Honda Research Institute EU
Honda Research Institute Europe GmbH
https://www.honda-ri.de/

Assessing Transferability in Reinforcement Learning from Randomized Simulations

Fabio Muratore, Michael Gienger, Jan Peters

2019

Preprint:

This is an accepted article published in Multidisciplinary Conference on Reinforcement Learning and Decision Making. The final authenticated version is available online at: https://doi.org/[DOI not available]

Assessing Transferability in Reinforcement Learning from Randomized Simulations

Anonymous Author(s)
Affiliation
Address
anonymous@cs.cranberry-lemon.edu

Abstract

Exploration-based reinforcement learning of control policies on physical systems is generally time-intensive and can lead to catastrophic failures. Therefore, simulation-based policy search appears to be an appealing alternative. Unfortunately, running policy search on a slightly faulty simulator can easily lead to the maximization of the 'Simulation Optimization Bias' (SOB), where the policy exploits modeling errors of the simulator such that the resulting behavior can potentially damage the device. For this reason, much work in reinforcement learning has focused on model-free methods. The resulting lack of safe simulation-based policy learning techniques imposes severe limitations on the application of reinforcement learning to real-world systems.
In this paper, we explore how physics simulations can be utilized for a robust policy optimization by randomizing the simulator's parameters and training from model ensembles. We propose an algorithm called Simulation-based Policy Optimization with Transferability Assessment (SPOTA) that uses an estimator of the SOB to formulate a stopping criterion for training. We show that the simulation-based policy search algorithm is able to learn a control policy exclusively from a randomized simulator that can be applied directly to a different system without using any data from the latter.

Keywords: simulation optimization bias, domain randomization,
leaning from pysics simulations, reinforcement learning

1 Problem Statement

We consider a time-discrete dynamical system given by

$$\boldsymbol{s}_{t+1} \sim \mathcal{P}_{\boldsymbol{\xi}}\left(\boldsymbol{s}_{t+1} \middle| \boldsymbol{s}_t, \boldsymbol{a}_t, \boldsymbol{\xi}\right), \quad \boldsymbol{a}_t \sim \pi_{\boldsymbol{\xi}}\left(\boldsymbol{a}_t \middle| \boldsymbol{s}_t, \boldsymbol{\xi}; \boldsymbol{\theta}\right), \quad \boldsymbol{s}_0 \sim \mu_{0,\boldsymbol{\xi}}\left(\boldsymbol{s}_0 \middle| \boldsymbol{\xi}\right),$$

with the continuous state $\boldsymbol{s}_t \in \mathcal{S}_{\boldsymbol{\xi}} \subseteq \mathbb{R}^{n_s}$, and continuous action $\boldsymbol{a}_t \in \mathcal{A}_{\boldsymbol{\xi}} \subseteq \mathbb{R}^{n_a}$ at time step t. The parameters $\boldsymbol{\xi} \in \mathbb{R}^{n_\xi}$ (e.g., masses, friction coefficients, or time delays) define the environment a.k.a. the domain. They also parametrize the transition probability density function $\mathcal{P}_{\boldsymbol{\xi}} \colon \mathcal{S}_{\boldsymbol{\xi}} \times \mathcal{A}_{\boldsymbol{\xi}} \times \mathcal{S}_{\boldsymbol{\xi}} \to \mathbb{R}^+$ which describes the system's stochastic dynamics. The initial state $\boldsymbol{s}_0$ is drawn from the distribution $\mu_{0,\boldsymbol{\xi}} \colon \mathcal{S}_{\boldsymbol{\xi}} \to \mathbb{R}^+$. We further define a deterministic reward function $r \colon \mathcal{S}_{\boldsymbol{\xi}} \times \mathcal{A}_{\boldsymbol{\xi}} \to \mathbb{R}$, and a discount factor $\gamma \in [0, 1]$. Finally, the Markov decision process is fully described by the tuple $\mathcal{M}_{\boldsymbol{\xi}} \coloneqq \langle \mathcal{S}_{\boldsymbol{\xi}}, \mathcal{A}_{\boldsymbol{\xi}}, \mathcal{P}_{\boldsymbol{\xi}}, \mu_{0,\boldsymbol{\xi}}, r, \gamma \rangle$.

Simulators can be obtained by implementing a set of physical laws and estimating their associated parameters by system identification. It is important to keep in mind that even if this procedure yields a very accurate model parameter estimate, simulations are nevertheless just approximations of the real world and are thus always flawed. Here, the physics parameters are drawn from a probability distribution $\boldsymbol{\xi} \sim p(\boldsymbol{\xi}; \psi)$, parametrized by ψ (e.g., mean, variance). As done in [9, 14, 13, 12], we use this distribution as a prior which ensures the physical plausibility of each parameter. Additionally, using the Gauss-Markov theorem one could also compute the parameters' covariance and hence construct a normal distribution for each domain parameter. Either way, specifying the distribution $p(\boldsymbol{\xi}; \psi)$ in the current state-of-the-art requires the researcher to make design decisions.

In general, the goal of a Reinforcement Learning (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 stochastic policy $\pi(\boldsymbol{a}_t | \boldsymbol{s}_t; \boldsymbol{\theta})$, characterized by its parameters $\boldsymbol{\theta} \in \mathbb{R}^{n_\theta}$, is defined as

$$J(\boldsymbol{\theta}, \boldsymbol{\xi}) = \mathbb{E}_{\boldsymbol{\tau}}\left[\sum_{t=0}^{T-1} \gamma^t r(\boldsymbol{s}_t, \boldsymbol{a}_t) \middle| \boldsymbol{\theta}, \boldsymbol{\xi}\right]. \tag{1}$$

The resulting state-action pairs are collected in trajectories a.k.a. rollouts $\boldsymbol{\tau} = \{\boldsymbol{s}_t, \boldsymbol{a}_t\}_{t=0}^{T-1}$. We aim at augmenting the standard RL setting with the concept of domain randomization, i.e. we want to maximize the expectation of the expected discounted return over all (feasible) realizations of the source domain

$$J(\boldsymbol{\theta}) = \mathbb{E}_{\boldsymbol{\xi}}[J(\boldsymbol{\theta}, \boldsymbol{\xi})]. \tag{2}$$

This score quantifies how well the policy would perform over an infinitely large ensemble of variations of the nominal domain $\mathcal{M}_{\bar{\boldsymbol{\xi}}}$. When training exclusively in simulation, we do not know the true physics model, and thus do not have access to the true $J(\boldsymbol{\theta})$ from (2). Instead, we maximize the estimated expected return from samples obtained by a randomized physics simulator. Thereby, we update the policy parameters $\boldsymbol{\theta}$ according to a policy optimization algorithm. The inevitable imperfections of physics simulations will be exploited by any policy search method if it could thereby achieve a 'virtual' improvement, i.e., an increase of $J(\boldsymbol{\theta})$ in simulation. To counter this undesirable behavior, we define a Stochastic Program (SP)

$$J(\boldsymbol{\theta}^\star) = \max_{\boldsymbol{\theta} \in \Theta} \mathbb{E}_{\boldsymbol{\xi}}[J(\boldsymbol{\theta}, \boldsymbol{\xi})],$$

where $\boldsymbol{\theta}$ is the decision variable, $\Theta \subseteq \mathbb{R}^{n_\theta}$ is the associated feasible set, $\boldsymbol{\xi}$ is a random variable, and $J(\boldsymbol{\theta}, \boldsymbol{\xi})$ is a real-valued function.

The SP above can be approximated by

$$\hat{J}_n(\boldsymbol{\theta}_n^\star) = \max_{\boldsymbol{\theta} \in \Theta} \hat{J}_n(\boldsymbol{\theta}) = \max_{\boldsymbol{\theta} \in \Theta} \frac{1}{n} \sum_{i=1}^{n} J(\boldsymbol{\theta}, \boldsymbol{\xi}[i]),$$

where the expectation is replaced by Monte-Carlo average over the sampled parameters $\boldsymbol{\xi}_1, \ldots, \boldsymbol{\xi}_n$, and $\boldsymbol{\theta}_n^\star$ is the solution to the approximated SP. Note that $\hat{J}_n(\boldsymbol{\theta})$ is an unbiased estimator of $J(\boldsymbol{\theta})$.

Referring to [4], sample-based optimization is guaranteed to be optimistically biased in the RL setting. In order to quantify this effect, we define the Simulation Optimization Bias (SOB) to be

$$\mathrm{b}\left[\hat{J}_n(\boldsymbol{\theta}_n^\star)\right] = \mathbb{E}_{\boldsymbol{\xi}}\left[\max_{\hat{\boldsymbol{\theta}} \in \Theta} \hat{J}_n(\hat{\boldsymbol{\theta}})\right] - \max_{\boldsymbol{\theta} \in \Theta} \mathbb{E}_{\boldsymbol{\xi}}\left[J(\boldsymbol{\theta}, \boldsymbol{\xi})\right] \geq 0,$$

where the first term is the optimum for the samples and the second term is the true optimum. Intuitively, we want to minimize the SOB in order to achieve the highest transferability of the policy. A visualization of the SOB is depicted in Figure 1.

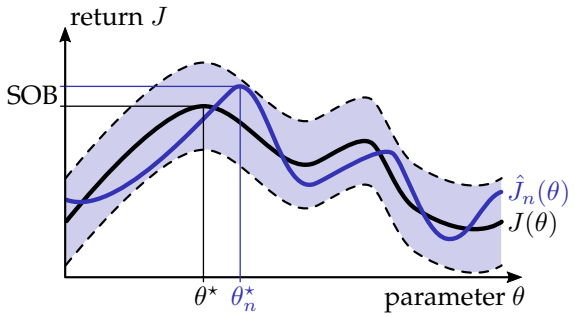

Figure 1: Simulation Optimization Bias (SOB) between the true optimum $\theta^\star$ and the sample-based optimum $\theta_n^\star$. The shaded region visualizes the standard deviation around $J(\theta)$, and $\hat{J}_n(\theta)$ is determined by a particular set of n sampled domains.

Since computing the SOB is intractable, the proposed method resides to calculating the Optimality Gap (OG). At a solution candidate $\boldsymbol{\theta}^c$, the OG is defined as

$$G(\boldsymbol{\theta}^c) = \max_{\boldsymbol{\theta} \in \Theta} \mathbb{E}_{\boldsymbol{\xi}}[J(\boldsymbol{\theta}, \boldsymbol{\xi})] - \mathbb{E}_{\boldsymbol{\xi}}[J(\boldsymbol{\theta}^c, \boldsymbol{\xi})] \geq 0, \tag{3}$$

where the first term is the SP's optimal objective function value and the second term is the SP's objective function evaluated at the candidate solution. The relation between the OG and the SOB, is explained in [10]. In order to compute the OG, we estimate an upper bound

$$\hat{G}_n(\boldsymbol{\theta}^c) = \max_{\boldsymbol{\theta} \in \Theta} \hat{J}_n(\boldsymbol{\theta}) - \hat{J}_n(\boldsymbol{\theta}^c) \geq G(\boldsymbol{\theta}^c)\,, \tag{4}$$

applying the Monte-Carlo approximation of $J(\boldsymbol{\theta})$ and $J(\boldsymbol{\theta}^c)$ with n i.i.d. samples of the domain parameters $\boldsymbol{\xi}$. The complete derivation of the estimated upper bound on the OG $\hat{G}_n(\boldsymbol{\theta}^c)$ can be found in [10]. Additionally, its monotonous decrease with increasing sample size is shown. Note that the OG always exists unless the solution candidate $\boldsymbol{\theta}^c$ is a global optimum. The difference $G(\boldsymbol{\theta}^c) - \hat{G}_n(\boldsymbol{\theta}^c)$ will only diminish for an infinite number of samples, i.e., $n \to \infty$.
Framing the RL problem in (1) as a SP (2), allows for the utilization of an Upper Confidence Bound on the Optimality Gap (UCBOG) (4) as the convergence criterion for a policy search algorithm. Furthermore, the UCBOG quantifies the inevitable over-fitting to the domains experienced during training.
Finally, we want to point out that (source) domain randomization can be seen as a form of uncertainty representation. If a control policy is trained successfully on multiple variations of the scenario, i.e., an ensemble of models, it is legitimate to assume that this policy will be able to handle modeling errors better than policies that have only been trained on the nominal model $\bar{\boldsymbol{\xi}}$.

2 Simulation-Based Policy Optimization with Transferability Assessment

In [10], the policy search meta-algorithm Simulation-based Policy Optimization with Transferability Assessment (SPOTA) is introduced. SPOTA returns a set of policy parameters such that the resulting policy is able to directly transfer from an ensemble of source domains to an unseen target domain. The goal of SPOTA is not only to maximize the agent's expected discounted return under the influence of perturbed physics simulations, but also to provide an approximate probabilistic guarantee on the loss in terms of expected discounted return when applying the obtained policy to a different domain. The key novelty is the utilization of an Upper Confidence Bound on the Optimality Gap (UCBOG) as a stopping criterion for the training procedure of an RL agent. A pseudo-code description of SPOTA as well as details on its implementation are presented in [10].

3 Experiments

We introduce the ball-on-plate task (Figure 2), a balancing task in which the agent has to stabilize a ball at the center of a plate, attached to a robotic arm. The agent sends task-space acceleration commands to the simulated robot, while the robot's joints are controlled by an inverse kinematics algorithm and low-level PD-controllers. To demonstrate the applicability of SPOTA, we compare it against a Linear-Quadratic Regulator (LQR) and neural network policies trained by Trust Region Policy Optimization (TRPO) [16] as well as Ensemble Policy Optimization (EPOpt) [14]. The description of the system's modeling as well as the conducted experiments can be found in [10]. The following figures summarize parts of the results obtained from the experiments described in [10]. Additional videos can be found at `URL_is_not_anonymous`.

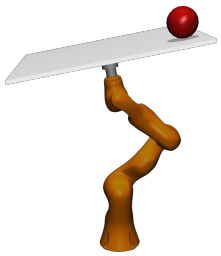

Figure 2: Illustration of the ball-on-plate task

Experiment 1 provides a comparison of different control policies' robustness against model uncertainties. Figure 3 shows the dependency of the achieved return on varying a set of selected simulator parameters. On the LQR side, there is potentially high performance, but also total trust in the dynamics model. This can be observed regarding the action delay (Figure 3 – right) which is assumed to be zero in the LQR model. The TRPO policy trained without domain randomization behaves similarly to the LQR in most cases. In contrast, the SPOTA policy is able to maintain its performance across a wider range of parameter values. Regarding the variation of the ball's rolling friction coefficient (Figure 3 – middle), it can be seen that the risk-averse EPOpt procedure leads to higher robustness for a limited subset of the possible problem instances (e.g., very low rolling friction). This effect can be explained by the fact that EPOpt optimizes the conditional value at risk of the return [14]. On the other side, the risk-neutral approaches outperform the EPOpt in most other cases.
Experiment 2 investigates if the application of domain randomization improves the transferability of a control policy between two different physics engines. The results in Figure 4 confirm two hypotheses. First, learned policies perform worse when tested using a different physics engine. Second, domain randomization alleviates this effect. The LQR baseline does well in both evaluations since the depicted rollouts are based on the nominal parameter values, and the LQR's feedback gains are not learned from samples, i.e., independent of physics engine. Compared to policy trained

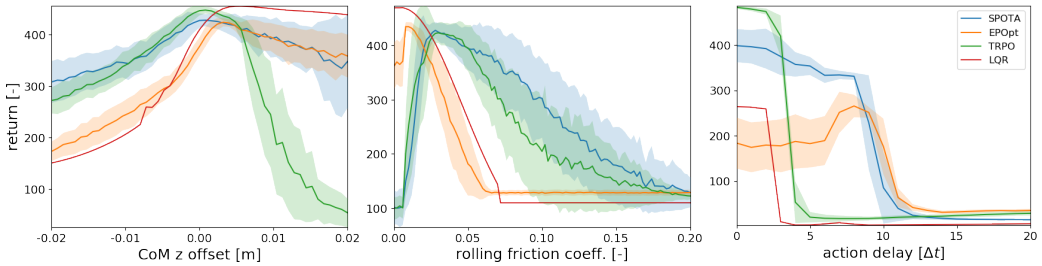

Figure 3: Performance measured in return of SPOTA, EPOpt, TRPO, and LQR policies when varying the ball's CoM offset in z direction (left), the ball's rolling friction coefficient (middle), and the policy's action delay (right).

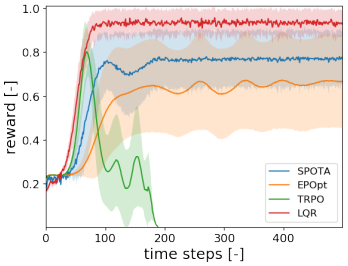

Figure 4: Cross-evaluation of SPOTA, EPOpt TRPO, and LQR policies trained in Vortex and then tested in Bullet. The simulators were set up to maximize the similarity between the physics engines as much as possible. Moreover, the simulator parameters used for evaluating are the same as for determining the LQR and TRPO policy, and equal to the nominal parameters used for the SPOTA as well as EPOpt procedure.[1]

with SPOTA or EPOpt, the TRPO policy is not able to maintain the level of performance. Note that the OG definition (3) requires the candidate's and references' simulator parameters to be from the same distribution. Practically, this assumption is violated as soon as one switches the physics engine, since the some parameters, e.g., the friction coefficients, are processed differently.

4 Related Work

The presented work is particularly related to research on the concept of the Optimality Gap (OG) in Stochastic Programs (SPs), and the application of domain randomization while learning in physics simulations.

Optimality Gap. Hobbs and Hepenstal [4] proved for linear programs that optimization is optimistically biased, given that there are errors in estimating the objective function coefficients. The SP introduced in Section 1 is guaranteed to fulfill the requirements stated in [4].
The most common approaches to solve convex SPs are Sample Average Approximation (SAA) methods, including: (i) the Multiple Replications Procedure and its derivatives [8, 3] which assess a solution's quality by comparing with sampled alternative solutions, (ii) Retrospective Approximation [11, 6] which iteratively improved the solution by lowering the error tolerance. Bastin, Cirillo, and Toint [2] extended the existing convergence guarantees from convex to non-convex SPs, showing almost sure convergence of the SAA problem's minimizers.
Domain Randomization. While the idea of randomizing the sensors and actuators dates back to at least 1995 [5], the systematic analysis of perturbed simulations in RL is a relatively new research direction [1, 9, 14, 12, 17, 13, 7, 15]. Regarding RL, recent domain randomization methods focus on perturbing the parameters defining the system dynamics. Approaches cover: (i) trajectory optimization on finite model-ensembles [9] (ii) learning a neural network policy for an under-actuated problem [1], (iii) using a risk-averse objective function [14], (iv) employing recurrent NN policies trained with experience replay [12], (v) optimizing a policy from samples of a model randomly chosen from an ensemble which is repeatedly fitted to world data [7].
Domain randomization is also applied in computer vision. One example is the work by Tobin et al. [17] in which an object detector for robot grasping is trained using multiple variants of the environment and applied to the real world. The approach presented by Pinto et al. [13] combines the concepts of randomized environments and actor-critic training, enabling the direct sim-to-real transfer of the abilities to pick, push, or move objects. Sadeghi and Levine [15] achieved the sim-to-real transfer by learning to fly a drone in visually randomized environments. The resulting deep neural network policy was able to map from monocular images to normalized 3D drone velocities.

5 Conclusion and Future Work

We presented a policy search meta-algorithm called Simulation-based Policy Optimization with Transferability Assessment (SPOTA) which is able to learn a control policy that directly transfers from a randomized source domain to an unseen target domain. The gist is to frame the training over an ensemble of models as a stochastic program and to use an upper confidence bound on the estimated optimality gap as stopping criterion for the training process. Furthermore, the resulting optimality gap can be interpreted as a measure of the obtained policy's robustness to variations of the source domain. To the best of our knowledge, SPOTA is the only domain randomization approach that provides this quantita-

tive measure for over-fitting to the domains experienced during the training phase. This measure is of high importance, since sample-based optimization is always optimistically biased [4, 8].
Our method as well as three baselines were evaluated on a simulated robotic balancing task in [10]. The results show that policies trained with SPOTA are able to generalize to unknown target domains, while baselines acquired without domain randomizations fail.

We are currently evaluating SPOTA on real-world devices, i.e., quantifying the resulting policy's transferability experimentally. In future work we will, we aim at answering the following questions: (i) how does the sampling strategy of the domains impact the learning procedure?, and (ii) what is the influence of the underlying policy optimization algorithm, i.e., do we need new algorithms tailored to the domain randomization setting?

References

[1] Rika Antonova and Silvia Cruciani. "Unlocking the Potential of Simulators: Design with RL in Mind." In: *ArXiv e-prints* (2017). URL: http://arxiv.org/abs/1706.02501.

[2] Fabian Bastin, Cinzia Cirillo, and Philippe L. Toint. "Convergence theory for nonconvex stochastic programming with an application to mixed logit." In: *Math. Program.* 108.2-3 (2006), pp. 207–234. DOI: 10.1007/s10107-006-0708-6. URL: https://doi.org/10.1007/s10107-006-0708-6.

[3] Güzin Bayraksan and David P. Morton. "Assessing solution quality in stochastic programs." In: *Math. Program.* 108.2-3 (2006), pp. 495–514. DOI: 10.1007/s10107-006-0720-x. URL: https://doi.org/10.1007/s10107-006-0720-x.

[4] Benjamin F. Hobbs and Ann Hepenstal. "Is optimization optimistically biased?" In: *Water Resources Research* 25.2 (1989), pp. 152–160.

[5] Nick Jakobi, Phil Husbands, and Inman Harvey. "Noise and the Reality Gap: The Use of Simulation in Evolutionary Robotics." In: *Advances in Artificial Life, Third European Conference on Artificial Life, Granada, Spain, June 4-6, 1995, Proceedings*. Springer, 1995, pp. 704–720. DOI: 10.1007/3-540-59496-5_337. URL: https://doi.org/10.1007/3-540-59496-5_337.

[6] Sujin Kim, Raghu Pasupathy, and Shane G. Henderson. "A guide to sample average approximation." In: *Handbook of Simulation Optimization*. Springer, 2015, pp. 207–243.

[7] Thanard Kurutach et al. "Model-Ensemble Trust-Region Policy Optimization." In: *ArXiv e-prints* (2018). URL: http://arxiv.org/abs/1802.10592.

[8] Wai-Kei Mak, David P. Morton, and R. Kevin Wood. "Monte Carlo bounding techniques for determining solution quality in stochastic programs." In: *Oper. Res. Lett.* 24.1-2 (1999), pp. 47–56. DOI: 10.1016/S0167-6377(98)00054-6. URL: https://doi.org/10.1016/S0167-6377(98)00054-6.

[9] Igor Mordatch, Kendall Lowrey, and Emanuel Todorov. "Ensemble-CIO: Full-body dynamic motion planning that transfers to physical humanoids." In: *IROS, Hamburg, Germany, September 28 - October 2*. 2015, pp. 5307–5314. DOI: 10.1109/IROS.2015.7354126. URL: https://doi.org/10.1109/IROS.2015.7354126.

[10] Fabio Muratore et al. "Domain Randomization for Simulation-Based Policy Optimization with Transferability Assessment." In: *2nd Annual Conference on Robot Learning, CoRL 2018, Zürich, Switzerland, 29-31 October 2018, Proceedings*. 2018, pp. 700–713. URL: http://proceedings.mlr.press/v87/muratore18a.html.

[11] Raghu Pasupathy and Bruce W. Schmeiser. "Retrospective-Approximation Algorithms for the Multidimensional Stochastic Root-Finding Problem." In: *ACM Trans. Model. Comput. Simul.* 19.2 (2009), 5:1–5:36. DOI: 10.1145/1502787.1502788. URL: http://doi.acm.org/10.1145/1502787.1502788.

[12] Xue Bin Peng et al. "Sim-to-Real Transfer of Robotic Control with Dynamics Randomization." In: *ArXiv e-prints* (2017). URL: http://arxiv.org/abs/1710.06537.

[13] Lerrel Pinto et al. "Asymmetric Actor Critic for Image-Based Robot Learning." In: *ArXiv e-prints* (2017). URL: http://arxiv.org/abs/1710.06542.

[14] Aravind Rajeswaran et al. "EPOpt: Learning Robust Neural Network Policies Using Model Ensembles." In: *ArXiv e-prints* 1610.01283 (2016). URL: http://arxiv.org/abs/1610.01283.

[15] Fereshteh Sadeghi and Sergey Levine. "CAD2RL: Real Single-Image Flight Without a Single Real Image." In: *RSS XIII, Massachusetts Institute of Technology, Cambridge, Massachusetts, USA, July 12-16*. 2017. URL: http://www.roboticsproceedings.org/rss13/p34.html.

[16] John Schulman et al. "Trust Region Policy Optimization." In: *ICML, Lille, France, July 6-11*. Vol. 37. 2015, pp. 1889–1897. URL: http://jmlr.org/proceedings/papers/v37/schulman15.html.

[17] Josh Tobin et al. "Domain randomization for transferring deep neural networks from simulation to the real world." In: *IROS, Vancouver, BC, Canada, September 24-28*. 2017, pp. 23–30. DOI: 10.1109/IROS.2017.8202133. URL: https://doi.org/10.1109/IROS.2017.8202133.