Is Curiosity All You Need? On the Utility of Emergent Behaviours from Curious Exploration

Oliver Groth[1,2] **Markus Wulfmeier**[1] **Giulia Vezzani**[1] **Vibhavari Dasagi**[1,3]
Tim Hertweck[1] **Roland Hafner**[1] **Nicolas Heess**[1] **Martin Riedmiller**[1]
[1]DeepMind, London, United Kingdom
[2]Department of Engineering Science, University of Oxford, United Kingdom
[3]Queensland University of Technology, Brisbane, Australia
ogroth@robots.ox.ac.uk

Abstract

Curiosity-based reward schemes can present powerful exploration mechanisms which facilitate the discovery of solutions for complex, sparse or long-horizon tasks. However, as the agent learns to reach previously unexplored spaces and the objective adapts to reward new areas, many behaviours emerge only to disappear due to being overwritten by the constantly shifting objective. We argue that merely using curiosity for fast environment exploration or as a bonus reward for a specific task does not harness the full potential of this technique and misses useful skills. Instead, we propose to shift the focus towards retaining the behaviours which emerge *during* curiosity-based learning. We posit that these self-discovered behaviours serve as valuable skills in an agent's repertoire to solve related tasks. Our experiments demonstrate the continuous shift in behaviour throughout training and the benefits of a simple policy snapshot method to reuse discovered behaviour for transfer tasks.

1 Introduction

Intrinsic motivation [15, 11, 12, 2, 16] can be a powerful concept to endow an agent with an automated mechanism to continuously explore its environment in the absence of task information. One common way to implement intrinsic motivation is to train a predictive model alongside the agent's policy and use the model's prediction error as a reward signal for the agent encouraging the exploration of previously unfamiliar transitions in the environment - a method also known as *curiosity learning* [13]. Curiosity-esque reward schemes have been used in different ways to facilitate exploration in sparse tasks [9, 4] or pre-train policy networks before fine-tuning them on difficult downstream tasks [17]. In environments where the main task objective is highly correlated with thorough exploration, curiosity-based approaches have also been shown to solve the main task without any additional reward signal [3].

However, in more realistic environments with multiple possible tasks – e.g. in manipulation scenarios where objects could be interacted with or re-arranged in different ways – not only the final behaviour of a curiousity experiment might be of interest, but intermediate behaviours can correlate with solutions to different tasks. Naturally, the constantly changing curiosity objective leads to the emergence of diverse behaviours during training – much akin to the learning process of infants which develop useful skills by playing [6]. Yet, only a fraction of this diversity is ultimately retained in the final policy. Working towards retaining and leveraging emergent behaviour can help to change our perspective of curiosity-based exploration from a tool for task-driven reinforcement learning to a rich continual learning setting [7] in itself which constantly discovers novel behaviour to be leveraged by downstream applications.

NeurIPS 2021 Workshop on Robot Learning: Self-Supervised and Lifelong Learning, Virtual, Virtual

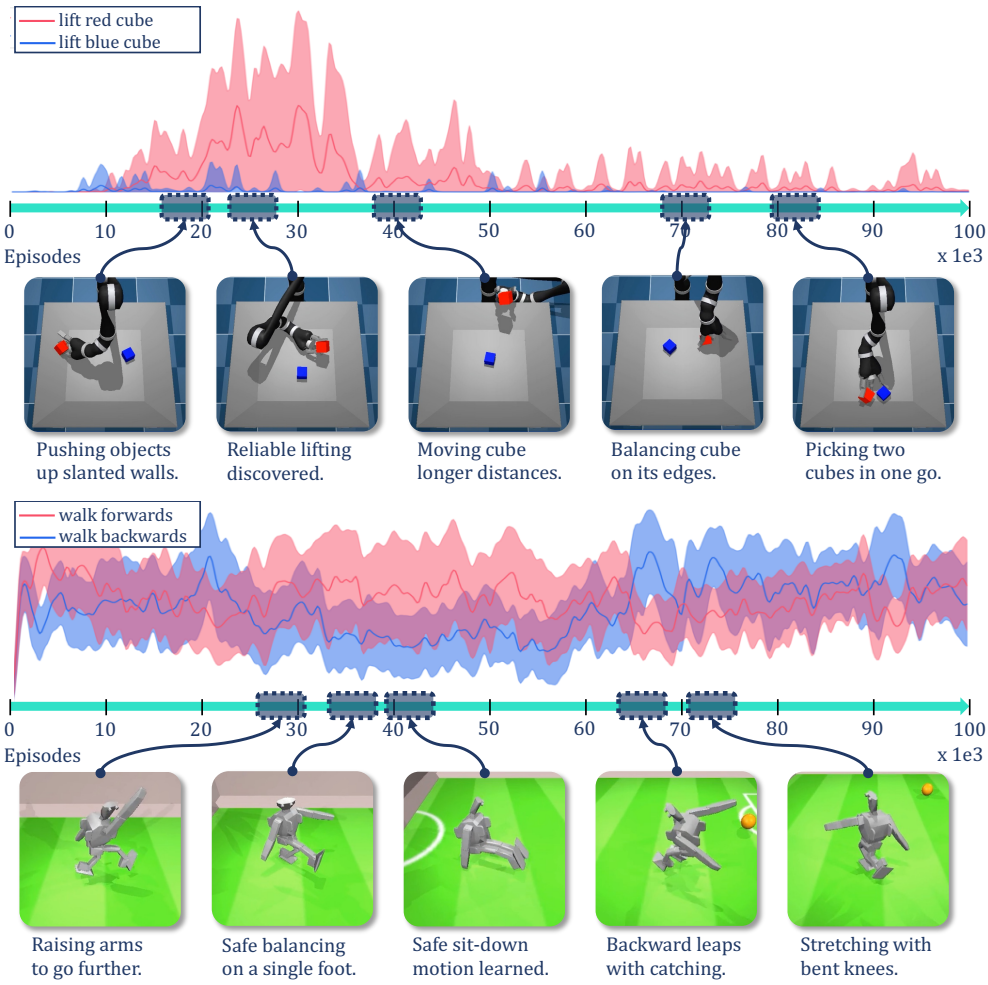

Figure 1: Two example timelines depicting the emergence and disappearance of behaviour while pursuing a constantly changing curiosity objective on a 9-DoF JACO arm (top) and on a 20-DoF OP3 humanoid robot (bottom). Each timeline represents the evolution of behaviour from a single random seed on a single simulated actor. The corresponding line plots represent evaluations which have been performed during the curiosity experiment to gauge the policy's performance in certain, hand-defined tasks like lifting a specific object or walking in a particular direction. A detailed description of the emergent behaviour in this experiment and the corresponding quantitative results is provided in Appendix C.

The application of self-induced curricula of skills holds a tantalising prospect for an agent's ability to solve broad sets of long-horizon tasks when it is able to draw upon potentially useful skills. Related work focusing on *skill diversity* [5, 19, 18] shows that skill spaces induced by diverse self-discovered behaviours can be leveraged for task planning. Other recent works [14, 8, 21] have studied the influence of specificly designed auxiliary tasks on the learning success of complex manipulation policies. A curiosity-based approach could reduce the effort of designing a curriculum of tasks and reward functions by providing a set of self-discovered skills which can be leveraged for efficient exploration and compositional task transfer.

In this paper, we make the following contributions: We introduce *SelMo*, an off-policy realisation of curiosity-based exploration and apply it to two robotic manipulation and locomotion domains in simulation. We demonstrate that our model yields meaningful and diverse emergent behaviour in complex 3D environments. Finally, we emphasise that useful behaviours emerge just to disappear later during training and propose to extend the focus of curiosity-based learning towards the retention of intermediate behaviours which can serve as a valuable skill set for compositional learning methods.

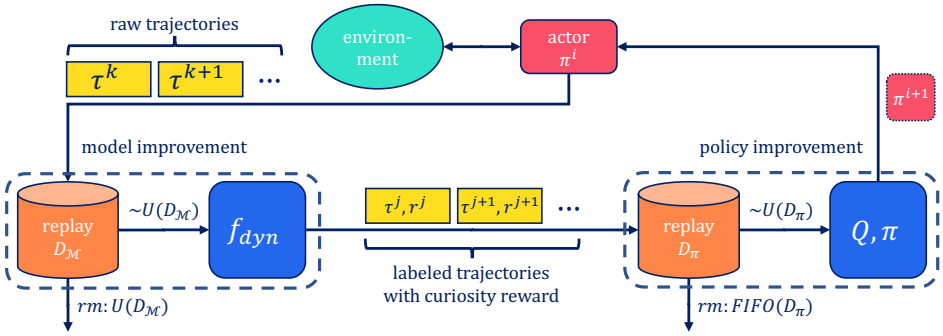

Figure 2: An overview over the SELMO system architecture. The agent collects trajectories $\tau^k, \tau^{k+1}, \ldots$ in the environment using its current policy π^i and stores it in a model replay buffer $D_{\mathcal{M}}$. When $D_{\mathcal{M}}$ is full, trajectories are replaced with a uniform removal strategy. The dynamics model f_{dyn} samples uniformly from this buffer and updates its parameters for forward prediction using *stochastic gradient descent* (SGD). The sampled trajectories $\tau^j, \tau^{j+1}, \ldots$ are then assigned a curiosity reward $r^j, r^{j+1}, \ldots$ scaled by their respective prediction error under the current $f_{dyn}^{(j)}$. The labeled trajectories are passed on to the policy replay buffer D_π which runs a $FIFO$ removal strategy. *Maximum a posteriori policy optimisation* (MPO) [1] is used to fit Q-function and policy π based on uniformly drawn samples from the policy replay. The resulting policy π^{i+1} is then synced back into the actor. Note that both model and policy learning is executed in independent loops.

2 Method

In this section we present SELMO – a self-motivated exploration method which optimises a curiosity objective in an off-policy fashion. Our system is designed around two key components: A forward dynamics model $f_{dyn}: S \times A \mapsto S$ which aims to approximate the state transition function of the environment and a policy $\pi(a_t|s_t)$ which aims to take transitions in the environment for which the prediction error of f_{dyn} is high. Crucially, our setup deviates from recent approaches (e.g. [13, 3, 17]) for curiosity-based rewards in two important aspects: First, we optimise in an off-policy fashion based on a diverse set of experienced transitions in the environment. Second, we employ an approximate but efficient data labelling strategy which only assigns curiosity rewards to trajectories when they are used to update the dynamics model, but refrain from relabeling all trajectories in the policy replay after every model update. We provide a general overview over the whole system in Figure 2.

We describe the environment in which the agent operates as $\mathcal{E} = (S, A, P)$ with state and action spaces S and A as well as a state transition function $s_{t+1} = P(s_t, a_t)$ over discrete time steps. The forward-predictive model f_{dyn} with parameters θ approximates the environment's transition dynamics as:

$$\hat{s}_{t+1} = f_{dyn}(s_t, a_t; \theta) \tag{1}$$

When a transition (s_t, a_t, s_{t+1}) is evaluated by the model, the assigned reward is scaled by the model's current prediction error:

$$r^{(j)}(s_t, a_t, s_{t+1}) = \tanh(\eta_r * (f_{dyn}^{(j)}(s_t, a_t) - s_{t+1})^2) \tag{2}$$

The 'state' of the model is indicated by the number of gradient updates j which have been performed on it so far. We scale the reward via a hyper-parameter η_r and pass it through a $\tanh$ to keep it bounded for the downstream policy learning procedure. When a new batch of data $\mathcal{B}$ is sampled from the model replay, two operations are performed: First, each $\tau \in \mathcal{B}$ is assigned curiosity rewards r^C according to Equation (2). Second, one gradient update is performed by minimising the model's prediction loss:

$$\mathcal{L}_{dyn}^{(j)}(\mathcal{B}) = \sum_{\tau \in \mathcal{B}} \sum_{(s_t, a_t, s_{t+1}) \in \tau} (f_{dyn}^{(j)}(s_t, a_t) - s_{t+1})^2 \tag{3}$$

After one model update, the labeled batch of trajectories $\tilde{\mathcal{B}}$ is stored in the policy replay buffer D_π.

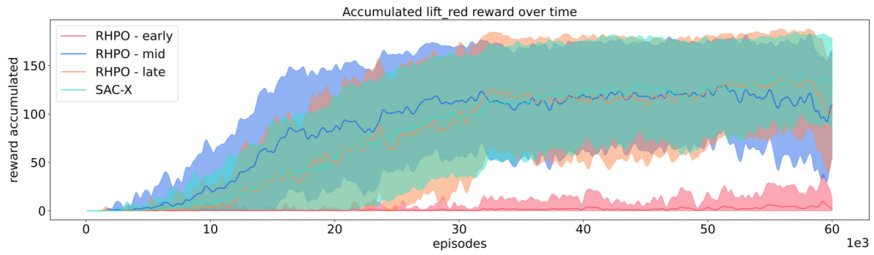

Figure 3: Learning curves for hierarchical skill learning of `lift_red` in the JACO environment. Each RHPO run uses five randomly sampled SELMO policies from the respective intervals as auxiliary skills (cf. Section 3). The SAC-X baseline uses hand-designed auxiliary reward functions `{reach,move}_red`. Mean and standard deviation are plotted for five random seeds for each model and the plot is smoothed with an exponential filter of $\sigma = 1.5$.

3 Utilisation of Emergent Behaviour

As we have highlighted in Figure 1, the constantly evolving curiosity policy develops behaviours which can correlate to the solutions of concrete, human interpretable tasks. In order to retain those diverse behaviours to accelerate the learning of new tasks, we run an experiment where self-discovered behaviour is reused. For instance, an agent which can reuse policies to reach and grasp an object can potentially learn how to lift an object much more quickly. For this experiment, we employ *Regularized Hierarchical Policy Optimization* (RHPO) [20] as this framework allows us to compose multiple policies in a hierarchical manner.

We begin by running a SELMO experiment where we optimise solely for the curiosity objective and save a snapshot of the curiosity policy every 100 episodes. Then, the RHPO experiment samples five SELMO policy snapshots and utilises the behaviour exhibited by them to assist the exploration for the desired downstream task. In the JACO environment (cf. Appendix A) we define the target task to be `lift_red`. While the policy for the target task is randomly initialised, the auxiliary policies are randomly sampled from the SELMO snapshots and kept fixed during the entire RHPO training. We differentiate between three different phases from which the SELMO snapshots are chosen: `early` snapshots have been trained for up to 10K episodes, `mid` snapshots are from the interval [10K, 20K) and `late` snapshots are from [20K, 30K) training episodes. In Figure 3 we compare the learning progress of the downstream task with the sampled SELMO auxiliary skills against a baseline featuring a hand-designed task curriculum in an SAC-X framework [14]. Specifically, the baseline uses auxiliary reward functions which help to `reach` and `move` the `red` cube.

We find that SELMO auxiliaries from the `mid` and `late` exploration periods give the learning of the lifting policy a significant boost which is commensurate with a tuned SAC-X baseline featuring multiple auxiliary rewards which have been hand-designed to facilitate the learning of `lift_red`. This experiment shows that even a simple behaviour retention strategy like policy snapshotting can already provide clear benefits for downstream learning similar to a specifically designed curriculum of reward functions. This is a promising result suggesting that independent curious exploration could be used in lieu of human-engineered task curricula in complex manipulation scenarios.

4 Conclusion

In this paper we have studied the emergence and disappearance of behaviour when optimising an exploration policy for a curiosity objective derived from a forward-predictive dynamics model. To this end, we have presented SELMO, a curiosity-based, off-policy exploration approach and applied it in two continuous control domains: a simulated robotic arm and humanoid robot. We have observed that complex behaviour emerges and disappears in both settings and provided a baseline for the utilisation of self-discovered behaviour in a modular downstream learning scenario. The included experiments only cover a naive form of retaining behaviours throughout an experiment but we believe that the automatic identification and retention of useful emerging behaviour from curious exploration can be a fruitful avenue of future investigation in unsupervised reinforcement learning.

References

[1] Abbas Abdolmaleki, Jost Tobias Springenberg, Yuval Tassa, Remi Munos, Nicolas Heess, and Martin Riedmiller. Maximum a posteriori policy optimisation. *arXiv preprint arXiv:1806.06920*, 2018.

[2] Adrien Baranes and Pierre-Yves Oudeyer. R-iac: Robust intrinsically motivated exploration and active learning. *IEEE Transactions on Autonomous Mental Development*, 1(3):155–169, 2009.

[3] Yuri Burda, Harri Edwards, Deepak Pathak, Amos Storkey, Trevor Darrell, and Alexei A Efros. Large-scale study of curiosity-driven learning. *arXiv preprint arXiv:1808.04355*, 2018.

[4] Yuri Burda, Harrison Edwards, Amos Storkey, and Oleg Klimov. Exploration by random network distillation. *arXiv preprint arXiv:1810.12894*, 2018.

[5] Benjamin Eysenbach, Abhishek Gupta, Julian Ibarz, and Sergey Levine. Diversity is all you need: Learning skills without a reward function. *arXiv preprint arXiv:1802.06070*, 2018.

[6] Nick Haber, Damian Mrowca, Stephanie Wang, Fei-Fei Li, and Daniel L. Yamins. Learning to play with intrinsically-motivated, self-aware agents. In *NeurIPS*, pages 8398–8409, 2018.

[7] Raia Hadsell, Dushyant Rao, Andrei A Rusu, and Razvan Pascanu. Embracing change: Continual learning in deep neural networks. *Trends in cognitive sciences*, 2020.

[8] Tim Hertweck, Martin Riedmiller, Michael Bloesch, Jost Tobias Springenberg, Noah Siegel, Markus Wulfmeier, Roland Hafner, and Nicolas Heess. Simple sensor intentions for exploration. *arXiv preprint arXiv:2005.07541*, 2020.

[9] Rein Houthooft, Xi Chen, Yan Duan, John Schulman, Filip De Turck, and Pieter Abbeel. Vime: Variational information maximizing exploration. *arXiv preprint arXiv:1605.09674*, 2016.

[10] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[11] Pierre-Yves Oudeyer, Frdric Kaplan, and Verena V Hafner. Intrinsic motivation systems for autonomous mental development. *IEEE transactions on evolutionary computation*, 11(2):265–286, 2007.

[12] Pierre-Yves Oudeyer and Frederic Kaplan. What is intrinsic motivation? a typology of computational approaches. *Frontiers in neurorobotics*, 1:6, 2009.

[13] Deepak Pathak, Pulkit Agrawal, Alexei A Efros, and Trevor Darrell. Curiosity-driven exploration by self-supervised prediction. In *International Conference on Machine Learning*, pages 2778–2787. PMLR, 2017.

[14] Martin Riedmiller, Roland Hafner, Thomas Lampe, Michael Neunert, Jonas Degrave, Tom Wiele, Vlad Mnih, Nicolas Heess, and Jost Tobias Springenberg. Learning by playing solving sparse reward tasks from scratch. In *International Conference on Machine Learning*, pages 4344–4353. PMLR, 2018.

[15] Jürgen Schmidhuber. A possibility for implementing curiosity and boredom in model-building neural controllers. In *Proc. of the international conference on simulation of adaptive behavior: From animals to animats*, pages 222–227, 1991.

[16] Jürgen Schmidhuber. Formal theory of creativity, fun, and intrinsic motivation (1990–2010). *IEEE Transactions on Autonomous Mental Development*, 2(3):230–247, 2010.

[17] Ramanan Sekar, Oleh Rybkin, Kostas Daniilidis, Pieter Abbeel, Danijar Hafner, and Deepak Pathak. Planning to explore via self-supervised world models. In *International Conference on Machine Learning*, pages 8583–8592. PMLR, 2020.

[18] Archit Sharma, Michael Ahn, Sergey Levine, Vikash Kumar, Karol Hausman, and Shixiang Gu. Emergent real-world robotic skills via unsupervised off-policy reinforcement learning. *arXiv preprint arXiv:2004.12974*, 2020.

[19] Archit Sharma, Shixiang Gu, Sergey Levine, Vikash Kumar, and Karol Hausman. Dynamics-aware unsupervised discovery of skills. *arXiv preprint arXiv:1907.01657*, 2019.

[20] Markus Wulfmeier, Abbas Abdolmaleki, Roland Hafner, Jost Tobias Springenberg, Michael Neunert, Tim Hertweck, Thomas Lampe, Noah Siegel, Nicolas Heess, and Martin Riedmiller. Compositional transfer in hierarchical reinforcement learning. In *Robotics: Science and Systems*. Robotics: Science and Systems Foundation, 2020.

[21] Markus Wulfmeier, Arunkumar Byravan, Tim Hertweck, Irina Higgins, Ankush Gupta, Tejas Kulkarni, Malcolm Reynolds, Denis Teplyashin, Roland Hafner, Thomas Lampe, et al. Representation matters: Improving perception and exploration for robotics. *arXiv preprint arXiv:2011.01758*, 2020.

A Simulation Environments

In this section we provide details about the two robotic simulation domains used in our experiments.

A.1 JACO Manipulation Environment

This environment is designed to study manipulation tasks like object lifting and stacking with a robotic arm (cf. Figure 4). The state observation space consists of 72 dimensions: 24 features are used to represent the robot's proprioception as well as the state of each object. The action space spans 9 dimensions: 6 concerning the arm and 3 concerning the three-point gripper. A detailed description of the environment is provided in Table 1. Interactions with the two objects (`O1` = red cube, `O2` = blue cube) are evaluated using the sparse reward functions `reach_{red,blue}` and `lift_{red,blue}`. Each episode in this environment lasts 20 seconds or 400 control timesteps.

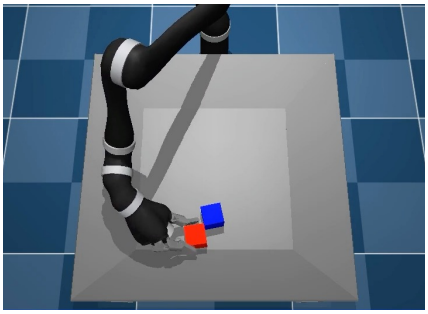

Figure 4: The JACO manipulation environment. The 6 DoF robot arm with a 3 DoF gripper can interact with multiple same-sized cubes in its workspace.

FEATURE	DIMENSION
`arm/joints_pos`	6
`arm/joints_vel`	6
`arm/hand/finger_joints_pos`	3
`arm/hand/finger_joints_vel`	3
`arm/hand/fingertip_sensors`	3
`arm/hand/pinch_site_pos`	3
proprioception	$\sum = 24$
`O<i>/rel_pos_wrt_tcp`	3
`O<i>/pos`	3
`O<i>/orientation`	4
`O<i>/linvel`	3
`O<i>/angvel`	3
`O<i>/proptype`	5
`O<i>/dimensions`	3
perception per object <i>	$\sum = 24$
`arm`	6
`hand`	3
action space	$\sum = 9$

Table 1: State and action space semantics of JACO environment.

A.2 OP3 Locomotion Environment

This environment is designed to study locomotion with a humanoid robot (cf. Figure 5). The state observation consists of 49 proprioceptive features. The 20-dimensional action space controls the orientations of the robot's head, ankle, elbow, hip, knee and shoulder. Actions passed to the robot are smoothed with an exponential filter to reduce motion jerk. A detailed description of the environment is provided in Table 2. The robot always spawns in a standing, upright position. Locomotion is evaluated by the dense reward functions `walk_{forward,backward}` for forward and backward walking gaits respectively. Each episode in this environment lasts 10 seconds or 200 control timesteps. If the robot's hip angle deviates more than $15°$ from an upright orientation, the simulation terminates early effectively preventing the robot from falling over.

B Model and Training Details

Across all experiments with the SelMo architecture, we consistently use the following hyper-parameters and model architectures. Each SelMo experiment is run with a single actor for $N = 1e5$ episodes.

Figure 5: The OP3 locomotion environment. The 20 DoF humanoid robot can walk around on a plane. The episode terminates early, if the OP3 is about to fall over.

FEATURE	DIMENSION
`walker/joints/pos`	20
`walker/imu/linear_acc`	3
`walker/imu/angular_vel`	3
`walker/imu/gravity`	3
`scaled/action_filter/state`	20
proprioception	$\sum = 49$
`head_{pan,tilt}`	2
`{l,r}_ankle_{pitch,roll}`	4
`{l,r}_elbow`	2
`{l,r}_hip_{pitch,roll,yaw}`	6
`{l,r}_knee`	2
`{l,r}_shoulder_{pitch,roll}`	4
action space	$\sum = 20$

Table 2: State and action space semantics of OP3 environment.

Dynamics Model f_{dyn} The world model is implemented as a *multi-layer perceptron* (MLP) with the following layer sizes and activation functions: `[FC(256), elu(·), FC(256), elu(·), FC(size_state)]` For each environment, `size_state` is the sum of the dimensions for proprioception and perception (cf. Appendix A). The world model is optimised using Adam [10] with a learning rate of $\eta_{\mathcal{M}} = 3e-4$.

Policy Both policy π and critic Q are implemented as two independent MLPs with the following layer sizes and activation functions:

- Q: `[tanh(·), FC(512), elu(·), FC(512), elu(·), FC(256), FC(1)]`
- π: `[FC(256), elu(·), FC(256), elu(·), FC(128), FC(size_action)]`

The `size_action` is different for each environment (cf. Appendix A). The policy is optimised using Adam [10] with a learning rate of $\eta_\pi = 3e-4$. The reward scale is set to $\eta_r = 10.0$ across all experiments.

Replays The replays $D_{\mathcal{M}}$ and D_π store trajectories with a length of $T = 50$ transitions. The buffer sizes used are $|D_{\mathcal{M}}| = |D_\pi| = 5e4$ and each trajectory in the buffers can be sampled at most $n_{max} = m_{max} = 32$ times (cf. Section 2). The batch size of samples drawn from the replays is set to $B = 64$.

C Emergence of Behaviour

In this section we present an analysis of the behaviour which emerges in our two simulated domains as depicted in Figure 1. The JACO domain features a 9 DoF robotic arm and two cubes; the OP3 domain features a 20 DoF humanoid robot. For this experiment, we run the SelMo learning loop (cf. Figure 2) with a single actor for 100K episodes on each of the environments. During training, the agent solely optimises its curiosity objective which is defined by Equation (2). The visual inspection of the experiments reveal that in both cases, diverse sets of human-interpretable behaviour emerge consistently and are exhibited by the agent for extended periods of time before the ever-changing curiosity reward function shifts the learning towards a new behaviour. Below, we describe the observed behaviours in each domain in greater detail.

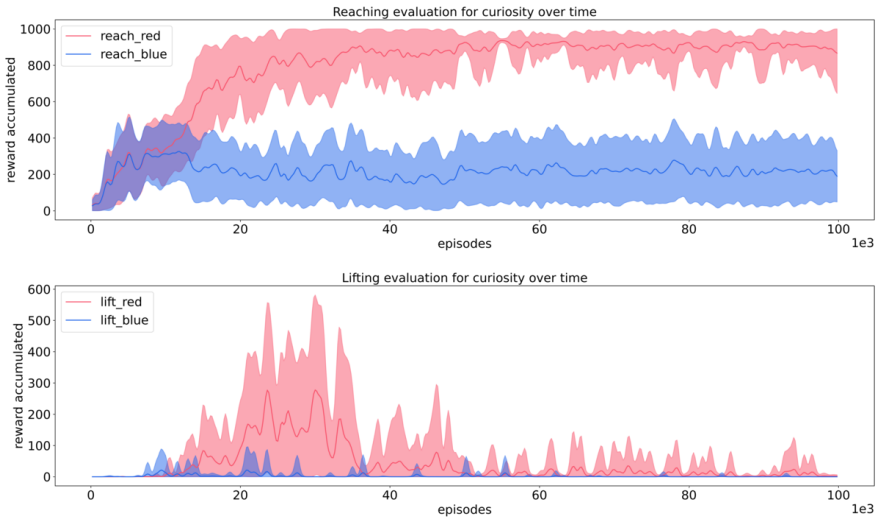

Figure 6: Manipulation task evaluation in the JACO environment over the lifetime of one experiment while the agent is only trained on the curiosity objective (cf. Equation (2)). A snapshot of the curiosity policy is saved every 100 episodes and evaluated on reaching and lifting the red and blue cubes respectively. Mean and standard deviation are plotted over 20 evaluation runs per policy snapshot and the plot is smoothed with an exponential filter of $\sigma = 1.5$.

C.1 Emergent Manipulation Behaviour on JACO

A qualitative example timeline of emerging behaviours on the JACO arm is depicted in Figure 1 (top) and supplemented by a plot evaluating reaching and lifting behaviour during the run in Figure 6. We find that the agent is very quickly driven towards both cubes with equal attention and starts interacting with them by pushing them around. Soon thereafter, it discovers that pushing them up the slanted walls of the bin facilitates picking them up before it stably latches onto a mode in which it prefers manipulating the red cube over the blue one after approximately 15K episodes. This also coincides with a first period of sustained lifting of the red cube. We hypothesise that the discovery of lifting said cube reinforces the interaction with it as it opens up a new dimension along which model prediction error can be rewarded: the height of the object. After about 25K episodes, it has learned to pick up an object reliably even without the help of the slopes.

After approximately 40K episodes, the curiosity objective pushes the agent to deliberately take objects outside of the workspace and perform pick-and-place operations which move a cube over a long distance but at a lower height. Interestingly, the policy does not degenerate into extreme behaviours like spinning motions which have been observed in related work [17] but stays focused on the objects and keeps exploring their physical properties. For instance, at around 70K episodes, the policy investigates the stability of cube poses in a targeted way by balancing them on their edges and corners. Finally, after about 80K episodes, it starts exploring the possibilities of moving both cubes simultaneously.

C.2 Emergent Locomotion Behaviour on OP3

Similar to the JACO arm, we present a timeline of emerging behaviour on the OP3 in Figure 1 (bottom) and a corresponding evaluation of locomotion behaviour in Figure 7. Unsurprisingly, the agent spends roughly the first 2K episodes – indicated by the steep rise in walking rewards – just on learning a sense of balance because an episode is terminated early when the torso constraint (cf. Appendix A) is violated and the agent is about to fall. This also corresponds to maximising the experienced episode length because this increases the chances of further increasing the accumulated reward. This finding is in line with earlier work [13] which has shown that the avoidance of a 'death' event is a natural by-product of curiosity-driven learning with a positive reward and favourably shapes the emerging policy.

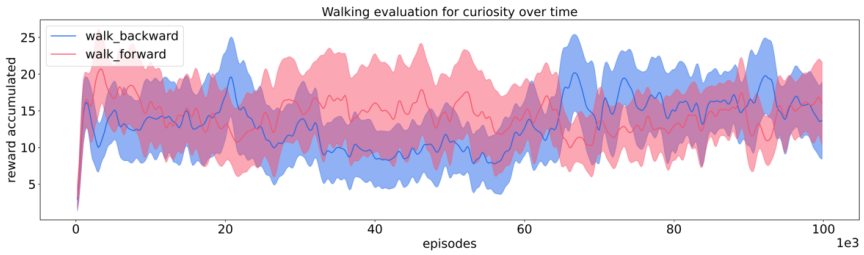

Figure 7: Task evaluation in the OP3 environment over the lifetime of one experiment while the agent is only trained on the curiosity objective (cf. Equation (2)). A snapshot of the curiosity policy is saved every 100 episodes and evaluated on locomotion (`walk_{forward,backward}`) tasks. Mean and standard deviation are plotted over 20 evaluation runs per policy snapshot and the plot is smoothed with an exponential filter of $\sigma = 1.5$.

Once the agent has learned to stay upright, it slowly starts to develop basic locomotion in the form of stumbling forwards and backwards with only small foot lifting heights which is also reflected in minor oscillations during the evaluation of the walking rewards in Figure 7. Interestingly, after around 30K episodes, the agent has discovered to swing its arms to take bigger steps. This is most impressively first demonstrated after nearly 40K episodes when the agent balances on one foot while stretching out the other leg using its arms for counterbalancing moves. Using the arms also opens new avenues for exploration. Approximately 40K episodes into training, the agent has learned to catch itself when falling backwards. This leads to the discovery of a sit-down behaviour which does not violate the environment's torso constraints. After the agent has explored various 'ground exercises' it switches back to walking gaits at around 55K episodes. Then, the whole body movement has become considerably more nimble and its movement repertoire now features quick turns, stumbling reflexes and even safe backward leaps. After about 70K episodes the agent starts revisiting earlier behaviour, e.g. the balancing skill, but keeps adding variations to it like knee-bending or stretching.