Robot Learning from Randomized Simulations: A Review

Fabio Muratore, Fabio Ramos, Wenhao Yu, Greg Turk, Michael Gienger, Jan Peters

2022

Preprint:

This is an accepted article published in Frontiers Robotics and AI. The final authenticated version is available online at:
https://doi.org/10.3389/frobt.2022.799893

frontiers | Frontiers in Robotics and AI

REVIEW
published: 11 April 2022
doi: 10.3389/frobt.2022.799893

Robot Learning From Randomized Simulations: A Review

Fabio Muratore[1,2]*, *Fabio Ramos*[3,4], *Greg Turk*[5], *Wenhao Yu*[6], *Michael Gienger*[2] and *Jan Peters*[1]

[1]Intelligent Autonomous Systems Group, Technical University of Darmstadt, Darmstadt, Germany, [2]Honda Research Institute Europe, Offenbach am Main, Germany, [3]School of Computer Science, University of Sydney, Sydney, NSW, Australia, [4]NVIDIA, Seattle, WA, United States, [5]Georgia Institute of Technology, Atlanta, GA, United States, [6]Robotics at Google, Mountain View, CA, United States

The rise of deep learning has caused a paradigm shift in robotics research, favoring methods that require large amounts of data. Unfortunately, it is prohibitively expensive to generate such data sets on a physical platform. Therefore, state-of-the-art approaches learn in simulation where data generation is fast as well as inexpensive and subsequently transfer the knowledge to the real robot (sim-to-real). Despite becoming increasingly realistic, all simulators are by construction based on models, hence inevitably imperfect. This raises the question of how simulators can be modified to facilitate learning robot control policies and overcome the mismatch between simulation and reality, often called the "reality gap." We provide a comprehensive review of sim-to-real research for robotics, focusing on a technique named "domain randomization" which is a method for learning from randomized simulations.

Keywords: robotics, simulation, reality gap, simulation optimization bias, reinforcement learning, domain randomization, sim-to-real

OPEN ACCESS

Edited by:
Antonio Fernández-Caballero, University of Castilla-La Mancha, Spain

Reviewed by:
Akansel Cosgun, Monash University, Australia
Konstantinos Chatzilygeroudis, University of Patras, Greece

***Correspondence:**
Fabio Muratore
fabio@robot-learning.de

Specialty section:
This article was submitted to Robot Learning and Evolution, a section of the journal Frontiers in Robotics and AI

Received: *22 October 2021*
Accepted: *21 January 2022*
Published: *11 April 2022*

Citation:
Muratore F, Ramos F, Turk G, Yu W, Gienger M and Peters J (2022) Robot Learning From Randomized Simulations: A Review. Front. Robot. AI 9:799893. doi: 10.3389/frobt.2022.799893

1 INTRODUCTION

Given that machine learning has achieved super-human performance in image classification (Ciresan et al., 2012; Krizhevsky et al., 2012) and games (Mnih et al., 2015; Silver et al., 2016), the question arises why we do not see similar results in robotics. There are several reasons for this. First, learning to act in the physical world is orders of magnitude more difficult. While the data required by modern (deep) learning algorithms could be acquired directly on a real robot (Levine et al., 2018), this solution is too expensive in terms of time and resources to scale up. Alternatively, the data can be generated in simulation faster, cheaper, safer, and with unmatched diversity. In doing so, we have to cope with unavoidable approximation errors that we make when modeling reality. These errors, often referred to as the "reality gap," originate from omitting physical phenomena, inaccurate parameter estimation, or the discretized numerical integration in typical solvers. Compounding this issue, state-of-the-art (deep) learning methods are known to be brittle (Szegedy et al., 2014; Goodfellow et al., 2015; Huang et al., 2017), that is, sensitive to shifts in their input domains. Additionally, the learner is free to exploit the simulator, overfitting to features which do not occur in the real world. For example, Baker et al. (2020) noticed that the agents learned to exploit the physics engine to gain an unexpected advantage. While this exploitation is an interesting observation for studies made entirely in simulation, it is highly undesirable in sim-to-real scenarios. In the best case, the reality gap manifests itself as a performance drop, giving a lower success rate or reduced tracking accuracy. More likely, the learned policy is not transferable to the robot because of unknown physical effects. One effect that is difficult to model is friction, often leading to an underestimation thereof in

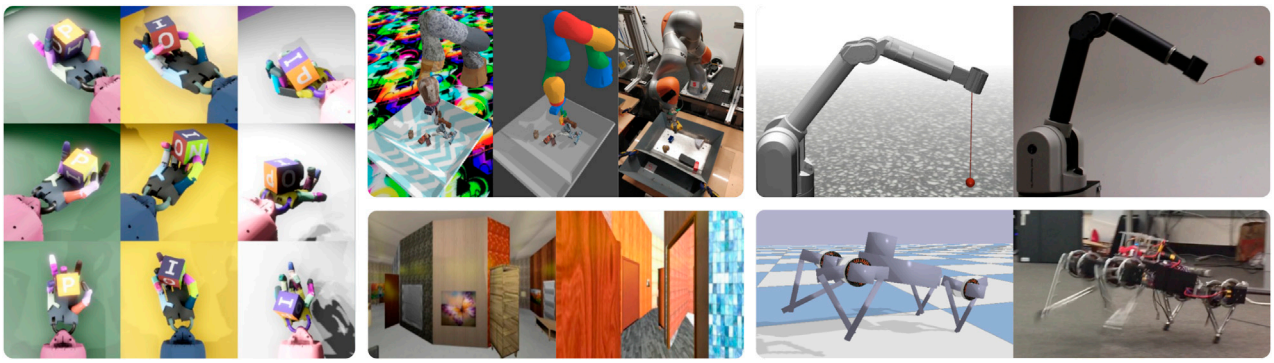

FIGURE 1 | Examples of sim-to-real robot learning research using domain randomization: (left) Multiple simulation instances of robotic in-hand manipulation (OpenAI et al., 2020), (middle top) transformation to a canonical simulation (James et al., 2019), (middle bottom) synthetic 3D hallways generated for indoor drone flight (Sadeghi and Levine, 2017), (right top) ball-in-a-cup task solved with adaptive dynamics randomization (Muratore et al., 2021a), (right bottom) quadruped locomotion (Tan et al., 2018).

simulation, which can result in motor commands that are not strong enough to get the robot moving. Another reason for failure are parameter estimation errors, which can quickly lead to unstable system dynamics. This case is particularly dangerous for the human and the robot. For these reasons, bridging the reality gap is the essential step to endow robots with the ability to learn from simulated experience.

There is a consensus that further increasing the simulator's accuracy alone will not bridge this gap (Höfer et al., 2020). Looking at breakthroughs in machine learning, we see that deep models in combination with large and diverse data sets lead to better generalization (Russakovsky et al., 2015; Radford et al., 2019). In a similar spirit, a technique called domain randomization has recently gained momentum (**Figure 1**). The common characteristic of such approaches is the perturbation of simulator parameters, state observations, or applied actions. Typical quantities to randomize include the bodies' inertia and geometry, the parameters of the friction and contact models, possible delays in the actuation, efficiency coefficients of motors, levels of sensor noise, as well as visual properties such as colors, illumination, position and orientation of a camera, or additional artifacts to the image (e.g., glare). Domain randomization can be seen as a regularization method that prevents the learner from overfitting to individual simulation instances. From the Bayesian perspective, we can interpret the distribution over simulators as a representation of uncertainty.

In this paper, we first introduce the necessary nomenclature and mathematical fundamentals for the problem (**Section 2**). Next, we review early approaches for learning from randomized simulations, state the practical requirements, and describe measures for sim-to-real transferability (**Section 3**). Subsequently, we discuss the connections between research on sim-to-real transfer and related fields (**Section 4**). Moreover, we introduce a taxonomy for domain randomization and categorize the current state of the art (**Section 5**). Finally, we conclude and outline possible future research directions (**Section 6**). For those who want to first become more familiar with robot policy learning as well as policy search, we recommend these surveys: Kober et al. (2013), Deisenroth et al. (2013), and Chatzilygeroudis et al. (2020).

2 PROBLEM FORMULATION AND NOMENCLATURE

We begin our discussion by defining critical concepts and nomenclature used throughout this article.

Markov Decision Processes (MDPs): Consider a discrete-time dynamical system

$$\boldsymbol{s}_{t+1} \sim \mathcal{P}_{\xi}(\boldsymbol{s}_{t+1}|\boldsymbol{s}_t, \boldsymbol{a}_t), \quad \boldsymbol{s}_0 \sim \mu_{\xi}(\boldsymbol{s}_0), \quad \boldsymbol{a}_t \sim \pi_{\boldsymbol{\theta}}(\boldsymbol{a}_t|\boldsymbol{s}_t), \quad \boldsymbol{\xi} \sim p(\boldsymbol{\xi}), \tag{1}$$

with the continuous state $\boldsymbol{s}_t \in \mathcal{S}_{\xi} \subseteq \mathbb{R}^{n_s}$ and continuous action $\boldsymbol{a}_t \in \mathcal{A}_{\xi} \subseteq \mathbb{R}^{n_a}$ at time step t. The environment, also called domain, is characterized by its parameters $\boldsymbol{\xi} \in \mathbb{R}^{n_{\xi}}$ (e.g., masses, friction coefficients, time delays, or surface appearance properties) which are in general assumed to be random variables distributed according to an unknown probability distribution $p(\boldsymbol{\xi})$: $\mathbb{R}^{n_{\xi}} \to \mathbb{R}^{+}$. A special case of this is the common assumption that the domain parameters obey a parametric distribution $p_{\boldsymbol{\phi}}(\boldsymbol{\xi})$ with unknown parameters $\boldsymbol{\phi}$ (e.g., mean and variance). The domain parameters determine the transition probability density function $\mathcal{P}_{\xi}$: $\mathcal{S}_{\xi} \times \mathcal{A}_{\xi} \times \mathcal{S}_{\xi} \to \mathbb{R}^{+}$ that describes the system's stochastic dynamics. The initial state $\boldsymbol{s}_0$ is drawn from the start state distribution μ_{ξ}: $\mathcal{S}_{\xi} \to \mathbb{R}^{+}$. In general, the instantaneous reward is a random variable depending on the current state and action as well as the next state. Here we make the common simplification that the reward is a deterministic function of the current state and action r_{ξ}: $\mathcal{S}_{\xi} \times \mathcal{A}_{\xi} \to \mathbb{R}$ Together with the temporal discount factor $\gamma \in [0, 1]$, the system forms a MDP described by the tuple $\mathcal{M}_{\xi} = \langle \mathcal{S}_{\xi}, \mathcal{A}_{\xi}, \mathcal{P}_{\xi}, \mu_{\xi}, r_{\xi}, \gamma \rangle$.

Reinforcement Learning (RL): The goal of a RL agent is to maximize the expected (discounted) return, a numeric scoring function which measures the policy's performance. The expected discounted return of a policy $\pi_{\boldsymbol{\theta}}(\boldsymbol{a}_t|\boldsymbol{s}_t)$ with the parameters $\boldsymbol{\theta} \in \Theta \subseteq \mathbb{R}^{n_{\theta}}$ is defined as

$$J(\boldsymbol{\theta},\boldsymbol{\xi}) = \mathbb{E}_{\boldsymbol{s}_0\sim\mu_\xi(\boldsymbol{s}_0)}\Big[\mathbb{E}_{\boldsymbol{s}_{t+1}\sim\mathcal{P}_\xi(\boldsymbol{s}_t,\boldsymbol{a}_t),\ \boldsymbol{a}_t\sim\pi_\theta(\boldsymbol{a}_t|\boldsymbol{s}_t)}\Big[\sum_{t=0}^{T-1}\gamma_\xi^t(\boldsymbol{s}_t,\boldsymbol{a}_t)\,|\,\boldsymbol{\theta},\boldsymbol{\xi},\boldsymbol{s}_0\Big]\Big]. \tag{2}$$

While learning from experience, the agent adapts its policy parameters. The resulting state-action-reward tuples are collected in trajectories, a.k.a. rollouts, $\boldsymbol{\tau} = \{\boldsymbol{s}_t, \boldsymbol{a}_t, r_t\}_{t=0}^{T-1} \in \mathcal{T}$ with $r_t = r_\xi(\boldsymbol{s}_t, \boldsymbol{a}_t)$. In a partially observable MDP, the policy's input would not be the state but observations there of $\boldsymbol{o}_t \in \mathcal{O}_\xi \subseteq \mathbb{R}^{n_o}$, which are obtained through an environment-specific mapping $\boldsymbol{o}_t = f_{\text{obs}}(\boldsymbol{s}_t)$.

Domain randomization: When augmenting the RL setting with domain randomization, the goal becomes to maximize the expected (discounted) return for a distribution of domain parameters

$$\begin{aligned} J(\boldsymbol{\theta}) &= \mathbb{E}_{\boldsymbol{\xi}\sim p(\xi)}[J(\boldsymbol{\theta},\boldsymbol{\xi})] \\ &= \mathbb{E}_{\boldsymbol{\xi}\sim p(\xi)}\Big[\mathbb{E}_{\boldsymbol{\tau}\sim p(\boldsymbol{\tau})}\Big[\sum_{t=0}^{T-1}\gamma_\xi^t(\boldsymbol{s}_t,\boldsymbol{a}_t)\,|\,\boldsymbol{\theta},\boldsymbol{\xi},\boldsymbol{s}_0\Big]\Big]. \end{aligned} \tag{3}$$

The outer expectation with respect to the domain parameter distribution $p(\boldsymbol{\xi})$ is the key difference compared to the standard MDP formulation. It enables the learning of robust policies, in the sense that these policies work for a whole set of environments instead of overfitting to a particular problem instance.

3 FOUNDATIONS OF SIM-TO-REAL TRANSFER

Modern research on learning from (randomized) physics simulations is based on solid foundation of prior work (**Section 3.1**). Parametric simulators are the core component of every sim-to-real method (**Section 3.2**). Even though the details of their randomization are crucial, they are rarely discussed (**Section 3.3**). Estimating the sim-to-real transferability during or after learning allows one to assess or predict the policy's performance in the target domain (**Section 3.4**).

3.1 Early Methods

The roots of randomized simulations trace back to the invention of the Monte Carlo method (Metropolis and Ulam, 1949), which computes its results based on repeated random sampling and subsequent statistical analysis. Later, the concept of common random numbers, also called correlated sampling, was developed as a variance reduction technique (Kahn and Marshall, 1953; Wright and Ramsay, 1979). The idea is to synchronize the random numbers for all stochastic events across the simulation runs to achieve a (desirably positive) correlation between random variables reducing the variance of an estimator based on a combination of them. Many of the sim-to-real challenges which are currently tackled have already been identified by Brooks (1992). In particular, Brooks addresses the overfitting to effects which only occur in simulation as well as the idealized modeling on sensing and actuation. To avoid overfitting, he advocated for reactive behavior-based programming which is deeply rooted in, hence tailored to, the embodiment. Focusing on RL, Sutton (1991) introduced the Dyna architecture which revolves around predicting from a learned world model and updating the policy from this hypothetical experience. Viewing the data generated from randomized simulators as "imaginary," emphasizes the parallels of domain randomization to Dyna. As stated by Sutton, the usage of "mental rehearsal" to predict and reason about the effect of actions dates back even further in other fields of research such as psychology (Craik, 1943; Dennett, 1975). Instead of querying a learned internal model, Jakobi et al. (1995) added random noise the sensors and actuators while learning, achieving the arguably first sim-to-real transfer in robotics. In follow-up work, Jakobi (1997) formulated the radical envelope of noise hypothesis which states that "it does not matter how inaccurate or incomplete [the simulations] are: controllers that have evolved to be reliably fit in simulation will still transfer into reality." Picking up on the idea of common random numbers, Ng and Jordan (2000) suggested to explicitly control the randomness of a simulator, i.e., the random number generator's state, rendering the simulator deterministic. This way the same initial configurations can be (re-)used for Monte Carlo estimations of different policies' value functions, allowing one to conduct policy search in partially observable problems. Bongard et al. (2006) bridged the sim-to-real gap through iterating model generation and selection depending on the short-term state-action history. This process is repeated for a given number of iterations, and then yields the self-model, i.e., a simulator, which best explains the observed data.

Inspired by these early approaches, the systematic analysis of randomized simulations for robot learning has become a highly active research direction. Moreover, the prior work above also falsifies the common belief that domain randomization originated recently with the rise of deep learning. Nevertheless, the current popularity of domain randomization can be explained by its widespread use in the computer vision and locomotion communities as well as its synergies with deep learning methods. The key difference between the early and the recent domain randomization methods (**Section 5**) is that the latter (directly) manipulate the simulators' parameters.

3.2 Constructing Stochastic Simulators

Simulators can be obtained by implementing a set of physical laws for a particular system. Given the challenges in implementing an efficient simulator for complex systems, it is common to use general purpose physics engines such as ODE, DART, Bullet, Newton, SimBody, Vortex, MuJoCo, Havok, Chrono, RaiSim, PhysX, FleX, or Brax. These simulators are parameterized generative models, which describe how multiple bodies or particles evolve over time by interacting with each other. The associated physics parameters can be estimated by system identification (**Section 4.6**), which generally involves executing experiments on the physical platform and recording associated measurement. Additionally, using the Gauss-Markov theorem one could also compute the parameters' covariance and hence construct a normal distribution for each domain parameter. Differentiable simulators facilitate deep learning for robotics (Degrave et al., 2019; Coumans, 2020; Heiden et al., 2021) by

propagating the gradients though the dynamics. Current research extends the differentiability to soft body dynamics (Hu et al., 2019). Alternatively, the system dynamics can be captured using nonparametric methods like Gaussian Processes (GPs) (Rasmussen and Williams, 2006) as for example demonstrated by Calandra et al. (2015). It is important to keep in mind that even if the domain parameters have been identified very accurately, simulators are nevertheless just approximations of the real world and are thus always imperfect.

Several comparisons between various physics engines were made (Ivaldi et al., 2014; Erez et al., 2015; Chung and Pollard, 2016; Collins et al., 2019; Körber et al., 2021). However, note that these results become outdated quickly due to the rapid development in the field, or are often limited to very few scenarios and partially introduce custom metrics to measure their performance or accuracy.

Apart from the physics engines listed above, there is an orthogonal research direction investigating human-inspired learning of the physics laws from visual input (Battaglia et al., 2013; Wu et al., 2015) as well as physical reasoning given a configuration of bodies (Battaglia et al., 2016), which is out of the scope of this review.

3.3 Randomizing a Simulator

Learning from randomized simulations entails significant design decisions:

Which parameters should be randomized? Depending on the problem, some domain parameters have no influence (e.g., the mass of an idealized rolling ball) while others are pivotal (e.g., the pendulum length for a stabilization task). It is recommended to first identify the essential parameters (Xie et al., 2020). For example, most robot locomotion papers highlight the importance of varying the terrain and contact models, while applications such as drone control benefit from adding perturbations, e.g., to simulate a gust of wind. Injecting random latency and noise to the actuation is another frequent modeling choice. Starting from a small set of randomized domain parameters, identified from prior knowledge, has the additional benefit of shortening the evaluation time which involves approximating an expectation over domains, which scales exponentially with the number of parameters. Moreover, including at least one visually observable parameter (e.g., an extent of a body) helps to verify if the values are set as expected.

When should the parameters be randomized? Episodic dynamics randomization, without a rigorous theoretical justification, is the most common approach. Randomizing the domain parameters at every time step instead would drastically increase the variance, and pose a challenge to the implementations since this typically implies recreating the simulation at every step. Imagine a stack of cubes standing on the ground. If we now vary the cubes' side lengths individually while keeping their absolute positions fixed, they will either lose contact or intersect with their neighboring cube(s). In order to keep the stack intact, we need to randomize the cubes with respect to their neighbors, additionally moving them in space. Executing this once at the beginning is fine, but doing this at every step creates artificial "movement" which would almost certainly be detrimental. Orthogonal to the argumentation above, alternative approaches apply random disturbance forces and torques at every time step. In these cases, the distribution over disturbance magnitudes is chosen to be constant until the randomization scheme is updated. To the best of our knowledge, event-triggered randomization has not been explored yet.

How should the parameters be randomized? Answering this question is what characterizes a domain randomization method (**Section 5**). There are a few aspects that needs to be considered in practice when designing a domain randomization scheme, such as the numerical stability of the simulation instances. Low masses for example quickly lead to stiff differential equations which might require a different (implicit) integrator. Furthermore, the noise level of the introduced randomness needs to match the precision of the state estimation. If the noise is too low, the randomization is pointless. On the other side, if the noise level is too high, the learning procedure will fail. To find the right balance between these considerations, we can start by statistically analyzing the incoming measurement signals.

What about physical plausibility? The application of pseudo-random color patterns, e.g., Perlin noise (Perlin, 2002), has become a frequent choice for computer vision applications. Despite that these patterns do not occur on real-world objects, this technique has improved the robustness of object detectors (James et al., 2017; Pinto et al., 2018). Regarding the randomization of dynamics parameters, no research has so far hinted that physically implausible simulations (e.g., containing bodies with negative masses) are useful. On the other hand, it is safe to say that these can cause numerical instabilities. Thus, ensuring feasibility of the resulting simulator is highly desirable. One solution is to project the domain parameters into a different space, guaranteeing physical plausibility via the inverse projection. For example, a body's mass could be learned in the log-space such that the subsequent exp-transformation, applied before setting the new parameter value, yields strictly positive numbers. However, most of the existing domain randomization approaches can not guarantee physical plausibility.

Even in the case of rigid body dynamics there are notable differences between physics engines, as was observed by Muratore et al. (2018) when transferring a robot control policy trained using Vortex to Bullet and vice versa. Typical sources for deviations are different coordinate representations, numerical solvers, friction and contact models. Especially the latter two are decisive for robot manipulation. For vision-based tasks, Alghonaim and Johns (2020) found a strong correlation between the renderer's quality and sim-to-real transferability. Additionally, the authors emphasize the importance of randomizing both distractor objects and background textures for generalizing to unseen environments.

3.4 Measuring and Predicting the Reality Gap

Coining the term "reality gap," Koos et al. (2010) hypothesize that the fittest solutions in simulation often rely on poorly simulated phenomena. From this, they derive a multi-objective formulation for sim-to-real transfer where performance and transferability

need to be balanced. In subsequent work, Koos et al. (2013) defined a transferability function that maps controller parameters to their estimated target domain performance. A surrogate model of this function is regressed from the real-world fitness values that are obtained by executing the controllers found in simulation.

The Simulation Optimization Bias (SOB) (Muratore et al., 2018; 2021b) is a quantitative measure for the transferability of a control policy from a set of source domains to a different target domain originating from the same distribution. Building on the formulation of the optimality gap from convex optimization (Mak et al., 1999; Bayraksan and Morton, 2006), Muratore et al. (2018) proposed a Monte Carlo estimator of the SOB as well as an upper confidence bound, tailored to reinforcement learning settings. This bound can be used as an indicator to stop training when the predicted transferability exceeds a threshold. Moreover, the authors show that the SOB is always positive, i.e., optimistic, and in expectation monotonically decreases with an increasing number of domains.

Collins et al. (2019) quantify the accuracy of ODE, (Py)Bullet, Newton, Vortex, and MuJoCo in a real-world robotic setup. The accuracy is defined as the accumulated mean-squared error between the Cartesian ground truth position, tracked by a motion capture system, and the simulators' prediction. Based on this measure, they conclude that simulators are able to model the control and kinematics accurately, but show deficits during dynamic robot-object interactions.

To obtain a quantitative estimate of the transferability, Zhang et al. (2020) suggest to learn a probabilistic dynamics model which is evaluated on a static set of target domain trajectories. This dynamics model is trained jointly with the policy in the same randomized simulator. The transferability score is chosen to be the average negative log-likelihood of the model's output given temporal state differences from the real-world trajectories. Thus, the proposed method requires a set of pre-recorded target domain trajectories, and makes the assumption that for a given domain the model's prediction accuracy correlates with the policy performance.

With robot navigation in mind, Kadian et al. (2020) define the Sim-vs-Real Correlation Coefficient (SRCC) to be the Pearson correlation coefficient on data pairs of scalar performance metrics. The data pairs consist of the policy performance achieved in a simulator instance as well as in the real counterpart. Therefore, in contrast to the SOB (Muratore et al., 2018), the SRCC requires real-world rollouts. A high SRCC value, i.e., close to 1, predicts good transferability, while low values, i.e., close to 0, indicates that the agent is exploited the simulation during learning. Kadian et al. (2020) also report tuning the domain parameters with grid search to increase the SRCC. By using the Pearson correlation, the SRCC is restricted to linear correlation, which might not be a notable restriction in practice.

4 RELATION OF SIM-TO-REAL TO OTHER FIELDS

There are several research areas that overlap with sim-to-real in robot learning, more specifically domain randomization (**Figure 2**). In the following, we describe those that either share the same goal, or employ conceptually similar methods.

4.1 Curriculum Learning

The key idea behind curriculum learning is to increase the sample efficiency by scheduling the training process such that the agent first encounters "easier" tasks and gradually progresses to "harder" ones. Hence, the agent can bootstrap from the knowledge it gained at the beginning, before learning to solve more difficult task instances. Widely known in supervised learning (Bengio et al., 2009; Kumar et al., 2010), curriculum learning has been applied to RL (Asada et al., 1996; Erez and Smart, 2008; Klink et al., 2019, 2021). The connection between curriculum learning and domain randomization can be highlighted by viewing the task as a part of the domain, i.e., the MDP, rendering the task parameters a subspace of the domain parameters. From this point of view, the curriculum learning schedule describes how the domain parameter distribution is updated. There are several challenges to using a curriculum learning approach for sim-to-real transfer. Three such challenges are: 1) we can not always assume to have an assessment of the difficulty level of individual domain parameter configurations, 2) curriculum learning does not aim at finding solutions robust to model uncertainty, and 3) curriculum learning methods may require a target distribution which is not defined in the domain randomization setting. However, adjustments can be made to circumvent these problems. OpenAI et al. (2019) suggested a heuristic for the domain randomization schedule that increases the boundaries of each domain parameter individually until the return drops more than a predefined threshold. Executing this approach on a computing cluster, the authors managed to train a policy and a vision system which in combination solve a Rubik's cube with a tendon-driven robotic hand. Another intersection point of curriculum learning and sim-to-real transfer is the work by Morere et al. (2019), where a hierarchical planning method for discrete domains with unknown dynamics is proposed. Learning abstract skills based on a curriculum enables the algorithm to outperform planning and RL baselines, even in domains with a very large number of possible states.

4.2 Meta Learning

Inspired by the human ability to quickly master new tasks by leveraging the knowledge extracted from solving other tasks, meta learning (Santoro et al., 2016; Finn et al., 2017) seeks to make use of prior experiences gained from conceptually similar tasks. The field of meta learning currently enjoys high popularity, leading to abundant follow-up work. Grant et al. (2018) for example casts meta learning as hierarchical Bayesian inference. Furthermore, the meta learning framework has been adapted to the RL setting (Wang et al., 2017; Nagabandi et al., 2019). The optimization over an ensemble of tasks can be translated to the optimization over an ensemble of domain instances, modeled by different MDPs (**Section 2**). *Via* this duality one can view

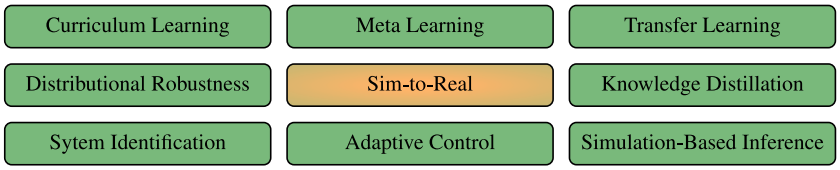

FIGURE 2 | Topological overview of the sim-to-real research and a selection of related fields.

domain randomization as a special form of meta learning where the robot's task remains qualitatively unchanged but the environment varies. Thus, the tasks seen during the meta training phase are analogous to domain instances experienced earlier in the training process. However, when looking at the complete procedure, meta learning and domain randomization are fundamentally different. The goal of meta learning, i.e., Finn et al. (2017), is to find a suitable set of initial weights, which when updated generalizes well to a new task. Domain randomization on the other hand strives to directly solve a single task, generalizing over domain instances.

4.3 Transfer Learning

The term transfer learning covers a wide range of machine learning research, aiming at using knowledge learned in the source domain to solve a task in the target domain. Rooted in classification, transfer learning is categorized in several subfields by for example differentiating 1) if labeled data is available in the source or target domain, and 2) if the tasks in both domains are the same (Pan and Yang, 2010; Zhuang et al., 2021). Domain adaptation is one of the resulting subfields, specifying the case where ground truth information is only available in the target domain which is not equal to the source domain while the task remains the same. Thus, domain adaptation methods are in general suitable to tackle sim-to-real problems. However, the research fields evolved at different times in different communities, with different goals in mind. The keyword "sim-to-real" specifically concerns regression and control problems where the focus lies on overcoming the mismatch between simulation and reality. In contrast, most domain adaptation techniques are not designed for a dynamical system as the target domain.

4.4 Knowledge Distillation

When executing a controller on a physical device operating at high frequencies, it is of utmost importance that the forward pass finishes with the given time frame. With deep Neural Network (NN) policies, and especially with ensembles of these, this requirement can become challenging to meet. Distilling the knowledge of a larger network into a smaller one reduces the evaluation time. Knowledge distillation (Hinton et al., 2015) has been successfully applied to several machine learning applications such as natural language processing (Cui et al., 2017), and object detection (Chen et al., 2017). In the context of RL, knowledge distillation techniques can be used to compress the learned behavior of one or more teachers into a single student (Rusu et al., 2016a). Based on samples generated by the teachers, the student is trained in a supervised manner to imitate them. This idea can be applied to sim-to-real robot learning in a straightforward manner, where the teachers can be policies optimal for specific domain instances (Brosseit et al., 2021). Complementarily, knowledge distillation has been applied to multitask learning (Parisotto et al., 2016; Teh et al., 2017), promising to improve sample efficiency when learning a new task. A technical comparison of policy distillation methods for RL is provided by Czarnecki et al. (2019).

4.5 Distributional Robustness

The term robustness is overloaded with different meanings, such as the ability to (quickly) counteract external disturbances, or the resilience against uncertainties in the underlying model's parameters. The field of robust control aims at designing controllers that explicitly deal with these uncertainties (Zhou and Doyle, 1998). Within this field, distributional robust optimization is a framework to find the worst-case probabilistic model from a so-called ambiguity set, and subsequently set a policy which acts optimally in this worst case. Mathematically, the problem is formulated as bilevel optimization, which is solved iteratively in practice. By restricting the model selection to the ambiguity set, distributional robust optimization regularizes the adversary to prevent the process from yielding solutions that are overly conservative policies. Under the lens of domain randomization, the ambiguity set closely relates to the distribution over domain parameters. Abdulsamad et al. (2021) for example define the ambiguity set as a Kullback-Leibler (KL) ball the nominal distribution. Other approaches use a moment-based ambiguity set (Delage and Ye, 2010) or introduce chance constrains (Van Parys et al., 2016). For a review of distributional robust optimization, see Zhen et al. (2021). Chatzilygeroudis et al. (2020) point out that performing policy search under an uncertain model is equivalent to finding a policy that can perform well under various dynamics models. Hence, they argue that "model-based policy search with probabilistic models is performing something similar to dynamics randomization."

4.6 System Identification

The goal of system identification is to find the set of model parameters which fit the observed data best, typically by minimizing the prediction-dependent loss such as the mean-squared error. Since the simulator is the pivotal element in every domain randomization method, the accessible parameters and

their nominal values are of critical importance. When a manufacturer does not provide data for all model parameters, or when an engineer wants to deploy a new model, system identification is typically the first measure to obtain an estimate of the domain parameters. In principle, a number of approaches can be applied depending on the assumptions on the internal structure of the simulator. The earliest approaches in robotics recognized the linearity of the rigid body dynamics with respect to combinations of physics parameters such as masses, moments of inertia, and link lengths, thus proposed to use linear regression (Atkeson et al., 1986), and later Bayesian linear regression (Ting et al., 2006). However, it was quickly observed that the inferred parameters may be physically implausible, leading to the development of methods that can account for this (Ting et al., 2011). With the advent of deep learning, such structured physics-based approaches have been enhanced with NNs, yielding nonlinear system identification methods such as the ones based on the Newton-Euler forward dynamics (Sutanto et al., 2020; Lutter et al., 2021b). Alternatively, the simulator can be augmented with a NN to learn the domain parameter residuals, minimizing the one step prediction error (Allevato et al., 2019). On another front, system identification based on the classification loss between simulated and real samples has been investigated (Du et al., 2021; Jiang et al., 2021). System identification can also be interpreted as an episodic RL problem by treating the trajectory mismatch as the cost function and iteratively updating a distribution over models (Chebotar et al., 2019). Recent simulation-based inference methods yield highly expressive posterior distributions that capture multi-modality as well as correlations between the domain parameters (**Section 4.8**).

4.7 Adaptive Control

The well-established field of adaptive control is concerned with the problem of adapting a controller's parameters at runtime to operate initially uncertain or varying systems (e.g., aircraft reaching supersonic speed). A prominent method is model reference adaptive control, which tracks a reference model's output specifying the desired closed-loop behavior. Model Identification Adaptive Control (MIAC) is a different variant, which includes an online system identification component that continuously estimates the system's parameters based on the prediction error of the output signal (Åström and Wittenmark, 2008; Landau et al., 2011). Given the identified system, the controller is updated subsequently. Similarly, there exists a line of sim-to-real reinforcement learning approaches that condition the policy on the estimated domain parameters (Yu et al., 2017, 2019b; Mozifian et al., 2020) or a latent representation thereof (Yu et al., 2019a; Peng et al., 2020; Kumar et al., 2021). The main difference to MIAC lies in the adaption mechanism. Adaptive control techniques typically define the parameters' gradient proportional to the prediction error, while the approaches referenced above make the domain parameters an input to the policy.

4.8 Simulation-Based Inference

Simulators are predominantly used as forward models, i.e., to make predictions. However, with the increasing fidelity and expressiveness of simulators, there is a growing interest to also use them for probabilistic inference (Cranmer et al., 2020). In the case of simulation-based inference, the simulator and its parameters define the statistical model. Inference tasks differ by the quantity to be inferred. Regarding sim-to-real transfer, the most frequent task is to infer the simulation parameters from real-world time series data. Similarly to system identification (**Section 4.6**), the result can be a point estimate, or a posterior distribution. Likelihood-Free Inference (LFI) methods are a type of simulation-based inference approaches which are particularly well-suited when we can make very little assumptions about the underlying generative model, treating it as an implicit function. These approaches only require samples from the model (e.g., a non-differentiable black-box simulator) and a measure of how likely real observations could have been generated from the simulator. Approximate Bayesian computation is well-known class of LFI methods that applies Monte Carlo sampling to infer the parameters by comparing summary statistics of synthetically generated and observed data. There exist plenty of variants for approximate Bayesian computation (Marjoram et al., 2003; Beaumont et al., 2009; Sunnåker et al., 2013) as well as studies on the design of low-dimensional summary statistics (Fearnhead and Prangle, 2012). In order to increase the efficiency and thereby scale LFI higher-dimensional problems, researchers investigated amortized approaches, which conduct the inference over multiple sequential rounds. Sequential neural posterior estimation approaches (Papamakarios and Murray, 2016; Lueckmann et al., 2017; Greenberg et al., 2019) approximate the conditional posterior, allowing for direct sampling from the posterior. Learning the likelihood (Papamakarios et al., 2019) can be useful in the context for hypothesis testing. Alternatively, posterior samples can be generated from likelihood-ratios (Durkan et al., 2020; Hermans et al., 2020). However, simulation-based inference does not explicitly consider policy optimization or domain randomization. Recent approaches connected all three techniques, and closed the reality gap by inferring a distribution over simulators while training policies in simulation (Ramos et al., 2019; Barcelos et al., 2020; Muratore et al., 2021c).

5 DOMAIN RANDOMIZATION FOR SIM-TO-REAL TRANSFER

We distinguish between static (**Section 5.1**), adaptive (**Section 5.2**), and adversarial (**Section 5.3**) domain randomization (**Figure 3**). Static, as well as adaptive, methods are characterized by randomly sampling a set of domain parameters $\boldsymbol{\xi} \sim p(\boldsymbol{\xi})$ at the beginning of

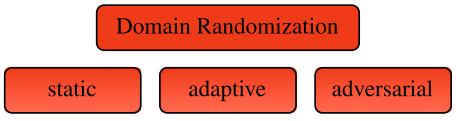

FIGURE 3 | Topological overview of domain randomization methods.

each simulated rollout. A randomization scheme is categorized as adaptive if the domain parameter distribution is updated during learning, otherwise the scheme is called static. The main advantage of adaptive schemes is that they alleviate the need for hand-tuning the distributions of the domain parameters, which is currently a decisive part of the hyper-parameter search in a static scheme. Nonetheless, the prior distributions still demand design decisions. On the downside, every form of adaptation requires data from the target domain, typically the real robot, which is significantly more expensive to obtain. Another approach for learning robust policies in simulation is to apply adversarial disturbances during the training process. We classify these perturbations as a form of domain randomization, since they either depend on a highly stochastic adversary learned jointly with the policy, or directly contain a random process controlling the application of the perturbation. Adversarial approaches may yield exceptionally robust control strategies. However, without any further restrictions, it is always possible to create scenarios in which the protagonist agent can never win, i.e., the policy can not learn the task. Balancing the adversary's power is pivotal to an adversarial domain randomization method, adding a sensitive hyper-parameter.

Another way to distinguish domain randomization concepts is the representation of the domain parameter distribution. The vast majority of algorithms assume a specific probability distribution (e.g., normal or uniform) independently for every parameter. This modeling decision has the benefit of greatly reducing the complexity, but at the same time severely limits the expressiveness. Novel LFI methods (**Section 5.2**) estimate the complete posterior, hence allow the recognition of correlations between the domain parameters, multi-modality, and skewness.

5.1 Static Domain Randomization

Approaches that sample from a fixed domain parameter distribution typically aim at performing sim-to-real transfer without using any real-world data (**Figure 4**). Since running the policy on a physical device is generally the most difficult and time-consuming part, static approaches promise quick and relatively easy to obtain results. In terms of final policy performance in the target domain, these methods are usually inferior to those that adapt the domain parameter distribution. Nevertheless, static domain randomization has bridged the reality gap in several cases.

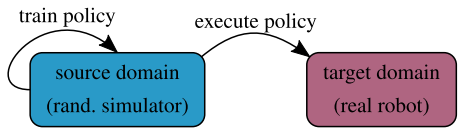

FIGURE 4 | Conceptual illustration of static domain randomization.

5.1.1 Randomizing Dynamics Without Using Real-World Data at Runtime

More than a decade ago, Wang et al. (2010) proposed to randomize the simulator in which the training data is generated. The authors examined the randomization of initial states, external disturbances, goals, and actuator noise, clearly showing an improved robustness of the learned locomotion controllers in simulated experiments (sim-to-sim). Mordatch et al. (2015) used a finite model ensembles to run (offline) trajectory optimization on a small-scale humanoid robot, achieving one of the first sim-to-real transfers in robotics powered by domain randomization. Similarly, Lowrey et al. (2018) employed the Natural Policy Gradient (Kakade, 2001) to learn a continuous controller for a three-finger positioning task, after carefully identifying the system's parameters. Conforming with Mordatch et al. (2015), their results showed that the policy learned from the identified model was able to perform the sim-to-real transfer, but the policies learned from an ensemble of models was more robust to modeling errors. In contrast, Peng et al. (2018) combined model-free RL with recurrent NN policies that were trained using hindsight experience replay (Andrychowicz et al., 2017) in order to push an object by controlling a robotic arm. Tan et al. (2018) presented an example for learning quadruped gaits from randomized simulations, where particular efforts were made to conduct a prior system identification. They empirically found that sampling domain parameters from a uniform distribution together with applying random forces and regularizing the observation space can be enough to cross the reality gap. For quadrotor control, Molchanov et al. (2019) trained feedforward NN policies which generalize over different physical drones. The suggested randomization includes a custom model for motor lag and noise based on an Ornstein-Uhlenbeck process. Rajeswaran et al. (2017) explored the use of a risk-averse objective function, optimizing a lower quantile of the return. The method was only evaluated on simulated MuJoCo tasks, however it was also one of the first methods that draws upon the Bayesian perspective. Moreover, this approach was employed as a baseline by Muratore et al. (2021b), who introduced a measure for the inter-domain transferability of controllers together with a risk-neutral randomization scheme. The resulting policies have the unique feature of providing a (probabilistic) guarantee on the estimated transferability and managed to directly transfer to the real platform in two different experiments. Siekmann et al. (2021) achieved the sim-to-real transfer of a recurrent NN policy for bipedal walking. The policy was trained using model-free RL in simulation with uniformly distributed dynamics parameters as well as randomized task-specific terrain. According to the authors, the recurrent architecture and the terrain randomization were pivotal.

5.1.2 Randomizing Dynamics Using Real-World Data at Runtime

The work by Cully et al. (2015) can be seen as both static and adaptive domain randomization, where a large set of hexapod locomotion policies is learned before execution on the physical robot, and subsequently evaluated in simulation. Every policy is associated with one configuration of the so-called behavioral descriptors, which can be interpreted as domain parameters. Instead of retraining or fine-tuning, the proposed algorithm

reacts to performance drops, e.g., due to damage, by querying Bayesian Optimization (BO) to sequentially select one of the pretrained policies and measure its performance on the robot. Instead of randomizing the simulator parameters, Cutler and How (2015) explored learning a probabilistic model, chosen to be a GP, of the environment using data from both simulated and real-world dynamics. A key feature of this method is to incorporate the simulator as a prior for the probabilistic model, and subsequently use this information of the policy updates with PILCO (Deisenroth and Rasmussen, 2011). The authors demonstrated policy transfer for a inverted pendulum task. In follow-up work, Cutler and How (2016) extended the algorithm to make a remote-controlled toy car learn how to drift in circles. Antonova et al. (2019) propose a sequential Variational AutoEncoder (VAE) to embed trajectories into a compressed latent space which is used with BO to search for controllers. The VAE and the domain-specific high-level controllers are learned jointly, while the randomization scheme is left unchanged. Leveraging a custom kernel which measures the KL divergence between trajectories and the data efficiency of BO, the authors report successful sim-to-real transfers after 10 target domain trials for a hexapod locomotion task as well as 20 trials for a manipulation task. Kumar et al. (2021) learned a quadruped locomotion policy that passed joint positions to a lower level PD controller without using any real-wold data. The essential components of this approach are the encoder that projects the domain parameters to a latent space and the adaption module which is trained to regress the latent state from the recent history of measured states and actions. The policy is conditioned on the current state, the previous actions, and the latent state which needs to be reconstructed during deployment in the physical world. Emphasizing the importance of the carefully engineered reward function, the authors demonstrate the method's ability to transfer from simulation to various outdoor terrains.

5.1.3 Randomizing Visual Appearance and Configurations

Tobin et al. (2017) learned an object detector for robot grasping using a fixed domain parameter distribution, and bridged the gap with a deep NN policy trained exclusively on simulated RGB images. Similarly, James et al. (2017) added various distracting shapes as well as structured noise (Perlin, 2002) when learning a robot manipulation task with an end-to-end controller that mapped pixels to motor velocities. The approach presented by Pinto et al. (2018) combines the concepts of static domain randomization and actor-critic training (Lillicrap et al., 2016), enabling the direct sim-to-real transfer of the abilities to pick, push, or move objects. While the critic has access to the simulator's full state, the policy only receives images of the environment, creating an information asymmetry. Matas et al. (2018) used the asymmetric actor-critic idea from Pinto et al. (2018) as well as several other improvements to train a deep NN policy end-to-end, seeded with prior demonstrations. Solving three variations of a tissue folding task, this work scales sim-to-real visuomotor manipulation to deformable objects. Purely visual domain randomization has also been applied to aerial robotics, where Sadeghi and Levine (2017) achieved sim-to-real transfer for learning to fly a drone through indoor environments. The resulting deep NN policy was able to map from monocular images to normalized 3D drone velocities. Similarly, Polvara et al. (2020) demonstrated landing of a quadrotor trained in end-to-end fashion using randomized environments. Dai et al. (2019) investigated the effect of domain randomization on visuomotor policies, and observed that this leads to more redundant and entangled representations accompanied with significant statistical changes in the weights. Yan et al. (2020) apply Model Predictive Control (MPC) to manipulate of deformable objects using a forward model based on visual input. The novelty of this approach is that the predictive model is trained jointly with an embedding to minimizing a contrastive loss (van den Oord et al., 2018) in the latent space. Finally, domain randomization was applied to transfer the behavior from simulation to the real robot.

5.1.4 Randomizing Dynamics, Randomizing Visual Appearance, and Configurations

Combining Generative Adversarial Networks (GANs) and domain randomization, Bousmalis et al. (2018) greatly reduced the number of necessary real-world samples for learning a robotic grasping task. The essence of their method is to transform simulated monocular RGB images in a way that is closely matched to the real counterpart. Extensive evaluation on the physical robot showed that domain randomization as well as the suggested pixel-level domain adaptation technique were important to successfully transfer. Despite the pixel-level domain adaptation technique being learned, the policy optimization in simulation is done with a fixed randomization scheme. In related work James et al. (2019) train a GAN to transform randomized images to so-called canonical images, such that a corresponding real image would be transformed to the same one. This approach allowed them to train purely from simulated images, and optionally fine-tune the policy on target domain data. Notably, the robotic in-hand manipulation conducted by OpenAI et al. (2020) demonstrated that domain randomization in combination with careful model engineering and the usage of recurrent NNs enables sim-to-real transfer on an unprecedentedly difficulty level.

5.2 Adaptive Domain Randomization

Static domain randomization (**Section 5.1**) is inherently limited and implicitly assumes knowledge of the true mean of the domain parameters or accepts biased samples (**Figure 5**). Adapting the randomization scheme allows the training to narrow or widen the search distribution in order to fulfill one or multiple criteria

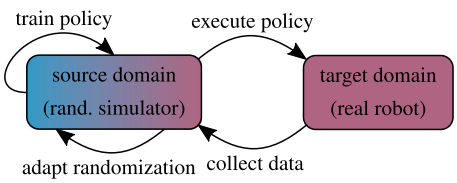

FIGURE 5 | Conceptual illustration of adaptive domain randomization.

which can be chosen freely. The mechanism devised for updating the domain parameter distribution as well as the procedure to collect meaningful target domain data are typically the center piece of adaptive randomization algorithms. In this process the execution of intermediate policies on the physical device is the most likely point of failure. However, approaches that update the distribution solely based on data from the source domain are less flexible and generally less effective.

5.2.1 Conditioning Policies on the Estimated Domain Parameters

Yu et al. (2017) suggested the use of a NN policy that is conditioned on the state and the domain parameters. Since these parameters are not assumed to be known, they have to be estimated, e.g., with online system identification. For this purpose, a second NN is trained to regress the domain parameters from the observed rollouts. By applying this approach to simulated continuous control tasks, the authors showed that adding the online system identification module can enable an adaption to sudden changes in the environment. In subsequent research, Yu et al. (2019a) intertwined policy optimization, system identification, and domain randomization. The proposed method first identifies bounds on the domain parameters which are later used for learning from the randomized simulator. In a departure from their previous approach, the policy is conditioned on a latent space projection of the domain parameters. After training in simulation, a second system identification step runs BO for a fixed number of iterations to find the most promising projected domain parameters. The algorithm was evaluated on sim-to-real bipedal robot walking. Mozifian et al. (2020) also introduce a dependence of the policy w.r.t. to the domain parameters. These are updated by gradient ascent on the average return over domains, regularized by a penalty proportional to the KL divergence. Similar to Ruiz et al. (2019), the authors update the domain parameter distribution using the score function gradient estimator. Mozifian et al. (2020) tested their method on sim-to-sim robot locomotion tasks. It remains unclear whether this approach scales to sim-to-real scenarios since the adaptation is done based on the return obtained in simulation, thus is not physically grounded. Bootstrapping from pre-recorded motion capture data of animals, Peng et al. (2020) learned quadruped locomotion skills with a synthesis of imitation learning, domain randomization, and domain adaptation (**Section 4.3**). The introduced method is conceptually related to the approach of Yu et al. (2019b), but adds an information bottleneck. According to the authors, this bottleneck is necessary because without it, the policy has access to the underlying dynamics parameters and becomes overly dependent on them, which leads to brittle behavior. To avoid this overfitting, Peng et al. (2020) limit the mutual information between the domain parameters and their encoding, realized as penalty on the KL divergence from a zero-mean Gaussian prior on the latent variable.

5.2.2 The Bilevel Optimization Perspective

Muratore et al. (2021a) formulated adaptive domain randomization as a bilevel optimization that consists of an upper and a lower level problem. In this framework, the upper level is concerned with finding the domain parameter distribution, which when used for training in simulation leads to a policy with maximal real-world return. The lower level problem seeks to find a policy in the current randomized source domain. Using BO for the upper level and model-free RL for the lower level, Muratore et al. (2021a) compare their method in two underactuated sim-to-real robotic tasks against two baselines. Picturing the real-world return analogous to the probability for optimality, this approach reveals parallels to control as inference (Rawlik et al., 2012; Levine and Koltun, 2013; Watson et al., 2021), where the control variates are the parameters of the domain distribution. BO has also been employed by Paul et al. (2019) to adapt the distribution of domain parameters such that using these for the subsequent training maximizes the policy's return. Their method models the relation between the current domain parameters, the current policy and the return of the updated policy with a GP. Choosing the domain parameters that maximize the return in simulation is critical, since this creates the possibility to adapt the environment such that it is easier for the agent to solve. This design decision requires the policy parameters to be fed into the GP which is prohibitively expensive if the full set of parameters are used. Therefore, abstractions of the policy, so-called fingerprints, are created. These handcrafted features, e.g., a Gaussian approximation of the stationary state distribution, replace the policy to reduce the input dimension. Paul et al. (2019) tested the suggested algorithm on three sim-to-sim tasks, focusing on the handling of so-called significant rare events. Embedding the domain parameters into the mean function of a GP which models the system dynamics, Chatzilygeroudis and Mouret (2018) extended a black-box policy search algorithm (Chatzilygeroudis et al., 2017) with a simulator as prior. The approach explicitly searches for parameters of the simulator that fit the real-world data in an upper level loop, while optimizing the GP's hyper-parameters in a lower level loop. This method allowed a damage hexapod robot to walk in less than 30 s. Ruiz et al. (2019) proposed a meta-algorithm which is based on a bilevel optimization problem and updates the domain parameter distribution using REINFORCE (Williams, 1992). The approach has been evaluated in simulation on synthetic data, except for a semantic segmentation task. Thus, there was no dynamics-dependent interaction of the learned policy with the real world. Mehta et al. (2019) also formulated the adaption of the domain parameter distribution as an RL problem where different simulation instances are sampled and compared against a reference environment based on the resulting trajectories. This comparison is done by a discriminator which yields rewards proportional to the difficulty of distinguishing the simulated and real environments, hence providing an incentive to generate distinct domains. Using this reward signal, the domain parameters of the simulation instances are updated *via* Stein Variational Policy Gradient (Liu et al., 2017). Mehta et al. (2019) evaluated their method in a sim-to-real experiment where a robotic arm had to reach a desired point. In contrast, Chebotar et al. (2019) presented a trajectory-based framework for closing the reality gap, and validated it on two sim-to-real

robotic manipulation tasks. The proposed procedure adapts the domain parameter distribution's parameters by minimizing discrepancy between observations from the real-world system and the simulation. To measure the discrepancy, Chebotar et al. (2019) use a linear combination of the L^1 and L^2 norm between simulated and real trajectories. These values are then plugged in as costs for Relative Entropy Policy Search (REPS) (Peters et al., 2010) to update the simulator's parameters, hence turning the simulator identification into an episodic RL problem. The policy optimization was done using Proximal Policy Optimization (PPO) (Schulman et al., 2017), a step-based model-free RL algorithm.

5.2.3 Removing Restrictions on the Domain Parameter Distribution

Ramos et al. (2019) perform a fully Bayesian treatment of the simulator's parameters by employing Likelihood-Free Inference (LFI) with a Mixture Density Network (MDN) as model for the density estimator. Analyzing the obtained posterior over domain parameters, they showed that the proposed method is, in a sim-to-sim scenario, able to simultaneously infer different parameter configurations which can explain the observed trajectories. An evaluation over a gird of domain parameters confirms that the policies trained with the inferred posterior are more robust model uncertainties. The key benefit over previous approaches is that the domain parameter distribution is not restricted to belong to a specific family, e.g., normal or uniform. Instead, the true posterior is approximated by the density estimator, fitted using LFI (Papamakarios and Murray, 2016). In follow-up work, Possas et al. (2020) addressed the problem of learning the behavioral policies which are required for the collection of target domain data. By describing the integration policy optimization *via* model-free RL, the authors created an online variant of the original method. The sim-to-real experiments were carried out using MPC where (only) the model parameters are updated based on the result from the LFI routine. Matl et al. (2020) scaled the Bayesian inference procedure of Ramos et al. (2019) to the simulation of granular media, estimating parameters such as friction and restitution coefficients. Barcelos et al. (2020) presented a method that interleaves domain randomization, LFI, and policy optimization. The controller is updated *via* nonlinear MPC while using the unscented transform to simulate different domain instances for the control horizon. Hence, this algorithm allows one to calibrate the uncertainty as the system evolves with the passage of time, attributing higher costs to more uncertain paths. For performing the essential LFI, the authors build upon the work of Ramos et al. (2019) to identify the posterior domain parameters, which are modeled by a mixture of Gaussians. The approach was validated on a simulated inverted pendulum swing-up task as well as a real trajectory following task using a wheeled robot. Since the density estimation problem is the center piece of LFI-based domain randomization, improving the estimator's flexibility is of great interest. Muratore et al. (2021c) employed a sequential neural posterior estimation algorithm (Greenberg et al., 2019) which uses normalizing flows to estimate the (conditional) posterior over simulators. In combination with a segment-wise synchronization between the simulations and the recorded real-world trajectories, Muratore et al. (2021c) demonstrated the neural inference method's ability to learn the posterior belief over contact-rich black-box simulations. Moreover, the proposed approach was evaluated with policy optimization in the loop on an underactuated swing-up and balancing task, showing improved results compared to BayesSim (Ramos et al., 2019) as well as Bayesian linear regression.

5.3 Adversarial Domain Randomization

Extensive prior studies have shown that deep NN classifiers are vulnerable to imperceptible perturbations their inputs, obtained *via* adversarial optimization, leading to significant drops in accuracy (Szegedy et al., 2014; Fawzi et al., 2015; Goodfellow et al., 2015; Kurakin et al., 2017; Ilyas et al., 2019). This line of research has been extended to reinforcement learning, showing that small (adversarial) perturbations are enough to significantly degrade the policy performance (Huang et al., 2017). To defend against such attacks, the training data can be augmented with adversarially-perturbed examples, or the adversarial inputs can be detected and neutralized at test-time (**Figure 6**). However, studies of existing defenses have shown that adversarial examples are harder to detect than originally believed (Carlini and Wagner, 2017). It is safe to assume that this insight gained from computer vision problems transfers to the RL setting, on which we focus here.

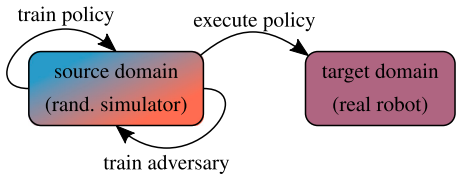

FIGURE 6 | Conceptual illustration of adversarial domain randomization.

5.3.1 Adversary Available Analytically

Mandlekar et al. (2017) proposed physically plausible perturbations by randomly deciding when to add a scaled gradient of the expected return w.r.t. the state. Their sim-to-sim evaluation on four MuJoCo tasks showed that agents trained with the suggested adversarial randomization generalize slightly better to domain parameter configurations than agents trained with a static randomization scheme. Lutter et al. (2021a) derived the optimal policy together with different optimal disturbances from the value function in a continuous state, action, and time RL setting. Despite outstanding sim-to-real transferability of the resulting policies, the presented approach is conceptually restricted by assuming access to a compact representation of the state domain, typically obtained through exhaustive sampling, which hinders the scalability to high-dimensional tasks.

5.3.2 Adversary Learned *via* Two-Player Games

Domain randomization can be described using a game theoretic framework. Focusing on two-player games for model-based RL, Rajeswaran et al. (2020) define a "policy player" which maximizes rewards in the learned model and a "model player" which minimizes prediction error of data collected by policy player.

This formulation can be transferred to the sim-to-real scenario in different ways. One example is to make the "policy player" model-agnostic and to let the "model player" control the domain parameters. Pinto et al. (2017) introduced the idea of a second agent whose goal it is to hinder the first agent from fulfilling its task. This adversary has the ability to apply force disturbances at predefined locations of the robot's body, while the domain parameters remain unchanged. Both agents are trained in alternation using RL make this a zero-sum game. Similarly, Zhang et al. (2021) aim to train an agent using adversarial examples such that it becomes robust against test-time attacks. As in the approach presented by Pinto et al. (2017), the adversary and the protagonist are trained alternately until convergence at every meta-iteration. Unlike prior work, Zhang et al. (2021) build on state-adversarial MDPs manipulating the observations but not the simulation state. Another key property of their approach is that the perturbations are applied after a projection to a bounded set. The proposed observation-based attack as well as training algorithm is supported by four sim-to-sim validations in MuJoCo environments. Jiang et al. (2021) employed GANs to distinguish between source and target domain dynamics, sharing the concept of a learned domain discriminator with Mehta et al. (2019). Moreover, the authors proposed to augment an analytical physics simulator with a NN that is trained to maximize the similarity between simulated and real trajectories, turning the identification of the hybrid simulator into an RL problem. The comparison on a sim-to-real quadruped locomotion task showed an advantage over static domain randomization baselines. On the other hand, this method added noise to the behavioral policy in order to obtain diverse target domain trajectories for the simulator identification, which can be considered dangerous.

6 DISCUSSION AND OUTLOOK

To conclude this review, we discuss practical aspects of choosing among the existing domain randomization approaches (**Section 6.1**), emphasizing that sim-to-real transfer can also be achieved without randomizing (**Section 6.2**). Finally, we sketch out several promising directions for future sim-to-real research (**Section 6.3**).

6.1 Choosing a Suitable Domain Randomization Approach

Every publication on sim-to-real robot learning presents an approach that surpasses its baselines. So, how should we select the right algorithm given a task? Up to now, there is no benchmark for sim-to-real methods based on the policy's target domain performance, and it is highly questionable if such a comparison could be fair, given that these algorithms have substantially different requirements and goals. The absence of one common benchmark is not necessarily bad, since bundling a set of environments to define a metric would bias research to pursue methods which optimize solely for that metric. A prominent example for this mechanism is the OpenAI Gym (Brockman et al., 2016), which became the *de facto* standard for RL. Contrarily, a similar development for sim-to-real research is not desirable since the overfitting to a small set of scenarios would be detrimental to the desired transferability and the vast amount of other scenarios.

When choosing from the published algorithms, the practitioner is advised to check if the approach has been tested on at least two different sim-to-real tasks, and if the (sometimes implicit) assumptions can be met. Adaptive domain randomization methods, for example, will require operating the physical device in order to collect real-world data. After all, we can expect that approaches with randomization will be more robust than the ones only trained on a nominal model. This has been shown consistently (**Section 5**). However, we can not expect that these approaches work out of the box on novel problems without adjusting the hyper-parameters. Another starting point could be the set of sim-to-sim benchmarks released by Mehta et al. (2020), targeting the problem of system identification for state-of-the-art domain randomization algorithms.

6.2 Sim-To-Real Transfer Without Domain Randomization

Domain randomization is one way to successfully transfer control policies learned in simulation to the physical device, but by no means the only way.

6.2.1 Action Transformation

In order to cope with the inaccuracies of a simulator, Christiano et al. (2016) propose to train a deep inverse dynamics model to map the action commanded by policy to a transformed action. When applying the original action to the real system and the transformed action to the simulated system, they would lead to the same next robot state, thus bridging the reality gap. To generate the data for training the inverse dynamics model, preliminary policies are augmented with hand-tuned exploration noise and executed in the target domain. Their approach is based on the observation that a policy's high-level strategy remains valid after sim-to-real transfer, and assumes that the simulator provides a reasonable estimate of the next state. With the same goal in mind, Hanna and Stone (2017) suggest an action transformation that is learned such that applying the transformed actions in simulation has the same effects as applying the original actions had on the real system. At the core approach is the estimation of neural forward and inverse models based on rollouts executed with the real robot.

6.2.2 Novel Neural Policy Architectures

Rusu et al. (2017) employ a progressively growing NN architecture (Rusu et al., 2016b) to learn an end-to-end approach mapping from pixels to discretized joint velocities. This NN framework enables the reuse of previously gained knowledge as well as the adaptation to new input modalities. The first part of the NN policy is trained in simulation, while the part added when transferring needs to be trained using real-world data. For a relatively simple reaching task, the authors reported requiring approximately 4 h of runtime on the physical robot.

6.2.3 Identifying and Improving the Simulator

Xie et al. (2019) describe an iterative process including motion tracking, system identification, RL, and knowledge distillation, to learn control policies for humanoid walking on the physical system. This way, the authors can rely on known building blocks resulting in initial and intermediate policies which are reasonably safe to execute. To run a policy on the real robot while learning without the risk of damaging or stopping the device, Kaspar et al. (2020) propose to combine operational space control and RL. After carefully identifying the simulator's parameters, the RL agent learns to control the end-effector *via* forces on a unit mass-spring-damper system. The constrains and nullspace behavior are abstracted away from the agent, making the RL problem easier and the policy more transferable.

6.3 Promising Future Research Directions

Learning from randomized simulations still offers abundant possibilities to enable or improve the sim-to-real transfer of control policies. In the following section, we describe multiple opportunities for future work in this area of research.

6.3.1 Real-To-Sim-To-Real Transfer

Creating randomizable simulation environments is time-intensive, and the initial guesses for the domain parameters as well as their variances are typically very inaccurate. It is of great interest to automate this process grounded by real-world data. One viable scenario could be to record an environment with a RGBD camera, and subsequently use the information to reconstruct the scene. Moreover, the recorded data can be processed to infer the domain parameters, which then specifies the domain parameter distributions. When devising such a framework, we could start from prior work on 3D scene reconstruction Kolev et al. (2009), Haefner et al. (2018) as well as methods to estimate the degrees of freedom for rigid bodies (Martin-Martin and Brock, 2014). A data-based automatic generation of simulation environments (real-to-sim-to-real) not only promises to reduce the workload, but would also yields a meaningful initialization for domain distribution parameters.

6.3.2 Policy Architectures With Inductive Biases

tDeep NNs are by far the most common policy type, favored because of their flexibility and expressiveness. However, they are also brittle w.r.t. changes in their inputs (Szegedy et al., 2014; Goodfellow et al., 2015; Huang et al., 2017). Due to the inevitable domain shift in sim-to-real scenarios this input sensitivity is magnified. The success of domain randomization methods for robot learning can largely be attributed to their ability of regularizing deep NN policies by diversifying the training data. Generally, one may also introduce regularization to the learning by designing alternative models for the control policies, e.g., linear combination of features and parameters, (time varying) mixtures of densities, or movement primitives. All of these have their individual strengths and weaknesses. We believe that pairing the expressiveness of deep NNs with physically-grounded prior knowledge leads to controllers that achieve high performance and suffer less from transferring to the real world, since they are able to bootstrap from their prior. There are multiple ways to incorporate abstract knowledge about physics. We can for example restrict the policy to obey stable system dynamics derived from first principles (Greydanus et al., 2019; Lutter et al., 2019). Another approach is to design the model class such that the closed-loop system is passive for all parameterizations of the learned policy, thus guaranteeing stability in the sense of Lyapunov as well as bounded output energy given bounded input energy (Brogliato et al., 2007; Yang et al., 2013; Dai et al., 2021). All these methods would require significant exploration in the environment, making it even more challenging to learn successful controllers in the real-world directly. Leveraging randomized simulation is likely going to be a critical component in demonstrating solving sequential problems on real robots.

6.3.3 Towards Dual Control *via* Neural Likelihood-Free Inference

Continuing the direction of adaptive domain randomization, we are convinced that neural LFI powered by normalizing flows are auspicious approaches. The combination of highly flexible density estimators with widely applicable and sample-efficient inference methods allows one to identify multi-modal distributions over simulators with very mild assumptions (Ramos et al., 2019; Barcelos et al., 2021; Muratore et al., 2021c). By introducing an auxiliary optimality variable and making the policy parameters subject to the inference, we obtain the posterior over policies quantifying their likelihood of being optimal. While this idea is well-known in the control-as-inference community (Rawlik et al., 2012; Levine and Koltun, 2013; Watson et al., 2021), prior methods were limited to less powerful density estimation procedures. Taking this idea one step further, we could additionally include the domain parameters for inference, and thereby establish connections to dual control (Feldbaum, 1960; Wittenmark, 1995).

6.3.4 Accounting for the Cost of Information Collection

Another promising direction for future research is the combination of simulated and real-world data collection with explicit consideration of the different costs when sampling from the two domains, subject to a restriction of the overall computational budget. One part of this problem was already addressed by Marco et al. (2017), showing how simulation can be used to alleviate the need for real-world samples when finding a set of policy parameters. However, the question of how to schedule the individual (simulated or real) experiments and when to stop the procedure, i.e., when does the cost of gathering information exceed its expected benefit, is not answered for sim-to-real transfer yet. This question relates to the problems of optimal stopping (Chow and Robbins, 1963) as well as multi-fidelity optimization (Forrester et al., 2007), and can be seen as a reformulation thereof in the context of simulation-based learning.

6.3.5 Solving Sequential Problems

The problem settings considered in the overwhelming majority of related publications, are (continuous) control tasks which do not have a sequential nature. In contrast, most real-world tasks such as the ones posed at the DARPA Robotics Challenge (Krotkov et al., 2017) consist of (disconnected) segments, e.g., a robot needs to turn the knob before it can open a door. One possible way to address these more complicated tasks is by splitting the control into high and low level policies, similar to the options framework (Sutton et al., 1999). The higher level policy is trained to orchestrate the low-level policies which could be learned or fixed. Existing approaches typically realize this with discrete switches between the low-level policies, leading to undesirable abrupt changes in the behavior. An alternative would be a continuous blending of policies, controlled by a special kind of recurrent NN which has originally been proposed by Amari (1977) to model activities in the human brain. Used as policy architectures they can be constructed to exhibit asymptotically stable nonlinear dynamics (Kishimoto and Amari, 1979). The main benefits of this structure are its easy interpretability via exhibition and inhibition of neural potentials, as well as the relatively low number of parameters necessary to create complex and adaptive behavior. A variation of this idea with hand-tuned parameters, i.e., without machine learning, has been applied by Luksch et al. (2012) to coordinate the activation pre-defined movement primitives.

SELECTION OF REFERENCES

We chose the references based on multiple criteria: 1) Our primary goal was to covering all milestones of the sim-to-real research for robotics. 2) In the process, we aimed at diversifying over subfields and research groups. 3) A large proportion of papers came to our attention by running Google Scholar alerts on "sim-to-real" and "reality gap" since 2017. 4) Another source were reverse searches starting from highly influential publications. 5) Some papers came to our attention because of citation notifications we received on our work. 6) Finally, a few of the selected publications are recommendations from reviewers, colleagues, or researchers met at conferences. 7) Peer-reviewed papers were strongly preferred over pre-prints.

AUTHOR CONTRIBUTIONS

FM: main author; FR: added and edited text, suggested publications, proofread; GT: added and edited text, suggested publications, proofread; WY: added and edited text, suggested publications, proofread; MG: edited text, proofread, (Ph.D. supervisor of FM); JP: added and edited text, suggested publications, proofread, (Ph.D. supervisor of FM).

FUNDING

FM gratefully acknowledges the financial support from Honda Research Institute Europe. JP received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 640554. WY and GT have been supported by NSF award IIS-1514258.

REFERENCES

Abdulsamad, H., Dorau, T., Belousov, B., Zhu, J., and Peters, J. (2021). Distributionally Robust Trajectory Optimization under Uncertain Dynamics via Relative-Entropy Trust Regions. arXiv 2103.15388

Alghonaim, R., and Johns, E. (2020). Benchmarking Domain Randomisation for Visual Sim-To-Real Transfer. arXiv 2011.07112

Allevato, A., Short, E. S., Pryor, M., and Thomaz, A. (2019). Tunenet: One-Shot Residual Tuning for System Identification and Sim-To-Real Robot Task Transfer. In Conference on Robot Learning (CoRL), Osaka, Japan, October 30 - November 1 (PMLR), vol. 100 of *Proc. Machine Learn. Res.*, 445–455.

Amari, S.-i. (1977). Dynamics of Pattern Formation in Lateral-Inhibition Type Neural fields. *Biol. Cybern.* 27, 77–87. doi:10.1007/bf00337259

Andrychowicz, M., Crow, D., Ray, A., Schneider, J., Fong, R., Welinder, P., et al. (2017). "Hindsight Experience Replay," in Conference on Neural Information Processing Systems (NIPS), December 4-9 (Long Beach, CA, USA, 5048–5058.

Andrychowicz, O. M., Baker, B., Chociej, M., Józefowicz, R., McGrew, B., Pachocki, J., et al. (2020). Learning Dexterous In-Hand Manipulation. *Int. J. Robotics Res.* 39, 3–20. doi:10.1177/0278364919887447

Antonova, R., Rai, A., Li, T., and Kragic, D. (2019). "Bayesian Optimization in Variational Latent Spaces with Dynamic Compression," in Conference on Robot Learning (CoRL), October 30 - November 1 (Osaka, Japan: PMLR), 456–465. of Proceedings of Machine Learning Research.100

Asada, M., Noda, S., Tawaratsumida, S., and Hosoda, K. (1996). Purposive Behavior Acquisition for a Real Robot by Vision-Based Reinforcement Learning. *Mach. Learn.* 23, 279–303. doi:10.1023/A:1018237008823 10.1007/bf00117447

Åström, K. J., and Wittenmark, B. (2008). *Adaptive Control*. 2 edn. Dover Publications.

Atkeson, C. G., H, C. A., and An, C. H. (1986). Estimation of Inertial Parameters of Manipulator Loads and Links. *Int. J. Robotics Res.* 5, 101–119. doi:10.1177/027836498600500306

Baker, B., Kanitscheider, I., Markov, T. M., Wu, Y., Powell, G., McGrew, B., et al. (2020). "Emergent Tool Use from Multi-Agent Autocurricula," in (Addis Ababa, Ethiopia. OpenReview.net.International Conference on Learning Representations (ICLR)April 26-30

Barcelos, L., Lambert, A., Oliveira, R., Borges, P., Boots, B., and Ramos, F. (2021). "Dual Online Stein Variational Inference for Control and Dynamics," in Robotics: Science and Systems (RSS), July 12-16. Virtual Event. doi:10.15607/RSS.2021.XVII.068

Barcelos, L., Oliveira, R., Possas, R., Ott, L., and Ramos, F. (2020). "DISCO: Double Likelihood-free Inference Stochastic Control," in International Conference on Robotics and Automation (ICRA), May 31 - August 31 (Paris, France: IEEE), 10969–10975. doi:10.1109/ICRA40945.2020.9196931

Battaglia, P. W., Hamrick, J. B., and Tenenbaum, J. B. (2013). Simulation as an Engine of Physical Scene Understanding. *Proc. Natl. Acad. Sci.* 110, 18327–18332. doi:10.1073/pnas.1306572110

Battaglia, P. W., Pascanu, R., Lai, M., Rezende, D. J., and Kavukcuoglu, K. (2016). "Interaction Networks for Learning about Objects, Relations and Physics," in Conference on Neural Information Processing Systems (NIPS), December 5-10 (Barcelona, Spain, 4502–4510.

Bayraksan, G., and Morton, D. P. (2006). Assessing Solution Quality in Stochastic Programs. *Math. Program* 108, 495–514. doi:10.1007/s10107-006-0720-x

Beaumont, M. A., Cornuet, J.-M., Marin, J.-M., and Robert, C. P. (2009). Adaptive Approximate Bayesian Computation. *Biometrika* 96, 983–990. doi:10.1093/biomet/asp052

Bengio, Y., Louradour, J., Collobert, R., and Weston, J. (2009). "Curriculum Learning," in International Conference on Machine Learning (ICML), June 14-18 (Montreal, Quebec, Canada: ACM), 41–48. of ACM International Conference Proceeding Series. doi:10.1145/1553374.1553380382

Bin Peng, X., Coumans, E., Zhang, T., Lee, T.-W., Tan, J., and Levine, S. (2020). "Learning Agile Robotic Locomotion Skills by Imitating Animals," in Robotics: Science and Systems (RSS), Virtual Event/Corvalis, July 12-16 (Oregon, USA. doi:10.15607/RSS.2020.XVI.064

Bongard, J., Zykov, V., and Lipson, H. (2006). Resilient Machines through Continuous Self-Modeling. *Science* 314, 1118–1121. doi:10.1126/science.1133687

Bousmalis, K., Irpan, A., Wohlhart, P., Bai, Y., Kelcey, M., Kalakrishnan, M., et al. (2018). "Using Simulation and Domain Adaptation to Improve Efficiency of Deep Robotic Grasping," in International Conference on Robotics and Automation, May 21-25 (Brisbane, Australia: ICRA), 4243–4250. doi:10.1109/ICRA.2018.8460875

Brockman, G., Cheung, V., Pettersson, L., Schneider, J., Schulman, J., Tang, J., et al. (2016). Openai Gym. *arXiv* 1606.01540

Brogliato, B., Maschke, B., Lozano, R., and Egeland, O. (2007). Dissipative Systems Analysis and Control. *Theor. Appl.* 2. doi:10.1007/978-1-84628-517-2

Brooks, R. A. (1992). "Artificial Life and Real Robots," in European Conference on Artificial Life (ECAL), December 11-13 (Paris, France, 3–10.

Brosseit, J., Hahner, B., Muratore, F., Gienger, M., and Peters, J. (2021). *Destilled Domain Randomization*, 2112, 03149.

Calandra, R., Ivaldi, S., Deisenroth, M. P., Rueckert, E., and Peters, J. (2015). "Learning Inverse Dynamics Models with Contacts," in International Conference on Robotics and Automation (ICRA), 26-30 May (Seattle, WA, USA: IEEE), 3186–3191. doi:10.1109/ICRA.2015.7139638

Carlini, N., and Wagner, D. (2017). "Adversarial Examples Are Not Easily Detected," in Workshop on Artificial Intelligence and Security (AISec), November 3 (Dallas, TX, USA: ACM), 3–14. doi:10.1145/3128572.3140444

Chatzilygeroudis, K., and Mouret, J.-B. (2018). "Using Parameterized Black-Box Priors to Scale up Model-Based Policy Search for Robotics," in International Conference on Robotics and Automation (ICRA), May 21-25 (Brisbane, Australia: IEEE), 1–9. doi:10.1109/ICRA.2018.8461083

Chatzilygeroudis, K., Rama, R., Kaushik, R., Goepp, D., Vassiliades, V., and Mouret, J.-B. (2017). "Black-box Data-Efficient Policy Search for Robotics," in International Conference on Intelligent Robots and Systems (IROS), September 24-28 (Vancouver, BC: CanadaIEEE), 51–58. doi:10.1109/IROS.2017.8202137

Chatzilygeroudis, K., Vassiliades, V., Stulp, F., Calinon, S., and Mouret, J.-B. (2020). A Survey on Policy Search Algorithms for Learning Robot Controllers in a Handful of Trials. *IEEE Trans. Robot.* 36, 328–347. doi:10.1109/TRO.2019.2958211

Chebotar, Y., Handa, A., Makoviychuk, V., Macklin, M., Issac, J., Ratliff, N., et al. (2019). "Closing the Sim-To-Real Loop: Adapting Simulation Randomization with Real World Experience," in International Conference on Robotics and Automation (ICRA), May 20-24 (Montreal, QC, Canada, 8973–8979. doi:10.1109/ICRA.2019.8793789

Chen, G., Choi, W., Yu, X., Han, T. X., and Chandraker, M. (2017). "Learning Efficient Object Detection Models with Knowledge Distillation," in Conference on Neural Information Processing Systems (NIPS), December 4-9 (Long Beach, CA, USA, 742–751.

Chow, Y. S., and Robbins, H. (1963). On Optimal Stopping Rules. *Z. Wahrscheinlichkeitstheorie Verw Gebiete* 2, 33–49. doi:10.1007/bf00535296

Christiano, P. F., Shah, Z., Mordatch, I., Schneider, J., Blackwell, T., Tobin, J., et al. (2016). Transfer from Simulation to Real World through Learning Deep Inverse Dynamics Model. *arXiv* 1610.03518

Chung, S.-J., and Pollard, N. (2016). Predictable Behavior during Contact Simulation: a Comparison of Selected Physics Engines. *Comp. Anim. Virtual Worlds* 27, 262–270. doi:10.1002/cav.1712

Ciresan, D., Meier, U., and Schmidhuber, J. (2012). "Multi-column Deep Neural Networks for Image Classification," in Conference on Computer Vision and Pattern Recognition (CVPR), June 16-21 (RI, USA: IEEE Computer Society), 3642–3649. doi:10.1109/CVPR.2012.6248110

Collins, J., Howard, D., and Leitner, J. (2019). "Quantifying the Reality gap in Robotic Manipulation Tasks," in International Conference on Robotics and Automation (ICRA), May 20-24 (Montreal, QC, Canada: IEEE), 6706–6712. doi:10.1109/ICRA.2019.8793591

Coumans, E. (2020). Tiny Differentiable Simulator. Available at: https://github.com/google-research/tiny-differentiable-simulator.

Craik, K. J. W. (1943). *The Nature of Explanation*.

Cranmer, K., Brehmer, J., and Louppe, G. (2020). The Frontier of Simulation-Based Inference. *Proc. Natl. Acad. Sci. USA* 117, 30055–30062. doi:10.1073/pnas.1912789117

Cui, J., Kingsbury, B., Ramabhadran, B., Saon, G., Sercu, T., Audhkhasi, K., et al. (2017). "Knowledge Distillation across Ensembles of Multilingual Models for Low-Resource Languages," in Knowledge distillation across ensembles of multilingual models for low-resource languages, March 5-9 (ICASSP, New Orleans, LA, USA: IEEE), 4825–4829. doi:10.1109/ICASSP.2017.7953073

Cully, A., Clune, J., Tarapore, D., and Mouret, J.-B. (2015). Robots that Can Adapt like Animals. *Nature* 521, 503–507. doi:10.1038/nature14422

Cutler, M., and How, J. P. (2016). "Autonomous Drifting Using Simulation-Aided Reinforcement Learning," in International Conference on Robotics and Automation (ICRA), May 16-21 (Stockholm, Sweden: IEEE), 5442–5448. doi:10.1109/ICRA.2016.7487756

Cutler, M., and How, J. P. (2015). "Efficient Reinforcement Learning for Robots Using Informative Simulated Priors," in International Conference on Robotics and Automation (ICRA), 26-30 May (Seattle, WA, USA: IEEE), 2605–2612. doi:10.1109/ICRA.2015.7139550

Czarnecki, W. M., Pascanu, R., Osindero, S., Jayakumar, S. M., Swirszcz, G., and Jaderberg, M. (2019). "Distilling Policy Distillation," in International Conference on Artificial Intelligence and Statistics (AISTATS) (Naha, Okinawa, Japan16-18 April: PMLR), 1331–1340. of Proceedings of Machine Learning Research.89

Dai, H., Landry, B., Yang, L., Pavone, M., and Tedrake, R. (2021). "Lyapunov-stable Neural-Network Control," in Robotics: Science and Systems (RSS), July 12-16. Virtual Event. doi:10.15607/RSS.2021.XVII.063

Dai, T., Arulkumaran, K., Tukra, S., Behbahani, F., and Bharath, A. A. (2019). Analysing Deep Reinforcement Learning Agents Trained with Domain Randomisation. *arXiv* 1912.08324

Degrave, J., Hermans, M., Dambre, J., and Wyffels, F. (2019). A Differentiable Physics Engine for Deep Learning in Robotics. *Front. Neurorobot.* 13, 6. doi:10.3389/fnbot.2019.00006

Deisenroth, M. P., Neumann, G., and Peters, J. (2011). A Survey on Policy Search for Robotics. *FNT in Robotics* 2, 1–142. doi:10.1561/2300000021

Deisenroth, M. P., and Rasmussen, C. E. (2011). "PILCO: a Model-Based and Data-Efficient Approach to Policy Search," in International Conference on Machine Learning (ICML), June 28 - July 2 (Bellevue, Washington, USA, 465–472.

Delage, E., and Ye, Y. (2010). Distributionally Robust Optimization under Moment Uncertainty with Application to Data-Driven Problems. *Operations Res.* 58, 595–612. doi:10.1287/opre.1090.0741

Dennett, D. C. (1975). Why the Law of Effect Will Not Go Away. *J. Theor. Soc. Behav.* doi:10.1111/j.1468-5914.1975.tb00350.x

Du, Y., Watkins, O., Darrell, T., Abbeel, P., and Pathak, D. (2021). *Auto-tuned Sim-To-Real Transfer. arXiv* 2104.07662.

Durkan, C., Murray, I., and Papamakarios, G. (2020). "On Contrastive Learning for Likelihood-free Inference," in International Conference on Machine Learning (ICML), July 13-18 (PMLR), 2771–2781. Virtual Eventof Proceedings of Machine Learning Research.119

Erez, T., and Smart, W. D. (2008). "What Does Shaping Mean for Computational Reinforcement Learning?," in International Conference on Development and Learning (ICDL) (Monterey, CA, USA: IEEE), 215–219.

Erez, T., Tassa, Y., and Todorov, E. (2015). "Simulation Tools for Model-Based Robotics: Comparison of Bullet, Havok, Mujoco, ODE and Physx," in International Conference on Robotics and Automation (ICRA), May 26-30 (Seattle, WA, USA, 4397–4404. doi:10.1109/ICRA.2015.7139807

Fawzi, A., Fawzi, O., and Frossard, P. (2015). "Fundamental Limits on Adversarial Robustness," in International Conference on Machine Learning (ICML), Workshop on Deep Learning.

Fearnhead, P., and Prangle, D. (2012). Constructing Summary Statistics for Approximate Bayesian Computation: Semi-automatic Approximate Bayesian Computation. *J. R. Stat. Soc.* 74, 419–474. doi:10.1111/j.1467-9868.2011.01010.x

Feldbaum, A. A. (1960). Dual Control Theory. I. *Avtomatika i Telemekhanika* 21, 1240–1249.

Finn, C., Abbeel, P., and Levine, S. (2017). "Model-agnostic Meta-Learning for Fast Adaptation of Deep Networks," in International Conference on Machine Learning, 6-11 August (Sydney, NSW, Australia: ICML), 1126–1135.

Forrester, A. I. J., Sóbester, A., and Keane, A. J. (2007). Multi-fidelity Optimization via Surrogate Modelling. *Proc. R. Soc. A.* 463, 3251–3269. doi:10.1098/rspa.2007.1900

Goodfellow, I. J., Shlens, J., and Szegedy, C. (2015). "Explaining and Harnessing Adversarial Examples," in International Conference on Learning Representations (ICLR), May 7-9 (San Diego, CA, USA. Conference Track.

Grant, E., Finn, C., Levine, S., Darrell, T., and Griffiths, T. L. (2018). "Recasting Gradient-Based Meta-Learning as Hierarchical Bayes," in International Conference on Learning Representations (ICLR), April 30 - May 3, 2018 (Vancouver, BC, Canada. Conference Track (OpenReview.net).

Greenberg, D. S., Nonnenmacher, M., and Macke, J. H. (2019). "Automatic Posterior Transformation for Likelihood-free Inference," in International Conference on Machine Learning (ICML), 9-15 June (Long Beach, California, USA: PMLR), 2404–2414. of Proceedings of Machine Learning Research.97

Greydanus, S., Dzamba, M., and Yosinski, J. (2019). "Hamiltonian Neural Networks," in Conference on Neural Information Processing Systems (NeurIPS), December 8-14 (Vancouver, BC, Canada, 15353–15363.

Höfer, S., Bekris, K. E., Handa, A., Higuera, J. C. G., Golemo, F., Mozifian, M., et al. (2020). Perspectives on Sim2real Transfer for Robotics: A Summary of the R:SS 2020 Workshop. *arXiv* 2012.03806.

Haefner, B., Queau, Y., Mollenhoff, T., and Cremers, D. (2018). "Fight Ill-Posedness with Ill-Posedness: Single-Shot Variational Depth Super-resolution from Shading," in Conference on Computer Vision and Pattern Recognition (CVPR), June 18-22 (Salt Lake City, UT, USA: IEEE Computer Society), 164–174. doi:10.1109/CVPR.2018.00025

Hanna, J. P., and Stone, P. (2017). "Grounded Action Transformation for Robot Learning in Simulation," in AAAI Conference on Artificial Intelligence, February 4-9 (San Francisco, California, USA, 3834–3840.

Heiden, E., Millard, D., Coumans, E., Sheng, Y., and Sukhatme, G. S. (2021). "NeuralSim: Augmenting Differentiable Simulators with Neural Networks," in International Conference on Robotics and Automation (ICRA), May 30 - June 5 (Xi'an, China. doi:10.1109/icra48506.2021.9560935

Hermans, J., Begy, V., and Louppe, G. (2020)., 119. PMLR, 4239–4248. of Proceedings of Machine Learning Research.Likelihood-free MCMC with Amortized Approximate Ratio EstimatorsInternational Conference on Machine Learning (ICML), Virtual Event13-18 July

Hinton, G. E., Vinyals, O., and Dean, J. (2015). Distilling the Knowledge in a Neural Network. *arXiv* 1503.02531.

Hu, Y., Liu, J., Spielberg, A., Tenenbaum, J. B., Freeman, W. T., Wu, J., et al. (2019). "Chainqueen: A Real-Time Differentiable Physical Simulator for Soft Robotics," in International Conference on Robotics and Automation (ICRA), May 20-24 (Montreal, QC, Canada: IEEE), 6265–6271. doi:10.1109/ICRA.2019.8794333

Huang, S. H., Papernot, N., Goodfellow, I. J., Duan, Y., and Abbeel, P. (2017). "Adversarial Attacks on Neural Network Policies, Workshop Track," in International Conference on Learning Representations (ICLR) Toulon, April 24-26 (France. OpenReview.net).

Ilyas, A., Santurkar, S., Tsipras, D., Engstrom, L., Tran, B., and Madry, A. (2019). "Adversarial Examples Are Not Bugs, They Are Features," in Conference on Neural Information Processing Systems (NeurIPS), December 8-14 (Vancouver, BC, Canada, 125–136.

Ivaldi, S., Peters, J., Padois, V., and Nori, F. (2014). "Tools for Simulating Humanoid Robot Dynamics: A Survey Based on User Feedback," in Tools for simulating humanoid robot dynamics: A survey based on user feedback, November 18-20 (Humanoids, Madrid, Spain: IEEE), 842–849. doi:10.1109/HUMANOIDS.2014.7041462

Jakobi, N. (1997). Evolutionary Robotics and the Radical Envelope-Of-Noise Hypothesis. *Adaptive Behav.* 6, 325–368. doi:10.1177/105971239700600205

Jakobi, N., Husbands, P., and Harvey, I. (1995). "Noise and the Reality gap: The Use of Simulation in Evolutionary Robotics," in Advances in Artificial Life, June 4-6 (Granada, Spain, 704–720. 704–720. doi:10.1007/3-540-59496-5_337

James, S., Davison, A. J., and Johns, E. (2017). "Transferring End-To-End Visuomotor Control from Simulation to Real World for a Multi-Stage Task," in Conference on Robot Learning (CoRL), November 13-15 (Mountain View, California, USA: PMLR), 334–343. of Proceedings of Machine Learning Research.78

James, S., Wohlhart, P., Kalakrishnan, M., Kalashnikov, D., Irpan, A., Ibarz, J., et al. (2019). "Sim-to-real via Sim-To-Sim: Data-Efficient Robotic Grasping via Randomized-To-Canonical Adaptation Networks," in Conference on Computer Vision and Pattern Recognition (CVPR), June 16-20 (Long Beach, CA, USA: Computer Vision Foundation/IEEE), 12627–12637. doi:10.1109/CVPR.2019.01291

Jiang, Y., Zhang, T., Ho, D., Bai, Y., Liu, C. K., Levine, S., et al. (2021). Simgan: Hybrid Simulator Identification for Domain Adaptation via Adversarial Reinforcement Learning. *arXiv* 2101.06005

Körber, M., Lange, J., Rediske, S., Steinmann, S., and Glück, R. (2021). Comparing Popular Simulation Environments in the Scope of Robotics and Reinforcement Learning. *arXiv* 2103.04616

Kadian, A., Truong, J., Gokaslan, A., Clegg, A., Wijmans, E., Lee, S., et al. (2020). Sim2real Predictivity: Does Evaluation in Simulation Predict Real-World Performance? *IEEE Robot. Autom. Lett.* 5, 6670–6677. doi:10.1109/LRA.2020.3013848

Kahn, H., and Marshall, A. W. (1953). Methods of Reducing Sample Size in Monte Carlo Computations. *Or* 1, 263–278. doi:10.1287/opre.1.5.263

Kakade, S. M. (2001). VancouverBritish Columbia, Canada, 1531–1538.A Natural Policy GradientConference on Neural Information Processing Systems (NIPS) December 3-8.

Kaspar, M., Munoz Osorio, J. D., and Bock, J. (2020). "Sim2real Transfer for Reinforcement Learning without Dynamics Randomization," in International Conference on Intelligent Robots and Systems (IROS), October 24 - January 24 (Las Vegas, NV, USA: IEEE), 4383–4388. doi:10.1109/IROS45743.2020.9341260

Kishimoto, K., and Amari, S. (1979). Existence and Stability of Local Excitations in Homogeneous Neural fields. *J. Math. Biol.* 7, 303–318. doi:10.1007/bf00275151

Klink, P., Abdulsamad, H., Belousov, B., D'Eramo, C., Peters, J., and Pajarinen, J. (2021). *A Probabilistic Interpretation of Self-Paced Learning with Applications to Reinforcement Learning*, 13176. *arXiv* 2102.

Klink, P., Abdulsamad, H., Belousov, B., and Peters, J. (2019). "Self-paced Contextual Reinforcement Learning," in Conference on Robot Learning (CoRL), October 30 - November 1 (Osaka, Japan: PMLR), 513–529. of Proceedings of Machine Learning Research.100.

Kober, J., Bagnell, J. A., and Peters, J. (2013). Reinforcement Learning in Robotics: A Survey. *Int. J. Robotics Res.* 32, 1238–1274. doi:10.1177/0278364913495721

Kolev, K., Klodt, M., Brox, T., and Cremers, D. (2009). Continuous Global Optimization in Multiview 3d Reconstruction. *Int. J. Comput. Vis.* 84, 80–96. doi:10.1007/s11263-009-0233-1

Koos, S., Mouret, J.-B., and Doncieux, S. (2010). "Crossing the Reality gap in Evolutionary Robotics by Promoting Transferable Controllers," in Genetic and Evolutionary Computation Conference (GECCO), July 7-11 (Portland, Oregon, USA: ACM), 119–126. doi:10.1145/1830483.1830505

Koos, S., Mouret, J.-B., and Doncieux, S. (2013). The Transferability Approach: Crossing the Reality gap in Evolutionary Robotics. *IEEE Trans. Evol. Computat.* 17, 122–145. doi:10.1109/TEVC.2012.2185849

Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2012). "Imagenet Classification with Deep Convolutional Neural Networks," in Conference on Neural Information Processing Systems (NIPS), 1106–1114.*Lake Tahoe, Nev. United States December*3-6

Krotkov, E., Hackett, D., Jackel, L., Perschbacher, M., Pippine, J., Strauss, J., et al. (2017). The DARPA Robotics challenge Finals: Results and Perspectives. *J. Field Robotics* 34, 229–240. doi:10.1002/rob.21683

Kumar, A., Fu, Z., Pathak, D., and Malik, J. (2021). "RMA: Rapid Motor Adaptation for Legged Robots," in Robotics: Science and Systems (RSS), Virtual Event, July 12-16. doi:10.15607/RSS.2021.XVII.011

Kumar, M. P., Packer, B., and Koller, D. (2010). "Self-paced Learning for Latent Variable Models," in Conference on Neural Information Processing Systems (NIPS), 6-9 December (Vancouver, British Columbia, Canada, 1189–1197.

Kurakin, A., Goodfellow, I. J., and Bengio, S. (2017). "Adversarial Examples in the Physical World," in International Conference on Learning Representations (ICLR) Toulon, April 24-26 (France. Workshop Track (OpenReview.net).

Landau, I. D., Lozano, R., M'Saad, M., and Karimi, A. (2011). *Adaptive Control: Algorithms, Analysis and Applications*. 2 edn. Springer Science & Business Media.

Levine, S., and Koltun, V. (2013). "Variational Policy Search via Trajectory Optimization," in Conference on Neural Information Processing Systems (NIPS), December 5-8 (Lake Tahoe, Nevada, USA, 207–215.

Levine, S., Pastor, P., Krizhevsky, A., Ibarz, J., and Quillen, D. (2018). Learning Hand-Eye Coordination for Robotic Grasping with Deep Learning and Large-Scale Data Collection. *Int. J. Robotics Res.* 37, 421–436. doi:10.1177/0278364917710318

Lillicrap, T. P., Hunt, J. J., Pritzel, A., Heess, N., Erez, T., Tassa, Y., et al. (2016). "Continuous Control with Deep Reinforcement Learning," in International Conference on Learning Representations (ICLR), May 2-4 (San Juan, Puerto Rico. Conference Track (OpenReview.net).

Liu, Y., Ramachandran, P., Liu, Q., and Peng, J. (2017). "Stein Variational Policy Gradient," in Association for Uncertainty in Artificial Intelligence (UAI) (Sydney, Australia, August 11–15.

Lowrey, K., Kolev, S., Dao, J., Rajeswaran, A., and Todorov, E. (2018). "Reinforcement Learning for Non-prehensile Manipulation: Transfer from Simulation to Physical System," in Simulation, Modeling, and Programming for Autonomous Robots (SIMPAR), May 16-19 (Brisbane, Australia, 35–42. doi:10.1109/SIMPAR.2018.8376268

Lueckmann, J., Gonçalves, P. J., Bassetto, G., Öcal, K., Nonnenmacher, M., and Macke, J. H. (2017). "Flexible Statistical Inference for Mechanistic Models of Neural Dynamics," in Conference on Neural Information Processing Systems, December 4-9 (Long Beach, CA, USA: NIPS), 1289–1299.

Luksch, T., Gienger, M., Mühlig, M., and Yoshiike, T. (2012). "Adaptive Movement Sequences and Predictive Decisions Based on Hierarchical Dynamical Systems," in International Conference on Intelligent Robots and Systems (IROS), October 7-12 (Vilamoura, Algarve, Portugal: IEEE), 2082–2088. doi:10.1109/iros.2012.6385651

Lutter, M., Mannor, S., Peters, J., Fox, D., and Garg, A. (2021a). *Robust Value Iteration for Continuous Control Tasks. arXiv* 2105.12189.

Lutter, M., Ritter, C., and Peters, J. (2019). "Deep Lagrangian Networks: Using Physics as Model Prior for Deep Learning," in International Conference on Learning Representations (ICLR), May 6-9 (New Orleans, LA, USA. Conference Track (OpenReview.net).

Lutter, M., Silberbauer, J., Watson, J., and Peters, J. (2021b). Differentiable Physics Models for Real-World Offline Model-Based Reinforcement Learning. *arXiv* 2011.01734

Mak, W.-K., Morton, D. P., and Wood, R. K. (1999). Monte Carlo Bounding Techniques for Determining Solution Quality in Stochastic Programs. *Operations Res. Lett.* 24, 47–56. doi:10.1016/S0167-6377(98)00054-6

Mandlekar, A., Zhu, Y., Garg, A., Fei-Fei, L., and Savarese, S. (2017). Vancouver, BC: Canada. September *24-28*. 3932–3939. doi:10.1109/IROS.2017.8206245Adversarially Robust Policy Learning: Active Construction of Physically-Plausible PerturbationsInternational Conference on Intelligent Robots and Systems (IROS)

Marco, A., Berkenkamp, F., Hennig, P., Schoellig, A. P., Krause, A., Schaal, S., et al. (2017). "Virtual vs. Real: Trading off Simulations and Physical Experiments in Reinforcement Learning with Bayesian Optimization," in International Conference on Robotics and Automation (ICRA), May 29 - Jun 3 (Marina Bay Sands, Singapore. doi:10.1109/icra.2017.7989186

Marjoram, P., Molitor, J., Plagnol, V., and Tavare, S. (2003). Markov Chain Monte Carlo without Likelihoods. *Proc. Natl. Acad. Sci.* 100, 15324–15328. doi:10.1073/pnas.0306899100

Martin Martin, R., and Brock, O. (2014). "Online Interactive Perception of Articulated Objects with Multi-Level Recursive Estimation Based on Task-specific Priors," in International Conference on Intelligent Robots and Systems (IROS), September 14-18 (Chicago, IL, USA: IEEE), 2494–2501. doi:10.1109/IROS.2014.6942902

Matas, J., James, S., and Davison, A. J. (2018). "Sim-to-real Reinforcement Learning for Deformable Object Manipulation," in Conference on Robot Learning (CoRL), October 29-31 (Zürich, Switzerland: PMLR), 734–743. of Proceedings of Machine Learning Research.87

Matl, C., Narang, Y., Bajcsy, R., Ramos, F., and Fox, D. (2020). "Inferring the Material Properties of Granular media for Robotic Tasks," in International Conference on Robotics and Automation (ICRA) (Paris, FranceMay 31 - August 31: IEEE), 2770–2777. doi:10.1109/ICRA40945.2020.9197063

Mehta, B., Diaz, M., Golemo, F., Pal, C. J., and Paull, L. (2019). "Active Domain Randomization," in Conference on Robot Learning (CoRL), October 30 - November 1 (Osaka, Japan: PMLR), 1162–1176. of Proceedings of Machine Learning Research.100

Mehta, B., Handa, A., Fox, D., and Ramos, F. (2020). "A User's Guide to Calibrating Robotics Simulators," in Conference on Robot Learning (CoRL), Virtual Event, November 16 - 18 (PMLR). Proceedings of Machine Learning Research.

Metropolis, N., and Ulam, S. (1949). The Monte Carlo Method. *J. Am. Stat. Assoc.* 44, 335–341. doi:10.1080/01621459.1949.10483310

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., et al. (2015). Human-level Control through Deep Reinforcement Learning. *Nature* 518, 529–533. doi:10.1038/nature14236

Molchanov, A., Chen, T., Honig, W., Preiss, J. A., Ayanian, N., and Sukhatme, G. S. (2019). "Sim-to-(multi)-real: Transfer of Low-Level Robust Control Policies to Multiple Quadrotors," in International Conference on Intelligent Robots and Systems (IROS), November 3-8 (Macau, SAR, China: IEEE), 59–66. doi:10.1109/IROS40897.2019.8967695

Mordatch, I., Lowrey, K., and Todorov, E. (2015). "Ensemble-cio: Full-Body Dynamic Motion Planning that Transfers to Physical Humanoids," in International Conference on Intelligent Robots and Systems (IROS), September 28 - October 2 (Hamburg, Germany, 5307–5314. doi:10.1109/IROS.2015.7354126

Morere, P., Ott, L., and Ramos, F. (2019). Learning to Plan Hierarchically from Curriculum. *IEEE Robot. Autom. Lett.* 4, 2815–2822. doi:10.1109/LRA.2019.2920285

Mozian, M., Camilo Gamboa Higuera, J., Meger, D., and Dudek, G. (2020). "Learning Domain Randomization Distributions for Training Robust Locomotion Policies," in International Conference on Intelligent Robots and Systems (IROS) Las Vegas, October 24 - January 24 (NV, USA: IEEE), 6112–6117. doi:10.1109/IROS45743.2020.9341019

Muratore, F., Eilers, C., Gienger, M., and Peters, J. (2021a). Data-efficient Domain Randomization with Bayesian Optimization. *IEEE Robot. Autom. Lett.* 6, 911–918. doi:10.1109/LRA.2021.3052391

Muratore, F., Gienger, M., and Peters, J. (2021b). Assessing Transferability from Simulation to Reality for Reinforcement Learning. *IEEE Trans. Pattern Anal. Mach. Intell.* 43, 1172–1183. doi:10.1109/TPAMI.2019.2952353

Muratore, F., Gruner, T., Wiese, F., Gienger, B. B. M., and Peters, J. (2021c). "Neural Posterior Domain Randomization," in Conference on Robot Learning (CoRL), Virtual Event, November 8-11 (London, England.

Muratore, F., Treede, F., Gienger, M., and Peters, J. (2018). "Domain Randomization for Simulation-Based Policy Optimization with Transferability Assessment," in Conference on Robot Learning (CoRL) (Zürich, SwitzerlandOctober 29-31: PMLR), 700–713. of Proceedings of Machine Learning Research.87

Nagabandi, A., Clavera, I., Liu, S., Fearing, R. S., Abbeel, P., Levine, S., et al. (2019). "Learning to Adapt in Dynamic, Real-World Environments through Meta-Reinforcement Learning," in International Conference on Learning Representations (ICLR) New Orleans, May 6-9 (LA, USA. OpenReview.net).

Ng, A. Y., and Jordan, M. I. (2000). "PEGASUS: a Policy Search Method for Large Mdps and Pomdps," in UAI, June 30 - July 3 (Stanford, California, USA: Morgan Kaufmann), 406–415.

OpenAIAkkaya, I., Andrychowicz, M., Chociej, M., Litwin, M., McGrew, B., et al. (2019). Solving Rubik's Cube with a Robot Hand. *arXiv* 1910.07113

Pan, S. J., and Yang, Q. (2010). A Survey on Transfer Learning. *IEEE Trans. Knowl. Data Eng.* 22, 1345–1359. doi:10.1109/TKDE.2009.191

Papamakarios, G., and Murray, I. (2016). "Fast ϵ-free Inference of Simulation Models with Bayesian Conditional Density Estimation," in Conference on Neural Information Processing Systems (NIPS), December 5-10 (Barcelona, Spain, 1028–1036.

Papamakarios, G., Sterratt, D. C., and Murray, I. (2019). "Sequential Neural Likelihood: Fast Likelihood-free Inference with Autoregressive Flows," in International Conference on Artificial Intelligence and Statistics (AISTATS), April 16-18 (Naha, Okinawa, Japan: PMLR), 837–848. of Proceedings of Machine Learning Research.89

Parisotto, E., Ba, L. J., and Salakhutdinov, R. (2016). "Actor-mimic: Deep Multitask and Transfer Reinforcement Learning," in International Conference on Learning Representations (ICLR) San Juan, May 2-4 (Puerto Rico. Conference Track.

Paul, S., Osborne, M. A., and Whiteson, S. (2019). Fingerprint Policy Optimisation for Robust Reinforcement Learning. In International Conference on Machine Learning (ICML), Long Beach California, USA, 9-15 June (PMLR), vol. 97

Peng, X. B., Andrychowicz, M., Zaremba, W., and Abbeel, P. (2018). "Sim-to-real Transfer of Robotic Control with Dynamics Randomization," in International Conference on Robotics and Automation (ICRA), May 21-25 (Brisbane, Australia, 1–8. doi:10.1109/ICRA.2018.8460528

Perlin, K. (2002). Improving Noise. *ACM Trans. Graph.* 21, 681–682. doi:10.1145/566654.566636

Peters, J., Mülling, K., and Altun, Y. (2010). "Relative Entropy Policy Search," in AAAI Conference on Artificial Intelligence, July 11-15 (Atlanta, Georgia, USA.

Pinto, L., Andrychowicz, M., Welinder, P., Zaremba, W., and Abbeel, P. (2018). "Asymmetric Actor Critic for Image-Based Robot Learning," in Robotics: Science and Systems (RSS), June 26-30 (Pittsburgh, Pennsylvania, USA. doi:10.15607/RSS.2018.XIV.008

Pinto, L., Davidson, J., Sukthankar, R., and Gupta, A. (2017). "Robust Adversarial Reinforcement Learning," in International Conference on Machine Learning (ICML), August 6-11 (Sydney, NSW, Australia: PMLR), 2817–2826.

Polvara, R., Patacchiola, M., Hanheide, M., and Neumann, G. (2020). Sim-to-real Quadrotor landing via Sequential Deep Q-Networks and Domain Randomization. *Robotics* 9, 8. doi:10.3390/robotics9010008

Possas, R., Barcelos, L., Oliveira, R., Fox, D., and Ramos, F. (2020). "Online Bayessim for Combined Simulator Parameter Inference and Policy Improvement," in International Conference on Intelligent Robots and Systems (IROS) Las Vegas, October 24 - January 24 (NV, USA: IEEE), 5445–5452. doi:10.1109/IROS45743.2020.9341401

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., and Sutskever, I. (2019). *Language Models Are Unsupervised Multitask Learners*.

Rajeswaran, A., Ghotra, S., Ravindran, B., and Levine, S. (2017). "Epopt: Learning Robust Neural Network Policies Using Model Ensembles," in International Conference on Learning Representations (ICLR), Toulon, April 24-26 (France. Conference Track (OpenReview.net).

Rajeswaran, A., Mordatch, I., and Kumar, V. (2020). "A Game Theoretic Framework for Model Based Reinforcement Learning," in International Conference on Machine Learning (ICML), Virtual Event, 13-18 July (PMLR), 7953–7963. of Proceedings of Machine Learning Research.119

Ramos, F., Possas, R., and Fox, D. (2019). "Bayessim: Adaptive Domain Randomization via Probabilistic Inference for Robotics Simulators," in Robotics: Science and Systems (RSS), June 22-26 (Germany: Freiburg im Breisgau). doi:10.15607/RSS.2019.XV.029

Rasmussen, C. E., and Williams, C. K. I. (2006). Gaussian Processes for Machine Learning. *Adaptive Computation and Machine Learning*. MIT Press.

Rawlik, K., Toussaint, M., and Vijayakumar, S. (2012). Sydney, NSW, Australia: RSS. doi:10.15607/RSS.2012.VIII.045On Stochastic Optimal Control and Reinforcement Learning by Approximate InferenceRobotics: Science and SystemsJuly 9-13

Ruiz, N., Schulter, S., and Chandraker, M. (2019). "Learning to Simulate," in International Conference on Learning Representations (ICLR), May 6-9 (New Orleans, LA, USA. (OpenReview.net).

Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., et al. (2015). Imagenet Large Scale Visual Recognition challenge. *Int. J. Comput. Vis.* 115, 211–252. doi:10.1007/s11263-015-0816-y

Rusu, A. A., Colmenarejo, S. G., Gülçehre, Ç., Desjardins, G., Kirkpatrick, J., Pascanu, R., et al. (2016a). "Policy Distillation," in (San Juan, Puerto Rico. Conference Track.International Conference on Learning Representations (ICLR)May 2-4

Rusu, A. A., Rabinowitz, N. C., Desjardins, G., Soyer, H., Kirkpatrick, J., Kavukcuoglu, K., et al. (2016b). Progressive Neural Networks. *arXiv* 1606.04671

Rusu, A. A., Vecerik, M., Rothörl, T., Heess, N., Pascanu, R., and Hadsell, R. (2017). "Sim-to-real Robot Learning from Pixels with Progressive Nets," in Conference on Robot Learning (CoRL), Mountain View, November 13-15 (California, USA: PMLR), 262–270. of Proceedings of Machine Learning Research.78

Sadeghi, F., and Levine, S. (2017). "CAD2RL: Real Single-Image Flight without a Single Real Image," in Robotics: Science and Systems (RSS), July 12-16 (Cambridge, Massachusetts, USA. doi:10.15607/RSS.2017.XIII.034

Santoro, A., Bartunov, S., Botvinick, M., Wierstra, D., and Lillicrap, T. P. (2016). "Meta-learning with Memory-Augmented Neural Networks," in International Conference on Machine Learning (ICML), June 19-24 (New York City, NY, USA: JMLR.org), 1842–1850.48

Schulman, J., Wolski, F., Dhariwal, P., Radford, A., and Klimov, O. (2017). *Proximal Policy Optimization Algorithms*. *arXiv* 1707.06347.

Siekmann, J., Green, K., Warila, J., Fern, A., and Hurst, J. (2021). "Blind Bipedal Stair Traversal via Sim-To-Real Reinforcement Learning," in Robotics: Science and Systems (RSS), Virtual Event, July 12-16. doi:10.15607/RSS.2021.XVII.061

Silver, D., Huang, A., Maddison, C. J., Guez, A., Sifre, L., van den Driessche, G., et al. (2016). Mastering the Game of Go with Deep Neural Networks and Tree Search. *Nature* 529, 484–489. doi:10.1038/nature16961

Sunnåker, M., Busetto, A. G., Numminen, E., Corander, J., Foll, M., and Dessimoz, C. (2013). Approximate Bayesian Computation. *Plos Comput. Biol.* 9, e1002803. doi:10.1371/journal.pcbi.1002803

Sutanto, G., Wang, A. S., Lin, Y., Mukadam, M., Sukhatme, G. S., Rai, A., et al. (2020). "Encoding Physical Constraints in Differentiable newton-euler Algorithm," in L4DC, Virtual Event, 11-12 June (Berkeley, CA, USA: PMLR), 804–813. of Proceedings of Machine Learning Research.120

Sutton, R. S. (1991). Dyna, an Integrated Architecture for Learning, Planning, and Reacting. *SIGART Bull.* 2, 160–163. doi:10.1145/122344.122377

Sutton, R. S., Precup, D., and Singh, S. (1999). Between Mdps and Semi-mdps: A Framework for Temporal Abstraction in Reinforcement Learning. *Artif. Intelligence* 112, 181–211. doi:10.1016/S0004-3702(99)00052-1

Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I. J., et al. (2014). "Intriguing Properties of Neural Networks," in (Banff, Canada. Conference Track.International Conference on Learning Representations (ICLR)April 14-16

Tan, J., Zhang, T., Coumans, E., Iscen, A., Bai, Y., Hafner, D., et al. (2018). "Sim-to-real: Learning Agile Locomotion for Quadruped Robots," in Robotics: Science and Systems (RSS), June 26-30 (Pittsburgh, Pennsylvania, USA. doi:10.15607/RSS.2018.XIV.010

Teh, Y. W., Bapst, V., Czarnecki, W. M., Quan, J., Kirkpatrick, J., Hadsell, R., et al. (2017). "Distral: Robust Multitask Reinforcement Learning," in Conference on Neural Information Processing Systems (NIPS) (Long Beach, CA, USA, 4496–4506.

Ting, J.-A., D'Souza, A., and Schaal, S. (2011). Bayesian Robot System Identification with Input and Output Noise. *Neural Networks* 24, 99–108. doi:10.1016/j.neunet.2010.08.011

Ting, J., Mistry, M., Peters, J., Schaal, S., and Nakanishi, J. (2006). "A Bayesian Approach to Nonlinear Parameter Identification for Rigid Body Dynamics," in Robotics: Science and Systems (RSS), August 16-19 (Philadelphia, Pennsylvania, USA: The MIT Press). doi:10.15607/RSS.2006.II.032

Tobin, J., Fong, R., Ray, A., Schneider, J., Zaremba, W., and Abbeel, P. (2017). "Domain Randomization for Transferring Deep Neural Networks from Simulation to the Real World," in International Conference on Intelligent Robots and Systems (IROS), September 24-28 (Vancouver, BC: Canada), 23–30. doi:10.1109/IROS.2017.8202133

van den Oord, A., Li, Y., and Vinyals, O. (2018). Representation Learning with Contrastive Predictive Coding. *arXiv* 1807.03748

Van Parys, B., Kuhn, D., Goulart, P., and Morari, M. (2015). Distributionally Robust Control of Constrained Stochastic Systems. *IEEE Trans. Automat. Contr.* 61, 1. doi:10.1109/TAC.2015.2444134

Wang, J., Kurth-Nelson, Z., Soyer, H., Leibo, J. Z., Tirumala, D., Munos, R., et al. (2017). "Learning to Reinforcement Learn," in Cognitive Science, 16-29 July (London, UK. cognitivesciencesociety.org).

Wang, J. M., Fleet, D. J., and Hertzmann, A. (2010). Optimizing Walking Controllers for Uncertain Inputs and Environments. *ACM Trans. Graphics* 29, 73–78. doi:10.1145/1833351.177881010.1145/1778765.1778810

Watson, J., Abdulsamad, H., Findeisen, R., and Peters, J. (2021). Stochastic Control through Approximate Bayesian Input Inference. *arXiv* 2105.07693

Williams, R. J. (1992). Simple Statistical Gradient-Following Algorithms for Connectionist Reinforcement Learning. *Mach Learn.* 8, 229–256. doi:10.1007/BF00992696

Wittenmark, B. (1995). "Adaptive Dual Control Methods: An Overview," in Adaptive Systems in Control and Signal Processing 1995 (Elsevier), 67–72. doi:10.1016/b978-0-08-042375-3.50010-x

Wright, R. D., and Ramsay, T. E. (1979). On the Effectiveness of Common Random Numbers. *Manag. Sci.* 25, 649–656. doi:10.1287/mnsc.25.7.649

Wu, J., Yildirim, I., Lim, J. J., Freeman, B., and Tenenbaum, J. B. (2015). "Galileo: Perceiving Physical Object Properties by Integrating a Physics Engine with Deep Learning," in Conference on Neural Information Processing Systems (NIPS), December 7-12 (Montreal, Quebec, Canada, 127–135.

Xie, Z., Clary, P., Dao, J., Morais, P., Hurst, J. W., and van de Panne, M. (2019). "Learning Locomotion Skills for Cassie: Iterative Design and Sim-To-Real," in Conference on Robot Learning (CoRL), October 30 - November 1 (Osaka, Japan: PMLR), 317–329. of Proceedings of Machine Learning Research.100

Xie, Z., Da, X., van de Panne, M., Babich, B., and Garg, A. (2020). Dynamics Randomization Revisited: A Case Study for Quadrupedal Locomotion. *arXiv* 2011.02404

Yan, W., Vangipuram, A., Abbeel, P., and Pinto, L. (2020). "Learning Predictive Representations for Deformable Objects Using Contrastive Estimation," in Conference on Robot Learning (CoRL), Virtual Event, November 16 - 18 (Virtual Event/Cambridge, MA, USA: PMLR), 564–574. of Proceedings of Machine Learning Research.155.

Yang, C., Sun, J., Zhang, Q., and Ma, X. (2013). Lyapunov Stability and strong Passivity Analysis for Nonlinear Descriptor Systems. *IEEE Trans. Circuits Syst.* 60, 1003–1012. doi:10.1109/TCSI.2012.2215396

Yu, W., Kumar, V. C., Turk, G., and Liu, C. K. (2019a). "Sim-to-real Transfer for Biped Locomotion," in International Conference on Intelligent Robots and Systems (IROS), November 3-8 (Macau, SAR, China: IEEE), 3503–3510. doi:10.1109/IROS40897.2019.8968053

Yu, W., Liu, C. K., and Turk, G. (2019b). "Policy Transfer with Strategy Optimization," in International Conference on Learning Representations (ICLR), May 6-9 (New Orleans, LA, USA. Conference Track (OpenReview.net).

Yu, W., Tan, J., Karen Liu, C., and Turk, G. (2017). "Preparing for the Unknown: Learning a Universal Policy with Online System Identification," in Robotics: Science and Systems (RSS), July 12-16 (Cambridge, Massachusetts, USA. doi:10.15607/RSS.2017.XIII.048

Zhang, H., Chen, H., Boning, D. S., and Hsieh, C. (2021). "Robust Reinforcement Learning on State Observations with Learned Optimal Adversary," in International Conference on Learning Representations (ICLR), Virtual Event, May 3-7 (Austria. OpenReview.net).

Zhang, L. M., Plappert, M., and Zaremba, W. (2020). *Predicting Sim-To-Real Transfer with Probabilistic Dynamics Models*, 12864. *arXiv* 2009.

Zhen, J., Kuhn, D., and Wiesemann, W. (2021). Mathematical Foundations of Robust and Distributionally Robust Optimization. *arXiv* 2105.00760

Zhou, K., and Doyle, J. C. (1998). *Essentials of Robust Control*, 104. Prentice-Hall.

Zhuang, F., Qi, Z., Duan, K., Xi, D., Zhu, Y., Zhu, H., et al. (2021). A Comprehensive Survey on Transfer Learning. *Proc. IEEE* 109, 43–76. doi:10.1109/JPROC.2020.3004555

Conflict of Interest: Author FM was employed by the Technical University of Darmstadt in collaboration with the Honda Research Institute Europe. Author FR was employed by NVIDIA. Author WY was employed by Google. Author MG was employed by the Honda Research Institute Europe.

The remaining authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

The authors declare that this study received funding from the Honda Research Institute Europe. The funder had the following involvement in the study: the structuring and improvement of this article jointly with the authors, and the decision to submit it for publication.

Publisher's Note: All claims expressed in this article are solely those of the authors and do not necessarily represent those of their affiliated organizations, or those of the publisher, the editors and the reviewers. Any product that may be evaluated in this article, or claim that may be made by its manufacturer, is not guaranteed or endorsed by the publisher.