



UvA-DARE (Digital Academic Repository)

A Performance-Based Start State Curriculum Framework for Reinforcement Learning

Wöhlke, J.; Schmitt, F.; van Hoof, H.

Publication date

2020

Document Version

Final published version

Published in

AAMAS'20

License

Article 25fa Dutch Copyright Act (<https://www.openaccess.nl/en/policies/open-access-in-dutch-copyright-law-taverne-amendment>)

[Link to publication](#)

Citation for published version (APA):

Wöhlke, J., Schmitt, F., & van Hoof, H. (2020). A Performance-Based Start State Curriculum Framework for Reinforcement Learning. In *AAMAS'20: proceedings of the 19th International Conference on Autonomous Agents and MultiAgent Systems : May 9-13, 2020, Auckland, New Zealand* (pp. 1503-1511). International Foundation for Autonomous Agents and Multiagent Systems. <https://dl.acm.org/doi/10.5555/3398761.3398934>

General rights

It is not permitted to download or to forward/distribute the text or part of it without the consent of the author(s) and/or copyright holder(s), other than for strictly personal, individual use, unless the work is under an open content license (like Creative Commons).

Disclaimer/Complaints regulations

If you believe that digital publication of certain material infringes any of your rights or (privacy) interests, please let the Library know, stating your reasons. In case of a legitimate complaint, the Library will make the material inaccessible and/or remove it from the website. Please Ask the Library: <https://uba.uva.nl/en/contact>, or a letter to: Library of the University of Amsterdam, Secretariat, Singel 425, 1012 WP Amsterdam, The Netherlands. You will be contacted as soon as possible.

UvA-DARE is a service provided by the library of the University of Amsterdam (<https://dare.uva.nl>)

A Performance-Based Start State Curriculum Framework for Reinforcement Learning

Jan Wöhlke
 Bosch Center for Artificial
 Intelligence
 Renningen, Germany
 University of Amsterdam
 JanGuenter.Woehlke@de.bosch.com

Felix Schmitt
 Bosch Center for Artificial
 Intelligence
 Renningen, Germany
 Felix.Schmitt@de.bosch.com

Herke van Hoof
 UvA-Bosch Delta Lab
 University of Amsterdam
 Amsterdam, Netherlands
 h.c.vanhoof@uva.nl

ABSTRACT

Sparse reward problems present a challenge for reinforcement learning (RL) agents. Previous work has shown that choosing start states according to a curriculum can significantly improve the learning performance. We observe that many existing curriculum generation algorithms rely on two key components: Performance measure estimation and a start selection policy. Therefore, we propose a unifying framework for performance-based start state curricula in RL, which allows to analyze and compare the performance influence of the two key components. Furthermore, a new start state selection policy using spatial performance measure gradients is introduced. We conduct extensive empirical evaluations to compare performance-based start state curricula and investigate the influence of performance measure model choice and estimation. Benchmarking on difficult robotic navigation tasks and a high-dimensional robotic manipulation task, we demonstrate state-of-the-art performance of our novel spatial gradient curriculum.

KEYWORDS

Reinforcement learning; Learning agent capabilities; Machine learning for robotics

ACM Reference Format:

Jan Wöhlke, Felix Schmitt, and Herke van Hoof. 2020. A Performance-Based Start State Curriculum Framework for Reinforcement Learning. In *Proc. of the 19th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2020)*, Auckland, New Zealand, May 9–13, 2020, IFAAMAS, 9 pages.

1 INTRODUCTION

Although reinforcement learning (RL) has shown remarkable success in playing games with super-human performance [13, 22, 36] as well as mastering robotic locomotion [35] or manipulation tasks [1, 21], sparse reward problems still present a challenge. Imagine an autonomous agent, which has to solve a spatial navigation task. The most simple and most precise objective definition for this task is to only reward the agent once it has reached the goal. However, this sparse reward definition comes at the cost of data-efficiency. The agent needs to carry out many rollouts in order to experience enough positively rewarded rollouts to improve its policy. Adding reward components by prior knowledge or using shaping leads to denser reward signals but carries a risk of resulting in undesired

behaviors of the agent [28]. For example, a straight-line distance reward signal may misguide an agent in a navigation task since it does not consider obstacles like walls in a maze-like environment. Ng et al. proposed potential-based reward shaping [25] to leave the optimal policy invariant. However, a suitable potential function might not always be obviously available: Reasonably rewarding intermediate configurations in real world robotic manipulation tasks may be difficult, although the desired target configuration is clear.

Previous works have suggested a variety of ideas to improve data-efficiency in sparse reward reinforcement learning [26]: Intrinsic motivation [5, 12, 15, 27, 32], diversity [7, 14], return decomposition [2], auxiliary tasks [16, 30], use of demonstrations [29, 31], or curriculum learning [8, 9, 19, 30].

Curriculum learning [6] is a general concept in machine learning. It is motivated by the way humans or animals learn. The idea is to accelerate learning by starting with simple problems and increasing the difficulty according to the learner’s capabilities. In RL, curricula can improve data-efficiency in sparse reward settings by deciding which “context” to train on next, with contexts being start states [9], goal states [8], or tasks [19, 30]. It is of specific interest not to hand-design but automatically generate the sequence of contexts.

We focus on generating curricula of starting states for sparse reward goal-based RL settings. This problem has already been studied in [9]. However, a unifying framework to gather the existing performance-based curricula from the supervised and reinforcement learning literature is missing. Furthermore, it is an open question how the different context selection strategies compare with respect to start state selection in RL using the same benchmark tasks and agent performance models. Besides that, existing start state curriculum generation algorithms make implicit assumptions on agent-environment dynamics or require additional rollouts [9].

In this work, we investigate performance-based start state curriculum generation using on-policy policy gradient methods, which are state-of-the-art in robotic tasks [21, 35]. Start state curricula are particularly appropriate for on-policy methods because changing the context has a direct influence on the current learning performance. Nevertheless, the presented curriculum ideas are potentially more broadly applicable in the off-policy setting. Our main contributions are:

- Introducing a formal framework for performance-based start state curricula in RL that consists of a performance measure model and a start state selection policy.
- Proposing a new, intuitive start state selection policy with state-of-the-art performance across different problem domains that builds on spatial performance measure gradients.

2 RELATED WORK

Curriculum learning has been shown to accelerate learning progress in supervised learning [6, 11, 17] as well as in reinforcement learning settings [8, 9, 19, 30]. A very general framework to propose increasingly difficult problems can be found in [33, 37]. The SAGGRIAC framework [4, 5] pursues a similar direction by generating “developmental trajectories” of increasingly difficult tasks by making use of intrinsic motivation concepts.

Even before the term was established, curriculum learning ideas have been considered in RL. Asada et al. [3] proposed with “Learning from Easy Missions (LEM)” an algorithm that schedules increasingly difficult start states. Thereby, they assume an “axis” along which states can be ordered by difficulty using prior knowledge about the problem. Kakade and Langford [18] also consider the possibility of modifying the start state distribution in RL.

Recent approaches automatically generate curricula for different types of contexts such as tasks [19, 30], start states [9], or goal states [8]. A common pattern is that Monte Carlo returns of past rollouts are used for determining how suitable a context is for the agent’s current learning process. To determine the next context, a continuously re-estimated Boltzmann distribution [19, 30], a heuristic that finds new contexts near previously “good” ones [9], or a trained generator that outputs suitable contexts [8] were used.

Klink et al. [19] adapt the concept of self-paced curriculum learning [17, 20] to relative entropy policy search by allowing the agent to control the intermediate task distribution with respect to its capabilities, which are represented as a value function. The optimization of the intermediate task distributions performs a trade-off between intermediate reward maximization and shifting the task distribution towards the desired target distribution.

The asymmetric self-play algorithm introduced in [38] puts two agents, Alice and Bob, into competition to implicitly generate a curriculum. During the self-supervised training, Alice starts in the goal configuration of the MDP and takes actions until it takes the STOP action. Then it is Bob’s turn to reset the MDP in its goal configuration starting from Alice’ final configuration. The policies of Alice and Bob are updated with intrinsic rewards. During the rest of the training, Bob trains on the original MDP.

Narvekar et al. [23, 24] introduce curriculum MDPs (cMDPs) to model the process of curriculum generation as an MDP itself. This way, the context can vary arbitrarily. However, the set of possible source tasks must be provided in advance. Furthermore, the recursive Monte-Carlo algorithm for curriculum generation in cMDPs [23] relies on hand-designed heuristics and lacks scalability as well as data-efficiency by requiring to train on every source task for some time in order to gauge it.

Several works have studied the important case of adaptively choosing starting states as a curriculum in the RL setting [9, 29, 31]. In [31] and [29] start state curricula are generated from a single demonstration. This assumes the availability of expert demonstrations, which is a strong assumption we want to omit.

Closest to our work is the “Reverse Curriculum” generation approach [9]. New start states are found nearby “good states” that have an intermediate probability of reaching the goal. Any state visited during random walks (“Brownian motion”) starting from these “good states” is a potential new start. This heuristic start state

generation implicitly assumes symmetry in the agent-environment dynamics in the way that reaching A from B and B from A is similarly difficult. Furthermore, additional rollouts are required for the random walks and hyper-parameters have to be tuned. Similar to our work, it is assumed that the agent can start in an arbitrary state of the MDP although possibly only a sub-space of the state space might be freely chosen.

In our work, we introduce a unifying framework for start state selection based on a Monte-Carlo return-based performance measure model. Our framework accommodates existing curriculum generation methods [9, 11, 19] and allows comparing them. Furthermore, we introduce a novel start state selection policy, that is based on a spatial gradient of the performance measure. This start state policy tends to select start states at the boundary of the agent’s capabilities. It neither requires additional policy rollouts nor does it assume symmetry in the agent-environment dynamics.

3 BACKGROUND AND PROBLEM STATEMENT

In this work, we consider sequential decision making problems that are modeled by means of a discrete-time finite-horizon Markov decision process (MDP) $\mathcal{M} = (\mathcal{S}, \mathcal{A}, \mathcal{P}, r, \gamma, T)$ with the set of states \mathcal{S} , the set of actions \mathcal{A} , the state transition dynamics $\mathcal{P} : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \rightarrow [0, 1]$, the reward function $r(s, a) : \mathcal{S} \times \mathcal{A} \rightarrow \mathbb{R}$, the discount factor γ , and the time horizon T . Furthermore, there is an initial state distribution $p_0 : \mathcal{S} \rightarrow [0, 1]$ which might be controlled by a curriculum generation algorithm.

We address the standard setting of policy gradient methods for RL which is the task of finding the parameters θ of a stochastic policy $\pi_\theta : \mathcal{S} \times \mathcal{A} \rightarrow [0, 1]$ that maximize the expected accumulated discounted rewards

$$\max_{\theta} \mathbb{E} \left[\sum_{t=0}^T \gamma^t r(s_t, a_t) \mid \mathcal{P}, \pi_\theta, p_0 \right]. \quad (1)$$

For this purpose, we define the value function $V^{\pi_\theta}(s)$ of the policy π_θ as

$$V^{\pi_\theta}(s) := \mathbb{E} \left[\sum_{t=0}^T \gamma^t r(s_t, a_t) \mid s_0 = s, \mathcal{P}, \pi_\theta \right]. \quad (2)$$

Goal-based Markov Decision Processes: Sparse reward tasks are often studied in goal-based MDPs [1, 9, 30]. Here, the objective is to reach a pre-defined terminal goal state $g \in \mathcal{S}$ for all feasible start states within the time horizon T . These MDPs are also important in the field of robotics, for example for modeling robotic navigation problems [9] or object reaching and manipulation tasks [1, 30].

Formally, the goal is reached once $d(s, g) \leq \epsilon$ for some distance measure $d : \mathcal{S} \times \mathcal{S} \rightarrow \mathbb{R}_+$ and a tolerance $\epsilon \in \mathbb{R}_+$. We define the goal-reaching-probability $\Omega(\pi_\theta, p_0)$ with respect to the policy π_θ and the start-state distribution p_0 as

$$\Omega(\pi_\theta, p_0) = \mathbb{E}_{s \sim p_0} \left[\mathbb{1}_{\exists t \leq T: d(s_t, g) \leq \epsilon} \mid s_0 = s, \mathcal{P}, \pi_\theta \right]. \quad (3)$$

To approximately solve the goal-based MDP, we search for policy parameters θ that maximize the goal-reaching probability under the uniform distribution over feasible states $\Omega(\pi_\theta, \mathcal{U}(\mathcal{S}))$. This can be cast in the previously defined formalism of (1) by defining a goal-based binary reward function $r_g(s, a) = \mathbb{1}_{d(s, g) \leq \epsilon}$ and assuming a discount of $\gamma = 1$.

Start State Curricula: Curriculum learning can be applied to RL for selecting the start states of the policy rollouts. This way, the start state distribution may depend on the training iteration i : $p_{0,i}$. The objective of start state curriculum generation is to improve the learning progress over the course of the training as well as the final goal-reaching-probability compared to training with the uniform start state distribution $p_0 = \mathcal{U}(\mathcal{S})$ by adapting $p_{0,i}$.

4 PERFORMANCE-BASED START STATE CURRICULUM FRAMEWORK

In the following, we propose a general framework for performance-based start state curriculum generation in reinforcement learning. The framework consists of two key components: A state-dependent performance measure (PM) $J(\pi_{\theta_i}, s)$ that assesses the agent’s capabilities and a start state selection policy, which is a distribution over start states that depends on the PM values.

Similar to earlier work on start state curriculum generation [9], we assume that the agent can reset to an arbitrary start configuration at any point in time and allow for the case that only the start state component \bar{s} in a sub-space $\bar{\mathcal{S}} \subset \mathcal{S}$ is freely choosable¹. For example, in robotic navigation tasks, the agent can start at any feasible x -, y -position whereas the initial velocity is always zero.

Given estimates of the performance measure $J(\pi_{\theta_i}, s)$ over the entire state space \mathcal{S} for potentially the entire history of training iterations \mathcal{H}_i , the start state selection policy π^{s_0} chooses start states s_0 for the environment rollouts during the RL training:

$$\pi^{s_0} \left(s_0 \mid \left(J(\pi_{\theta_j}, s) \right)_{s \in \mathcal{S}, j \in \mathcal{H}_i} \right). \quad (4)$$

As a result, the probability of sampling a state s as start state $s_{0,i}$ in iteration i is proportional to the value of a function G that is applied to the performance measure $J(\pi_{\theta_i}, s)$:

$$P_J(s_{0,i} = s) \propto G \left(\left(J(\pi_{\theta_j}, s) \right)_{s \in \mathcal{S}, j \in \mathcal{H}_i} \right). \quad (5)$$

This start state curriculum framework can accommodate existing curriculum generation approaches like [9], [19], or [11]. It furthermore allows us to formulate a novel start state curriculum generation algorithm utilizing spatial performance measure gradients.

Possible performance measure choices are discussed in Sec. 4.1. While Sec. 4.2 shows how existing curriculum generation approaches fit our framework, section 4.3 introduces our spatial gradient start state curriculum generation approach. Details regarding the algorithmic implementation of our framework are given in Sec. 4.4.

4.1 Performance Measure

In reinforcement learning, the typical measure of performance of a policy starting in a certain state s is its expected return, which is equivalent to the value-function $V^{\pi_{\theta}}(s)$ at state s (see Eq. 2). In the setting of goal-based MDPs, we can also express this performance measure using the goal-reaching probability:

$$J(\pi_{\theta_i}, s) = V^{\pi_{\theta_i}}(s) = \Omega(\pi_{\theta_i}, \mathbb{I}_{s_0=s}). \quad (6)$$

In contrast to supervised learning, the reinforcement learning performance measure $J(\pi_{\theta_i}, s)$ of the policy π_{θ_j} with respect to the

¹To improve readability, the equations in this section are stated with s and \mathcal{S} instead of \bar{s} and $\bar{\mathcal{S}}$, wherever it is sensible. A notable exception are implementations.

reward function $r(s, a)$ given a start state s is not readily available for all states $s \in \mathcal{S}$. Instead, the performance must be estimated by means of policy evaluation.

4.1.1 Performance Measure Estimation. A simple approach for estimating the performance measure is to roll out the policy π_{θ} several times starting in the states of interest and averaging the Monte Carlo returns of the resulting trajectories (see [9, 30]). However, this requires many interactions with the environment. For this reason, RL algorithms usually learn an estimate of the return of a policy given a state. Historically, tabular representations of the value function were used [39]. With increasingly large state spaces, function approximation techniques keep learning manageable [40]. Recently, neural networks have become a popular tool to model the complex shape of value functions in difficult tasks [10, 35].

4.1.2 Performance Measure Models. In principle, all models and architectures employed for expected return estimation can be employed as performance measure (PM) model. For our robotic evaluation tasks, we specifically use two different PM models $\hat{V}_{\phi}(\bar{s})$:

- **Performance Measure Map (PMM):** For robotic navigation tasks, we use a tabular model as experimentally justified in Sec. 5.2.1. We discretize the controllable 2D x -, y -position sub-space by applying a uniform grid. The performance measure estimate of a specific cell is the average of the undiscounted returns of all policy rollout trajectory states of the last l training iterations that fall into this cell. A major drawback of this model is that it suffers the curse of dimensionality, not scaling to higher-dimensional subspaces $\bar{\mathcal{S}}$ as e.g. encountered in robotic manipulation tasks.
- **Performance Measure Network (PMN):** This parametric neural network PM model scales better to higher dimensions. It is trained similar to a value function critic network in actor-critic RL but learns a model of the undiscounted returns given only the controllable component \bar{s} of the state as input.

The terms PMM and PMN indicate that these return prediction models are specifically estimated for the start state curricula. For policy optimization, an additional value function model using the full state and potentially employing discounting might be estimated.

4.2 Adapting Existing Curriculum Generation Approaches

In this section, we show how the previously introduced framework accommodates existing curriculum generation methods.

4.2.1 Good Starts/States (GS). Florensa et al. [9] propose to ideally draw start states uniformly from a set of “good” states that have a probability for reaching the goal that is neither too low nor too high. This serves the goal of collecting a set of rollouts with a good balance between trajectories that reached the goal and trajectories that failed to do so. In terms of start state sampling probability, the selection mechanism can be written as

$$P_J(s_0 = s) \propto \mathbb{I}_{a < J(\pi_{\theta}, s) < b}. \quad (7)$$

with a and b denoting the lower and upper threshold on the performance measure, respectively.

4.2.2 *Self-Paced Contextual RL (SPCRL)*. The RL adaption of self-paced curriculum learning [17] of Klink et al. [19] can be represented in our framework of start state curricula for RL:

$$P_J(s_0 = s) \propto e^{\frac{1}{\eta} J(\pi_{\theta}, s)}. \quad (8)$$

The temperature parameter η is increased towards infinity over the course of the training in order to recover the original objective of uniform start sampling, in the limit.

4.2.3 *Temporal Prediction Gain (TPG)*. Graves et al. [11] investigated curriculum generation for supervised learning based on maximizing the log-likelihood $L(x, \theta)$. In their study, the gain in prediction performance $L(x, \theta') - L(x, \theta)$, where θ and θ' are the network parameters before and after training on x , turned out to be the best criterion to select the next training sample x . This curriculum generation approach translates to the RL setting by choosing starting states proportional to the temporal improvement of the performance measure $J(\pi_{\theta_i}, s)$ across the last l training iterations with a subsequently applied Boltzmann distribution:

$$P_J(s_0 = s) \propto e^{\frac{1}{\eta} (J(\pi_{\theta_i}, s) - J(\pi_{\theta_{i-l}}, s))}. \quad (9)$$

4.3 Spatial Gradient Curriculum

We introduce a novel criterion for selecting start states, which we term the spatial gradient (SG) curriculum. First, we assume that:

- The sub-space \bar{S} is a Euclidean vector space.
- States that are close in the Euclidean norm are easily reachable from one another.

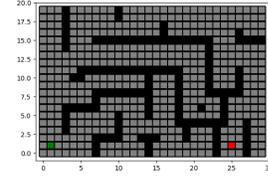
Given the assumptions, we propose to sample start states proportional to the Euclidean norm of the gradient of the performance measure $J(\pi_{\theta}, s)$ with respect to the state s :

$$P_J(s_0 = s) \propto \|\nabla_s J(\pi_{\theta}, s)\|_2 \quad (10)$$

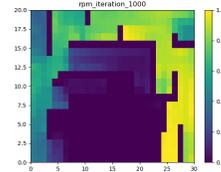
The SG curriculum can be motivated by spatial navigation problems: Imagine two neighboring states whereby the probability to reach the goal state if starting in the state is high for one of the states and low for the other. Once the policy learns how to reach the state with the high probability from the state with the low probability, the latter state will also have a high goal-reaching probability. This procedure casts the task of improving the global goal-reaching capabilities of the policy to a much easier local improvement task.

The spatial gradient exhibits large values at states that lie close to the boundary of the policy’s goal-reaching capabilities. An exemplary visualization is given in Fig. 1: Figure 1b shows the goal-reaching probabilities for every state (tile) after training an RL agent using the SG curriculum for 1000 iterations in the discrete gridworld depicted in Fig. 1a. The corresponding spatial gradient values for every state are shown in Fig. 1c.

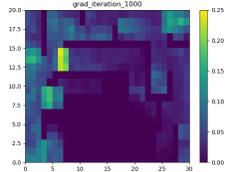
While the assumptions of the SG curriculum may seem restrictive, the empirical evaluation in Sec. 5 demonstrates that the approach suits a variety of robotic tasks, achieving state-of-the-art performance. Our method seems to work well in practice if one can easily transition between pairs of states that are within a small distance of Δ from each other. For our robotic insertion task, this is the case for the angle components of the non-Euclidean joint space.



(a) Gridworld Layout



(b) PMM, Iter=1000



(c) SG, Iter=1000

Figure 1: PMM and Corresponding Spatial Gradients

4.4 Algorithmic Implementation Details

Alg. 1 shows the combination of our framework with on-policy RL.

Algorithm 1: PM Based Start State Selection for On-Policy RL

Input: On-policy RL algorithm \mathbf{A} , initial policy π_{θ_0} , start state selection policy π^{s_0} , model of performance measure (PM) $\hat{V}_{\phi}(\bar{s})$, empty set of trajectory data D

```

1 for  $i = 1$  to  $i_{max}$  do
2   Calculate start state selection policy based on PM model
    $\pi^{s_0, i}(s_0, i | (\hat{V}_{\phi_j})_{j \in \mathcal{H}_i})$ 
3   Collect policy rollout data  $\tau$  and update policy  $\pi_{\theta}$ 
    $\pi_{\theta_i}, \tau \leftarrow \mathbf{A}(\pi_{\theta_{i-1}}, \pi^{s_0, i})$ 
4    $D \leftarrow D \cup \tau$ 
5   if  $i \bmod l = 0$  then
6     Update parameters  $\phi$  of PM model  $\hat{V}_{\phi}(\bar{s})$  using  $D$ 
7      $D \leftarrow \emptyset$ 

```

We implement our SG start state selection policy as a finite-difference approximation of Eq. 10. For training iteration i , it denotes

$$\pi^{s_0, i}(s_0 | (J(\pi_{\theta_j}, \bar{s}))_{\bar{s} \in \bar{S}, j \in \mathcal{H}_i}) = \frac{\sqrt{\sum_{d=1}^{\dim(\bar{S})} (J(\pi_{\theta_i}, \bar{s}_{d,+}) - J(\pi_{\theta_i}, \bar{s}_{d,-}))^2}}{\sum_{\hat{s} \in \bar{S}} \sqrt{\sum_{d=1}^{\dim(\bar{S})} (J(\pi_{\theta_i}, \hat{s}_{d,+}) - J(\pi_{\theta_i}, \hat{s}_{d,-}))^2}} \quad (11)$$

where $\bar{s}_{d,-}$ and $\bar{s}_{d,+}$ denote the state \bar{s} where the d -th (dimension) entry is decremented or incremented by a chosen scalar Δ . In case of the PMM model, Δ is chosen such that $\bar{s}_{d,-}$ and $\bar{s}_{d,+}$ fall into the respective neighboring grid cells. In case of the PMN model, Δ is a hyper-parameter and start states are sampled from a number of proposal states which are uniformly pre-sampled from \bar{S} .

5 EMPIRICAL EVALUATION

In Sec. 5.1 we analyze the two key components of our start state curriculum framework, the performance measure (PM) and the start state selection policies, while eliminating the influence of state space discretization and PM estimation. Afterwards, we benchmark the presented performance-based start state curricula, including our spatial gradient (SG) curriculum, on different continuous dynamics spatial navigation tasks, in Sec. 5.2. Finally, we demonstrate that our SG curriculum generalizes to a high-dimensional robotic key insertion task while outperforming previous approaches.

Since all of our benchmark tasks are goal-based MDPs with a binary reward for reaching the goal, our evaluation metric is the agent’s goal-reaching probability with respect to a uniform start distribution: $\Omega(\pi_{\theta_i}, \mathcal{U}(S))$. Every l training iterations, 10 (100 in case of robotic key insertion task) rollouts starting in uniformly sampled feasible states are carried out using the current policy π_{θ_i} to obtain an estimate $\hat{\Omega}(\pi_{\theta_i}, \mathcal{U}(S))$. Performance metric values are reported as mean (solid line in plots) \pm standard error (shaded area) across the indicated number of random seeds. Throughout all experiments, we use TRPO [34] for policy optimization.

5.1 Curriculum Generation Components

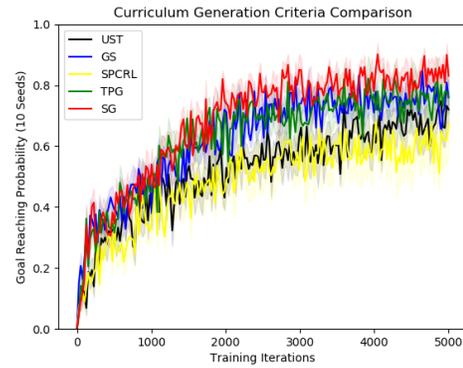
We presented a unified formulation of start state selection policies in Sec. 4. The important scientific question that naturally arises is: Which of the presented policies is most effective? Since the effectiveness of the start state selection policies depends on the accuracy of the PM, we answer this question in the first part of the section using a very accurate estimate of the PM. Afterwards, we quantify the effects of PM estimation on the learning performance.

5.1.1 Start State Selection Policy Comparison. In order to minimize effects of PM estimation errors, we use a specific experimental set-up: A discrete state and action space navigation scenario in a 30×20 discrete gridworld (see Fig. 1a). The state space has 10 dimensions: Agent x - and y -coordinate and 8 binary features indicating wall (1) or free space (0) for the neighboring tiles. The agent can take 4 actions: Up, right, down, or left. We combine the grid environment with a PMM-type PM model. The PMM grid cells correspond to the discrete environment states. Using rollouts starting 10 times from each state for PMM fitting results in very accurate goal-reaching probability estimates for all states.

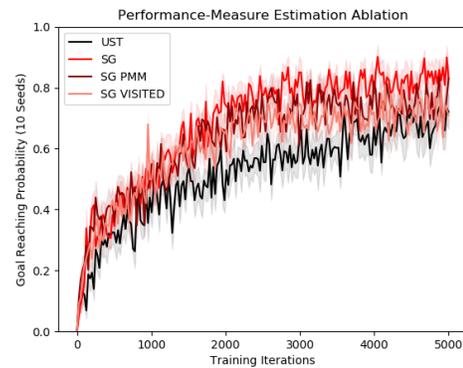
The start state selection policy comparison is visualized in Fig. 2a. The GS, TPG², and SG start state selection criteria clearly outperform the uniform start sampling (UST) baseline whereas the SPCRL start state selection is slightly worse than the baseline. The SG criterion performs best being slightly superior towards the end.

5.1.2 Effects of Performance Measure Estimation. The previous experiments showed that several choices for the start state selection policy can significantly outperform the UST baseline given a high accuracy “ground truth” PMM. Compared to the normal case, in a practical application, where the PMM must be estimated solely from anyways collected trajectory data, for data-efficiency reasons (compare Sec. 5.2), the “ground truth” PMM has two main advantages for the curricula: First, the precision at the individual

²for better performance only states with a positive temporal performance measure gain are considered for start selection



(a) Curriculum Generation Criteria Comparison



(b) Performance-Measure Estimation Ablation

Figure 2: Experiments on Discrete Gridworld Environment

grid cells is high, which allows the start state selection policies to exploit subtle differences in goal-reaching probability. Second, estimates of the goal-reaching probability are available over the entire state space, even for states that the policy has not visited yet. Consequently, starting states can be selected over the *entire* state space to optimally boost the learning progress. To understand to which extent these two properties influence the performance of the curricula, we conduct an ablation study of three SG variants:

- **SG:** PMM estimated by specifically conducting 10 rollouts from every feasible state in the state space for PMM estimation only (significantly reduced data-efficiency)
- **SG PMM:** PMM estimated from the anyways collected rollout data of the last 5 RL training iterations (data-efficient)
- **SG VISITED:** like SG but PMM values only provided for grid cells visited within the last 5 RL training iterations

Two things can be concluded from the results in Fig. 2b: First, the estimation accuracy for the visited grid cells is high enough using the training rollout data of the last 5 iterations, since SG PMM and SG VISITED perform similarly. No additional rollouts are necessary for sufficient estimation accuracy. Second, SG performs slightly better towards the end by having goal-reaching probability estimates of grid cells the agent has not visited recently during RL training, which is not practical with respect to data-efficiency.

5.2 Benchmarking on Spatial Navigation Tasks

The experiments in Sec. 5.1 showed that our novel SG start state curriculum achieves state-of-the-art performance given a global high accuracy PMM model estimate. Furthermore, we found out that the PMM can be estimated sufficiently accurate from rollout data collected in the inner RL loop. In the following, we compare the performance-based start state curricula using this data-efficient PMM estimation approach on challenging and realistic sparse reward spatial navigation tasks. We evaluate them on three different environments comparing to existing algorithms that can be used for automatic start state selection in RL, demonstrating performance gains of our SG curriculum. Prior to this, we evaluate which type of PM model gives best performance in spatial navigation tasks.

5.2.1 Performance Measure Model Evaluation. In the following, we want to find out whether the tabular PMM model, which was perfectly suited for the discrete state and action space maze navigation, or a neural network-based PMN model for the performance measure results in better performance in difficult continuous state and action space spatial navigation tasks. Therefore, we conduct experiments with an RL agent using the SG curriculum on a *continuous* state/action space and agent dynamics version of the *gridworld* depicted in Fig. 1a. The 12-dimensional state space consists of velocity in x - and y -direction, x - and y -position, and eight binary values to indicate obstacles on neighboring tiles. Actions are accelerations in x - and y -direction. The agent has to reach the red goal tile.

For the spatial gradient curriculum with the tabular PMM (SG PMM), the state space discretization necessary for the PMM estimation uses the same tiling as in the discrete state space setting before. SG PMN uses the value function critic of TRPO as performance measure model since no discounting is employed and only using the low-dimensional state space component \bar{s} did not turn out to be beneficial in this scenario. For SG PMN 100 start state candidates are pre-sampled uniformly and the Δ value is similar to the tile size 1 m. The comparison is visualized in Fig. 3.

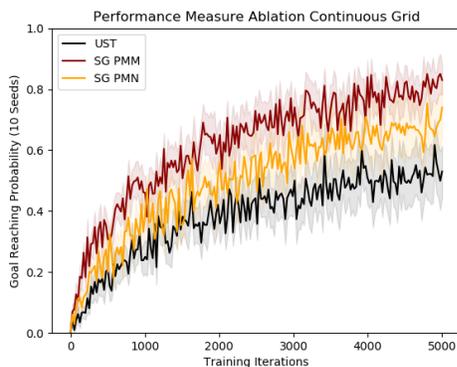


Figure 3: Performance Measure Choice on Continuous Grid

While both SG variants clearly outperform the uniform start sampling baseline, SG PMM clearly outperforms SG PMN³. A possible reason for this is that the PMM by simply averaging returns

³We obtained similar results for the MuJoCo Point Mass task (Sec. 5.2.5).

of recent rollout data for regions of the state space can quickly adapt to changes in the goal reaching capabilities of the agent whereas the neural network uses a gradient-based optimizer to continuously update its parameters, which additionally may have to fight local optima. Setting the reaching probability for tiles without available data by default to zero in case of the PMM model, which enforces gradients at boundaries towards unexplored regions of the state space, might be another aspect that explains the better performance of SG PMM. As a result, we will use the tabular PMM as our performance measure model in the remainder of this section.

5.2.2 Environments. The three spatial navigation scenarios are:

- A *continuous* state/action space and agent dynamics *gridworld* as presented in Sec. 5.2.1 ($T = 100$).
- Similar to [9] a *point mass* agent in a "G-shape" maze (see Fig. 4a) implemented using *MuJoCo* [41] ($T = 500$).
- Similar to [9] an "*ant*" agent in a "U-shape" maze (see Fig. 4b) implemented using *MuJoCo* [41] ($T = 2000$).

While all of them are spatial navigation tasks with continuous dynamics, the environments differ in maze complexity, dynamics complexity, and time horizon T . The number of policy training iterations between curriculum updates is $l = 5$ for all of them.

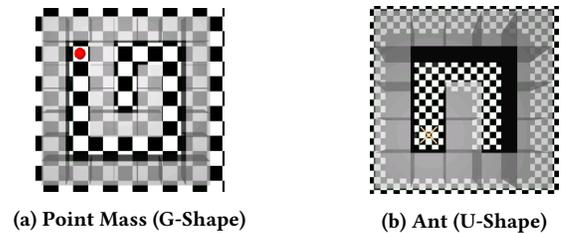
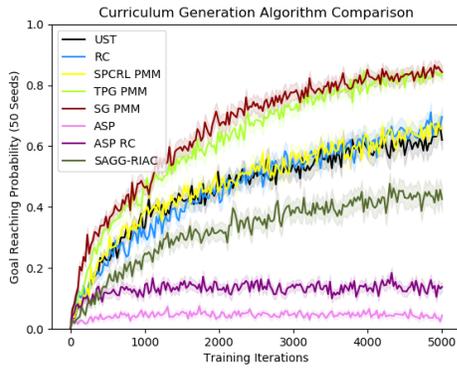


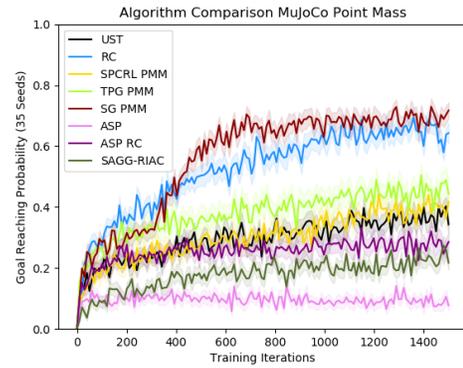
Figure 4: MuJoCo Spatial Navigation Environments

5.2.3 Baselines. Apart from the simple UST baseline, we compare the performance-based start state selection policies to:

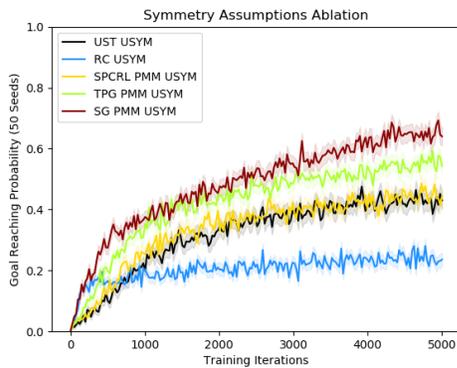
- *Reverse Curriculum Generation (RC)* [9]: RC has been proposed as a practical approximation of the GS curriculum that does not assume to know the agent's goal-reaching probability at unvisited states. It is currently the state-of-the-art in start state selection for RL and replaces GS from now on.
- *Asymmetric Self-Play (ASP)* [38]: We implement and evaluate two versions of this algorithm. As the first version, we implement the original algorithm as reported in [38] where during self-play Alice starting from the goal proposes starts for Bob. During the rest of the training time, Bob is trained on the original goal-based MDP under uniform start states. As the second version, ASP RC, we use a hybrid version of ASP and RC which was used as a baseline in [9]. ASP RC uses the self-play only to update Alice but not Bob. The starts Alice generates are used as the "good states" in the RC algorithm.
- *SAGG-RIAC* [5]: In [8] the SAGG-RIAC algorithm for automatic goal generation has been adapted to the modern batch reinforcement learning setting using TRPO as the "Low-Level Goal-Directed Exploration with Evolving Context". We use their implementation but modify it in order to generate start states instead of goal states.



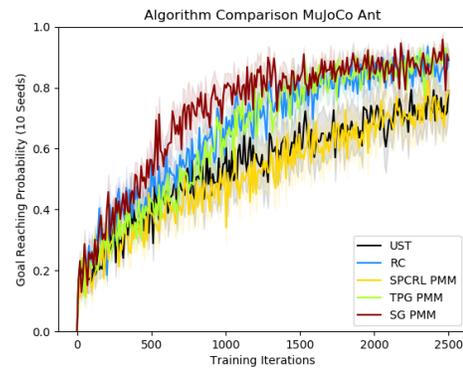
(a) Curriculum Generation Algorithm Comparison



(a) Point Mass (G-Shape)



(b) Symmetry Assumptions Ablation



(b) Ant (U-Shape)

Figure 5: Experiments on Continuous Gridworld

5.2.4 *Continuous Grid*. The results for the continuous gridworld environment are presented in Fig. 5a. SG PMM and TPG PMM perform best, clearly outperforming UST and all the other baselines. The two variants of the Asymmetric Self-Play algorithm perform particularly badly. It turns out that, in this experiment, the competition between the two RL agents Alice and Bob quickly collapses in the sense that one of the two gets much better than the other one, which is usually Bob, the goal-reaching agent of interest. The main reason for this might be that Bob gets more training since he does “normal” reinforcement learning besides the self-play.

5.2.5 *MuJoCo Point Mass*. The results of the MuJoCo point mass maze navigation task are presented in Fig. 6a. Again, SG PMM performs best, especially more quickly reaching its final performance level than RC, which catches up in the end. The performance of TPG PMM greatly depends on the Boltzmann distribution temperature: A value of 1.0 instead of 0.2 (gridworld) improves performance.

5.2.6 *MuJoCo Ant*. The results of the MuJoCo ant maze navigation task are shown in Fig. 6b. SG PMM performs best again. TPG PMM and RC reach the same final performance level in the end but show a slower increase in goal-reaching probability in the beginning. SAGG-RIAC and Asymmetric Self-Play are not considered any more due to their poor performance in the previous tasks.

Figure 6: MuJoCo Maze Navigation Experiments

5.2.7 *Symmetry Assumptions Ablation*. All spatial navigation tasks considered in this section have symmetric agent dynamics in common: With the equal magnitude of action, the agent moves the same distance in the corresponding direction. However, this may not hold for real world vehicles. A car, for example, has potentially different gear ratios as well as maximum speeds in forward and backward direction, respectively. Since the heuristic “Brownian motion” procedure to generate new starting states in RC assumes similar difficulty of getting from A to B and from B to A, respectively, we want to investigate RC and the performance-based start state curricula under “asymmetric” agent dynamics on the continuous gridworld environment, in the following.

We skewed the agent dynamics: While they are unchanged when moving in positive x - or y -direction, moving in negative direction only a quarter of the chosen acceleration is applied. By this, the “Brownian motion” of RC is biased towards positive direction. The results are visualized in Fig. 5b. While the final goal-reaching probability of SG PMM decreases by around 20 percentage points in comparison to the symmetric dynamics case, the performance of RC drops by more than 40 percentage points, being worse than the UST baseline. Also the performance of TPG PMM drops slightly more than SG PMM. This indicates that SG start state selection is more robust with respect to (arbitrary) agent-environment dynamics.

5.3 Generalization to Robotic Manipulation Task

Finally, we show that our spatial gradient curriculum is also applicable to higher-dimensional robotic manipulation tasks, achieving superior performance. We use a robotic key insertion task similar to the one presented in [9], which was implemented using *MuJoCo* [41]. The state space includes the 7 joint angles and 7 joint velocities of the robot as well as the 3D positions of 3 reference points on the key. The joint accelerations are applied as actions. For a successful key insertion, the key must be inserted with a specific orientation, rotated clockwise by 90° , pushed, rotated counterclockwise by 90° , and finally be within 3 cm of the target location. All robot configurations with the tip of the key being within 40 cm of the key-hole are considered feasible. The environment is depicted in Fig. 7.

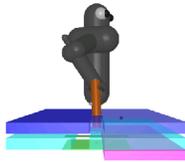


Figure 7: MuJoCo Key Insertion Task

The subspace \bar{S} to choose start states from is chosen as the 7D joint angle configuration. Therefore, we use a neural network-based PMN model for the performance measure since the PMM model does not scale to such high dimensionality. Each $l = 2$ iterations the sampling probabilities for 1000 uniformly sampled proposal starts are calculated. For SG PMN, the delta is chosen similar to the goal-reaching accuracy: $\Delta = 3$ cm.

The simulation results are presented in Fig. 8. While all performance-based start state curricula outperform the UST baseline, the spatial gradient curriculum (SG PMN) clearly performs best.

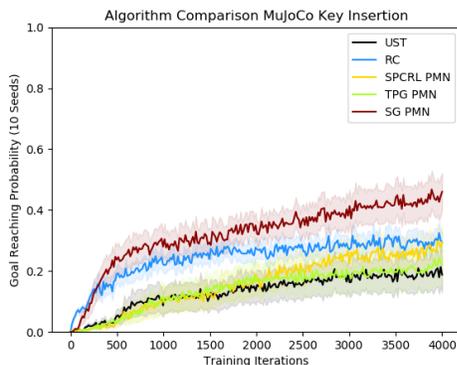


Figure 8: MuJoCo Key Insertion Task

6 CONCLUSION

In this work, we introduced a unifying framework for performance-based start state curriculum generation in reinforcement learning.

It consists of two key components: 1) A state-dependent performance measure, which estimates the policy’s current capabilities, e.g. the goal-reaching probability. 2) A start state selection policy that selects starts based on the PM. We showed how several existing automatic curriculum generation approaches fit in our framework. Furthermore, we introduced a novel start state selection policy that makes use of spatial performance measure gradients to find start states at the boundary of the policy’s current capabilities.

Comparing different start state selection policies using a “ground truth” PM model, our spatial gradient curriculum performed best. We found that for recently visited states the PM can sufficiently be estimated using rollout data from the RL training loop. The performance gap to the “ground truth” PM model is largely explained by the available global information at unvisited states.

In realistic continuous dynamics spatial navigation tasks, our proposed spatial gradient start state curriculum achieves state-of-the-art performance, performing best or among the best compared to the other curriculum generation algorithms, most notably outperforming the Reverse Curriculum approach. Additionally, our approach turned out to be more robust with respect to asymmetric dynamics than any other start state curriculum and does not need any additional rollouts to generate new start states.

Our proposed spatial gradient start state selection policy can be flexibly combined with arbitrary estimators of the performance measure. For spatial navigation tasks, a tabular representation empirically resulted in the best performance. Using a neural network model as performance measure, our spatial gradient start state curriculum is also applicable to high-dimensional robotic manipulation tasks, achieving state-of-the-art performance.

Applying our start state curriculum framework to MDPs with intermediate rewards is a potential direction for future work. The performance measure might either solely focus on goal reaching as before or be designed to also account for the intermediate rewards such that the curriculum selects starts to improve the overall performance. Another research direction may be to find out whether our SG curriculum can be modified to be applicable to more general graph structures than regular lattices by looking at PM differences between neighboring nodes rather than spatial derivatives.

A POLICY OPTIMIZATION PARAMETERS

Table 1: TRPO Parameters

Environment	Grid	MJ Point	MJ Ant	MJ Key
Max KL	5e-4	5e-4	1e-2	5e-4
Damping	5e-3	5e-3	1e-3	5e-3
Batch Size	3200	20000	80000	50000

Policy and value function critic are represented by fully connected neural networks with three hidden layers of 64 neurons.

For GS and RC, $a = 0.1$ and $b = 0.9$. The RC parameters are $N_{\text{new}} = 50$, $N_{\text{old}} = 25$, $T_B = 20$, $M = 1000$ in the gridworld experiments and similar to [9] in the *MuJoCo* experiments.

Code will be made available: <https://github.com/boschresearch/A-Performance-Based-Start-State-Curriculum-Framework-for-RL>

REFERENCES

- [1] Marcin Andrychowicz, Filip Wolski, Alex Ray, Jonas Schneider, Rachel Fong, Peter Welinder, Bob McGrew, Josh Tobin, OpenAI Pieter Abbeel, and Wojciech Zaremba. 2017. Hindsight experience replay. In *Advances in Neural Information Processing Systems*. 5048–5058.
- [2] Jose A Arjona-Medina, Michael Gillhofer, Michael Widrich, Thomas Unterthiner, Johannes Brandstetter, and Sepp Hochreiter. 2018. Rudder: Return decomposition for delayed rewards. *arXiv preprint arXiv:1806.07857* (2018).
- [3] Minoru Asada, Shoichi Noda, Sukoya Tawaratsumida, and Koh Hosoda. 1996. Purposive behavior acquisition for a real robot by vision-based reinforcement learning. *Machine learning* 23, 2-3 (1996), 279–303.
- [4] Adrien Baranes and Pierre-Yves Oudeyer. 2009. R-iac: Robust intrinsically motivated exploration and active learning. *IEEE Transactions on Autonomous Mental Development* 1, 3 (2009), 155–169.
- [5] Adrien Baranes and Pierre-Yves Oudeyer. 2013. Active learning of inverse models with intrinsically motivated goal exploration in robots. *Robotics and Autonomous Systems* 61, 1 (2013), 49–73.
- [6] Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. 2009. Curriculum learning. In *International Conference on Machine Learning*. ACM, 41–48.
- [7] Benjamin Eysenbach, Abhishek Gupta, Julian Ibarz, and Sergey Levine. 2018. Diversity is all you need: Learning skills without a reward function. *arXiv preprint arXiv:1802.06070* (2018).
- [8] Carlos Florensa, David Held, Xinyang Geng, and Pieter Abbeel. 2018. Automatic goal generation for reinforcement learning agents. In *International Conference on Machine Learning*. 1514–1523.
- [9] Carlos Florensa, David Held, Markus Wulfmeier, Michael Zhang, and Pieter Abbeel. 2017. Reverse Curriculum Generation for Reinforcement Learning. In *Conference on Robot Learning*. 482–495.
- [10] Scott Fujimoto, Herke Hoof, and David Meger. 2018. Addressing Function Approximation Error in Actor-Critic Methods. In *International Conference on Machine Learning*. 1582–1591.
- [11] Alex Graves, Marc G Bellemare, Jacob Menick, Remi Munos, and Koray Kavukcuoglu. 2017. Automated curriculum learning for neural networks. In *International Conference on Machine Learning*, Vol. 70. JMLR. org, 1311–1320.
- [12] Nick Haber, Damian Mrowca, Stephanie Wang, Li F Fei-Fei, and Daniel L Yamins. 2018. Learning to Play With Intrinsically-Motivated, Self-Aware Agents. In *Advances in Neural Information Processing Systems*. 8388–8399.
- [13] Matteo Hessel, Joseph Modayil, Hado Van Hasselt, Tom Schaul, Georg Ostrovski, Will Dabney, Dan Horgan, Bilal Piot, Mohammad Azar, and David Silver. 2018. Rainbow: Combining improvements in deep reinforcement learning. In *AAAI Conference on Artificial Intelligence*.
- [14] Zhang-Wei Hong, Tzu-Yun Shann, Shih-Yang Su, Yi-Hsiang Chang, Tsu-Jui Fu, and Chun-Yi Lee. 2018. Diversity-Driven Exploration Strategy for Deep Reinforcement Learning. In *Advances in Neural Information Processing Systems*. 10489–10500.
- [15] Rein Houthoofd, Xi Chen, Yan Duan, John Schulman, Filip De Turck, and Pieter Abbeel. 2016. Vime: Variational information maximizing exploration. In *Advances in Neural Information Processing Systems*. 1109–1117.
- [16] Max Jaderberg, Volodymyr Mnih, Wojciech Marian Czarnecki, Tom Schaul, Joel Z Leibo, David Silver, and Koray Kavukcuoglu. 2016. Reinforcement learning with unsupervised auxiliary tasks. *arXiv preprint arXiv:1611.05397* (2016).
- [17] Lu Jiang, Deyu Meng, Qian Zhao, Shiguang Shan, and Alexander G Hauptmann. 2015. Self-paced curriculum learning. In *AAAI Conference on Artificial Intelligence*.
- [18] Sham Kakade and John Langford. 2002. Approximately optimal approximate reinforcement learning. In *International Conference on Machine Learning*, Vol. 2. 267–274.
- [19] Pascal Klirk, Hany Abdulsamad, Boris Belousov, and Jan Peters. 2019. Self-Paced Contextual Reinforcement Learning. In *Conference on Robot Learning*.
- [20] M Pawan Kumar, Benjamin Packer, and Daphne Koller. 2010. Self-paced learning for latent variable models. In *Advances in Neural Information Processing Systems*. 1189–1197.
- [21] Sergey Levine, Chelsea Finn, Trevor Darrell, and Pieter Abbeel. 2016. End-to-end training of deep visuomotor policies. *The Journal of Machine Learning Research* 17, 1 (2016), 1334–1373.
- [22] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. 2015. Human-level control through deep reinforcement learning. *Nature* 518, 7540 (2015), 529.
- [23] Sanmit Narvekar, Jivko Sinapov, and Peter Stone. 2017. Autonomous Task Sequencing for Customized Curriculum Design in Reinforcement Learning. In *International Joint Conference on Artificial Intelligence*. 2536–2542.
- [24] Sanmit Narvekar and Peter Stone. 2019. Learning Curriculum Policies for Reinforcement Learning. In *Proceedings of the 18th International Conference on Autonomous Agents and MultiAgent Systems*. 25–33.
- [25] Andrew Y Ng, Daishi Harada, and Stuart Russell. 1999. Policy invariance under reward transformations: Theory and application to reward shaping. In *International Conference on Machine Learning*, Vol. 99. 278–287.
- [26] Ian Osband, Charles Blundell, Alexander Pritzel, and Benjamin Van Roy. 2016. Deep exploration via bootstrapped DQN. In *Advances in Neural Information Processing Systems*. 4026–4034.
- [27] Deepak Pathak, Pulkit Agrawal, Alexei A Efros, and Trevor Darrell. 2017. Curiosity-driven exploration by self-supervised prediction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*. 16–17.
- [28] Ivaylo Popov, Nicolas Heess, Timothy Lillicrap, Roland Hafner, Gabriel Barth-Maron, Matej Vecerik, Thomas Lampe, Yuval Tassa, Tom Erez, and Martin Riedmiller. 2017. Data-efficient deep reinforcement learning for dexterous manipulation. *arXiv preprint arXiv:1704.03073* (2017).
- [29] Cinjon Resnick, Roberta Raileanu, Sanyam Kapoor, Alex Peysakhovich, Kyunghyun Cho, and Joan Bruna. 2018. Backplay: “Man muss immer umkehren”. *arXiv preprint arXiv:1807.06919* (2018).
- [30] Martin Riedmiller, Roland Hafner, Thomas Lampe, Michael Neunert, Jonas Degraeve, Tom Wiele, Vlad Mnih, Nicolas Heess, and Jost Tobias Springenberg. 2018. Learning by Playing Solving Sparse Reward Tasks from Scratch. In *International Conference on Machine Learning*. 4341–4350.
- [31] Tim Salimans and Richard Chen. 2018. Learning Montezuma’s Revenge from a Single Demonstration. *arXiv preprint arXiv:1812.03381* (2018).
- [32] Jürgen Schmidhuber. 2010. Formal theory of creativity, fun, and intrinsic motivation (1990–2010). *IEEE Transactions on Autonomous Mental Development* 2, 3 (2010), 230–247.
- [33] Jürgen Schmidhuber. 2013. Powerplay: Training an increasingly general problem solver by continually searching for the simplest still unsolvable problem. *Frontiers in psychology* 4 (2013), 313.
- [34] John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. 2015. Trust region policy optimization. In *International Conference on Machine Learning*. 1889–1897.
- [35] John Schulman, Philipp Moritz, Sergey Levine, Michael Jordan, and Pieter Abbeel. 2016. High-Dimensional Continuous Control Using Generalized Advantage Estimation. In *International Conference on Learning Representations*.
- [36] David Silver, Thomas Hubert, Julian Schrittwieser, Ioannis Antonoglou, Matthew Lai, Arthur Guez, Marc Lanctot, Laurent Sifre, Dharmashan Kumaran, Thore Graepel, et al. 2018. A general reinforcement learning algorithm that masters chess, shogi, and Go through self-play. *Science* 362, 6419 (2018), 1140–1144.
- [37] Rupesh Kumar Srivastava, Bas R Steunebrink, Marijn Stollenga, and Jürgen Schmidhuber. 2012. Continually adding self-invented problems to the repertoire: first experiments with powerplay. In *IEEE International Conference on Development and Learning and Epigenetic Robotics (ICDL)*. IEEE, 1–6.
- [38] Sainbayar Sukhbaatar, Zeming Lin, Ilya Kostrikov, Gabriel Synnaeve, Arthur Szlam, and Rob Fergus. 2017. Intrinsic motivation and automatic curricula via asymmetric self-play. *arXiv preprint arXiv:1703.05407* (2017).
- [39] Richard S Sutton and Andrew G Barto. 2018. *Reinforcement learning: An introduction*. MIT press.
- [40] Richard S Sutton, David A McAllester, Satinder P Singh, and Yishay Mansour. 2000. Policy gradient methods for reinforcement learning with function approximation. In *Advances in Neural Information Processing Systems*. 1057–1063.
- [41] Emanuel Todorov, Tom Erez, and Yuval Tassa. 2012. Mujoco: A physics engine for model-based control. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 5026–5033.