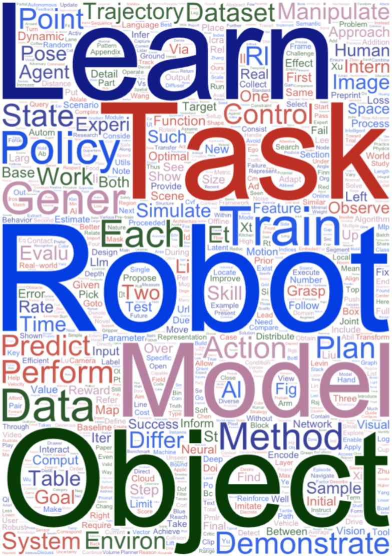


Robot Learning

RL II: Offline RL & Sim2Real

Marc Toussaint
Technical University of Berlin
Summer 2024

Outline

- Some RL application papers
- Offline RL (on-policy vs. off-policy)
- Sim2Real
  - Domain Randomization
  - Privileged Training & Imitation Learning
  - Domain Adaptation

Outline

- **Some RL application papers**
- Offline RL (on-policy vs. off-policy)
- Sim2Real
  - Domain Randomization
  - Privileged Training & Imitation Learning
  - Domain Adaptation

The International Journal of
Robotics Research
29(13) 1608–1639

DOI: 10.1177/0278364910371999
ijr.sagepub.com
SAGE

Autonomous Helicopter Aerobatics through Apprenticeship Learning

Pieter Abbeel[1], Adam Coates[2] and Andrew Y. Ng[2]

Abstract

Autonomous helicopter flight is widely regarded to be a highly challenging control problem. Despite this fact, human experts can reliably fly helicopters through a wide range of maneuvers, including aerobatic maneuvers at the edge of the helicopter's capabilities. We present apprenticeship learning algorithms, which leverage expert demonstrations to efficiently learn good controllers for tasks being demonstrated by an expert. These apprenticeship learning algorithms have enabled us to significantly extend the state of the art in autonomous helicopter aerobatics. Our experimental results include the first autonomous execution of a wide range of maneuvers, including but not limited to in-place flips, in-place rolls, loops and hurricanes, and even auto-rotation landings, chaos and tic-tocs, which only exceptional human pilots can perform. Our results also include complete airshows, which require autonomous transitions between many of these maneuvers. Our controllers perform as well as, and often even better than, our expert pilot.

P. Abbeel, A. Coates, and A. Y. Ng. Autonomous Helicopter Aerobatics through Apprenticeship Learning.
The International Journal of Robotics Research, 29(13):1608–1639, 2010-11-01.
URL: `https://delete_doi.org/10.1177/0278364910371999`

`http://heli.stanford.edu/`

Article

Outracing champion Gran Turismo drivers with deep reinforcement learning

https://doi.org/10.1038/s41586-021-04357-7

Received: 9 August 2021

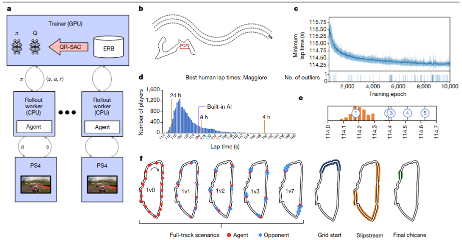
Accepted: 15 December 2021

Published online: 9 February 2022

Check for updates

Peter R. Wurman[1✉], Samuel Barrett[1], Kenta Kawamoto[2], James MacGlashan[1], Kaushik Subramanian[3], Thomas J. Walsh[1], Roberto Capobianco[3], Alisa Devlic[3], Franziska Eckert[3], Florian Fuchs[3], Leilani Gilpin[1], Piyush Khandelwal[1], Varun Kompella[1], HaoChih Lin[3], Patrick MacAlpine[1], Declan Oller[1], Takuma Seno[2], Craig Sherstan[1], Michael D. Thomure[1], Houmehr Aghabozorgi[1], Leon Barrett[1], Rory Douglas[1], Dion Whitehead[1], Peter Dürr[3], Peter Stone[1], Michael Spranger[2] & Hiroaki Kitano[2]

Many potential applications of artificial intelligence involve making real-time decisions in physical systems while interacting with humans. Automobile racing represents an extreme example of these conditions; drivers must execute complex tactical manoeuvres to pass or block opponents while operating their vehicles at their traction limits[1]. Racing simulations, such as the PlayStation game Gran Turismo, faithfully reproduce the non-linear control challenges of real race cars while also encapsulating the complex multi-agent interactions. Here we describe how we trained agents for Gran Turismo that can compete with the world's best e-sports drivers. We combine state-of-the-art, model-free, deep reinforcement learning algorithms with mixed-scenario training to learn an integrated control policy that combines exceptional speed with impressive tactics. In addition, we construct a reward function that enables the agent to be competitive while adhering to racing's important, but under-specified, sportsmanship rules. We demonstrate the capabilities of our agent, Gran Turismo Sophy, by winning a head-to-head competition against four of the world's best Gran Turismo drivers. By describing how we trained championship-level racers, we demonstrate the possibilities and challenges of using these techniques to control complex dynamical systems in domains where agents must respect imprecisely defined human norms.



https://sonyresearch.github.io/gt_sophy_public/

P. R. Wurman, S. Barrett, K. Kawamoto, J. MacGlashan, K. Subramanian, T. J. Walsh, R. Capobianco, A. Devlic, F. Eckert, F. Fuchs, L. Gilpin, P. Khandelwal, V. Kompella, H. Lin, P. MacAlpine, D. Oller, T. Seno, C. Sherstan, M. D. Thomure, H. Aghabozorgi, L. Barrett, R. Douglas, D. Whitehead, P. Dürr, P. Stone, M. Spranger, and H. Kitano.

Outracing champion Gran Turismo drivers with deep reinforcement learning. *Nature*, 602(7896):223–228, 2022-02. URL: https://www.nature.com/articles/s41586-021-04357-7

Article

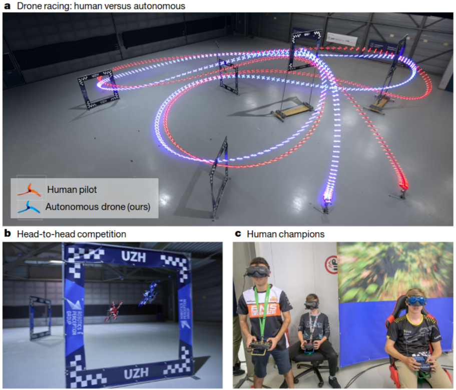
Champion-level drone racing using deep reinforcement learning

https://doi.org/10.1038/s41586-023-06419-4
Received: 5 January 2023
Accepted: 10 July 2023
Published online: 30 August 2023
Open access
Check for updates

Elia Kaufmann[1✉], Leonard Bauersfeld[1], Antonio Loquercio[1], Matthias Müller[2], Vladlen Koltun[3] & Davide Scaramuzza[1]

First-person view (FPV) drone racing is a televised sport in which professional competitors pilot high-speed aircraft through a 3D circuit. Each pilot sees the environment from the perspective of their drone by means of video streamed from an onboard camera. Reaching the level of professional pilots with an autonomous drone is challenging because the robot needs to fly at its physical limits while estimating its speed and location in the circuit exclusively from onboard sensors[1]. Here we introduce Swift, an autonomous system that can race physical vehicles at the level of the human world champions. The system combines deep reinforcement learning (RL) in simulation with data collected in the physical world. Swift competed against three human champions, including the world champions of two international leagues, in real-world head-to-head races. Swift won several races against each of the human champions and demonstrated the fastest recorded race time. This work represents a milestone for mobile robotics and machine intelligence[2], which may inspire the deployment of hybrid learning-based solutions in other physical systems.

E. Kaufmann, L. Bauersfeld, A. Loquercio, M. Müller, V. Koltun, and D. Scaramuzza.
Champion-level drone racing using deep reinforcement learning.
Nature, 620(7976):982–987, 2023-08.
URL: `https://www.nature.com/articles/s41586-023-06419-4`



`https://www.youtube.com/watch?v=fBiataDpGIo`

Outline

- Some RL application papers
- **Offline RL (on-policy vs. off-policy)**
- Sim2Real
  - Domain Randomization
  - Privileged Training & Imitation Learning
  - Domain Adaptation

On-Policy vs. Off-Policy Methods

- **On-policy:** estimate V^π or Q^π while executing π (e.g., Policy Evaluation)
  - The value-function updates directly depend on the policy π
- **Off-policy:** estimate Q^* while executing π (e.g., Q-learning)
  - The actually executed (data-collecting) policy π is also called "behavioral policy"
  - In contrast, values Q^* are estimated for the optimal policy π^*

On-Policy vs. Off-Policy Methods

- **On-policy:** estimate V^π or Q^π while executing π (e.g., Policy Evaluation)
  - The value-function updates directly depend on the policy π
- **Off-policy:** estimate Q^* while executing π (e.g., Q-learning)
  - The actually executed (data-collecting) policy π is also called “behavioral policy”
  - In contrast, values Q^* are estimated for the optimal policy π^*
- Off-policy is considered more efficient, as it can use off-policy-distribution data

On-Policy vs. Off-Policy Methods

- **On-policy:** estimate V^π or Q^π while executing π (e.g., Policy Evaluation)
  - The value-function updates directly depend on the policy π
- **Off-policy:** estimate Q^* while executing π (e.g., Q-learning)
  - The actually executed (data-collecting) policy π is also called "behavioral policy"
  - In contrast, values Q^* are estimated for the optimal policy π^*
- Off-policy is considered more efficient, as it can use off-policy-distribution data

[More technically: Consider you have data $D = \{(s_i, a_i, r_i, s_{i+1}, a_{i+1})\}_{i=0}^n$ collected with behavior policy π. When you make Q- or V-updates, do you take only expectations w.r.t. D? Or do you take conditional expectations $a_{i+1} \sim \pi^*(a|s_{i+1})$ w.r.t. another policy? (E.g. greedy policy.)]

[SAC is called off-policy, because when training V it takes expectations w.r.t. $a_t \sim \pi_\theta$ (instead of w.r.t. data collected previously).]

Offline RL

- Motivation:
  - Separation of Concerns!
  - Separate thinking about Data Collection, and thinking about **what best to make of given data**
  - Real-world data is expensive!
  - Data collection (exploration) in RL is an issue anyway
  - No matter how RL collects data, it makes sense to study what best to make of given data

Offline RL

- Motivation:
  - Separation of Concerns!
  - Separate thinking about Data Collection, and thinking about **what best to make of given data**
  - Real-world data is expensive!
  - Data collection (exploration) in RL is an issue anyway
  - No matter how RL collects data, it makes sense to study what best to make of given data

  - **The data could come from anywhere:** huge data sets of other observed agents, of human behavior, perhaps extracted from abundant video
  - The data is not collected by "our AI agent" itself – but can still be used to learn a Q^*-function and train our agent for optimal behavior

Offline RL

- Naive problem formulation: Given data $D = \{(s_i, a_i, r_i, s_{i+1}\}_{i=0}^n$, find θ to

$$\min_\theta \quad \mathbb{E}_{(s,a,r,s')\sim D}\{\ [Q_\theta(s,a) - r - \gamma Q_{\bar\theta}(s', \pi(s'))]^2\ \}$$
$$\text{s.t.}\quad \bar\theta \approx \theta$$
$$\pi \approx \underset{\pi}{\text{argmax}}\, \mathbb{E}_{(s,a)\sim D}\{Q_\theta(s,a)\}$$

In words:

– minimize the empirical Bellman residual, with delayed $Q_{\bar\theta}$-target

– ...where eventually π becomes optimal and $\bar\theta$ converges

Offline RL

- Naive problem formulation: Given data $D = \{(s_i, a_i, r_i, s_{i+1}\}_{i=0}^n$, find θ to

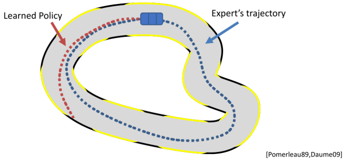
$$\min_\theta \quad \mathbb{E}_{(s,a,r,s')\sim D}\{\ [Q_\theta(s,a) - r - \gamma Q_{\bar\theta}(s', \pi(s'))]^2\ \}$$
$$\text{s.t. } \bar\theta \approx \theta$$
$$\pi \approx \operatorname*{argmax}_\pi \mathbb{E}_{(s,a)\sim D}\{Q_\theta(s,a)\}$$

  In words:
  - minimize the empirical Bellman residual, with delayed $Q_{\bar\theta}$-target
  - ...where eventually π becomes optimal and $\bar\theta$ converges

- That's a well-defined problem
  - We have gradients for everything: Bellman gradient, deterministic policy gradient – let's go!

Offline RL

- Resulting policy fails badly, due to distribution shift, just as in imitation learning:

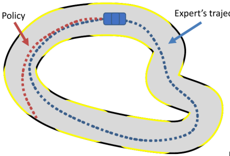


Also called **Compound Error** (Shi's lecture 5)

- In the naive problem formulation
  - there is no penalty for "dreaming" crazy Q-values outside the data distribution
  - the trained policy is likely to exploit these arbitrary Q-values

Offline RL

- Resulting policy fails badly, due to distribution shift, just as in imitation learning:

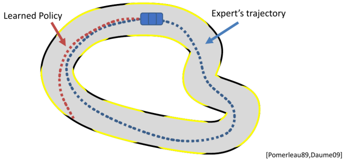


Also called **Compound Error** (Shi's lecture 5)

- In the naive problem formulation
  - there is no penalty for "dreaming" crazy Q-values outside the data distribution
  - the trained policy is likely to exploit these arbitrary Q-values
- We don't have the DAgger option: Can't collect more data to cover reached states!

Offline RL

- Resulting policy fails badly, due to distribution shift, just as in imitation learning:



Also called **Compound Error** (Shi's lecture 5)

- In the naive problem formulation
  - there is no penalty for "dreaming" crazy Q-values outside the data distribution
  - the trained policy is likely to exploit these arbitrary Q-values
- We don't have the DAgger option: Can't collect more data to cover reached states!

$\rightarrow$ We need to **add a penalty for leaving the data distribution**!

Offline RL

- We need to add a penalty for leaving the data distribution...
  - Many different ideas, incl. literally penalizing "distribution distance" (divergence regularization)
  - Modern versions found simple approaches:

TD3+BC

A Minimalist Approach to Offline Reinforcement Learning

Scott Fujimoto[1,2] **Shixiang Shane Gu**[2]
[1]Mila, McGill University
[2]Google Research, Brain Team
scott.fujimoto@mail.mcgill.ca

Abstract

Offline reinforcement learning (RL) defines the task of learning from a fixed batch of data. Due to errors in value estimation from out-of-distribution actions, most offline RL algorithms take the approach of constraining or regularizing the policy with the actions contained in the dataset. Built on pre-existing RL algorithms, modifications to make an RL algorithm work offline comes at the cost of additional complexity. Offline RL algorithms introduce new hyperparameters and often leverage secondary components such as generative models, while adjusting the underlying RL algorithm. In this paper we aim to make a deep RL algorithm work while making minimal changes. We find that we can match the performance of state-of-the-art offline RL algorithms by simply adding a behavior cloning term to the policy update of an online RL algorithm and normalizing the data. The resulting algorithm is a simple to implement and tune baseline, while more than halving the overall run time by removing the additional computational overheads of previous methods.

S. Fujimoto and S. S. Gu. A minimalist approach to offline reinforcement learning.
Advances in neural information processing systems, 34:20132–20145, 2021.
URL: https://proceedings.neurips.cc/paper_files/paper/2021/hash/a8166da05c5a094f7dc03724b41886e5-Abstract.html

- Use TD3 (twin delayed deep deterministic..)
- Simply add a BC term to the policy objective!

$$\pi \approx \operatorname*{argmax}_{\pi} \mathbb{E}_{(s,a)\sim D}\left\{\lambda Q_\theta(s,a) + (\pi(s) - a)^2\right\}$$

S4RL

S4RL: Surprisingly Simple Self-Supervision for Offline Reinforcement Learning in Robotics

Samarth Sinha[1,2*], **Ajay Mandlekar**[3], **Animesh Garg**[2,4]

[1] Facebook AI Research, [2]University of Toronto, Vector Institute, [3]Stanford University, [4]Nvidia

Abstract: Offline reinforcement learning proposes to learn policies from large collected datasets without interacting with the physical environment. These algorithms have made it possible to learn useful skills from data that can then be deployed in the environment in real-world settings where interactions may be costly or dangerous, such as autonomous driving or factories. However, offline agents are unable to access the environment to collect new data, and therefore are trained on a static dataset. In this paper, we study the effectiveness of performing data augmentations on the state space, and study 7 different augmentation schemes and how they behave with existing offline RL algorithms. We then combine the best data performing augmentation scheme with a state-of-the-art Q-learning technique, and improve the function approximation of the Q-networks by smoothening out the learned state-action space. We experimentally show that using this Surprisingly Simple Self-Supervision technique in RL (S4RL), we significantly improve over the current state-of-the-art algorithms on offline robot learning environments such as MetaWorld [1] and RoboSuite [2, 3], and benchmark datasets such as D4RL [4].

S. Sinha, A. Mandlekar, and A. Garg. S4rl: Surprisingly simple self-supervision for offline reinforcement learning in robotics.
In *Conference on Robot Learning*, pages 907–917, 2022.
URL: `https://proceedings.mlr.press/v164/sinha22a.html`

- Include a strong data augmentation in the Q-function loss

$$\min_\theta \mathbb{E}_{(s,a,r,s')\sim D}\left\{ \left[\frac{1}{I}\sum_i Q_\theta(\mathcal{T}_i(\tilde{s}|s),a) - r - \gamma\frac{1}{I}\sum_i Q_{\bar\theta}(\mathcal{T}_i(\tilde{s}'|s'),\pi(s'))\right]^2 \right\}$$

where $\mathcal{T}_i$ generates a variant of s (they propose 7 alternative, including spatial smoothing and adversarial)

Offline RL Application

Pre-Training for Robots: Offline RL Enables Learning New Tasks in a Handful of Trials

Aviral Kumar*[,1], Anikait Singh*[,1], Frederik Ebert*[,1], Mitsuhiko Nakamoto[1], Yanlai Yang[3], Chelsea Finn[2], Sergey Levine[1]
[1]UC Berkeley, [2]Stanford University, [3]New York University (*Equal contribution)

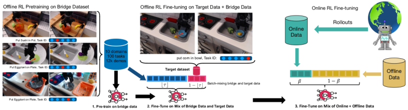
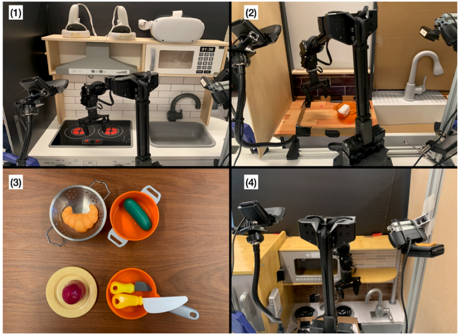


Fig. 1: **Overview of PTR:** We first perform general offline pre-training on diverse multi-task robot data and subsequently fine-tune on one or several target tasks while mixing batches between the prior data and the target dataset using a batch mixing ratio of τ. Additionally, a separate online fine-tuning phase can be done, where offline pre-training is done on a static dataset and an online replay buffer is collected using rollouts in the environment. The offline and online buffers are mixed per batch with a ratio of β.



A. Kumar, A. Singh, F. Ebert, M. Nakamoto, Y. Yang, C. Finn, and S. Levine. Pre-Training for Robots: Offline RL Enables Learning New Tasks from a Handful of Trials, 2023-09-23.
URL: http://arxiv.org/abs/2210.05178, arXiv:2210.05178

https://sites.google.com/view/ptr-final/

Offline RL Conclusions

- Scientifically important (separation of concerns)
- Opens new dimension: Train optimal behaviors from *any* data
- Promising future applications (leverage massive data, reward re-labelled data)

Outline

- Some RL application papers
- Offline RL (on-policy vs. off-policy)
- **Sim2Real** (slides based on Shi's lecture)
  - Domain Randomization
  - Privileged Training & Imitation Learning
  - Domain Adaptation

- Why train in Simulation?
  - Real-world data is expensive!
  - Many RL methods require millions of samples
  - Simulation is fast
  - Simulation is safe, can be fully explored
  - Simulation provides ground truth labels (e.g. train priviledged policy)
  - Simulations get better and better, including simulating sensors (image rendering)

Robot Simulators

- Simulator taxonomy by simulately.wiki

Simulator	Physics Engine	Rendering	Sensor😕	Dynamics	GPU-accelerated Simulation	Open-Source
IsaacSim	PhysX 5	Rasterization; RayTracing	RGBD; Lidar; Force; Effort; IMU; Contact; Proximity	Rigid;Soft;Cloth;Fluid	✓	✗
IsaacGym	PhysX 5, Flex	Rasterization;	RGBD; Force; Contact;	Rigid;Soft;Cloth	✓	✗
SAPIEN	PhysX 5, Warp	Rasterization; RayTracing⭐;	RGBD; Force; Contact;	Rigid;Soft;Fluid	✗	✓
Pybullet	Bullet	Rasterization;	RGBD; Force; IMU; Tactile;	Rigid;Soft;Cloth	✗	✓
MuJoCo	MuJoCo	Rasterization;	RGBD; Force; IMU; Tactile;	Rigid;Soft;Cloth	✓💡	✓
CoppeliaSim	MuJoCo; Bullet; ODE; Newton; Vortex	Rasterization; RayTracing🔶;	RGBD; Force; Contact;	Rigid;Soft;Cloth	✗	✓
Gazebo	Bullet; ODE; DART; Simbody	Rasterization;	RGBD; Lidar; Force; IMU;	Rigid;Soft;Cloth	✗	✓

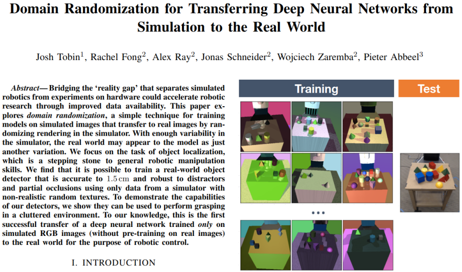
from Shi's lecture

- What are Sim2Real issues?
  - Simulation never matches real world exactly; policies overfit to simulation and fail in real
  - **Parameteric mismatches:** Other dynamics parameters, e.g. friction, inertias
  - **Non-parametric mismatches:** Physical effects not simulated: Wind, exact fluids, sand/dust

- What are Sim2Real issues?
  - Simulation never matches real world exactly; policies overfit to simulation and fail in real
  - **Parameteric mismatches:** Other dynamics parameters, e.g. friction, inertias
  - **Non-parameteric mismatches:** Physical effects not simulated: Wind, exact fluids, sand/dust

- Approaches to tackle this:
  - Domain Randomization
  - Privileged Training & Imitation Learning
  - Domain Adaptation

Domain Randomization



Domain Randomization for Transferring Deep Neural Networks from Simulation to the Real World

Josh Tobin[1], Rachel Fong[2], Alex Ray[2], Jonas Schneider[2], Wojciech Zaremba[2], Pieter Abbeel[3]

Abstract—Bridging the 'reality gap' that separates simulated robotics from experiments on hardware could accelerate robotic research through improved data availability. This paper explores *domain randomization*, a simple technique for training models on simulated images that transfer to real images by randomizing rendering in the simulator. With enough variability in the simulator, the real world may appear to the model as just another variation. We focus on the task of object localization, which is a stepping stone to general robotic manipulation skills. We find that it is possible to train a real-world object detector that is accurate to 1.5 cm and robust to distractors and partial occlusions using only data from a simulator with non-realistic random textures. To demonstrate the capabilities of our detectors, we show they can be used to perform grasping in a cluttered environment. To our knowledge, this is the first successful transfer of a deep neural network trained *only* on simulated RGB images (without pre-training on real images) to the real world for the purpose of robotic control.

I. INTRODUCTION

- Train a single policy to perform well in many domain variants
- Original paper focussed on perception, but works equally for any other parameter Θ

J. Tobin, R. Fong, A. Ray, J. Schneider, W. Zaremba, and P. Abbeel. Domain randomization for transferring deep neural networks from simulation to the real world. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 23–30, 2017.
URL: https://ieeexplore.ieee.org/abstract/document/8202133/

Domain Randomization

- Let Θ be a simulation parameter: $x_{t+1} = f(x_t, u_t; \Theta)$
- Randomly sample $\Theta \sim p(\Theta)$ at the start of each episode
- Otherwise, use standard RL
  - But since the world is "more uncertain", the RL problem becomes harder

- What if we train a policy $\hat{\pi}(s_t, \Theta)$ that get's Θ as input?

  Is that cheating? [2]

Privileged Training & Imitation Learning

Privileged Training & Imitation Learning

- Priviledged RL Training:
  - We first train $\hat{\pi}(s_t, \Theta)$ using standard RL
  - Much easier than without access to Θ

- Sensorimotor Imitation using DAgger:
  - Then we train a policy $\pi(s_t)$ to imitate $\hat{\pi}(s_t, \Theta)$
  - As we can query $\hat{\pi}(s_t, \Theta)$, we can use DAgger! Much more efficient than plain BC

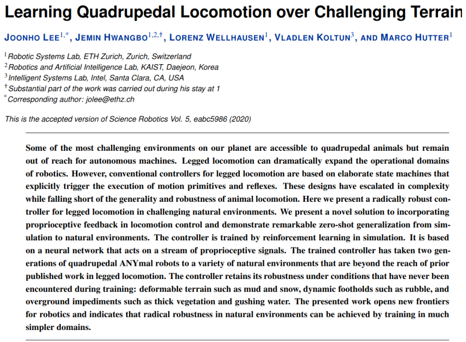
Privileged Training & Imitation Learning

- Priviledged RL Training:
  - We first train $\hat{\pi}(s_t, \Theta)$ using standard RL
  - Much easier than without access to Θ

- Sensorimotor Imitation using DAgger:
  - Then we train a policy $\pi(s_t)$ to imitate $\hat{\pi}(s_t, \Theta)$
  - As we can query $\hat{\pi}(s_t, \Theta)$, we can use DAgger! Much more efficient than plain BC

- This approach is a core paradigm beyond RL:
  - First develop a method to solve a problem using full information (could be a planner)
  - Then train a policy to imitate that method with only available (sensor) information

Privileged Training & Imitation Learning



Learning Quadrupedal Locomotion over Challenging Terrain

Joonho Lee[1,*], Jemin Hwangbo[1,2,†], Lorenz Wellhausen[1], Vladlen Koltun[3], and Marco Hutter[1]

[1] Robotic Systems Lab, ETH Zurich, Zurich, Switzerland
[2] Robotics and Artificial Intelligence Lab, KAIST, Daejeon, Korea
[3] Intelligent Systems Lab, Intel, Santa Clara, CA, USA
[†] Substantial part of the work was carried out during his stay at 1
[*] Corresponding author: jolee@ethz.ch

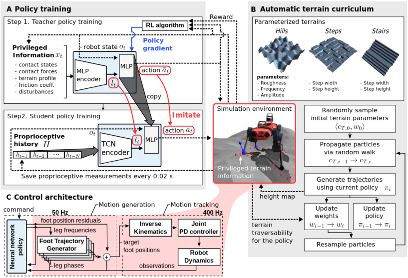
This is the accepted version of Science Robotics Vol. 5, eabc5986 (2020)

Some of the most challenging environments on our planet are accessible to quadrupedal animals but remain out of reach for autonomous machines. Legged locomotion can dramatically expand the operational domains of robotics. However, conventional controllers for legged locomotion are based on elaborate state machines that explicitly trigger the execution of motion primitives and reflexes. These designs have escalated in complexity while falling short of the generality and robustness of animal locomotion. Here we present a radically robust controller for legged locomotion in challenging natural environments. We present a novel solution to incorporating proprioceptive feedback in locomotion control and demonstrate remarkable zero-shot generalization from simulation to natural environments. The controller is trained by reinforcement learning in simulation. It is based on a neural network that acts on a stream of proprioceptive signals. The trained controller has taken two generations of quadrupedal ANYmal robots to a variety of natural environments that are beyond the reach of prior published work in legged locomotion. The controller retains its robustness under conditions that have never been encountered during training: deformable terrain such as mud and snow, dynamic footholds such as rubble, and overground impediments such as thick vegetation and gushing water. The presented work opens new frontiers for robotics and indicates that radical robustness in natural environments can be achieved by training in much simpler domains.

J. Lee, J. Hwangbo, L. Wellhausen, V. Koltun, and M. Hutter. Learning quadrupedal locomotion over challenging terrain.
Science Robotics, 5(47):eabc5986, 2020-10-21.
URL: `https://www.research-collection.ethz.ch/bitstream/handle/20.500.11850/448343/1/2020_science_robotics_lee_locomotion.pdf`

`https://youtu.be/txjqn8h6pjU`
`https://youtu.be/Xnn4sVSpSh0`

Privileged Training & Imitation Learning



- The privileged policy gets full information as input: Exact Θ and state s_t, including terrain model
- The sensorimotor policy only sensor obs. y_t $\to$ the sensorimotor policy needs to use the sequence $y_{0:t}$, e.g. recursive or transformer

- The sensorimotor policy uses full observation sequence $y_{0:t}$ to output controls u_t...
  - What else could it predict based on $y_{0:t}$?

- The sensorimotor policy uses full observation sequence $y_{0:t}$ to output controls u_t...
  - What else could it predict based on $y_{0:t}$?

The unobserved physics parameters Θ!

Adaptive Control

- Large area within Control Theory
- Assumes environment has *varying* parameters Θ (not directly observed)

Adaptive Control

- Large area within Control Theory
- Assumes environment has *varying* parameters Θ (not directly observed)

- One approach: Estimate Θ from past observations and use for control
- Robust control: Estimate posterior belief $p(\Theta|y_{0:T})$ over possible Θ and use control robust to all possibilities

Domain Adaptation

- In the Robot Learning community, the word *Domain Adaptation* is used for any controller that adapts (to varying unobserved Θ) based on past observations $y_{0:t}$.

Domain Adaptation

- In the Robot Learning community, the word *Domain Adaptation* is used for any controller that adapts (to varying unobserved Θ) based on past observations $y_{0:t}$.

- Explicit approach:
  - Train an estimator $\psi : y_{0:t} \mapsto \hat{\Theta}$
  - Then train a policy $\pi(y_{0:t}, \psi(y_{0:t}))$ for fixed ψ

- Implicit approach:
  - As in Lee et al'20
  - Just train $\pi(y_{0:t})$, but potentially imposing a representation that is also predictive for Θ

Sim2Real Conclusions

- (Pre-)Training in Sim became a standard in modern Robot Learning
- Sim2Real is not considered a blocker anymore:
  - Domain Randomization, Privileged Training & Sensorimotor are powerful approaches
  - Even if policies do not directly transfer $\rightarrow$ Real-World finetuning requires much less data

Side note: Privileged Training for Imitation Learning

- The paper below used same approach, but in the context of Imitation Learning:
  - The privileged policy imitated a human demonstrator using full access to the driving simulation
  - The sensorimotor policy imitated the privileged policy

D. Chen, B. Zhou, V. Koltun, and P. Krähenbühl. Learning by cheating.
In *Conference on Robot Learning*, pages 66–75, 2020.
URL: http://proceedings.mlr.press/v100/chen20a.html

[1] P. Abbeel, A. Coates, and A. Y. Ng.
Autonomous Helicopter Aerobatics through Apprenticeship Learning.
The International Journal of Robotics Research, 29(13):1608–1639, 2010-11-01.
URL: https://delete_doi.org/10.1177/0278364910371999.

[2] D. Chen, B. Zhou, V. Koltun, and P. Krähenbühl.
Learning by cheating.
In *Conference on Robot Learning*, pages 66–75, 2020.
URL: http://proceedings.mlr.press/v100/chen20a.html.

[3] S. Fujimoto and S. S. Gu.
A minimalist approach to offline reinforcement learning.
Advances in neural information processing systems, 34:20132–20145, 2021.
URL: https://proceedings.neurips.cc/paper_files/paper/2021/hash/a8166da05c5a094f7dc03724b41886e5-Abstract.html.

[4] E. Kaufmann, L. Bauersfeld, A. Loquercio, M. Müller, V. Koltun, and D. Scaramuzza.
Champion-level drone racing using deep reinforcement learning.
Nature, 620(7976):982–987, 2023-08.
URL: https://www.nature.com/articles/s41586-023-06419-4.

[5] A. Kumar, A. Singh, F. Ebert, M. Nakamoto, Y. Yang, C. Finn, and S. Levine.
Pre-Training for Robots: Offline RL Enables Learning New Tasks from a Handful of Trials, 2023-09-23.
URL: http://arxiv.org/abs/2210.05178, arXiv:2210.05178.

[6] J. Lee, J. Hwangbo, L. Wellhausen, V. Koltun, and M. Hutter.
Learning quadrupedal locomotion over challenging terrain.
Science Robotics, 5(47):eabc5986, 2020-10-21.
URL: https://www.research-collection.ethz.ch/bitstream/handle/20.500.11850/448343/1/2020_science_robotics_lee_locomotion.pdf.

[7] S. Sinha, A. Mandlekar, and A. Garg.
S4rl: Surprisingly simple self-supervision for offline reinforcement learning in robotics.
In *Conference on Robot Learning*, pages 907–917, 2022.
URL: `https://proceedings.mlr.press/v164/sinha22a.html`.

[8] J. Tobin, R. Fong, A. Ray, J. Schneider, W. Zaremba, and P. Abbeel.
Domain randomization for transferring deep neural networks from simulation to the real world.
In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 23–30, 2017.
URL: `https://ieeexplore.ieee.org/abstract/document/8202133/`.

[9] P. R. Wurman, S. Barrett, K. Kawamoto, J. MacGlashan, K. Subramanian, T. J. Walsh, R. Capobianco, A. Devlic, F. Eckert, F. Fuchs, L. Gilpin, P. Khandelwal, V. Kompella, H. Lin, P. MacAlpine, D. Oller, T. Seno, C. Sherstan, M. D. Thomure, H. Aghabozorgi, L. Barrett, R. Douglas, D. Whitehead, P. Dürr, P. Stone, M. Spranger, and H. Kitano.
Outracing champion Gran Turismo drivers with deep reinforcement learning.
Nature, 602(7896):223–228, 2022-02.
URL: `https://www.nature.com/articles/s41586-021-04357-7`.