meta-learning (learning to learn)...unique reward function for each environment.
- Other domains (3D parkour, autonomous driving, protein folding, search for chemical processes that solve unique problems).

References...

https://eng.uber.com/poet-open-ended-deep-learning/

https://arxiv.org/pdf/1901.01753.pdf