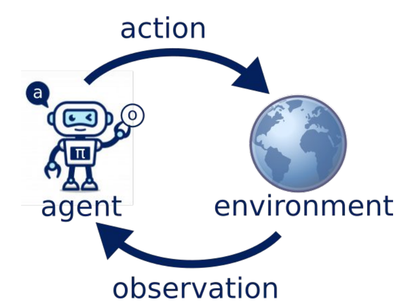
# Policy Manifold Search
## for Improving Diversity-based Neuroevolution

Nemanja **Rakicevic**
Antoine **Cully**
Petar **Kormushev**
Imperial College London

NEURAL INFORMATION PROCESSING SYSTEMS

## Reinforcement Learning Problem Setting

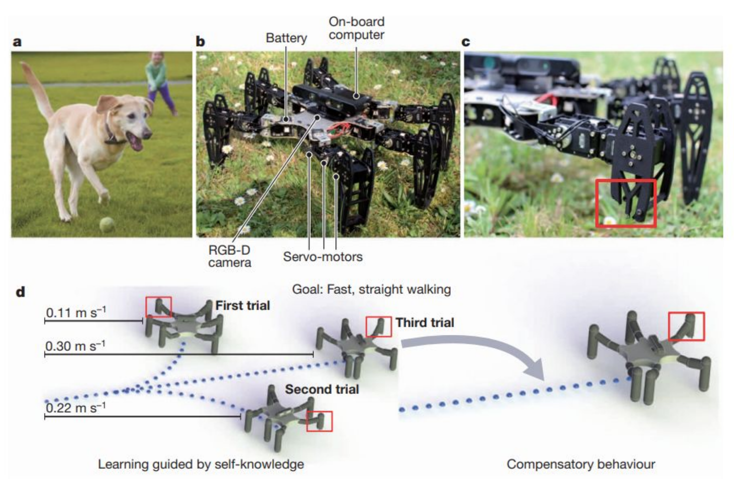



Agent, defined by a policy (**π**), interacts with the environment, by making observations (**o**) and determining the best action (**a**) to execute next.

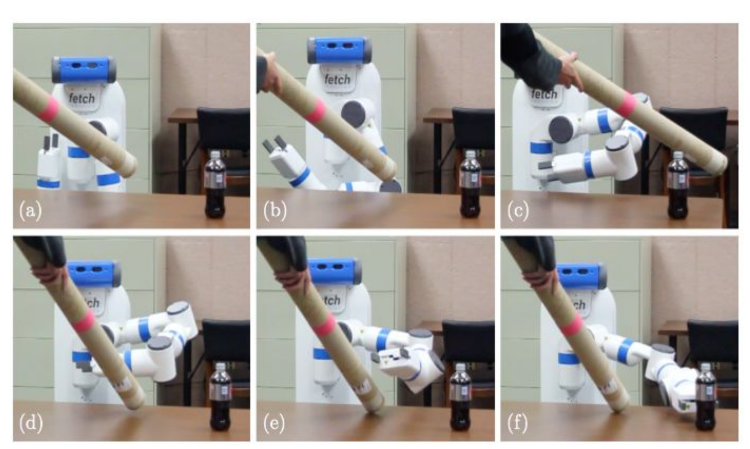
Standard RL approaches learn a single optimal policy.

## Why Behaviour Diversity?

If variations in the environment are too great, or if situations occur, which have not been seen sufficiently during training - a single policy might not be able to solve the task anymore.



Robot damage recovery [1]



Encountering obstacles not seen previously [2]

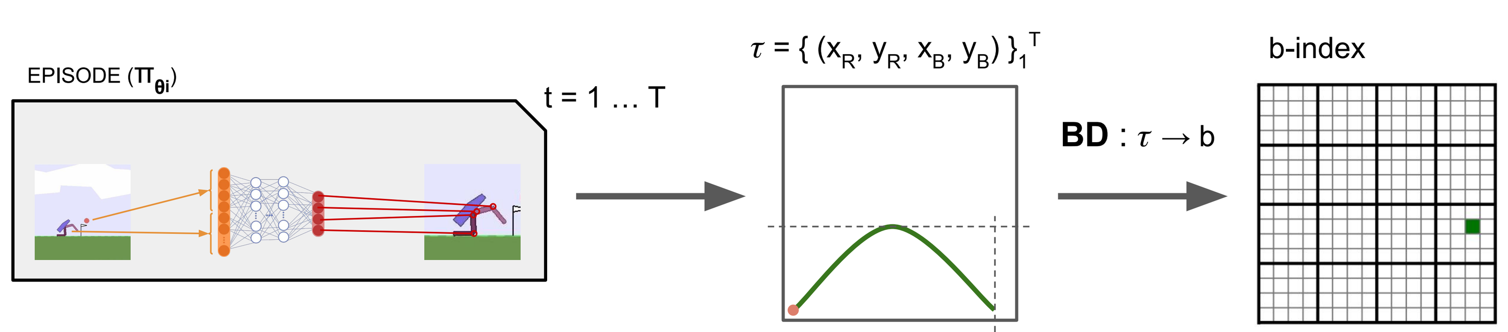
Having **multiple policies**, exhibiting **diverse behaviours** in the environment, would be useful in these situations.

Main approaches in the literature:
- Skill-conditioned policies [3, 4]
- Quality-Diversity [1, 5]

## Behaviour descriptor

**Behaviour descriptor** quantifies a policy behaviour, by mapping the trajectory generated by a policy in an environment, to a **behaviour index (b)**, which defines the position of the corresponding policy parameters in a **policy collection**.

Policy parameters are added to the policy collection, if they exhibit diverse behaviours in the environment during evaluation.




## Policy Manifold Search

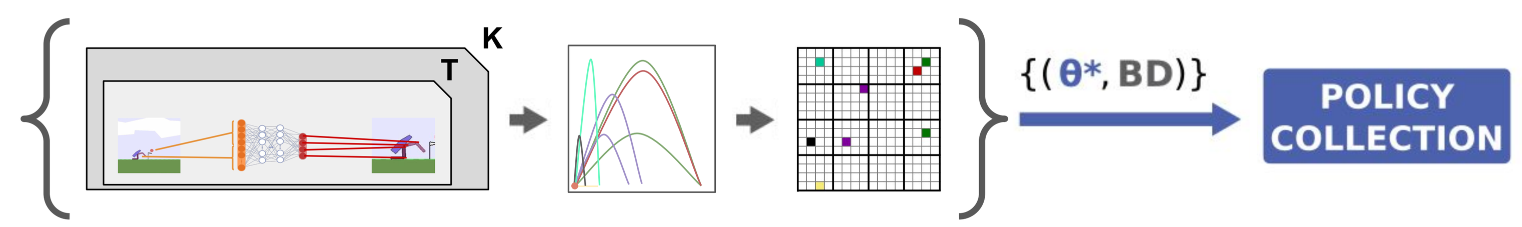
**Step 0)** Initialisation

Start by randomly sampling a set of policy parameters $\{\theta^*\}_k$

$$\{ N(0, I) \}_K \longrightarrow \{\theta^*\}_K$$

**Step 1)** Evaluate and Add to the Policy Collection

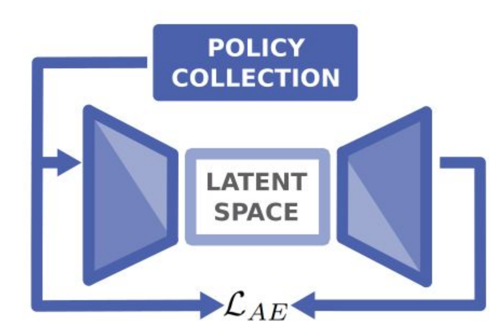
The generated set of policies is evaluated in the environment.




**HYPOTHESIS**
There exists a low-dimensional **manifold**, embedded in the high-dimensional **policy parameter space**, around which a **high-density of solutions** can be found.

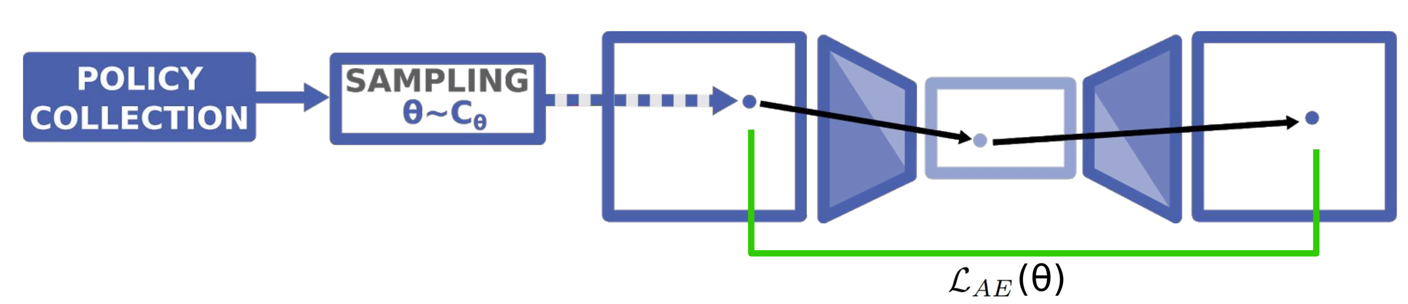
**Step 2)** Manifold Learning

Current policy collection is used to train the AE and obtain a latent representation of the policy parameter space.

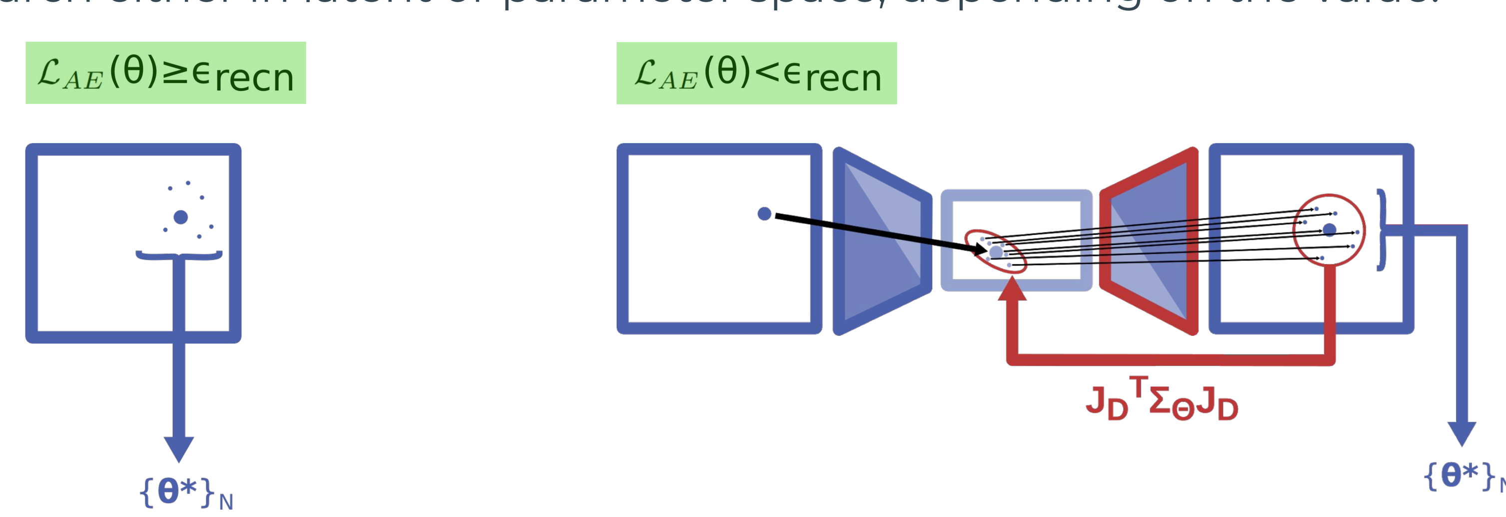



**Step 3)** Manifold Search

Sample the policy collection as in [3] and asses the reconstruction error of the selected sample, based on the current AE state.



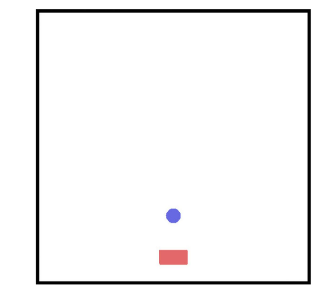

Search either in latent or parameter space, depending on the value.



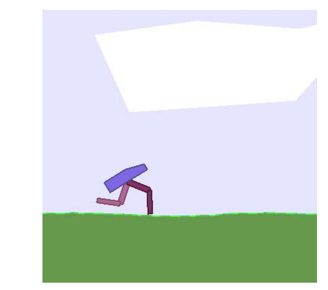

**Repeat Steps 1 - 3**, until stopping criteria are met.

## Experimental Evaluation

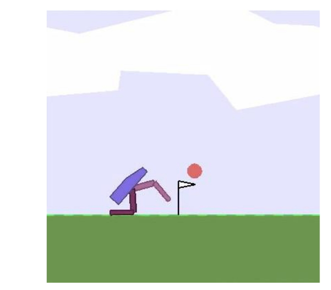
Environments



Striker

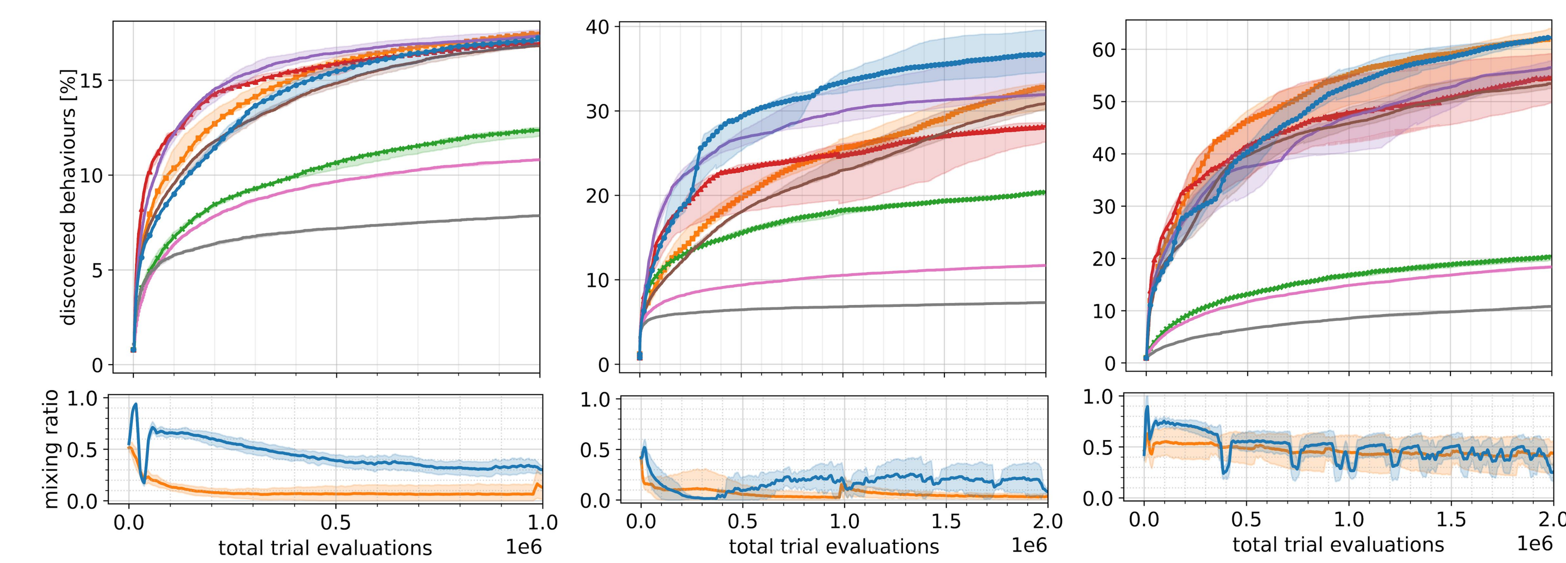


Bipedal Walker

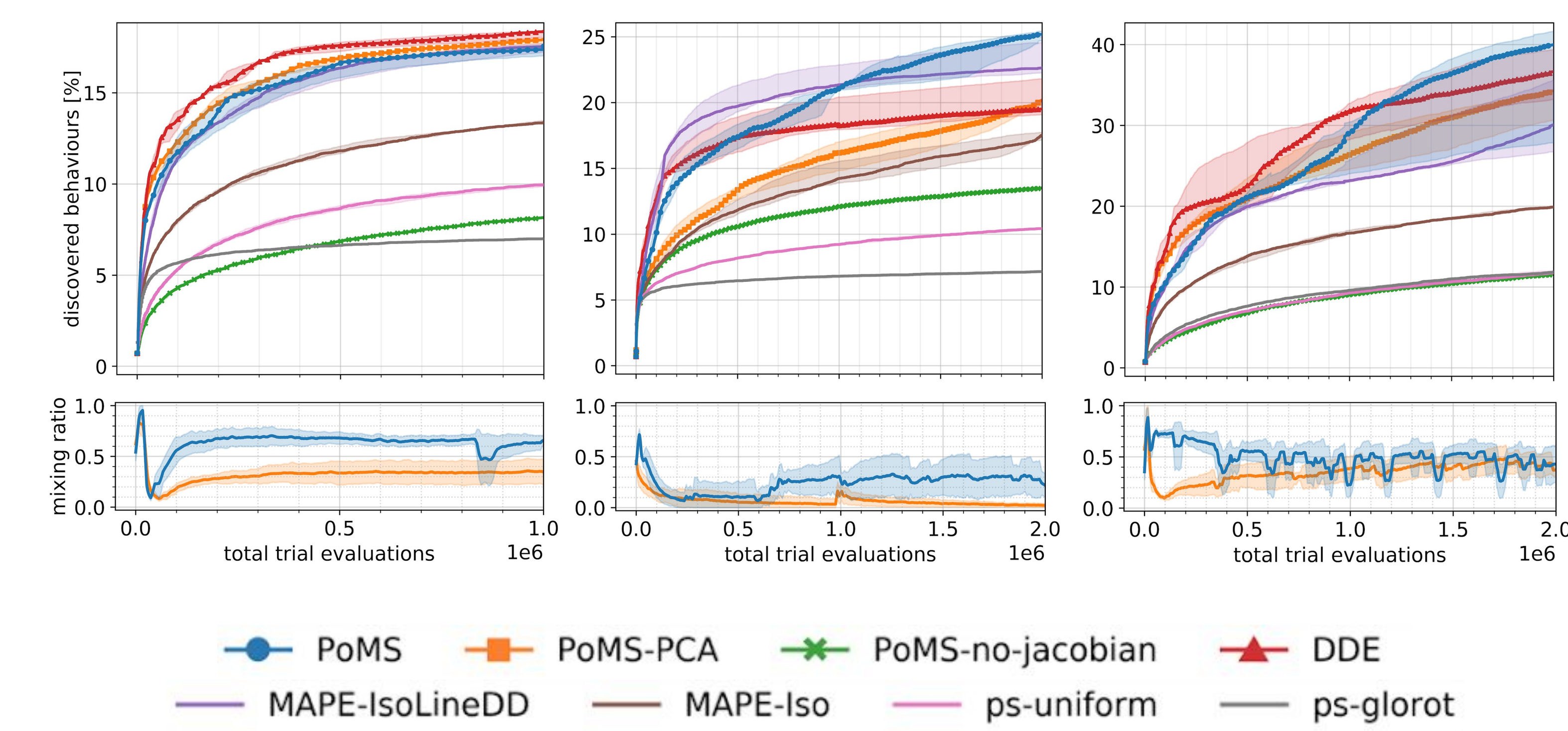


Bipedal Kicker

Normalized observations




Mixed-Scale observations




## Conclusions

- Search in the learned latent space (implicitly or explicitly) is better than in the original high-dimensional parameter space.
- Decoder Jacobian-based latent search scaling is crucial.
- Nonlinear representations are desired over linear representations.

### References

[1] Cully, Antoine et al. (2015). "Robots that can adapt like animals." In: Nature

[2] Ichnowski, Jeffrey el al. (2020). "Cloud-based motion plan computation for power-constrained robots." In: Algorithmic Foundations of Robotics XII

[3] Eysenbach, Benjamin, et al. (2018). "Diversity is All You Need: Learning Skills without a Reward Function". In: ICLR

[4] Hausman, Karol et al. (2018). "Learning an embedding space for transferable robot skills." In: ICLR

[5] Cully, Antoine and Yiannis Demiris (2017). "Quality and diversity optimization: A unifying modular framework." In: IEEE Transactions on Evolutionary Computation