Issues with Measuring Task Complexity via Random Policies in Robotic Tasks

Reabetswe M. Nkhumise
University of Sheffield
Sheffield, United Kingdom
rabs.mike@yahoo.com

Mohamed S. Talamali
University of Sheffield
Sheffield, United Kingdom
m.s.talamali@sheffield.ac.uk

Aditya Gilra
Wirtschaftsuniversität
Vienna, Austria
aditya.gilra@wu.ac.at

ABSTRACT

Reinforcement learning (RL) has enabled major advances in fields such as robotics and natural language processing. A key challenge in RL is measuring task complexity, which is essential for creating meaningful benchmarks and designing effective curricula. While there are numerous well-established metrics for assessing task complexity in tabular settings, relatively few exist in non-tabular domains. These include (i) Statistical analysis of the performance of random policies via Random Weight Guessing (RWG), and (ii) information-theoretic metrics Policy Information Capacity (PIC) and Policy-Optimal Information Capacity (POIC), which are reliant on RWG. In this paper, we evaluate these methods using progressively difficult robotic manipulation setups, with known relative complexity, with both dense and sparse reward formulations. Our empirical results reveal that measuring complexity is still nuanced. Specifically, under the same reward formulation, PIC suggests that a two-link robotic arm setup is easier than a single-link setup — which contradicts the robotic control and empirical RL perspective whereby the two-link setup is inherently more complex. Likewise, for the same setup, POIC estimates that tasks with sparse rewards are easier than those with dense rewards. Thus, we show that both PIC and POIC contradict typical understanding and empirical results from RL. These findings highlight the need to move beyond RWG-based metrics towards better metrics that can more reliably capture task complexity in non-tabular RL with our task framework as a starting point.

KEYWORDS

Reinforcement Learning, Task Complexity, Robotic Manipulation

ACM Reference Format:
Reabetswe M. Nkhumise, Mohamed S. Talamali, and Aditya Gilra. 2026. Issues with Measuring Task Complexity via Random Policies in Robotic Tasks. In *Proc. of the 25th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2026), Paphos, Cyprus, May 25 – 29, 2026*, IFAAMAS, 10 pages. https://doi.org/10.65109/FDIK3367

1 INTRODUCTION

Reinforcement Learning (RL) is a framework formulated to tackle sequential decision-making problems, which are at the heart of autonomous agents [42]. It has demonstrated remarkable success in constructing decision-making agents across various domains, including games [29, 52], robotics [28, 57] and energy systems [47, 61], amongst others. Novel RL algorithms are continually being developed to tackle increasingly complex real-world problems.

To evaluate progress, researchers employ a variety of RL *benchmarks* [6, 44, 56] — collections of tasks designed with varying levels of difficulty — to assess and compare algorithms [43]. When proposing a new method, it is standard practice to use such benchmarks to characterise its capabilities relative to existing approaches by measuring performance across tasks and examining how results scale with increasing difficulty. In addition, researchers also often leverage *curriculum learning*, where agents are trained on a sequence of progressively more difficult tasks to enable intermediate learning and gradual skills acquisition [40]. This approach has been shown to improve both learning and generalisability [12, 30, 46], with agents trained using curricula typically outperforming those trained directly on the most difficult tasks [7, 30, 40].

Measuring task difficulty is essential in both benchmarks and curriculum learning. In benchmarks, it ensures that tasks span a broad spectrum of challenges, providing comprehensive coverage that avoids sets that are uniformly trivial or overly difficult and thereby enables a more rigorous evaluation of algorithms' capabilities [21, 46]. In curriculum learning, by contrast, task difficulty supports the structured ranking of tasks, allowing agents to encounter them in a progression that reflects their true hardness [39].

RL tasks are commonly divided into tabular and non-tabular settings. Tabular RL assumes small, finite state-action spaces that can be explicitly enumerated, as in grid-world or bandit problems. Non-tabular RL, by contrast, involves large or continuous state-action spaces, typical of robotics control, autonomous driving, or energy management [22, 25, 33, 47]. While there are well-established metrics of task difficulty in tabular settings [1, 13], a unified task complexity framework for non-tabular tasks is lacking [13], largely since existing approaches rely on heuristics, make restrictive assumptions, or are computationally intractable [13, 14].

Nevertheless, two notable approaches have been proposed to broadly analyse task complexity in non-tabular domains. Both are based on the Random Weight Guessing (RWG) [48] process, in which untrained policies — initialised with random weights — are executed within tasks and their cumulative rewards (returns) are measured [43]. The first approach employs *statistical analysis* of the resulting return distributions to assess task difficulty [43]. The second approach adopts an *information-theoretic* perspective [36], introducing two metrics: *Policy Information Capacity* (PIC) and *Policy-Optimal Information Capacity* (POIC) [25].

PIC quantifies the mutual information between the policy weights (parameters) and the returns. In contrast, POIC measures the mutual information between the policy parameters and an optimality variable, which indicates whether the agent behaves optimally

Proc. of the 25th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2026), C. Amato, L. Dennis, V. Mascardi, J. Thangarajah (eds.), May 25 – 29, 2026, Paphos, Cyprus. © 2026 International Foundation for Autonomous Agents and Multiagent Systems (www.ifaamas.org). https://doi.org/10.65109/FDIK3367

throughout the episode [25]. A higher PIC value reflects a stronger dependence of returns on the random policy parameters, suggesting that the policy exerts a greater influence on performance and that the task is therefore easier. Similarly, high POIC values indicate that finding an optimal policy is relatively straightforward. Conversely, lower PIC and POIC values are associated with more difficult tasks.

While the statistical approach provides a *relative* measure of task complexity — indicating, for instance, that one task is more difficult than another based on the return statistics of randomly sampled policies — it does not quantify *how much harder* one task is compared to another. This limitation is addressed by the information-theoretic approach through the PIC and POIC metrics, which offer quantitative measures of relative task complexity.

In this work, we demonstrate that, despite the information-theoretic approach providing a more quantitative characterisation of relative task complexity, the resulting measures can be misleading in certain cases. Using tasks with **known relative complexity relationships**, we show that PIC and POIC can incorrectly capture task hardness. These tasks consist of simple robotic manipulation environments, *1-link* and *2-link* manipulators with one and two degrees of freedom (DoF) respectively — where the objective is to control a robotic arm to reach specified target positions. Two reward formulations are considered: a *dense* formulation, which provides incremental rewards as the arm approaches the target, and a *sparse* formulation, which provides a non-negative reward only upon reaching the target. In our experiments, both PIC and POIC produced results that contradict expectations — indicating, for instance, that the *2-link* manipulator task is easier than the *1-link* manipulator task under the same reward formulation, or that it is easier to find optimal policies in a sparse-reward setting than in a dense-reward one. These outcomes suggest that the PIC and POIC metrics may not be reliable indicators of task complexity, and that the question of reliable metrics remains open. We speculate that the inconsistencies of these metrics could be attributed to their reliance on the RWG process, which is known to be ineffective for tasks with sparse solution regions in the weight (parameter) space [48]. Our contributions are threefold:

- We propose a framework for assessing task complexity metrics, achieved by using environments and reward formulations of known relative complexity. Specifically, we employ structurally comparable robotic manipulation environments evaluated under different reward formulations.
- Using this framework, we show that PIC and POIC yield results that contradict these known complexity relationships, suggesting that these metrics do not remain valid in certain task settings.
- We highlight the need for continued research into developing more reliable and interpretable measures of task complexity.

The remainder of this paper is organised as follows. We briefly outline the methods used to assess task complexity — i.e. statistical analysis of return distributions of random policies, and PIC and POIC — in Section 2. Following this, we explicate the complexity of manipulation tasks in Section 3. Then, we present the results of task complexity analysis using the aforementioned methods on manipulation tasks in Section 4. Finally, we discuss the limitations of these task complexity metrics in Section 5.

2 TASK COMPLEXITY QUANTIFICATION FRAMEWORKS

In this section, we review the process used to determine the return statistics of random policies obtained through RWG, and the PIC and POIC task complexity metrics. Henceforth, we use the term *return* interchangeably with *performance*.

2.1 RWG and Statistical Analysis

The use of RWG, along with statistical analysis of performance, for analysing RL task complexity was introduced in [43]. In this method, a policy model is represented by a neural network architecture. The model's parameters are randomly sampled at the beginning of each run and remain fixed thereafter. The untrained policy is then executed within the environment, and the resulting episodic rewards are recorded — as outlined in Algorithm 1.

Algorithm 1: Task Evaluation with RWG

Input: Prior distribution of parameters $p(\theta) = \mathcal{N}(0, I)$, Number of samples N, Number of episodes M.
Output: episodic cumulative reward $S_{n,e}$

```
Initialize environment;
Create array S_{n,e} of size N × M;
for n = 1, 2, ..., N do
    Sample weights θ_n ~ p(θ);
    for e = 1, 2, ..., M do
        Reset the environment;
        Run episode with θ_n;
        Store cumulative episode reward in S_{n,e};
```

The prior distribution of parameters $p(\theta)$ is a multivariate normal distribution $\mathcal{N}(0, I)$, where $I \in \mathbb{R}^{d \times d}$ is an identity matrix over weight vectors $\theta_n \in \mathbb{R}^d$. N is the number of parameter sets of the policy model, i.e. number of (random) policies. M is the number of episodes per run. Performance of each policy indexed by n is aggregated by computing the mean M_n and variance V_n of the cumulative rewards over its trial set of episodes, using [43]:

$$M_n = \frac{1}{M} \sum_{e=1}^{M} S_{n,e} \tag{1}$$

$$V_n = \frac{1}{M-1} \sum_{e=1}^{M} (S_{n,e} - M_n)^2 \tag{2}$$

where $S_{n,e}$ is the episodic cumulative reward (i.e. performance sample) for the e^{th} episode of the n^{th} policy. For a given task environment, the aggregate performance of the policies is showcased in three plots:

(1) Log-scale histogram of M_n
(2) Mean performance M_n vs rank R_n
(3) Performance variance V_n vs mean performance M_n

where rank R_n sorts the policies according to performance, with 1 denoting the policy with the lowest mean performance and larger values representing policies with higher mean performances. If two policies rank the same, then the tie is broken by ranking them in the order in which their weights were sampled.

2.2 PIC and POIC

Mutual information is a quantity that measures the dependency between two random variables [17, 36]. Unlike the correlation coefficient, it is not limited to only linear relationships but can also describe nonlinear ones [36]. PIC is the mutual information between the policy model parameters and the corresponding episodic cumulative rewards (i.e. *return* samples). It is given by [25],

$$\mathcal{I}(R;\Theta) = \mathcal{H}(R) - \mathbb{E}_{p(\theta)}[\mathcal{H}(R|\Theta=\theta)] \tag{3}$$

where $\mathcal{H}(\cdot)$ is Shannon entropy, R is the episodic cumulative reward random variable, and Θ is the random variable of policy model parameters. Intuitively, if the parameters Θ do not tightly determine R (i.e. have little effect on the reward signal), then the first term and second term in Equation 3 will be approximately equal. That is, $\mathcal{H}(R) \approx \mathbb{E}_{p(\theta)}[\mathcal{H}(R|\Theta=\theta)]$ and hence PIC ≈ 0. In that case, the task is relatively hard and therefore, PIC $\rightarrow 0$ as tasks become harder.

POIC is the mutual information between the policy model parameters and the *optimality variable*. An optimality variable represents whether the agent behaves optimally during the entire episode [25]. For instance, when POIC is expressed as [25],

$$\mathcal{I}(\mathbb{O};\Theta) = \mathcal{H}(\mathbb{O}) - \mathbb{E}_{p(\theta)}[\mathcal{H}(\mathbb{O}|\Theta=\theta)] \tag{4}$$

$\mathbb{O}$ is the optimality variable which $\mathbb{O}=1$ when the agent behaves optimally during the episode, and $\mathbb{O}=0$ otherwise. If parameters Θ have significant effect on the agent's optimal performance, then the difference between the first and second terms in Equation 4 will be large. This would highlight the ease of acting optimally in the task.

Note that PIC and POIC are aligned. They respectively represent the influence of the policy model parameters Θ on rewards and optimal behaviour. Furthermore, they can be viewed as the remaining randomness in rewards or optimality after accounting for the randomness caused by the parameters Θ. The residual randomness reflects variability inherent to the environment. Both PIC and POIC make use of performance samples generated via Algorithm 1. In practice, PIC and POIC are empirically estimated by discretising the return distribution $p(R)$ and each conditional distribution $p(R|\theta_i)$ into B identical bins, as follows — starting with PIC [25]:

$$\begin{aligned}\hat{\mathcal{I}}(R;\theta) = &-\sum_{b=1}^{B}\hat{p}(R_b)\log(\hat{p}(R_b)) \\ &+\frac{1}{N}\sum_{n=1}^{N}\sum_{b=1}^{B}\hat{p}(R_b|\theta_n)\log(\hat{p}(R_b|\theta_n))\end{aligned} \tag{5}$$

where N is the number of random policies and $\hat{p}(R_b)$ estimates a portion of return samples in bin b with respect to the total number of return samples. For a given θ_n, the fraction of return samples in bin b relative to all return samples is $\hat{p}(R_b|\theta_n)$. POIC is estimated using [25]:

$$\begin{aligned}\hat{\mathcal{I}}(O;\theta) = &-\hat{p}_1\log(\hat{p}_1) - (1-\hat{p}_1)\log(1-\hat{p}_1) \\ &+\frac{1}{N}\sum_{n=1}^{N}[\hat{p}_{1n}\log(\hat{p}_{1n}) + (1-\hat{p}_{1n})\log(1-\hat{p}_{1n})]\end{aligned} \tag{6}$$

where $\hat{p}_1 \doteq p(\mathbb{O}=1) \approx \frac{1}{N}\sum_{n=1}^{N}\hat{p}_{1n}$ and $\hat{p}_{1n} \doteq p(\mathbb{O}=1|\theta_n) \approx \frac{1}{M}\sum_{e=1}^{M}\exp\left(\frac{S_{n,e}-S_{max}}{\lambda}\right)$. Note that λ is a temperature parameter and $S_{max} = \max[S_{n,e}, S^*]$, where S^* is the episodic cumulative reward of an optimal policy π^*. Minimum and maximum values in the return samples are set as limits and divided into B equal parts for the calculations.

We now consider the complexity of robotic reaching tasks that we will use to evaluate the task complexity metrics presented in this section.

3 COMPLEXITY OF ROBOTIC REACHING TASKS

Robot manipulation is a classic problem that has been studied rigorously in control theory [35, 50]. Within manipulation, reaching tasks are defined by the objective of moving the end-effector to desired positions. We consider such tasks along with fully actuated serial manipulators shown in Figure 1, where the motion of each joint is directly controllable. A manipulator with n number of links or joints, often referred to as an *n-link* arm, has the dynamic model [16]:

$$M(q)\ddot{q} + C(q,\dot{q})\dot{q} + G(q) + F(\dot{q}) + J(q)^T\eta = \tau \tag{7}$$

where $q, \dot{q}, \ddot{q} \in \mathbb{R}^n$ are joint position, velocity and acceleration

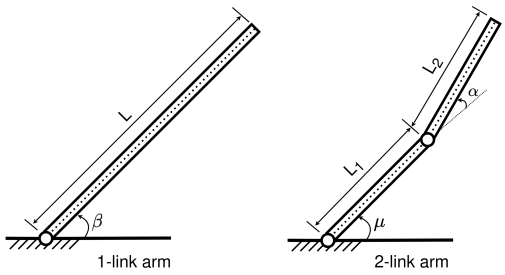

Figure 1: Illustration of manipulators.

vectors, respectively. $\tau \in \mathbb{R}^n$ is actuator joint torque vector, $M(q) \in \mathbb{R}^{n\times n}$ is inertia matrix, $C(q,\dot{q})\dot{q} \in \mathbb{R}^n$ is Coriolis and centrifugal vector, $G(q) \in \mathbb{R}^n$ is gravity vector, and $F(\dot{q}) \in \mathbb{R}^n$ is friction torque. $J(q)^T$ is the transpose of the Jacobian matrix that relates $\eta \in \mathbb{R}^6$ forces at the end-effector to joint torques.

Note that a fully actuated *n-link* arm has n degrees-of-freedom (DoF), i.e. n-DoF. Higher n-DoF enable agile, precise and energy-efficient robot motions [37], but consist of larger state-action space dimensionality. This leads to more complex dynamics [15] that contribute to the task complexity [27]. For instance, as n increases the system characteristics are impacted as follows:

(1) The number of coupled terms in $M(q)$, $C(q,\dot{q})$, $F(\dot{q})$ and $G(q)$ increases [37]. This means the robot dynamics become highly nonlinear and more complex, resulting in unpredictable behaviour under perturbations [49, 53].

(2) The coordination of multiple joints becomes more intricate and necessary to avoid factors such as link collisions, joint limits and singularity [10, 38, 51]. This means controlling the system becomes more difficult.

(3) The ability of the robot to move in arbitrary directions, called manipulability, increases [31, 58, 62]. This means the set of possible

target positions for the end-effector grows. This comes with high computational control effort [38], since the algorithms are burdened with learning more target positions.

Solving Equation 7 to compute joint motion requires algorithm complexity of $O(n)$ [24]. This illustrates how the computational costs of the dynamics scale with n-DoF. In general, it can be declared that controlling an *n+1-link* manipulator is inherently more difficult than controlling an *n-link* manipulator. In summary,

$$C(\textit{n-link manipulator}) < C(\textit{n+1-link manipulator}) \tag{8}$$

where $C(\cdot)$ denotes the hardness of controlling the system. Although Equation 8 is not quantitative, it is useful for sanity check. In the next section, ranking of tasks via Equation 8 will be compared with those provided by the earlier task complexity metrics.

4 EXPERIMENTAL EVALUATION

In this section, the methods described in Section 2 are evaluated in a class of reaching tasks[1]. The purpose of the experiments is to answer the following questions: (1) *Can the statistical analysis of the performance of random policies and PIC/POIC effectively capture the task complexity of reaching tasks?* (2) *Do task difficulty levels ranked by PIC/POIC align with the ranking suggested by Equation 8?*

Section 4.1 describes the experimental setup, while Section 4.2 introduces the task framework used for assessing the accuracy of the task complexity metrics. In Section 4.3, we train RL agents on the tasks defined within this framework and use their performance to verify the tasks' complexity. The resulting measures are then compared with known complexity rankings from robotics. Section 4.4 examines the limitations of the task complexity metrics, showing that PIC and POIC can inaccurately measure task complexity.

4.1 Experimental Setup

We employ manipulators shown in Figure 1 in six task settings (discussed in Section 4.2). For each task, both the end-effector and its target position are randomly initialised at the start of each episode. This is a good training practice that prevents environment overfitting when training RL agents [60]. Friction is ignored, states are assumed to be fully observable, and the arms are operated on a horizontal plane; hence, the effects of gravity are not considered.

The arm configurations are evaluated on dense- and sparse-rewards. In dense-reward settings, the reward function is

$$r = -\omega_1 \|P_{ee} - P_g\|_2^2 - \omega_2 \|action\|_2^2 \tag{9}$$

where $[\omega_1, \omega_2] = [1, 1]$ are distance and control weights. P_{ee} and P_g are respectively end-effector and target/goal positions. In sparse-reward settings, $r = 0$ when the end-effector is within the threshold distance from target (< 0.05 meters), otherwise $r = -1$. All the tasks have a maximum of 50 steps per episode, 500 training episodes and $N = 10^4$ samples (i.e. random policies). To match the experimental setup in [25], the policy network consists of 2 hidden layers with 32 neurons each and the number of discretisation bins $B = 10^5$.

[1]Code and Supplementary material are available at: https://github.com/nkhumisa-rea/task_complexity.git

4.2 Tasks framework

Our framework consists of six tasks that include three arm setups, each with dense- and sparse-rewards. The arm setups include: (1) *1-link* arm with link length $L = 1.00$ meter, (2) *1-link* arm with link length $L = 1.65$ meters, and (3) *2-link* arm with link lengths $L_1 = 0.95$ and $L_2 = 0.70$ meters. We ensured that the link lengths of *2-link* arm sum to 1.65 m, to have the same total length as the *1-link* arm in (2) above. This makes these two arms (2) and (3) have equivalent magnitude of error at the end-effector, leading to rewards that can be directly comparable, while varying in complexity only due to the number of links/joints.

The error in the end-effector arises from the errors in the arm joint angles being amplified by the link lengths, which results in higher positional errors at the end-effector for longer link lengths, as captured by (for small errors):

$$\|\delta x\|_2 = \left\| \sum_{k=1}^{n} J_k(\theta) \delta\theta_k \right\|_2 \leq \epsilon \sum_{i=1}^{n} i l_i \tag{10}$$

where δx is error at the end-effector, $\delta\theta_k$ is angle error of the $k-$th joint, and $J_k(\theta)$ is the $k-$th column of the Jacobian matrix. n is the number of DoF, while l_i is the $i-$th link on the arm. ϵ is the worst-case joint error, i.e. $|\delta\theta_k| \leq \epsilon$ for $k = 1, \cdots, n$ (see Supplementary material[1] for details on Equation 10). Generally, tasks with higher error rates present greater learning challenges for RL algorithms, thus resulting in reduced rewards or longer learning time [23, 59].

We selected the three arm configurations to study independently the effects of altering link lengths and number of joints. This simplifies task comparison and ensures task structural homogeneity. Following Equation 8, we expect *1-link* arm tasks to be easier than the *2-link* arm task under the same reward formulation. From Equation 10, we expect the *1-link* arm task with a shorter link length to be easier than the *1-link* arm task with a longer link length. Additionally, we expect consistency with RL literature [5, 45, 55], where tasks in dense-reward settings are easier than those in sparse-reward settings.

Importance of structurally similar tasks. Compared to most RL benchmarks, often the tasks are varied (as they should) but not structurally related [6, 21], e.g. Cartpole and MountainCar in OpenAI Gym [9]. As such, most benchmarks can be unsuitable for reliably assessing new task complexity methods. To ensure meaningful validation, methods should first be evaluated on families of closely related tasks with known relative complexity (such as our setup). This enables sanity checking, clearer interpretation of results, and easier characterisation of the proposed methods prior to applying them to large heterogeneous benchmarks.

We start by comparing learning curves across our task framework for a baseline algorithm, to confirm if these match our expectations.

4.3 Reinforcement Learning of Tasks

We compare learning curves of a state-of-the-art algorithm Soft Actor Critic (SAC) [26] across the tasks in our task framework. Studying how the algorithm performs during training provides us with information about convergence, such as the optimal return and convergence time for the tasks. The learning curves are presented

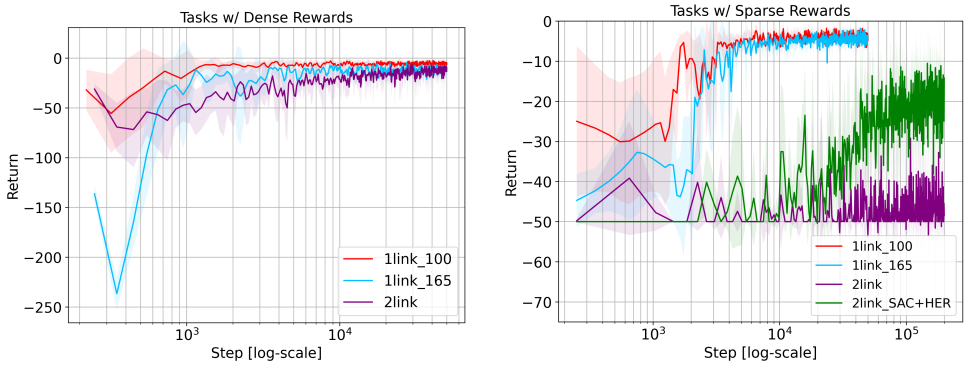

Figure 2: Learning curves of SAC algorithm across the six tasks. The left panel depicts agent performance in dense-reward settings, while the right panel is in sparse-reward settings. To accommodate wide and varying ranges of steps, results are plotted on a logarithmic scale to enhance interpretability. In the *2-link* arm with sparse rewards, SAC results are presented with HER [5] augmentation (SAC+HER) and without it. The results are obtained via evaluation of each task over 5 runs.

in Figure 2. Details about the SAC architecture and its hyperparameters are provided in the Supplementary material[1]. Note that the same network model for SAC was employed in all the tasks.

Dense-reward settings. We note in Figure 2 that *1-link* arm with link length $L = 1.0$ m converges faster and to a higher return than *1-link* arm with link length $L = 1.65$ m. Similarly, in the *1-link* arm with link length $L = 1.65$ m, the agent converges faster than in the *2-link* arm task. This highlights the order of tasks based on their hardness (from easiest to hardest) to be *1-link* arm ($L = 1$ m), *1-link* arm ($L = 1.65$ m) and *2-link* arm. This is consistent with our expectations as supported by Equations 8 and 10.

Sparse-reward settings. In Figure 2, we see that pure SAC is unable to solve the *2-link* arm task, showing that *2-link* arm task is harder than the *1-link* arm tasks, aligning with Equation 8. Even SAC augmented with Hindsight Experience Replay (HER) [5] takes significantly longer to converge in the *2-link* arm task compared with *1-link* arm tasks. Further, SAC converges faster in the *1-link* arm ($L = 1$ m) task than in *1-link* arm ($L = 1.65$ m) task. The ordering of these three arms is consistent with our expectations, in this sparse-reward settings as well.

Dense-reward vs Sparse-reward settings. By comparing the dense- and sparse-reward settings, we observe that the algorithm converges quicker in dense-reward settings than in sparse-reward settings. Furthermore, SAC performance is noisier in the sparse-reward settings than its counterpart. This coincides with intuition that tasks with dense rewards are easier than with sparse rewards.

REMARK 1. *In our settings, SAC(+HER) could solve the tasks (verified by demonstrations). However, in general, algorithms may fail to solve tasks or perform optimally. This restricts the usage of learning curves in assessing task complexity.*

In the subsequent section, we compare methods introduced in Section 2 for quantifying task complexity. These methods are independent of specific RL algorithms.

4.4 Task Complexity Analysis

In this section, we categorise our results into examining how task complexity is influenced by a) link length, b) number of DoF, and c) reward formulation. We compare measures of task complexity beginning with statistical analysis of performance, and then following with PIC and POIC.

(I) Statistical analysis of performance. To carry out this analysis, we used Algorithm 1 to capture the cumulative rewards (returns or performance) of $N = 10^4$ randomly sampled policies via RWG [43]. The performance was then aggregated into mean M_n and variance V_n using Equations 1 and 2. We visualise the aggregated performance in three plots: *mean performance histograms* (Log-scale histogram of M_n), *mean performance curve* (M_n vs R_n plot), and *variance distribution* ($\sqrt{V_n}$ vs M_n plot). Note that R_n is rank, from lowest to highest mean performance.

Figure 3 presents the performance plots, where the left, middle and right columns, respectively, depict the *mean performance histograms*, *mean performance curves*, and *variance distributions*. We normalised the mean M_n and variance V_n in each task to avoid scale-induced bias and ensure commensurability of performance [2, 11]. We used *min-max scaling* [36],

$$x' = \frac{x - \min[x]}{\max[x] - \min[x]} \tag{11}$$

where x is the variable being scaled.

Overall description. We notice in Figure 3 that the histograms (across the tasks) have an overall shape that approximates a Gaussian distribution that does not span the entire range of mean scores. The performance curves (black curves) have smooth slopes without jumps, and the variance distributions mostly reveal that performance consistency is not uniform across the mean score range. We discuss the plots in detail below.

L = 1.0 vs L = 1.65m (*1-link*, dense-rewards). In dense-reward settings for *1-link* arms, where one has $L = 1$ m and the other has $L = 1.65$ m, we make the following observations,

Mean Performance histograms: In both tasks, no single random policy achieved mean performance M_n near the maximum return. This exhibits that the tasks are not trivial [43]. The performance distributions in both arms are similar. This shows that the tasks are structurally equivalent. This aligns with intuition, as the link

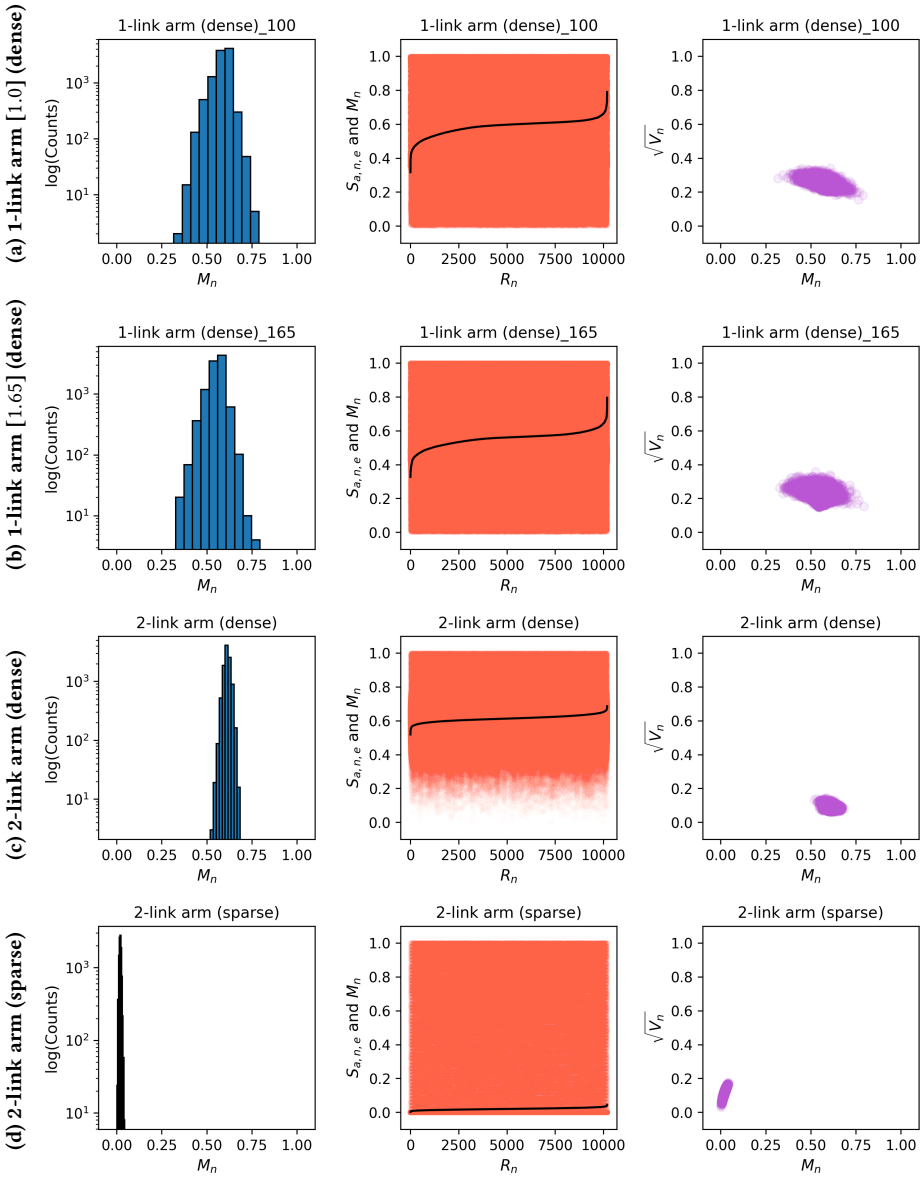

Figure 3: Performance distribution plots for the tasks: (a) *1-link (L=1.0m)*, (b) *1-link (L=1.65m)*, (c) *2-link* arms with dense rewards, and (d) *2-link* arm with sparse rewards. The left column shows a histogram of mean performances of the random policies (*Log-scale histogram of* M_n). The middle column depicts *mean performance curves* in black, i.e. mean performance M_n vs rank R_n. Moreover, all the cumulative rewards of the policies $S_{a,n,e}$ across the trials are represented by red dots (behind the black curve), where a represents the neural network architecture (same for all panels). The right column displays plots of standard deviation $\sqrt{V_n}$ vs mean performance M_n (often referred to as *variance distribution*). The plots were made using 10^4 random policies.

lengths impacts reward scales (see Figure 2), but do not alter the task structure, as seen after normalisation of rewards.

Performance curves: Interestingly in both tasks, the random policies managed to attain episodic cumulative rewards $S_{n,e}$ that are nearly the maximum return (shown by red dots at 1.0) in a few episodes. This is sensible since both the initial end-effector and target positions are random in every episode. With 500 episodes, it is likely that initial and target positions of the end-effector were in close proximity in some episodes — which simplifies the task in those episodes.

Variance distributions: Both arms have similar variance distributions, with wide spread about the middle mean score (i.e. $M_n = 0.5$), which slowly narrows towards the limits of the range of mean performance. This indicates that the majority of policies attain their

mean performance by succeeding on some episodes and failing at others, leading to higher variance of scores across episodes. However, policies seem to consistently fail for lower mean scores and consistently succeed for higher mean scores, leading to lower variance [43]. Note that apart from Figure 3 revealing that the *1-link* arm tasks are similar, it is not clear which task is easier or harder between the two.

1-Link vs 2-Link (dense rewards). Figure 3 (2^{nd} and 3^{rd} rows) displays how the histogram has a narrower width for *2-link* arm (than for *1-link* arm) while the median M_n nearly remains consistent across the tasks. The performance curve in *2-link* arm has a low slope than that in the *1-link* arm task. Furthermore, the variance of the performance in the *2-link* arm task is smaller. These denote that the *2-link* arm task is harder than the *1-link* arm task. The reason is that harder tasks often provide higher rewards only when a coherent sequence of successful actions is executed, which untrained random policies are unlikely to achieve, hence reduced variability in performance.

Dense vs Sparse rewards (2-link). Figure 3 (3^{rd} and 4^{th} rows) portrays a drastic drop in peak M_n and variance V_n, from dense- to sparse-reward settings. The performance curve slope further decreased (almost zero) in the sparse-reward setting compared to the dense-reward setting. This highlights a lack of diversity in the performance of the random policies. In the variance plots for the sparse-reward setting, we notice that random policies fail to succeed in the task regardless of initial conditions. We can conclude from these results that the dense-reward setting is easier than the sparse-reward setting.

REMARK 2. *Although the statistical analysis and visualisation of performance provide some insights about the task characteristics and relative hardness, they fail to quantitatively measure task difficulty, i.e. the approach is qualitative. This makes it inapplicable to RL benchmarks and curriculum learning, where relative hardness amongst tasks needs to be quantified. Moreover, performance distributions that are similar across tasks can potentially make the plots less informative in comparing the tasks. For these reasons, we now examine quantitative metrics PIC and POIC.*

(II) PIC/POIC. The quantitative representations of task complexity offered by PIC and POIC are exhibited for our six tasks in Table 1. We checked the statistical robustness of the results in Table 1 by quantifying their uncertainty using bootstrapped confidence intervals [20]. These estimate the uncertainty by repeatedly resampling the data. In our context, the data are episodic cumulative rewards of random policies constructed via RWG. We resampled the data 1,000 times with replacement and computed the values presented in Table 1.

We then applied the Welch's t-test [19] to evaluate the statistical significance in the differences between values in Table 1, using the same 1,000 resamples. Consistently in all cases, the p-values of the t-statistic of the Welch's t-test are in the orders of 10^{-5}, below a typical cut-off p-value = 0.005 which indicate strong statistical significance [8]. Outcomes of the Welch's t-test and the confidence intervals of PIC and POIC can be found in Supplementary material[1].

It should be noted that the values in Table 1 were gathered using a policy network of 2 hidden layers, each with 32 neurons. 10^4 untrained policies were sampled from a multivariate normal prior distribution. We also confirmed that RWG sampled from different prior distributions and policy network architectures did not change our results (see Supplementary materials[1]).

Table 1: PIC and POIC values with $N = 10^4$ samples (random policies). High PIC and POIC values correspond to easier tasks, while low values correspond to harder tasks.

Rewards	Arm [dim]	PIC $(\times 10^{-3})$	POIC $(\times 10^{-3})$
Dense	1-link [1.0]	4005 ± 8.5	2.628 ± 0.056
	1-link [1.65]	4153 ± 8.4	4.105 ± 0.085
	2-link [0.95,1.7]	4200 ± 6.1	0.725 ± 0.011
Sparse	1-link [1.0]	85.11 ± 0.4	1.958 ± 0.034
	1-link [1.65]	71.21 ± 0.2	1.197 ± 0.031
	2-link [0.95,1.7]	45.95 ± 0.0	0.946 ± 0.0079

Dense-reward settings. According to the PIC values under dense-reward settings (in Table 1), the *2-link* arm task is the easiest task (highest PIC) and *1-link* arm ($L = 1$ m) task is the hardest task (lowest PIC). This contradicts expectations based on Equations 8 and 10, and our empirical RL results. For instance, we showed using learning curves of trained agents that the *2-link* arm task is the hardest, while *1-link* arm ($L = 1$ m) task is the easiest. This is further corroborated by performance distributions of random policies in Figure 3. On the POIC side, *1-link* arm ($L = 1.65$ m) task is easier than *1-link* arm ($L = 1$ m) task. This does not align with Equation 10.

Sparse-reward settings. In these settings, both the PIC and POIC order of task difficulty across the tasks seems correct. The tasks are ordered from easiest to hardest as *1-link* arm ($L = 1$ m), *1-link* arm ($L = 1.65$ m) and *2-link* arm. This aligns with our intuition.

When we compare across dense- and sparse-reward settings, POIC values for the *2-link* arm task suggest that the dense-reward setting is harder than the sparse-reward setting. In this instance, POIC values contradict our expectations, as we showed empirically in Section 4.3 that dense-reward settings are easier than the sparse-reward settings. This inconsistency of PIC and POIC is further observed in an additional task setting that places an obstacle in the workspace of the *2-link* arm with dense rewards. Details are provided in Supplementary material[1] (Section D).

To investigate these incorrect PIC and POIC instances, we decomposed the individual entropy terms in the metrics. Figure 4 displays normalised scores (i.e. performance) against POIC, PIC, individual entropy terms, and the variance of cumulative rewards. The normalised scores use *min-max scaling* (Equation 11) over the performance samples of the random policies.

POIC related plots. The first three columns in Figure 4 portray POIC and entropies of the optimality variable. We observe that $\hat{\mathcal{H}}(O)$ and $\hat{\mathcal{H}}(O \mid \Theta)$ are closely approximate, which produces POIC $\hat{I}(O;\Theta)$ values of small magnitude, similar to the work that introduced POIC [25]. There are multiple strong linear correlations between normalised scores and the other quantities (given by Pearson correlation coefficients above the plots), however they are not statistically significant.

PIC related and Variance plots. In Figure 4, the last four columns display PIC, entropies of the cumulative reward variable, and variance of returns. We note that $\hat{\mathcal{H}}(R)$ and $\hat{\mathcal{H}}(R \mid \Theta)$ differ. This is responsible for larger magnitude values of PIC $\hat{I}(R;\Theta)$.

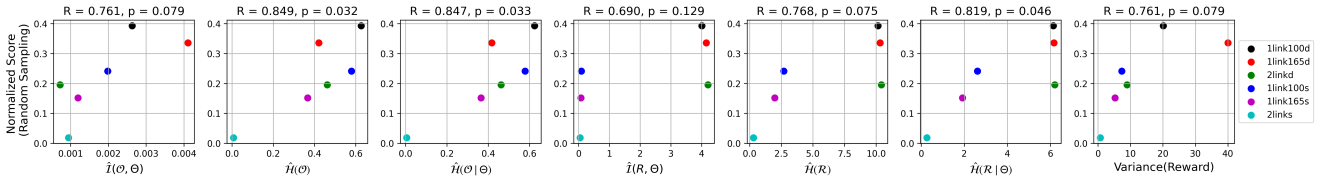

Figure 4: 2D-scatter plots with Normalised scores (performance) computed using *min-max scaling* (Equation 11) over the returns of untrained random policies. The Normalised scores are plotted against PIC, POIC, variance of returns, along with entropies of optimality variable and cumulative reward (return) variable.

It seems dense-reward settings have more variability in returns, than sparse-reward settings. This aligns with results presented in Figure 3. It seems generally that in our setup, tasks with dense-rewards enjoy higher normalised scores than sparse-rewards coinciding with our expectations. Figure 4 does not provide further insights about why PIC and POIC values in Table 1 do not match expectations. We discuss further possibilities in the next section.

5 DISCUSSION & LIMITATIONS

Finally, we explore potential reasons for the inconsistencies observed with RWG-based metrics PIC and POIC compared to expectations derived from robotic control and verified with empirical RL. These issues stem from (1) the dependence on randomly generated parameters on the prior distribution $p(\theta)$, and (2) the lack of consideration for training and exploration.

While statistical analysis of performance of RWG-generated policies has been consistent with intuition, it can be challenging to effectively communicate the degree of difference in task difficulty. Moreover, if tasks produce nearly similar performance distributions, then this approach might be less informative for comparative analysis.

PIC and POIC are dependent on the prior distribution of parameters $p(\theta)$, as highlighted in [25]. The prior $p(\theta)$ can be interpreted as the *effective search area* in the parameter (policy) space [3]. For problems where high-performing policies are sparsely distributed in the parameter space, the effective search area is likely to cover mainly low-performing regions. Policies sampled from these regions are limited to yielding mean performance M_n far from the peak return as shown in Figure 3. This observation reinforces the limitation of RWG originally noted by [48], namely its ineffectiveness in tasks with sparse solution regions in the weight space.

Another limitation arises from the fact that RWG does not involve training. This implies that the effective search area remains static once $p(\theta)$ is selected. In contrast, training involves exploration of the policy space [34, 41, 55] — where the effective search area (of the learning algorithm) is dynamically moved around in the policy space. It is also important to note that neither statistical analysis of the performance of random policies, nor PIC and POIC metrics, consider the visitation complexity [13] of the tasks, which measures the difficulty in exploring the state space of the environment. This implies that the way actions influence state transitions during exploration in the learning phase is not accounted for by these task complexity methods [25]. Several methods that aim to capture exploration effort [4, 34, 41] have been investigated; however, none have been applied to task complexity. This makes for an interesting future direction of work.

It is clear from our experimental results that PIC and POIC can be misleading in capturing task complexity. Inconsistencies in these metrics can be challenging to notice in most RL benchmarks, especially if they have tasks with heterogeneous structure. Our task framework offers tasks with structural homogeneity and known relative task complexity, thus enabling a more reliable assessment of these metrics.

The results presented in this article showcase the need for continued work in task complexity for deep RL, especially for the case of robotic tasks, where our task framework could be a starting point. We propose the following directions for improving PIC and POIC:

(1) A key drawback of the PIC/POIC metrics is the limited effective search space sampled by RWG. We can replace the standard multilayer perceptron (MLP) policy with an architecture which introduces parameterised inductive biases relevant to the tasks. These biases are aimed at maximally covering the state-action space. By setting the parameters of these inductive biases via RWG, we can cover a wider search space. In robotics, the inductive bias can be a policy composed of dynamic movement primitives [54], skills [18] or normalising flows [32]. The drawback of this approach is that the resulting complexity measure would be dependent on the selected inductive biases.

(2) A second issue with PIC and POIC metrics is that they have a static effective search area in the parameter space due to the lack of exploration. In devising new metrics, we want the effective search area to be dynamic by including exploration. We can perform RWG after every k updates of a RL algorithm and compute corresponding PIC/POIC values at each stage. Ultimately, use the mean PIC/POIC values across the entire learning path as the metric for task complexity. This would entail sampling weights via RWG, at every k update of the policy during RL training θ_n using $\theta_n = \theta_k + \epsilon, \epsilon \sim \mathcal{N}(0, I)$ where θ_k are parameters after the k^{th} policy model update. With every RWG set, we compute PIC_k, and finally, determine the average across the trajectory and use it as a measure of task complexity. The drawback of this approach is that the task complexity metric(s) would be dependent on the exploration strategy of the learning algorithm.

(3) When the optimal policy is known, we can compute the distribution of distances of any RWG policy to the optimal policy using optimal transport [41]. Task complexity could then be defined by the mean and variance of these distances, thereby capturing the expected effort required to move from random policies to the optimal policy. The limitation of this method is that it requires prior knowledge of the optimal policy.

In principle, all three of these suggestions could be combined.

ACKNOWLEDGMENTS

R. Nkhumise was supported by the EPSRC Doctoral Training Partnership (DTP) - Early Career Researcher funding awarded to A. Gilra. A. Gilra acknowledges the CHIST-ERA grant for the Causal Explanations in Reinforcement Learning (CausalXRL) project (CHIST-ERA-19-XAI-002), by the Engineering and Physical Sciences Research Council, United Kingdom (grant reference EP/V055720/1) for supporting the work.

REFERENCES

[1] D. Abel, C. Allen, D. Arumugam, D. E. Hershkowitz, M. L. Littman, and L. L. S. Wong. 2021. Bad-policy density: A measure of reinforcement learning hardness. *arXiv preprint arXiv:2110.03424* (2021). arXiv:2110.03424

[2] R. Agarwal, M. Schwarzer, P. S. Castro, A. C. Courville, and M. Bellemare. 2021. Deep reinforcement learning at the edge of the statistical precipice. *Advances in neural information processing systems* 34 (2021), 29304–29320.

[3] M. Aleksandrowicz and J. Jaworek-Korjakowska. 2023. Metrics for assessing generalization of deep reinforcement learning in parameterized environments. *JAISCR* 14, 1 (2023), 45–61.

[4] S. Amin, M. Gomrokchi, H. Satija, H. van Hoof, and D. Precup. 2021. A Survey of Exploration Methods in Reinforcement Learning. *arXiv preprint arXiv:2109.00157* (2021). arXiv:2109.00157

[5] M. Andrychowicz, F. Wolski, A. Ray, J. Schneider, R. Fong, P. Welinder, B. McGrew, J. Tobin, OpenAI A. Pieter Abbeel, and W. Zaremba. 2017. Hindsight Experience Replay. In *Advances in Neural Information Processing Systems*, Vol. 30. Curran Associates, Inc.

[6] M. G. Bellemare, Y. Naddaf, J. Veness, and M. Bowling. 2013. The arcade learning environment: An evaluation platform for general agents. *Journal of artificial intelligence research* 47 (2013), 253–279.

[7] Y. Bengio, J. Louradour, R. Collobert, and J. Weston. 2009. Curriculum learning. In *Proceedings of the 26th annual international conference on machine learning*. 41–48.

[8] D. J. Benjamin, J. O. Berger, M. Johannesson, B. A. Nosek, E. J. Wagenmakers, R. Berk, K. A. Bollen, B. Brembs, L. Brown, C. Camerer, et al. 2018. Redefine statistical significance. *Nature human behaviour* 2, 1 (2018), 6–10.

[9] G. Brockman, V. Cheung, L. Pettersson, J. Schneider, J. Schulman, J. Tang, and W. Zaremba. 2016. OpenAI Gym. *arXiv preprint arXiv:1606.01540* (2016). arXiv:1606.01540 https://www.gymlibrary.dev/environments/classic_control/mountain_car/

[10] S. Chiaverini. 2002. Singularity-robust task-priority redundancy resolution for real-time kinematic control of robot manipulators. *IEEE Transactions on Robotics and Automation* 13, 3 (2002), 398–410.

[11] K. Cobbe, C. Hesse, J. Hilton, and J. Schulman. 2020. Leveraging procedural generation to benchmark reinforcement learning. In *International conference on machine learning*.

[12] K. Cobbe, O. Klimov, C. Hesse, T. Kim, and J. Schulman. 2019. Quantifying generalization in reinforcement learning. In *International conference on machine learning*.

[13] M. Conserva and P. Rauber. 2022. Hardness in Markov Decision Processes: Theory and Practice. *36th Conference on Neural Information Processing Systems* (2022).

[14] M. Conserva, R. Sasso, and P. Rauber. 2025. On the Limits of Tabular Hardness Metrics for Deep RL: A Study with the Pharos Benchmark. *arXiv preprint arXiv:2509.17092* (2025). arXiv:2509.17092

[15] C. Copot, C. Muresan, C.-M. Ionescu, S. Vanlanduit, and R. De Keyser. 2018. Calibration of UR10 robot controller through simple auto-tuning approach. *Robotics* 7, 3 (2018). https://www.mdpi.com/2218-6581/7/3/35

[16] P. Corke. 2011. *Robotics, vision and control*. Springer Berlin, Heidelberg.

[17] T. M. Cover and J. A. Thomas. 2006. *Elements of Information Theory* (2nd ed.). John Wiley & Sons, Ltd.

[18] M. Dalal, D. Pathak, and R. R. Salakhutdinov. 2021. Accelerating robotic reinforcement learning via parameterized action primitives. *Advances in Neural Information Processing Systems* 34 (2021), 21847–21859.

[19] M. Delacre, D. Lakens, and C. Leys. 2017. Why psychologists should by default use Welch's t-test instead of Student's t-test. *International Review of Social Psychology* 30, 1 (2017), 92–101.

[20] T. J. DiCiccio and B. Efron. 1996. Bootstrap confidence intervals. *Statistical science* 11, 3 (1996), 189–228.

[21] Y. Duan, X. Chen, R. Houthooft, J. Schulman, and P. Abbeel. 2016. Benchmarking Deep Reinforcement Learning for Continuous Control. *Proceedings of the 33 rd International Conference on Machine Learning* 48 (2016), 1329––1338.

[22] G. Dulac-Arnold, N. Levine, D. J. Mankowitz, J. Li, C. Paduraru, S. Gowal, and T. Hester. 2020. An empirical investigation of the challenges of real-world reinforcement learning. *arXiv preprint arXiv:2003.11881* (2020). arXiv:2003.11881

[23] A. Edmondson and R. P. A. Petrick. 2025. Navigating Errors: The Tolerance of Reinforcement Learning Algorithms to Misleading Heuristics. *Association for the Advancement of Artificial Intelligence* (2025).

[24] R. Featherstone. 2008. *Rigid body dynamics algorithms*. Springer.

[25] H. Furuta, T. Matsushima, T. Kozuno, Y. Matsuo, S. Levine, O. Nachum, and S. S. Gu. 2021. Policy Information Capacity: Information-Theoretic Measure for Task Complexity in Deep Reinforcement Learning. *Proceedings of the 38th International Conference on Machine Learning* 139 (2021), 3541–3552.

[26] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine. 2018. Soft Actor-Critic: Off-Policy Maximum Entropy Deep Reinforcement Learning with a Stochastic Actor. In *Proceedings of the 35th International Conference on Machine Learning*.

[27] A. Hentout, A. Maoudj, and M. Aouache. 2023. A review of the literature on fuzzy-logic approaches for collision-free path planning of manipulator robots. *Artificial Intelligence Review* 56, 4 (2023), 3369–3444.

[28] J. Ibarz, J. Tan, C. Finn, M. Kalakrishnan, P. Pastor, and S. Levine. 2021. How to train your robot with deep reinforcement learning: lessons we have learned. *The International Journal of Robotics Research* 40, 4-5 (2021), 698–721.

[29] M. Jaderberg, W. M. Czarnecki, I. Dunning, L. Marris, G. Lever, A. G. Castaneda, C. Beattie, N. C. Rabinowitz, A. S. Morcos, A. Ruderman, N. Sonnerat, T. Green, L. Deason, J. Z. Leibo, D. Silver, D. Hassabis, K. Kavukcuoglu, and T. Graepel. 2019. Human-level performance in 3D multiplayer games with population-based reinforcement learning. *Science* 364, 6443 (2019), 859–865.

[30] N. Justesen, R. R. Torrado, P. Bontrager, A. Khalifa, J. Togelius, and S. Risi. 2018. Illuminating generalization in deep reinforcement learning through procedural level generation. *arXiv preprint arXiv:1806.10729* (2018). arXiv:1806.10729

[31] M. Khadem, L. Da Cruz, and C. Bergeles. 2018. Force/velocity manipulability analysis for 3d continuum robots. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 4920–4926.

[32] S. A. Khader, H. Yin, P. Falco, and D. Kragic. 2021. Learning stable normalizing-flow control for robotic manipulation. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 1644–1650.

[33] P. Kormushev, S. Calinon, and D. G. Caldwell. 2013. Reinforcement learning in robotics: Applications and real-world challenges. *Robotics* 2, 3 (2013), 122–148.

[34] P. Ladosz, L. Weng, M. Kim, and H. Oh. 2022. Exploration in Deep Reinforcement Learning: A Survey. *Information Fusion* 85 (2022), 1–22.

[35] M. T. Mason. 2018. Toward robotic manipulation. *Annual Review of Control, Robotics, and Autonomous Systems* 1, 1 (2018), 1–28.

[36] K. P. Murphy. 2012. *Machine learning: a probabilistic perspective*. The MIT Press.

[37] R. M. Murray, Z. Li, and S. S. Sastry. 2017. *A mathematical introduction to robotic manipulation*. CRC press.

[38] J. Nakanishi, R. Cory, M. Mistry, J. Peters, and S. Schaal. 2008. Operational space control: A theoretical and empirical comparison. *The International Journal of Robotics Research* 27, 6 (2008), 737–757.

[39] S. Narvekar, B. Peng, M. Leonetti, J. Sinapov, M. E. Taylor, and P. Stone. 2020. Curriculum learning for reinforcement learning domains: A framework and survey. *Journal of Machine Learning Research* 21, 181 (2020), 1–50.

[40] S. S. Narvekar. 2021. *Curriculum learning in reinforcement learning*. Ph.D. Dissertation. The University of Texas at Austin. Order Number: 29605085.

[41] R. M. Nkhumise, D. Basu, T. J. Prescott, and A. Gilra. 2025. Studying Exploration in RL: An Optimal Transport Analysis of Occupancy Measure Trajectories. *Transactions on Machine Learning Research (TMLR)* (2025).

[42] J. Obando-Ceron, J. G. M. Araújo, A. Courville, and P. S. Castro. 2024. On the consistency of hyper-parameter selection in value-based deep reinforcement learning. In *Reinforcement Learning Conference (RLC)*. Reinforcement Learning Journal (RLJ).

[43] D. Oller, T. Glasmachers, and G. Cuccu. 2020. Analyzing reinforcement learning benchmarks with random weight guessing. In *Proceedings of the 19th International Conference on Autonomous Agents and MultiAgent Systems*. 975–982.

[44] I. Osband, Y. Doron, M. Hessel, J. Aslanides, E. Sezener, A. Saraiva, K. McKinney, T. Lattimore, C. Szepesvari, S. Singh, B. van Roy, R. Sutton, D. Silver, and H. van Hassel. 2019. Behaviour suite for reinforcement learning. *arXiv preprint arXiv:1908.03568* (2019). arXiv:1908.03568

[45] D. Pathak, P. Agrawal, A. A. Efros, and T. Darrell. 2017. Curiosity-driven exploration by self-supervised prediction. In *International conference on machine learning*. PMLR, 2778–2787.

[46] R. Rajan, J. L. B. Diaz, S. Guttikonda, F. Ferreira, A. Biedenkapp, J. O. von Hartz, and F. Hutter. 2023. MDP playground: An analysis and debug testbed for reinforcement learning. *Journal of Artificial Intelligence Research* 77 (2023), 821–890.

[47] R. Rocchetta, L. Bellani, M. Compare, E. Zio, and E. Patelli. 2019. A reinforcement learning framework for optimal operation and maintenance of power grids. *Applied energy* 241 (2019), 291–301.

[48] J. Schmidhuber, S. Hochreiter, and Y. Bengio. 1999. Evaluating benchmark problems by random guessing. In *A Field Guide to Dynamical Recurrent Networks*. Wiley, 1329––1338.

[49] L. Sciavicco and B. Siciliano. 2012. *Modelling and control of robot manipulators*. Springer.

[50] B. Siciliano, O. Khatib, and T. Kröger. 2008. *Springer handbook of robotics*. Vol. 200. Springer.

[51] B. Siciliano, L. Sciavicco, L. Villani, and G. Oriolo. 2009. *Robotics: modelling, planning and control*. Springer.
[52] D. Silver, J. Schrittwieser, K. Simonyan, I. Antonoglou, A. Huang, A. Guez, T. Hubert, L. Baker, M. Lai, A. Bolton, Y. Chen, T. Lillicrap, F. Hui, L. Sifre, G. van den Driessche, T. Graepel, and D. Hassabis. 2017. Mastering the game of go without human knowledge. *nature* 550, 7676 (2017), 354–359.
[53] M. W. Spong, S. Hutchinson, and M. Vidyasagar. 2006. *Robot modeling and control*. Vol. 3. John Wiley & Sons.
[54] F. Stulp and S. Schaal. 2011. Hierarchical reinforcement learning with movement primitives. In *2011 11th IEEE-RAS International Conference on Humanoid Robots*. IEEE, 231–238.
[55] R. S. Sutton and A. G. Barto. 2018. *Reinforcement Learning: An Introduction* (2 ed.). MIT Press.
[56] Y. Tassa, Y. Doron, A. Muldal, T. Erez, Y. Li, D. de Las Casas, D. Budden, A. Abdolmaleki, J. Merel, A. Lefrancqm, T. Lillicrap, and M. Reidmiller. 2018. Deepmind control suite. *arXiv preprint arXiv:1801.00690* (2018). arXiv:1801.00690
[57] G. Thomas, M. Chien, A. Tamar, J. A. Ojea, and P. Abbeel. 2018. Learning robotic assembly from cad. In *IEEE International Conference on Robotics and Automation (ICRA)*. 3524–3531.
[58] N. Vahrenkamp, T. Asfour, G. Metta, G. Sandini, and R. Dillmann. 2012. Manipulability analysis. In *12th IEEE-RAS international conference on humanoid robots (humanoids 2012)*. IEEE, 568–573.
[59] J. Wang, Y. Liu, and B. Li. 2020. Reinforcement learning with perturbed rewards. In *Proceedings of the AAAI conference on artificial intelligence*, Vol. 34. 6202–6209.
[60] S. Whiteson, B. Tanner, M. E. Taylor, and P. Stone. 2011. Protecting against evaluation overfitting in empirical reinforcement learning. In *2011 IEEE symposium on adaptive dynamic programming and reinforcement learning (ADPRL)*. IEEE, 120–127.
[61] T. Yang, L. Zhao, W. Li, and A. Y. Zomaya. 2020. Reinforcement learning in sustainable energy and electric systems: A survey. *Annual Reviews in Control* 49 (2020), 145–163.
[62] T. Yoshikawa. 1985. Manipulability of robotic mechanisms. *The international journal of Robotics Research* 4, 2 (1985), 3–9.