CuNAS - CUriosity-driven Neural-Augmented Simulator

Sharath Chandra Raparthy*†§, Melissa Mozifian*‡, Liam Paull*†, and Florian Golemo*†
*University of Montreal, DIRO
†Mila - The Quebec AI Institute
‡McGill University
§Corresponding author: sharathraparthy@gmail.com

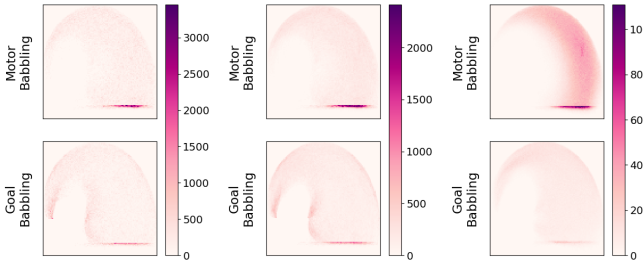
***Abstract*—Transfer of policies from simulation to physical robots is an important open problem in deep reinforcement learning. Prior work has introduced the model-based Neural-Augmented Simulator (NAS) method, which uses task-independent data to create a model of the differences between simulated and real robot. In this work, we show that this method is sensitive to the sampling of motor actions and the control frequency. To overcome this problem, we propose a simple extension based on artificial curiosity. We demonstrate on a physical robot, that this leads to a better exploration of the state space and consequently better transfer performance when compared to the NAS baseline.**

***Index Terms*—Sim2Real, Domain Adaptation, Intrinsic Motivation**

I. Introduction

Despite the success in Deep Reinforcement Learning (RL), training policies directly on physical robots remains a difficult problem due to concerns about amount of human intervention required and potential damage to the robot and environment. Since simulators offer a test bed for training and evaluating policies, training the policies in a simulated environment and then deploying them on the real robot is common practice. However, due to various factors (noise, modelling errors, unmodelled effects, etc.), there always exists a gap between simulation and reality. To close this ”sim2real gap”, two families of methods are usually employed: Domain Adaptation/Randomization (e.g. [12, 7, 10, 8]) and Model-based methods.

In the model-based family of methods, commonly a model of the discrepancies between simulation and reality is learned and then used to adapt the simulation or policy. [1], for example, proposed a hybrid dynamics model, Simulation-Augmented Interaction Networks (SAIN), where the authors incorporated Interaction Networks [2] into a physics engine for solving real world complex robotics control tasks. [4] uses a Recurrent Neural Network (RNN) to learn a forward dynamics model by predicting the state offset between simulation and real world trajectories. This discrepancy measure is used as a correction term to estimate the current state in the real environment given the simulation state. However, many existing model-based sim2real methods do not consider the state-space coverage during the data collection phase. Assuring a wide range of states covered in these model-based methods allows for a wider range of policies to be transferred with accuracy.



(a) 1Hz (b) 2 Hz (c) 10 Hz

Fig. 1: Heatmaps of end-effector positons comparing motor babbling and goal babbling under different control frequencies. Under lower frequencies, both methods show a low absolute coverage of the state space, but in all cases, the goal babbling method leads to a wider relative coverage.

In this work, (a) we show how transfer performance is affected by state space coverage and (b) in order to reach better state space coverage, we combine an off-the-shelf intrinsic motivation method, goal babbling [9], with a state-of-the-art model-based sim2real method NAS and demonstrate its increased performance on a real robot transfer task.

II. Method

The Neural-Augmented Simulator (NAS) method works in two phases:

- **Model Learning Phase:** During this phase, it learns the forward dynamics model between simulation and the real world from matching trajectories by predicting the discrepancies between them using an LSTM [5]
- **Policy Learning Phase:** This learned LSTM model is applied at every simulation step to update the simulated state. Using this augmented simulator, a policy can be learned using an Reinforcement Learning (RL) algorithm.

NAS gathers the trajectories during the learning phase through random movements (i.e. under a random policy). This can be seen as a naive exploration strategy and this is known as ”Motor Babbling (MB)”, where the commands that are sent to the robot are the randomization objective. Benureau [3] has demonstrated that this exploration mechanism does not lead to a uniform state space exploration and hence limits the robustness of a forward model. The authors suggested a

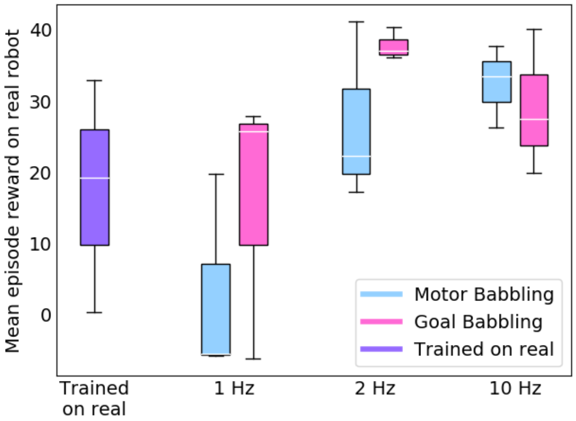


Fig. 2: **Real robot evaluation results** comparing motor babbling and goal babbling performance after transfer. Higher is better. Across different control frequencies, goal babbling outperforms motor babbling and the baseline in terms of median. The median difference in the 10Hz case is not statistically significant.

better, curiosity-inspired exploration strategy, Goal Babbling (GB) [13]. This method works by randomizing over goals (e.g. end effector positions) instead of motor commands. This can be achieved through an inverse model but since this is not available for every robot, a simple nearest-neighbor lookup table that is extended with each rollout is a suitable replacement[1].

As extension of NAS, we propose the **CUriosity-driven Neural-Augmented Simulator** (CuNAS) which uses the *Goal Babbling* exploration strategy for collecting trajectories during the model learning phase. We use the non-parametric Nearest Neighbors (NN) algorithm for GB to find the action that came closest to the goal given the history of past observations. A small amount of noise is drawn from a uniform distribution and added to the best action while ensuring it does not exceed the motor ranges. During the initial exploratory phase, the history ob observations is bootstrapped with MB.The model is otherwise trained identically to the LSTM-based model described in [4].

III. Experiments

We evaluated our framework in *ErgoReacher* environment [4], a 4-DOF robotic arm simulated in the Bullet Physics Engine, using the Poppy ErgoJr robot arm[6]. The end-effector of the arm has to reach multiple goals sampled along the plane perpendicular to the axis of rotation. For the **model learning phase** of NAS, we collected 10K real trajectories and simulated them out according to NAS. The list of actions was obtained either from MB or GB policies. We collected these trajectories at three different control frequencies, 1Hz, 2Hz, and 10Hz, i.e. a new action command is sent once a second, twice a second or ten times a second and in between, the robot has time to move its joints via a PID controller to reach the angles specified in the control commands[2]. In the **policy learning phase**, we augmented the PyBullet simulation with the learned LSTM and used PPO [11] to train policies in the source environment. We trained 3 policies with different seeds for each LSTM (i.e. a total of 18 policies, 3 for each one of the 3 control frequency and MB/GB cases). We rolled out these simulation-trained policies on the real robot and evaluated GB against the MB baseline and against a set of 3 policies that were trained directly on the real robot. For each policy, we rolled out 25 episodes. In Fig. 2, both the individual episode results are plotted as well as the mean for each condition. We found that GB-based methods consistently outperformed both the baseline that was trained directly on the target environment[3] and the MB-based methods. We also found evidence that the transferred performance is sensitive to the frequency at which the data is collected.

IV. Conclusions

We introduced an easy yet powerful extension of the Neural-Augmented Simulator. With this extension we were able to achieve significantly better results on the transfer task with no overhead in terms of physical experiments and next to no overhead in terms of total experiment runtime. As such, we recommend CuNAS as drop-in replacement for NAS. We recognize that even though experimenting on even a single physical robot is difficult and time-consuming, we have yet to prove the generalization of this method to other robotic platforms and tasks. We hope to be able to address this when we can return to our lab after the current pandemic. Furthermore, we took goal babbling as an assumption when designing this method for it's simplicity and low overhead. In future work, we would like to evaluate different alternative exploration strategies as well. Additionally, in upcoming work, we would like to investigate a metric that indicates pre-transfer the likelihood of successful sim2real transfer, based on the coverage of the state space of the robot during the exploration and model learning phase and with respect to much of that state space is visited by the target policy.

Acknowledgments

This work was funded by CIFAR. Thanks to the IDT team at Mila for giving the robots a place to live.

[1] As demonstrated here: http://fabien.benureau.com/recode/benureau2015_gb/benureau2015_gb.html

[2] Internally, the simulated and real robot operate at 100Hz and the action is simply repeated between new action commands.

[3] This is an artifact (also discussed in [4]) of how, when training directly on the real robot, the robot sometimes gets stuck on the table or environment and therefore the training is significantly harder than in simulation where the reset function is reliable.

References

[1] Anurag Ajay, Maria Bauzá, Jiajun Wu, Nima Fazeli, Joshua B. Tenenbaum, Alberto Rodriguez, and Leslie Pack Kaelbling. Combining physical simulators and object-based networks for control. *CoRR*, abs/1904.06580, 2019. URL http://arxiv.org/abs/1904.06580.

[2] Peter W. Battaglia, Razvan Pascanu, Matthew Lai, Danilo Jimenez Rezende, and Koray Kavukcuoglu. Interaction networks for learning about objects, relations and physics. *CoRR*, abs/1612.00222, 2016. URL http://arxiv.org/abs/1612.00222.

[3] Fabien Benureau. *Self-Exploration of Sensorimotor Spaces in Robots*. PhD thesis, Universit 'e de Bordeaux, May 2015.

[4] Florian Golemo, Adrien Ali Taiga, Aaron Courville, and Pierre-Yves Oudeyer. Sim-to-real transfer with neural-augmented robot simulation. In Aude Billard, Anca Dragan, Jan Peters, and Jun Morimoto, editors, *Proceedings of The 2nd Conference on Robot Learning*, volume 87 of *Proceedings of Machine Learning Research*, pages 817–828. PMLR, 29–31 Oct 2018. URL http://proceedings.mlr.press/v87/golemo18a.html.

[5] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Comput.*, 9(8):1735–1780, November 1997. ISSN 0899-7667. doi: 10.1162/neco.1997.9.8.1735. URL https://doi.org/10.1162/neco.1997.9.8.1735.

[6] Matthieu Lapeyre, Pierre Rouanet, Jonathan Grizou, Steve Nguyen, Fabien Depraetre, Alexandre Le Falher, and Pierre-Yves Oudeyer. Poppy Project: Open-Source Fabrication of 3D Printed Humanoid Robot for Science, Education and Art. In *Digital Intelligence 2014*, page 6, Nantes, France, September 2014. URL https://hal.inria.fr/hal-01096338.

[7] Bhairav Mehta, Manfred Diaz, Florian Golemo, Christopher J. Pal, and Liam Paull. Active domain randomization. *CoRR*, abs/1904.04762, 2019. URL http://arxiv.org/abs/1904.04762.

[8] Melissa Mozifian, Juan Camilo Gamboa Higuera, David Meger, and Gregory Dudek. Learning domain randomization distributions for transfer of locomotion policies. *CoRR*, abs/1906.00410, 2019. URL http://arxiv.org/abs/1906.00410.

[9] Pierre-Yves Oudeyer and Frédéric Kaplan. What is intrinsic motivation? a typology of computational approaches. *Frontiers in Neurorobotics*, 1, 2007.

[10] Xue Bin Peng, Marcin Andrychowicz, Wojciech Zaremba, and Pieter Abbeel. Sim-to-real transfer of robotic control with dynamics randomization. *CoRR*, abs/1710.06537, 2017. URL http://arxiv.org/abs/1710.06537.

[11] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *CoRR*, abs/1707.06347, 2017. URL http://arxiv.org/abs/1707.06347.

[12] Joshua Tobin, Rachel Fong, Alex Ray, Jonas Schneider, Wojciech Zaremba, and Pieter Abbeel. Domain randomization for transferring deep neural networks from simulation to the real world. *CoRR*, abs/1703.06907, 2017. URL http://arxiv.org/abs/1703.06907.

[13] E Ugur, Y Nagai, E Oztop, and M Asada. Goal babbling: a new concept for early sensorimotor exploration. 2012.