Evaluating Planning for Policy Search

Jakob J. Hollenstein
Dept. of Computer Science
University of Innsbruck
Innsbruck, Austria
jakob.hollenstein@uibk.ac.at

Justus Piater
Dept. of Computer Science
University of Innsbruck
Innsbruck, Austria
justus.piater@uibk.ac.at

***Abstract*—Training using randomized simulations can be seen as a model-based reinforcement learning method. By providing a model, this training blends planning and reinforcement learning.**

We propose the use of kinodynamic planning methods as part of a domain-randomized, model-based reinforcement learning method and to learn in an off-policy fashion from solved planning instances.

In order for this to be an improvement, using the planning method needs to have beneficial properties over pure reinforcement learning. On a simple toy domain, we show an improved state space coverage over PPO, while still also finding trajectories with good returns.

***Index Terms*—kinodynamic planning, domain randomization, reinforcement learning**

I. Introduction

Human environments are less structured, more varied and more quickly changing than typical automated manufacturing environments. This presents challenges for robots and makes learning robots an interesting and promising direction.

Learning and trying out actions directly on a real robot is time-consuming and potentially dangerous to the environment as well as to the robot. Physically-based simulation potentially provides the benefit of faster, cheaper and safer ways for robot learning. However, simulation has to model physical phenomena which often requires approximations. Therefore the experience in simulation and the real world will differ. This difference is called the *reality gap*.

Domain randomization[16, 12, 4] is the concept of bridging this gap, by using a distribution of simulation models with varying properties. The idea is to make policies robust to the differences within this distribution, thereby increasing robustness against the difference between the training distribution and the target domain.

II. Method

In preliminary experiments, not included in this paper, performed on a simulated 7-DoF robot arm for a pushing (non-prehensile manipulation) task, we found that D-RL algorithms (DDPG[6], PPO[13], from [1]) were not able to learn the given tasks. The algorithms were also not reaching relevant parts of the state space. Plappert et al.[11] report similar problems on the *HalfCheetah* environment where the algorithms converge to a local optimum corresponding to the cheetah wiggling on its back. They alleviated this problem by a different exploration scheme.

The research leading to these results has received funding from the European Unions Horizon 2020 research and innovation programme under grant agreement no. 731761, IMAGINE.

Consequently we assume that part of the problem of failing to learn the tasks is related to insufficient exploration. Two potential remedies could be taking larger samples while keeping the exploration probability high, or using a more principled exploration approach.

While domain randomization has shown promising results, as shown by OpenAI[9], results such as these usually require an even larger number of simulation samples. If we could reduce the number of samples needed to reach relevant state space areas, this reduction should also help when domain-randomized training is used.

A sample-efficient type of model-based deep reinforcement learning is guided policy search[5], where rollouts from a deep neural network controller are optimised by an optimal control method such as the iLQR[17, 15] method. However, guided policy search is, depending on the task, usually initialized by demonstrations since the exploration capabilities of the underlying optimization method (iLQR) are limited. Furthermore, the optimization method requires an applicable cost function which is able to guide the search procedure towards relevant solutions.

Planning methods that seek to maximize state-space exploration, and thus are less dependant on shaped rewards, are for example rapidly exploring random trees. AQR-RRT[2] or LQR-RRT[10] are examples of RRT methods, which use a dynamics-based cost metric to guide the tree extension, making this methods able to deal with kinodynamic planning problems. If planning methods are able to reach sufficiently good solutions with fewer samples and explore the state space in a more directed way, then these aspects could help reduce the aforementioned problems in deep reinforcement learning and domain-randomized training.

The drawback of these methods are their requirements which make them unsuitable for direct application to the real world. For example, RRT requires restarting from already-visited points in state space, which is only possible by using a model, and even if a model is given, may take too much time to be applicable in the real world (although work is done to alleviate this problem, e.g. [18]).

We propose to take advantage of the benefits the aforementioned methods provide while tackling the problem of speed

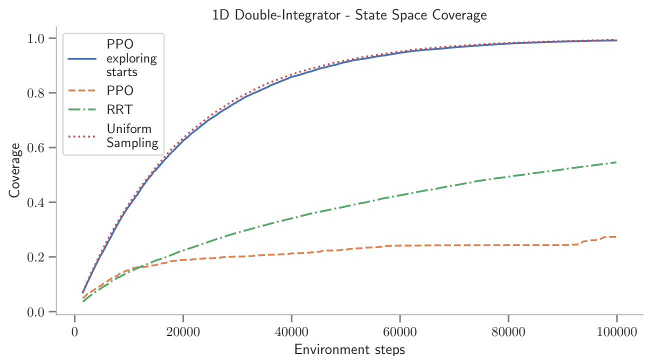

Fig. 1. Coverage of the state space: calculated as the percentage of non-empty uniformly shaped bins. The number of bins is equal along each state space dimension and is set to $\left(\frac{10^5}{5}\right)^{\frac{1}{2}}$, i.e. such that, in the uniform case, we expect five data points in each bin on average. The "uniform sampling" curve is included to ease interpretation and shows the coverage function for samples drawn uniformly from the state space.

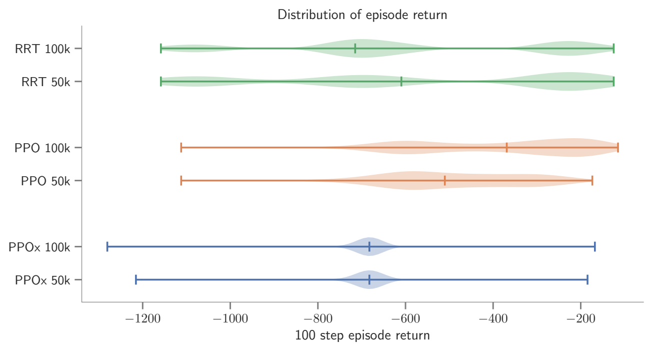

Fig. 2. Distribution of achieved returns of 100 steps long trajectories: data are collected over 100k (respectively 50k) environment steps. For RRT these are the data stored in its tree. For PPO these data constitute the union of the rollouts of the exploring PPO policies generated during their learning process. PPOx denotes PPO with exploring starts. Note that the results for "50k" are a subset of the results for "100k" steps.

by synthesizing the planning results into a policy. The use of planning methods, such as LQR-RRT, is possible in domain-randomized training, since, by the definition of domain randomization, simulation models are available. This essentially makes our method a model-based method[14] which employs the domain-randomized simulation as a model. We will refer to this method as *planning for policy search*.

In order for this approach to be meaningful, the exploration and data generation properties have to compare favorably to those of methods such as PPO which do not use models and directed exploration.

III. Experiments

In this experiment we compare the data generation properties of PPO[13, 3] against a kinodynamic search variant of RRT based on the LQR distance metric proposed by[10]. For simplicity this evaluation is performed on a 1D double-integrator environment. The environment's goal is to maneuver a point mass to a goal position by applying a (limited) force. The reward is based on the squared distance to a goal point position. Both algorithms are run for 10^5 environment steps.

Figure 1 shows the coverage of the visited state space. Note how the RRT algorithm keeps increasing the state space coverage while the PPO agent levels out. In the beginning the PPO agent covers slightly more space than the RRT method, we attribute this to properties of our RRT method which does not yet use minimal length trajectories to reach to the RRT-extension point, but a fixed-length trajectory controlled by an infinite horizon LQR controller. When exploring starts are used for the PPO agent (PPOx), the state space coverage of PPOx follows the maximum expected coverage (i.e. of uniform sampling) very closely.

Figure 2 shows the distribution of achieved returns of 100 steps long trajectories. Note that while the median return is not high, the RRT method still finds good trajectories in the -200 return area. This is an important indication that the planning method is able to reach into high reward areas even though it maximizes exploration. This compares favorably to the returns of PPOx which covers the search space in an undirected way and thus fails to find trajectories with good returns and subsequently is unable to learn a successful policy. The data generated by PPO shows a shift (50k to 100k) in the weight towards higher return trajectories, this indicates the successful learning of the PPO policy.

IV. Discussion

We conclude from this preliminary data, that RRT improves state space coverage, while also generating more relevant data than PPO with the exploring-starts setting. This presumably is because RRT explores the state space from the initial point also used for evaluation – a setting which is more similar to typical robotic manipulation setups.

The environment used is deterministic and not yet randomized. Possible ways to includes randomization might be to treat each instance of the domain-randomized model as a separate model instance and combine the data gathered from these separate models, or alternatively by using a probabilistic RRT variant (e.g. particle RRT[7]).

While the solutions found by RRT in this experiment still achieve good returns, the solutions are worse than the solutions found by PPO (as expected). A potential solution to this problem [8] uses model-based RL and then perform model-free fine tuning to improve the policies.

Our next step is to learn a policy from the data generated by the planning method.

References

[1] Prafulla Dhariwal et al. "OpenAI Baselines". In: *GitHub repository* (2017). URL: https://github.com/openai/baselines.

[2] Elena Glassman and Russ Tedrake. "A Quadratic Regulator-Based Heuristic for Rapidly Exploring State Space". In: *2010 IEEE International Conference on Robotics and Automation*. IEEE, 2010, pp. 5021–5028.

[3] Ashley Hill et al. "Stable Baselines". In: *GitHub repository* (2018). URL: https://github.com/hill-a/stable-baselines.

[4] Stephen James, Andrew J. Davison, and Edward Johns. "Transferring End-to-End Visuomotor Control from Simulation to Real World for a Multi-Stage Task". In: *CoRR* abs/1707.02267 (2017). URL: http://arxiv.org/abs/1707.02267.

[5] Sergey Levine and Vladlen Koltun. "Guided Policy Search". In: *Proceedings of the 30th International Conference on Machine Learning (ICML-13)*. 2013, pp. 1–9. URL: http://www.jmlr.org/proceedings/papers/v28/levine13.pdf (visited on 06/27/2017).

[6] Timothy P. Lillicrap et al. "Continuous Control with Deep Reinforcement Learning". In: (Sept. 9, 2015). arXiv: 1509.02971 [cs, stat]. URL: http://arxiv.org/abs/1509.02971 (visited on 06/04/2018).

[7] N. A. Melchior and R. Simmons. "Particle RRT for Path Planning with Uncertainty". In: *Proceedings 2007 IEEE International Conference on Robotics and Automation*. Proceedings 2007 IEEE International Conference on Robotics and Automation. Apr. 2007, pp. 1617–1624. DOI: 10.1109/ROBOT.2007.363555.

[8] A. Nagabandi et al. "Neural Network Dynamics for Model-Based Deep Reinforcement Learning with Model-Free Fine-Tuning". In: *2018 IEEE International Conference on Robotics and Automation (ICRA)*. May 2018, pp. 7559–7566. DOI: 10.1109/ICRA.2018.8463189.

[9] OpenAI. "Learning Dexterous In-Hand Manipulation". In: (Aug. 1, 2018). arXiv: 1808.00177 [cs, stat]. URL: http://arxiv.org/abs/1808.00177 (visited on 08/06/2018).

[10] A. Perez et al. "LQR-RRT*: Optimal Sampling-Based Motion Planning with Automatically Derived Extension Heuristics". In: *2012 IEEE International Conference on Robotics and Automation*. 2012 IEEE International Conference on Robotics and Automation. May 2012, pp. 2537–2542. DOI: 10.1109/ICRA.2012.6225177.

[11] Matthias Plappert et al. "Parameter Space Noise for Exploration". In: (June 6, 2017). arXiv: 1706.01905 [cs, stat]. URL: http://arxiv.org/abs/1706.01905 (visited on 06/05/2018).

[12] Fereshteh Sadeghi and Sergey Levine. "CAD2RL: Real Single-Image Flight Without a Single Real Image". In: *Robotics: Science and Systems*. 2017.

[13] John Schulman et al. "Proximal Policy Optimization Algorithms". In: *CoRR* abs/1707.06347 (2017).

[14] Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. 2*. Vol. 1. 1. MIT press Cambridge, 2016. URL: http://www.cell.com/trends/cognitive-sciences/pdf/S1364-6613(99)01331-5.pdf (visited on 07/03/2017).

[15] Yuval Tassa, Tom Erez, and Emanuel Todorov. "Synthesis and Stabilization of Complex Behaviors through Online Trajectory Optimization". In: *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference On*. IEEE, 2012, pp. 4906–4913.

[16] Josh Tobin et al. "Domain Randomization for Transferring Deep Neural Networks from Simulation to the Real World". In: *Intelligent Robots and Systems (IROS), 2017 IEEE/RSJ International Conference On*. IEEE, 2017, pp. 23–30.

[17] E. Todorov and Weiwei Li. "A Generalized Iterative LQG Method for Locally-Optimal Feedback Control of Constrained Nonlinear Stochastic Systems". In: *Proceedings of the 2005, American Control Conference, 2005*. Proceedings of the 2005, American Control Conference, 2005. Portland, OR, USA: IEEE, 2005, pp. 300–306. ISBN: 978-0-7803-9098-0. DOI: 10.1109/ACC.2005.1469949. URL: http://ieeexplore.ieee.org/document/1469949/ (visited on 05/22/2019).

[18] W. J. Wolfslag et al. "RRT-CoLearn: Towards Kinodynamic Planning Without Numerical Trajectory Optimization". In: *IEEE Robotics and Automation Letters* 3.3 (July 2018), pp. 1655–1662. ISSN: 2377-3766. DOI: 10.1109/LRA.2018.2801470.