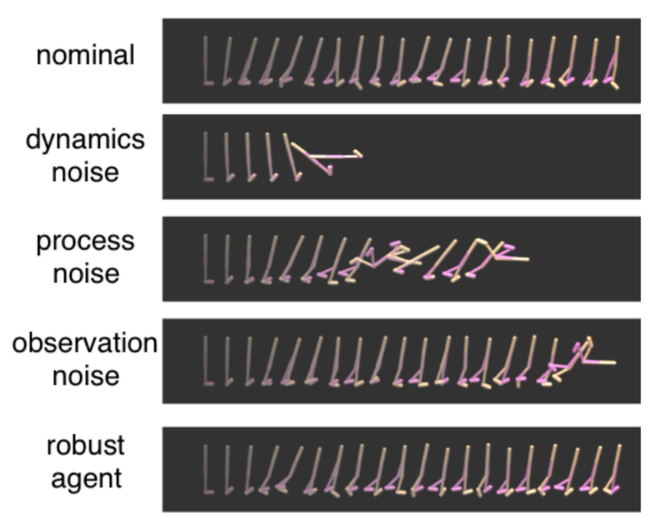
Adversarially Robust Policy Learning through Active Construction of Physically-Plausible Perturbations

Ajay Mandlekar, Yuke Zhu, Animesh Garg, Li Fei-Fei, Silvio Savarese
Department of Computer Science, Stanford University

Introduction

- As we move towards deploying learned controllers on physical systems around us, robust performance is not only a desired property but also a design requirement to ensure the safety of both users and the system itself.
- We demonstrate that Deep RL methods are susceptible to adversarial perturbations in states, model parameters, and observations.
- We introduce Adversarially Robust Policy Learning (ARPL) - an algorithm that leverages active computation of physically-plausible adversarial examples during training in order to enable robust performance under both random and adversarial perturbations of the system.



Physically-Plausible Threat Model

Dynamics Noise · Process Noise

Dynamics $\quad x_{t+1} = f(x_t, u_t; \boxed{\mu}) + \boxed{\nu}$

Observation $\quad z_t = g(x_t) + \boxed{\omega}$

Observation Noise

Key Idea: Can we use Adversarial Perturbations?

ARPL Algorithm

ARPL is a basic augmentation for any policy gradient method. Every iteration consists of policy evaluation and improvement.

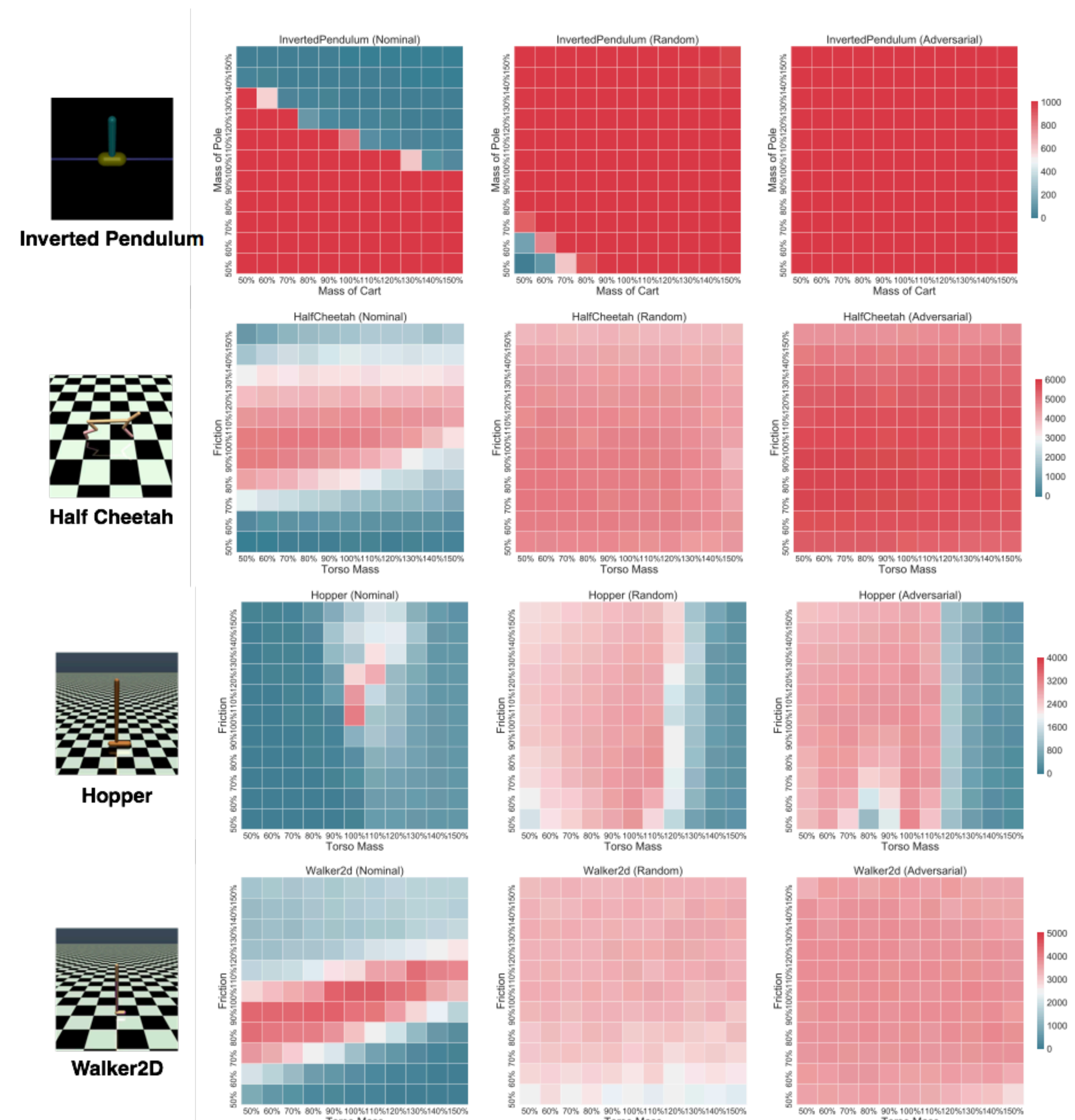
During **policy evaluation**, we collect N trajectories via policy rollouts. For every observation during a rollout, we **adversarially perturb** the state with **probability ϕ** and **magnitude ε** using the following perturbation

$$\delta = \varepsilon \, \nabla_s \eta(\pi_\theta(s))$$

where **η** is the L2 norm of the control produced by the policy **π**.

Then, we run **policy improvement**, as prescribed by the policy gradient method. Our implementation uses TRPO.

Demonstrated Robustness in Physical Dynamics Parameters



Experimental Setup

Process Noise - We perturb the original environment state and explicitly set this as the environment state in the simulator. We also feed this perturbed state to the agent as an observation.

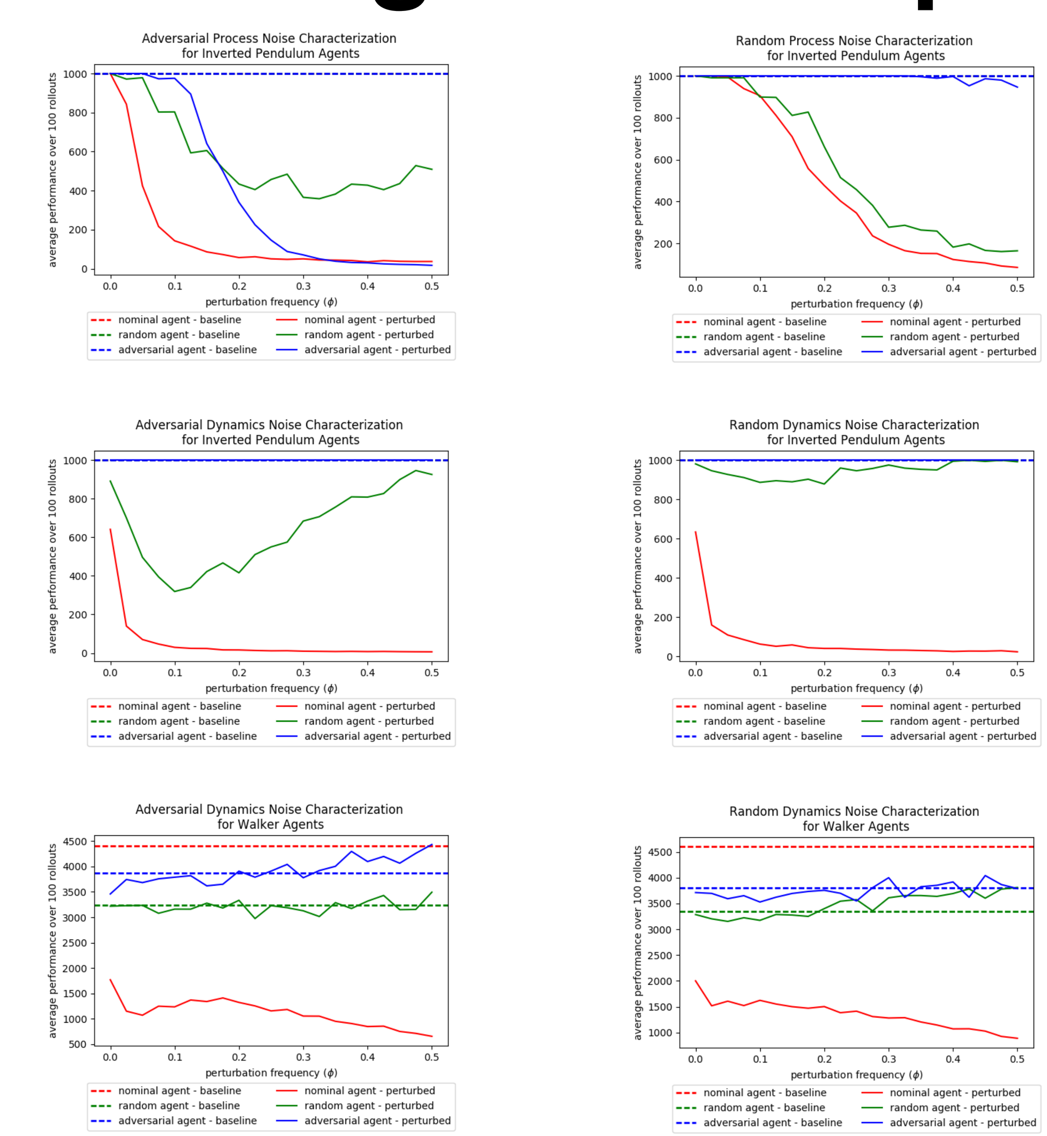
Dynamics Noise - We augment the agent's observation with environment dynamics parameters during training and use this part of the observation vector to compute perturbations for these parameters. They are then updated in the environment.

Observation Noise - Identical to process noise, but the agent receives the unperturbed state as the observation.

We experimented with the **perturbation type** (process, dynamics, observation), the **perturbation frequency**, controlled by **ϕ**, the probability of a perturbation at every time step, and whether the perturbation is generated **randomly** or **adversarially**.

We evaluate ARPL on 4 continuous control tasks using MuJoCo and Gym.

ARPL Agent Examples



References

S. Huang, N. Papernot, I. Goodfellow, Y. Duan, and P. Abbeel, "Adversarial attacks on neural network policies", Feb. 8, 2017. arXiv: 1702.02284v1 [cs.LG]

J. Schulman, S. Levine, P. Moritz, M. Jordan, and P. Abbeel, "Trust region policy optimization", *ICML*, 2015

C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus, "Intriguing properties of neural networks", Dec. 21, 2013. arXiv: 1312.6199v4 [cs.CV]

E. Todorov, T. Erez, and Y. Tassa, "Mujoco: A physics engine for model-based control", in *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*, IEEE, 2012, pp. 5026–5033