Rapidly Adaptable Legged Robots via Evolutionary Meta-Learning

Xingyou Song Yuxiang Yang Krzysztof Choromanski
Ken Caluwaerts Wenbo Gao Chelsea Finn Jie Tan

IROS 2020

Google AI

COLUMBIA UNIVERSITY
IN LVMINE TVO VIDEBIMVS LVMEN

Locomotion

Locomotion is one of the most fundamental skills of all land animals:

Cheetahs

Elephants

Humans

Robot Locomotion

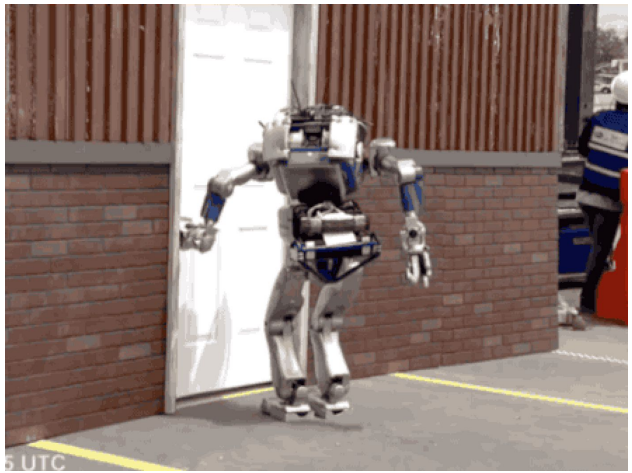
So far, Robot Locomotion research has displayed an impressive set of results to reproduce this natural skill.

Boston Dynamics Spot

MIT Cheetah

Slight Changes in Dynamics

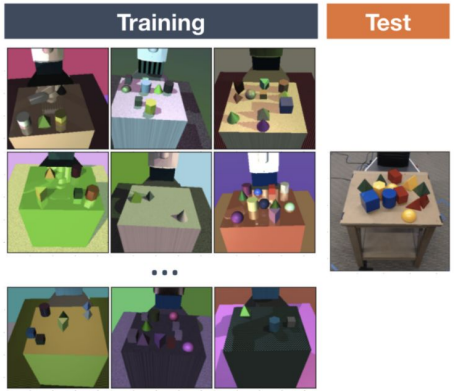
But unfortunately, robots can be fragile to slight changes in dynamics.



DARPA Robotics Challenge, 2015

Asimo Robot, 2006

Prior Works: Robustness to Real World Changes

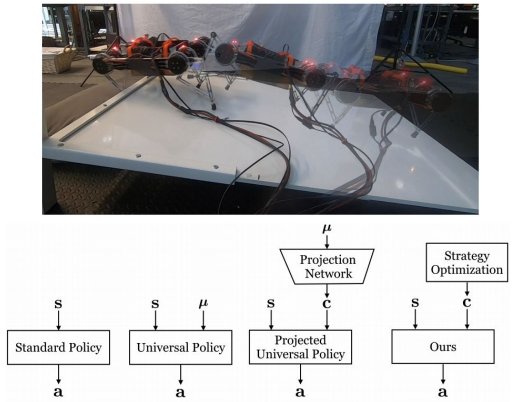


Domain Randomization (Tobin et al, 2017)

- Only trains in simulation
- Assumes all tasks possess same optimal policy.

Model Based Adaptation (Nagabandi et al, 2019)

- Dynamics models can have compounding error problems.
- Acquiring an accurate model can be difficult.

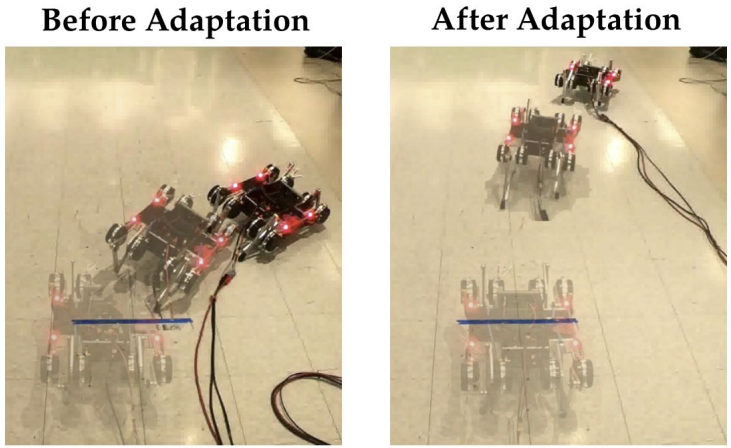


Meta Strategy Optimization (Yu et al, 2020)

- Latent context vectors appended to the state
- Context may not contain necessary information

Our Strategy: Real World Data for Fast Adaptation

- **Model-Free:** Only needs to train a feed-forward policy mapping **state -> action**.
- Instead of training only in simulation (Domain Randomization), use s a small amount of **real-world data** at **test time,** since it is the ground truth.



Minitaur Robot adapts to mass imbalances and voltage changes.

Challenges

1. **Sample Efficiency for Adaptation:** Real world data is expensive. We should only use a handful of episodes.
2. How does a pre-trained policy in simulation **use real world data** to adapt?

This leads us to Model Agnostic Meta-Learning or MAML (Finn et al, 2017).

Model-Agnostic Meta-Learning for Fast Adaptation of Deep Networks

Chelsea Finn [1] **Pieter Abbeel** [1 2] **Sergey Levine** [1]

Abstract

We propose an algorithm for meta-learning that is model-agnostic, in the sense that it is compatible with any model trained with gradient descent and applicable to a variety of different learning problems, including classification, regression, and reinforcement learning. The goal of meta-learning is to train a model on a variety of learning tasks, such that it can solve new learning tasks using only a small number of training samples. In our approach, the parameters of the model are explicitly trained such that a small number of gradient steps with a small amount of training data from a new task will produce good generalization performance on that task. In effect, our method trains the model to be easy to fine-tune. We demonstrate that this approach leads to state-of-the-art performance on two few-shot image classification benchmarks, produces good results on few-shot regression, and accelerates fine-tuning for policy gradient reinforcement learning with neural network policies.

the form of computation required to complete the task.

In this work, we propose a meta-learning algorithm that is general and model-agnostic, in the sense that it can be directly applied to any learning problem and model that is trained with a gradient descent procedure. Our focus is on deep neural network models, but we illustrate how our approach can easily handle different architectures and different problem settings, including classification, regression, and policy gradient reinforcement learning, with minimal modification. In meta-learning, the goal of the trained model is to quickly learn a new task from a small amount of new data, and the model is trained by the meta-learner to be able to learn on a large number of different tasks. The key idea underlying our method is to train the model's initial parameters such that the model has maximal performance on a new task after the parameters have been updated through one or more gradient steps computed with a small amount of data from that new task. Unlike prior meta-learning methods that learn an update function or learning rule (Schmidhuber, 1987; Bengio et al., 1992; Andrychowicz et al., 2016; Ravi & Larochelle, 2017), our algorithm does not expand the number of learned param-

Meta-Learning for Robotics in a Nutshell:

- Train **meta-policy** that can be easily turned into an **adapted policy** for a particular task from a large set of tasks in few amounts of data
- Training meta-policy in a standard meta-learning framework can be expressed as:

$$\max_{\theta} J(\theta) := \mathbb{E}_{T \sim \mathcal{P}(\mathcal{T})}\left[\mathbb{E}_{\tau' \sim \mathcal{P}_T(\tau'|\theta')}\left[R_T(\tau')\right]\right]$$

bilevel optimization formulation

$$\theta' = U(\theta, T) = \theta + \alpha \nabla_{\theta} \mathbb{E}_{\tau \sim \mathcal{P}_T(\tau|\theta)}\left[R(\tau)\right]$$

one-shot gradient-based adaptation operator

$\mathcal{P}_T(\cdot|\eta)$ - **distribution over trajectories given a task and conditioned on a policy**

- Standard MAML uses gradient-based operator for both adaptation and meta-policy optimization.

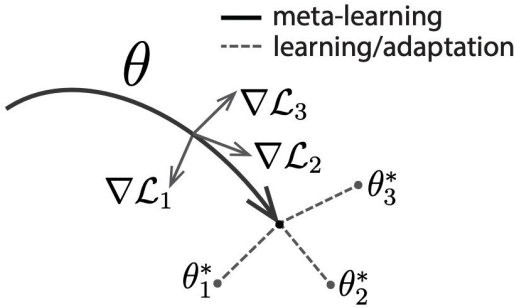
Gradient-based Meta Learning:

- Compute gradient of **J** as follows [the so-called **PG-MAML** algorithm]:

$$\nabla_\theta J(\theta) = \mathbb{E}_{T\sim\mathcal{P}(\mathcal{T})}[\mathbb{E}_{\tau'\sim\mathcal{P}_T(\tau'|\theta')}[\nabla_{\theta'}\log\mathcal{P}_T(\tau'|\theta')R(\tau')\nabla_\theta U(\theta,T)]]$$

$$\nabla_\theta U = \mathbf{I} + \alpha\int\mathcal{P}_T(\tau|\theta)\nabla^2_\theta\log\pi_\theta(\tau)R(\tau)d\tau + \alpha\int\mathcal{P}_T(\tau|\theta)\nabla_\theta\log\pi_\theta(\tau)\nabla_\theta\log\pi_\theta(\tau)^T R(\tau)d\tau$$

- **Challenge:** Unbiased estimation of the above gradient is non-trivial.
- **Model Limitations:** The above computational framework **excludes non-differentiable operators** U



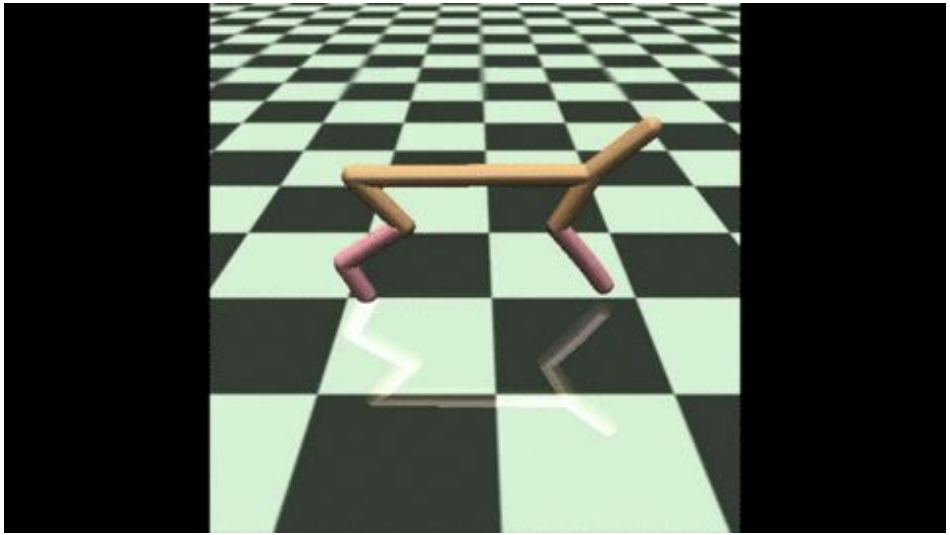
standard MAML

$$\max_\theta J(\theta) := \mathbb{E}_{T\sim\mathcal{P}(\mathcal{T})}[\mathbb{E}_{\tau'\sim\mathcal{P}_T(\tau'|\theta')}[R_T(\tau')]]$$

$$\theta' = U(\theta,T) = \theta + \alpha\nabla_\theta\mathbb{E}_{\tau\sim\mathcal{P}_T(\tau|\theta)}[R(\tau)]$$

PG-MAML Results

Originally, MAML in Reinforcement Learning used standard Mujoco benchmarks, and modified them via their reward functions to shape behaviors, such as Forwards + Backwards HalfCheetah:



https://github.com/tristandeleu/pytorch-maml-rl

PG-MAML for Legged Robots - Challenges

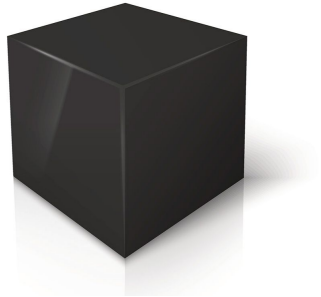
- PG-MAML performs task exploration via sampling actions from the policy's output distribution, but **jerky random actions are bad** for real robots.

$$a \sim \pi_\theta(s) = \mathcal{N}(\mu, \sigma)$$

- Noisy environment gives inaccurate return measurement. If $f(\theta)$ is (expected) cumulative reward, we get **noisy observations**:

$$\widetilde{f}(\theta, \varepsilon) = f(\theta) + \varepsilon$$

Evolutionary Meta Learning (ES-MAML)



Evolutionary Strategies (ES):

1. Treat total reward as **blackbox function**
2. Estimate gradients via local perturbations

$$\nabla_\theta \tilde{f}_\sigma(\theta) = \frac{1}{\sigma}\mathbb{E}_{\mathbf{g}\sim\mathcal{N}(0,\mathbf{I}_d)}[f(\theta+\sigma\mathbf{g})\mathbf{g}]$$

can be interpreted as gradients of Gaussian smoothings and approximated via Monte Carlo methods

ESGrad (f, θ, n, σ)

inputs: function f, policy θ, number of perturbations n, precision σ

Sample n i.i.d $N(0, I)$ vectors $g_1, \ldots, g_n$;

return $\frac{1}{n\sigma}\sum_{i=1}^{n} f(\theta+\sigma g_i)g_i$;

Features of ES algorithms:

- Work particularly well for continuous control tasks.
- Allow ***deterministic*** policies, which do not exacerbate the noise problem.
- Provide parameter-based rather than action-based exploration (as PG algorithms).

Evolutionary Meta Learning (ES-MAML)

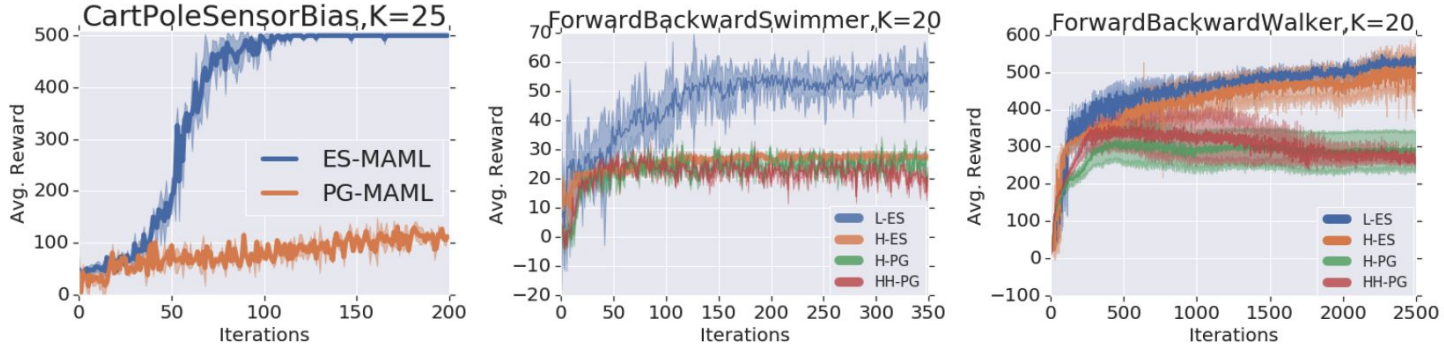
ES-MAML: Estimate the meta gradient using ES. (Song et al, 2019)

for $t = 0, 1, \ldots$ **do**
  Sample n tasks $T_1, \ldots, T_n$ and iid vectors $\mathbf{g}_1, \ldots, \mathbf{g}_n \sim \mathcal{N}(0, \mathbf{I})$;
  foreach $(T_i, \mathbf{g}_i)$ **do**
    $v_i \leftarrow f^{T_i}(U(\theta_t + \sigma \mathbf{g}_i, T_i))$
  end
  $\theta_{t+1} \leftarrow \theta_t + \frac{\beta}{\sigma n} \sum_{i=1}^{n} v_i \mathbf{g}_i$
end

- Can use non-differentiable adaptation operator *U*.
- Example: **Hill-climbing**, which enforces **monotonic improvement** (in Deterministic Environments).

ES-MAML: Continuous Control Benefits

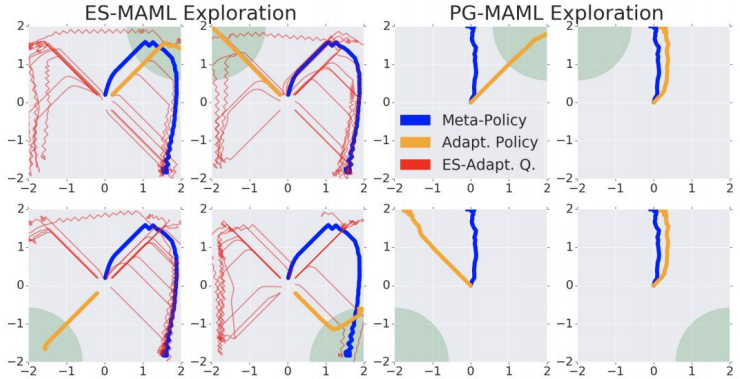
Figure 4: Stability comparisons of ES and PG on the Biased-Sensor CartPole and Swimmer, Walker2d environments. (L), (H), and (HH) denote linear, one- and two-hidden layer policies.



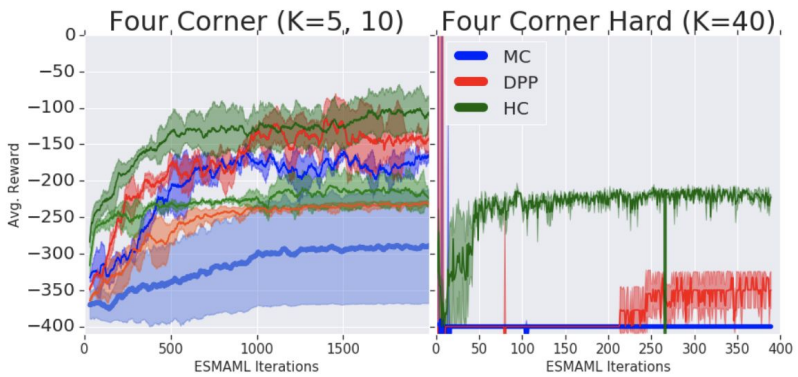
ES-MAML: Exploration Benefits - 4 Corners

- Exploration via **parameter space** solves hard tasks for PG-MAML.
- **Hill-Climbing** is strongest adaptation operator.

Figure 1: (a) ES-MAML and PG-MAML exploration behavior. (b) Different exploration methods when K is limited ($K = 5$ plotted with lighter colors) or large penalties are added on wrong goals.



(a)



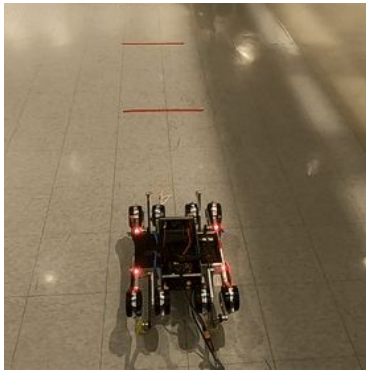
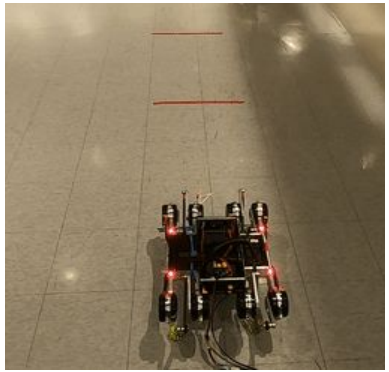
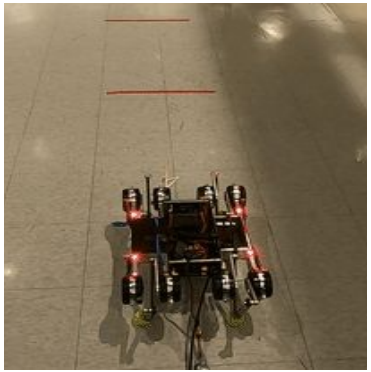
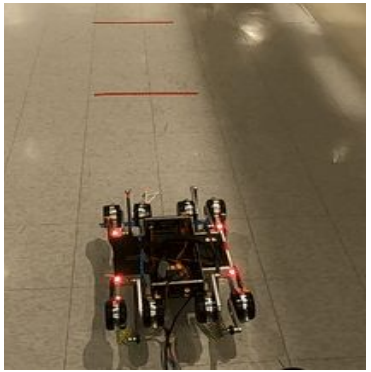
(b)

What about the real world?

Adaptation in the noisy real world

When there is noise:

$$\widetilde{f}(\theta, \varepsilon) = f(\theta) + \varepsilon$$

Which hill-climbing operator should we use?

Sequential Hill-Climbing

Sequential (Original):

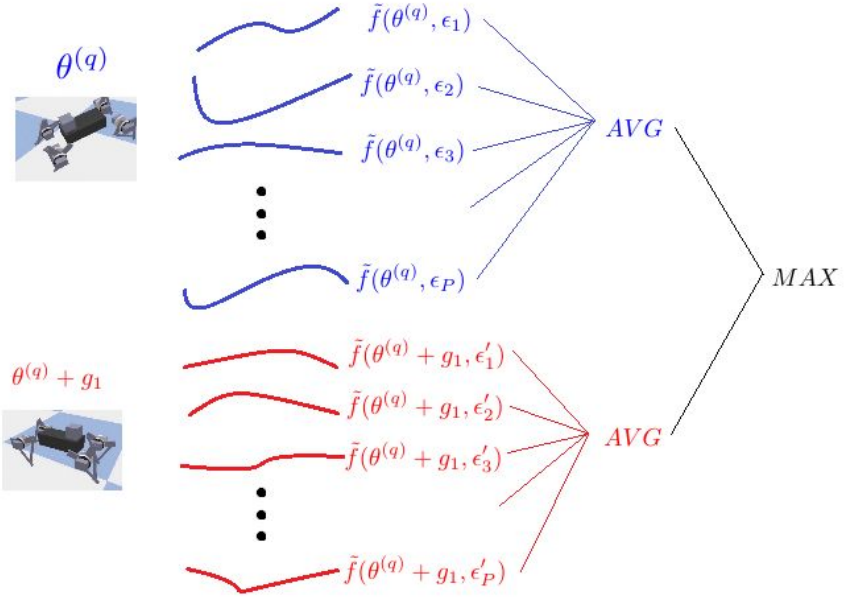
- Monotonic increase only in the deterministic case.
- Susceptible to noise in the real world.

$$\theta^{(q+1)} = \underset{\theta \in \{\theta^{(q)}, \theta^{(q)} + \alpha \mathbf{g}\}}{\operatorname{argmax}} f(\theta)$$

Average Hill-Climbing

Average evaluation over *P* trials - Approximate the **expected objective**:

- Low sample efficiency - multiple rollouts committed to single parameter
- Fails when noise is:
  - **Not IID**. Example: Robot motor overheats throughout testing.
  - **Not zero mean**. Example: Robot can fall during testing.

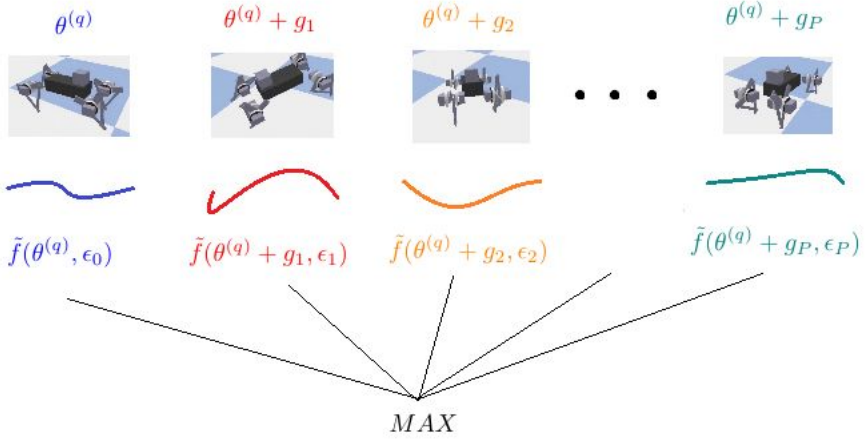


$$\theta^{(q+1)} = \underset{\theta \in \{\theta^{(q)}, \theta^{(q)} + \alpha \mathbf{g}\}}{\operatorname{argmax}} \frac{1}{P} \sum_{i=1}^{P} \tilde{f}(\theta, \varepsilon_i)$$

Batch Hill-Climbing

Batch evaluation over *P* perturbed trials - Take the best trial, **even if noisy**:

- Sample efficient - P diverse parameter samples.
- Works even in the case of adversarial noise - does not require strict noise assumptions!



$$\theta^{(q+1)} = \underset{\theta \in \{\theta^{(q)}, \theta^{(q)} + \alpha \mathbf{g}_1, \ldots, \theta^{(q)} + \alpha \mathbf{g}_P\}}{\operatorname{argmax}} \tilde{f}(\theta, \varepsilon)$$

The Mathematics of Batch Hill-Climbing

Batch Hill-Climbing:

- producing strong convergence (see: right) with high probability even if substantial number of measurements is **arbitrarily corrupted**

standard averaging-operator is not resistant to arbitrary noise

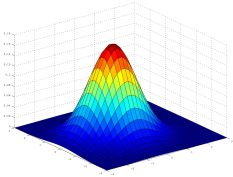
$f : \mathbb{R}^d \to \mathbb{R}$ *is a* (μ, ρ)*-strong concave function*

diameter of f-domain

$$\frac{\sum_{t=0}^{T-1}\left(f(\theta^{\mathrm{opt}}) - f(\theta_t)\right)}{T} \leq \frac{1}{\sqrt{T}}\left(\frac{D^2}{2} + \left(L + \frac{4L^2}{\sqrt{T}}\right)^2 + 8DL^2\right) + D\phi$$

number of iterations of the algorithm

upper-bound on the norm on the L2-norm of f-gradient



any constant satisfying:

$$\phi > \frac{4(\rho-\mu)\sqrt{T}d\sigma}{7} + \frac{16\Lambda\sqrt{T}}{7\sigma\sqrt{d}}$$

upper bound on the measurement error of small-error measurements

any constant satisfying:

$$\sigma \leq 4\sqrt{\frac{\Lambda}{7\sqrt{d}}}$$

$$or\ |f(\theta^{\mathrm{opt}}) - f(\theta_i)| \leq D\phi\ for\ some\ \theta_i$$

Experimental Framework

Reinforcement Learning Framework

In the standard RL framework, the Minitaur task is a Markov Decision Process (MDP), consisting of:

- **Observation:** Roll + Pitch Angle, 8 Motor Angles, and sin/cos phase variable
- **Action:** Swing and Extension of each leg
- **Reward:** Encourages gradually increasing velocity, reduce energy (torque * angular velocity)

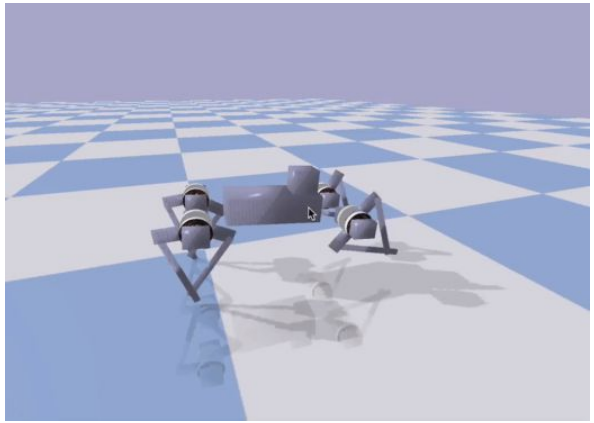
$$r(t) = \min(v, v_{\max})dt - 0.005 \sum_{i=1}^{8} \tau_i \omega_i dt$$

Simulation Experiment Setup

We train the meta policy in simulation using Pybullet.

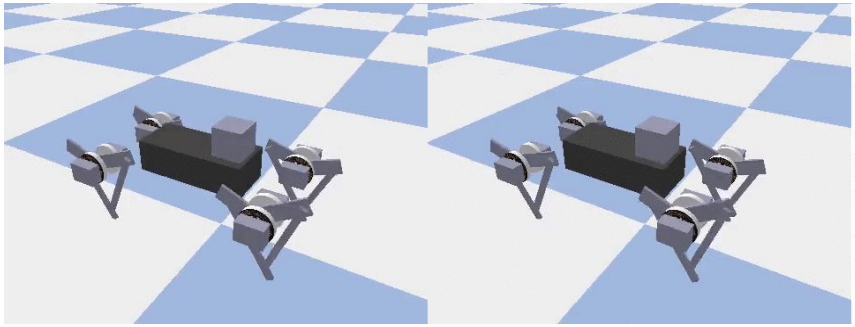
Each task samples a different combination of physics parameters:

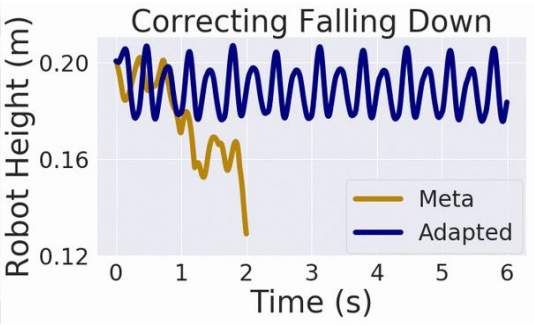
- Body and Leg Mass
- Battery Voltage, Foot Friction
- Motor Damping, Motor Strength, Control Latency



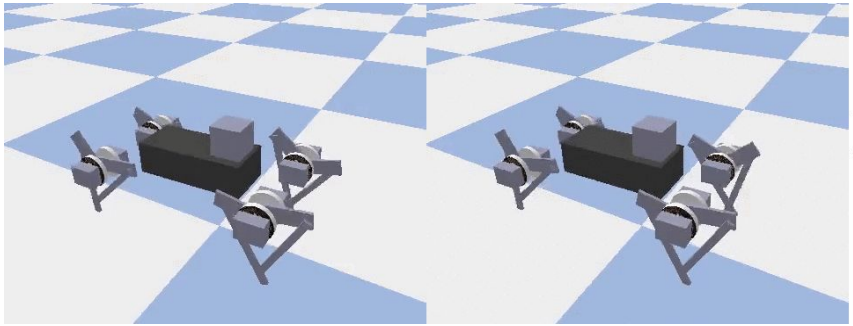
Simulation Results: Qualitative Changes

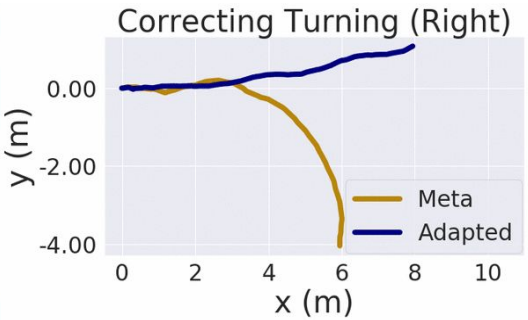
Correction from falling:





Correction of walking direction:

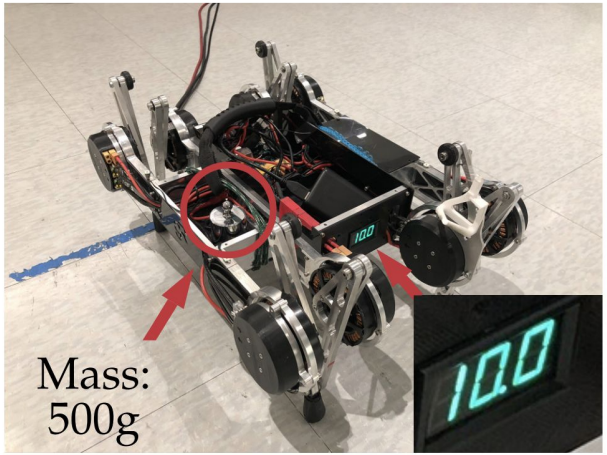




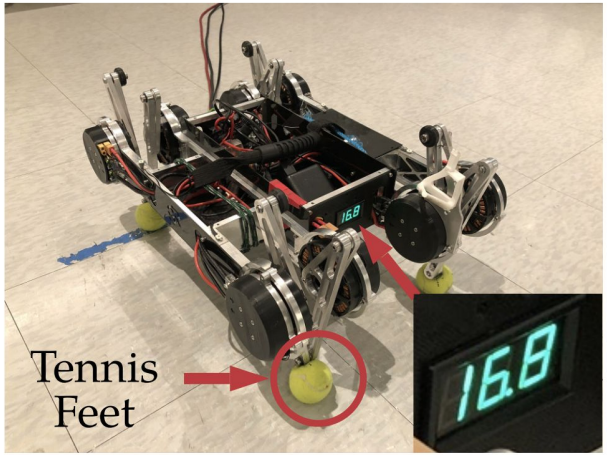
Real-Robot Experiments

Task Setup

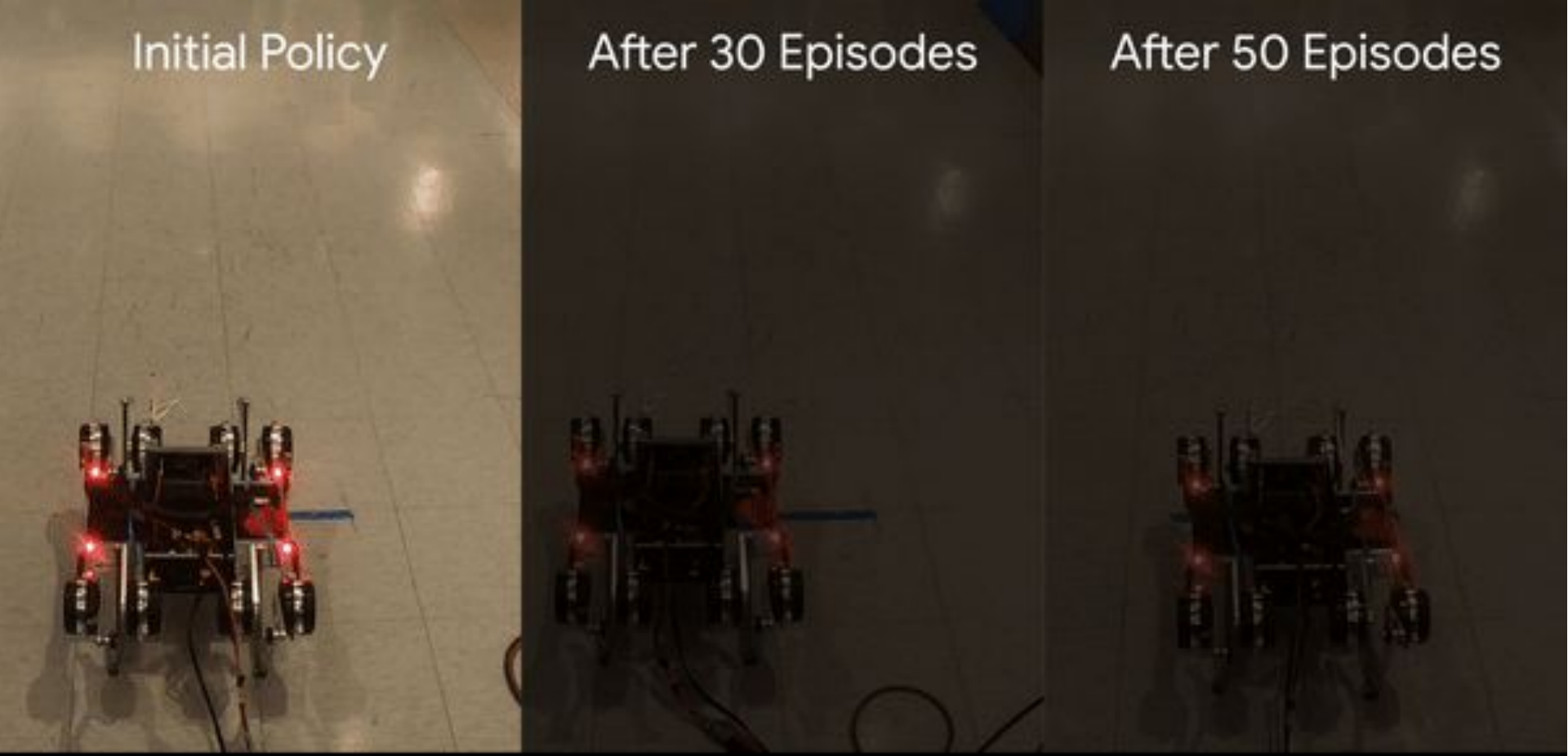
Mass-Voltage Task



Friction Task

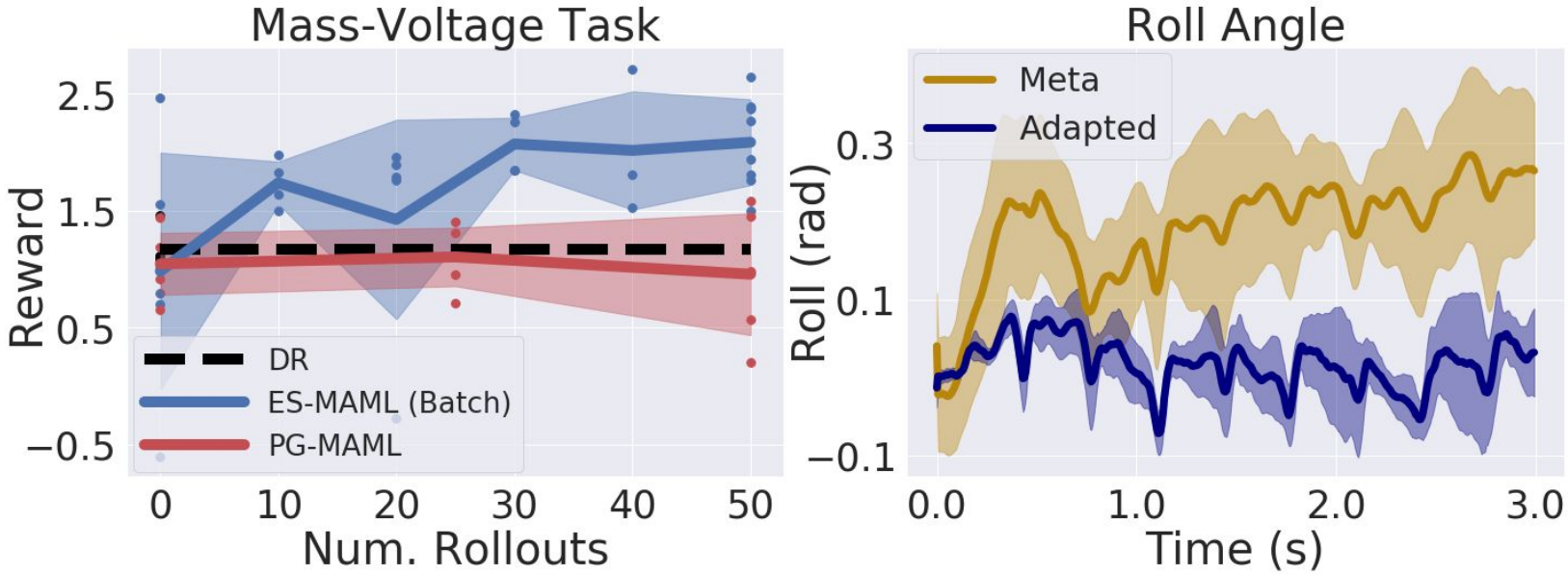


- Mass Voltage: 500g mass on side, voltage reduced to disrupt leg synchronization
- Friction: Tennis Balls on feet, to reduce gait via slipping.

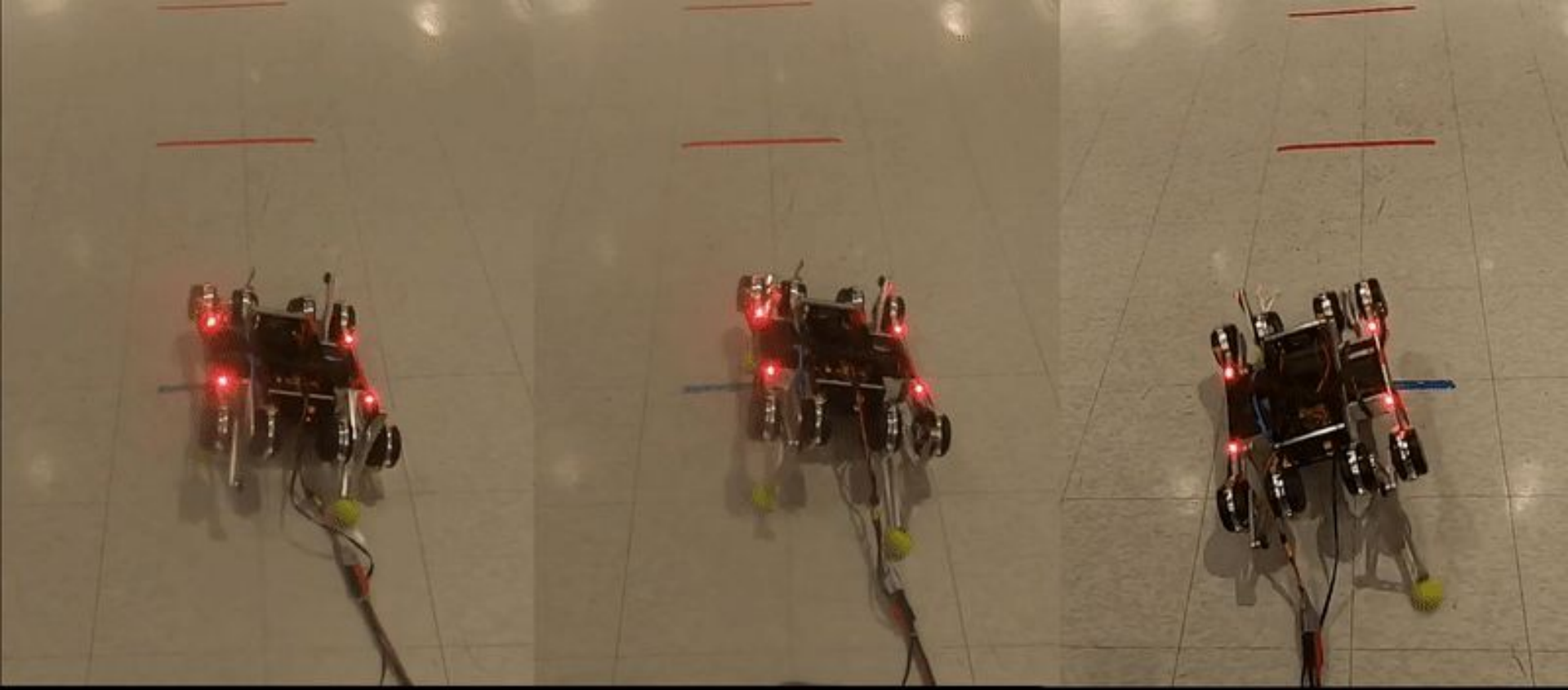


The initial policy shifts to the right.

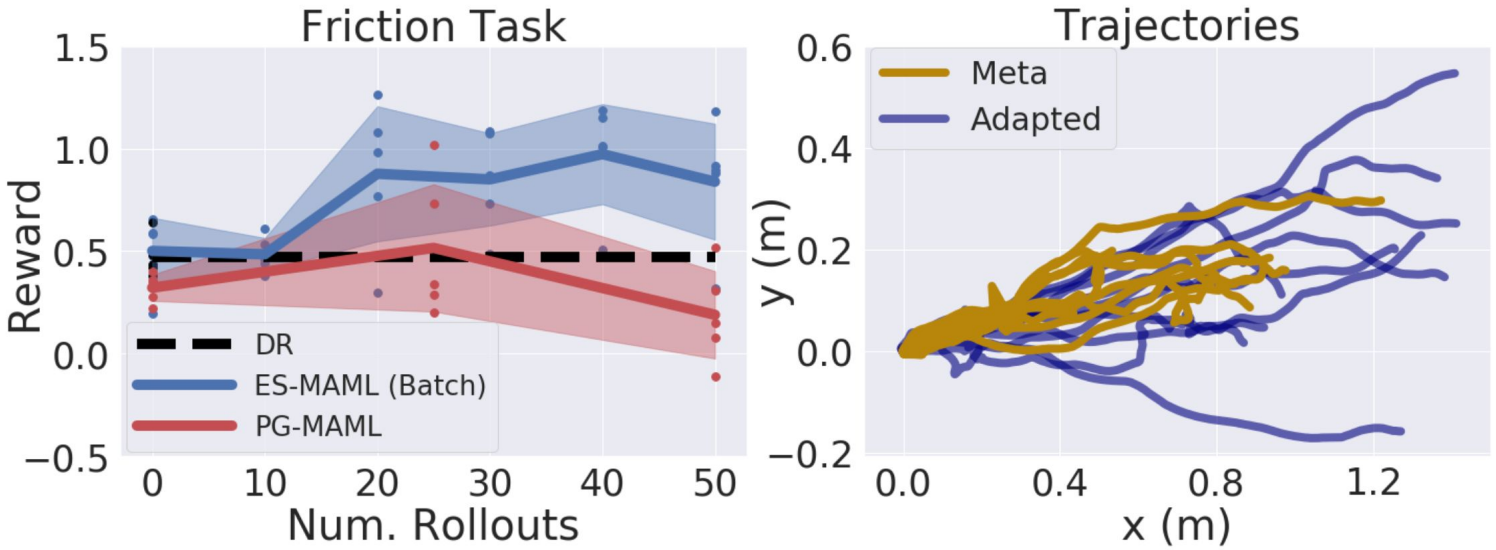
Mass-Voltage Task



- ES-MAML outperforms PG-MAML and Domain Randomization (DR)
- ES-MAML stabilizes the roll angle to 0 after adaptation.


Domain Randomization
PG-MAML
Our Method

Friction Task



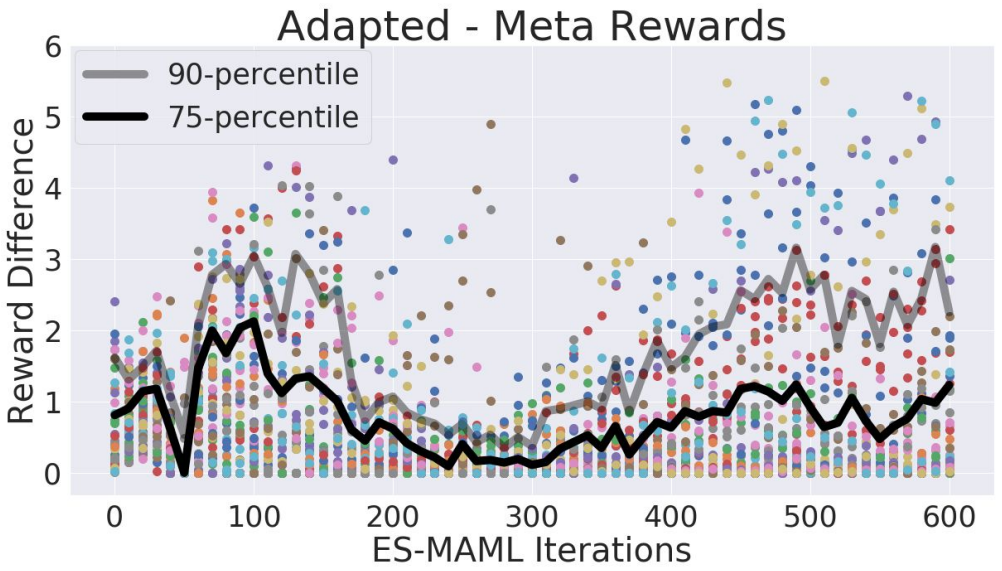
- ES-MAML outperforms PG-MAML and Domain Randomization (DR)
- Despite the noisy environment, ES-MAML consistently produces longer and more stable trajectories.

Ablation Studies in Simulation

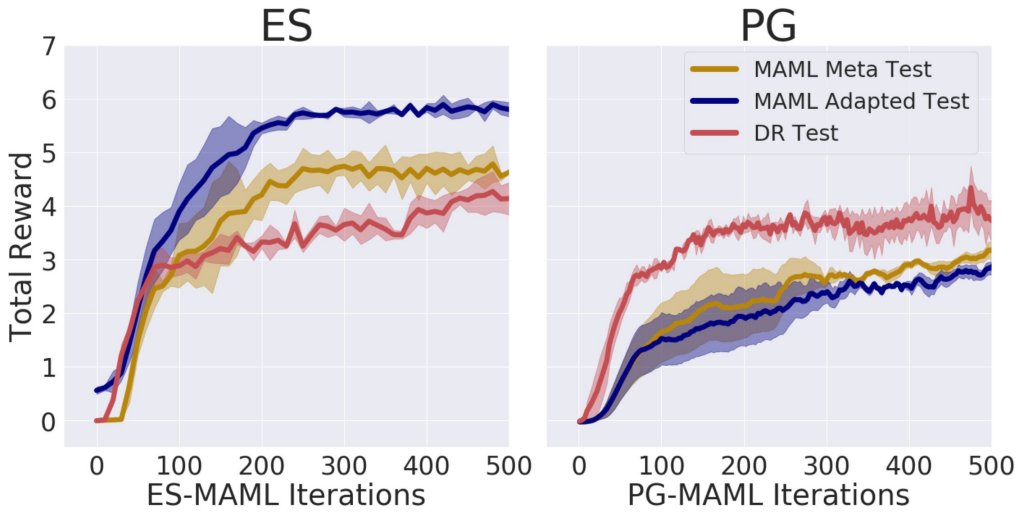
Simulation Results: Distribution Across Tasks

Is adaptation even needed for this benchmark?

Yes! Scatterplot shows multiple tasks where ES-MAML adaptation increases reward drastically



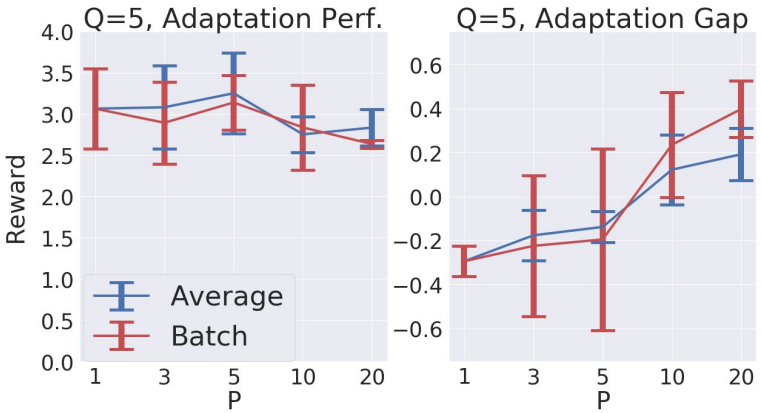
Simulation Results: ES-MAML vs PG-MAML



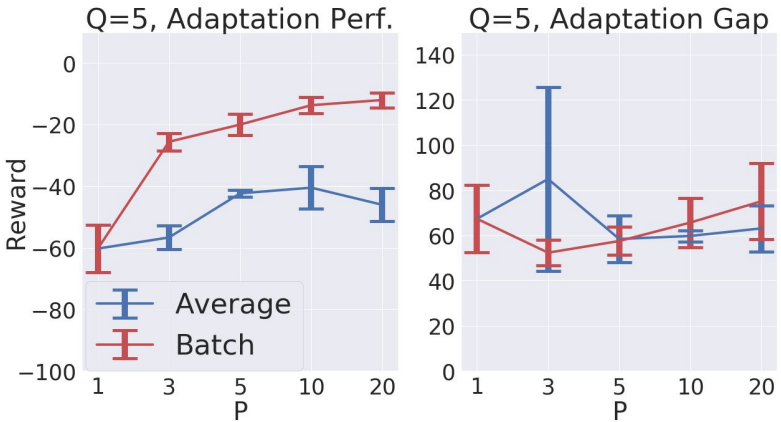
- ES-MAML consistently outperforms PG-MAML and Domain Randomization (DR)
- Hill-Climbing enforces Adapted > Meta, while PG-MAML has no guarantees.

Simulation Results: Average vs Batch

- Given same number of parameter changes (Q) and parallel (P) rollouts:
  - **(Left):** On **Noisy Minitaur,** Batch produces higher adaptation gap.
  - **(Right):** On **Noisy Nav-2D** (toy env. from (Finn et al, 2017)), Batch Produces higher raw adaptation performance.



Noisy Minitaur Environment



Noisy Nav-2D Environment

Conclusion

- We have demonstrated a **successful application of MAML on a real robot**.
- ES-MAML + Batch Hill-Climbing (our method) enables fast adaptation on robots.
  - Less sensitive to noisy robot experiments
  - Allows all the benefits of ES for robotics:
    - Deterministic, stable policies
    - Exploration via parameter space

Future Work

- Continuous Adaptation:
  - How can the robot adapt to constantly changing environments?
- Improving Sample Efficiency:
  - Can we use less data for adaptation by using model-based techniques?
  - Are there better adaptation operators for the model-free case?

More Details

Please see our following links for more information:

- arXiv: https://arxiv.org/abs/2003.01239
- Google AI Blog: https://ai.googleblog.com/2020/04/exploring-evolutionary-meta-learning-in.html
- Video: https://youtu.be/_QPMCDdFC3E
- Code: https://github.com/google-research/google-research/tree/master/es_maml

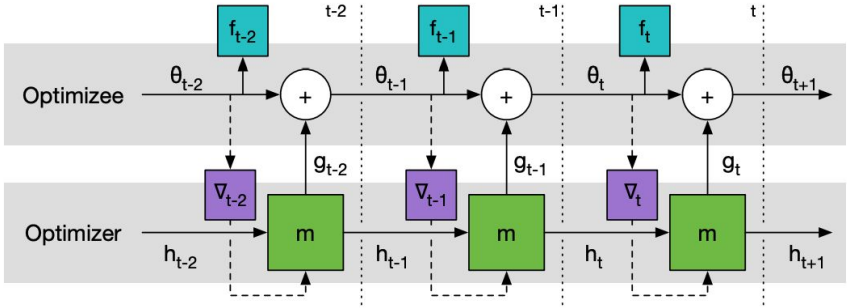
Thank you!

UNUSED SLIDES

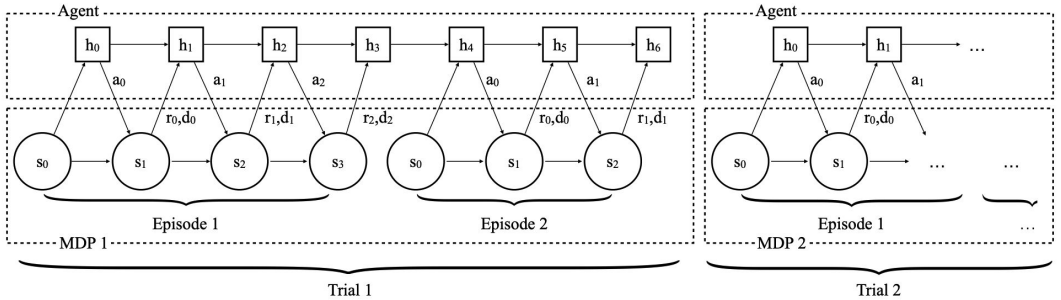
Meta Learning - Learning the Learning Process

Meta-Learning, or “Learning to Learn” has appeared in a variety of works:

Learning to Learn by Gradient Descent by Gradient Descent (Andrychowicz et al, 2016)



RL^2 (Duan et al, 2017)



MAML: Model Agnostic Meta Learning (Finn. et al 2017)

Meta Training

For each task i

$$\theta_i = \theta - \alpha \nabla_\theta \mathcal{L}_{\mathcal{T}_i}(\theta)$$

$$\theta = \arg\min_\theta \mathbb{E}_{\mathcal{T}_i \sim p(\mathcal{T})} \mathcal{L}_{\mathcal{T}_i}(\theta_i)$$

Testing

For test task T^{test}

$$\theta_{\text{test}} = \theta - \alpha \nabla_\theta \mathcal{L}_{\mathcal{T}^{\text{test}}}(\theta)$$