Closing the Sim-To-Real Gap with Evolutionary Meta-Learning

Xingyou (Richard) Song Yuxiang Yang Krzysztof Choromanski
Ken Caluwaerts Wenbo Gao Chelsea Finn Jie Tan
Aldo Pacchiano Yunhao Tang

Locomotion

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

Cheetahs

Elephants

Humans

Robot Locomotion

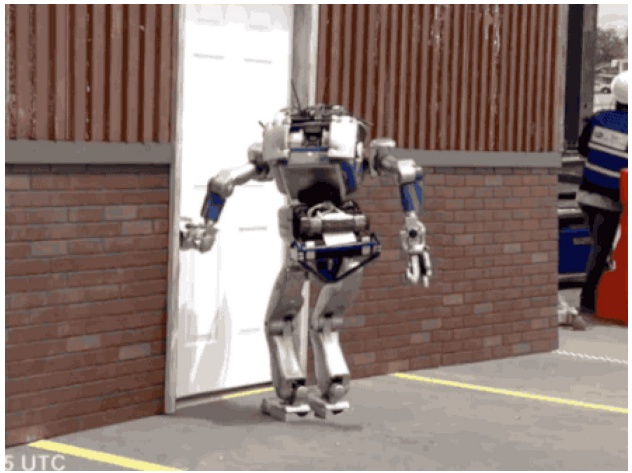
So far, Robot Locomotion research has displayed an impressive set of results to reproduce this natural skill.

Boston Dynamics Spot

MIT Cheetah

Slight Changes in Dynamics

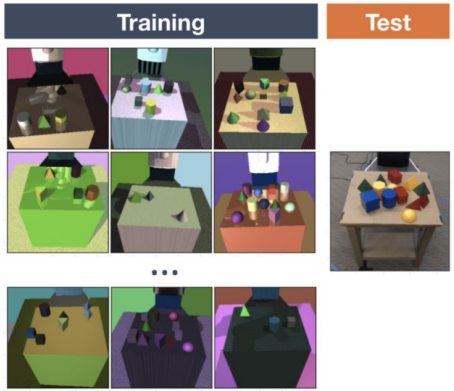
But unfortunately, robots can be fragile to slight changes in dynamics.



DARPA Robotics Challenge, 2015

Asimo Robot, 2006

Prior Works: Robustness to Real World Changes

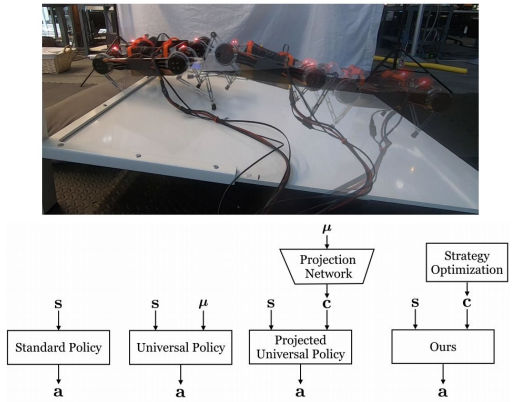


Domain Randomization (Tobin et al, 2017)

- Only trains in sim
- Assumes all tasks use same optimal policy.

Model Based Adaptation (Nagabandi et al, 2019)

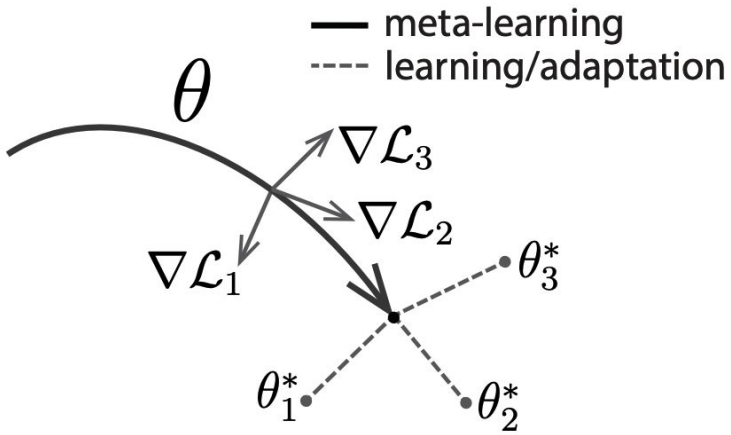
- Compounding error with dynamics models
- Acquiring an accurate model can be difficult.



Meta Strategy Optimization (Yu et al, 2020)

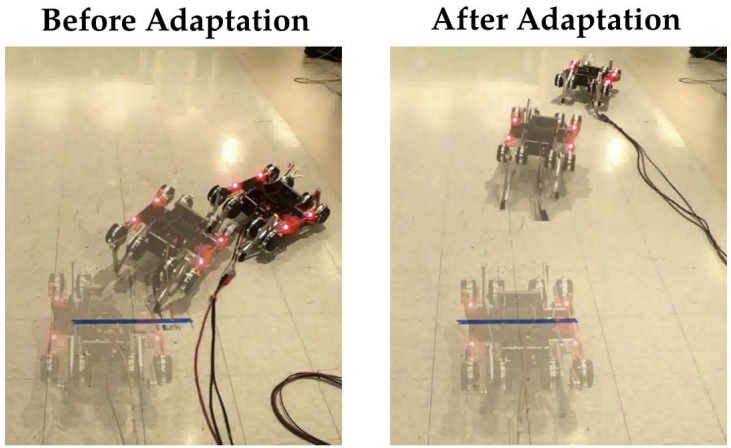
- Latent context vectors appended to the state
- Context may not contain necessary information

But what about MAML?



Our Strategy: MAML + Adaptation w/ Real world Data

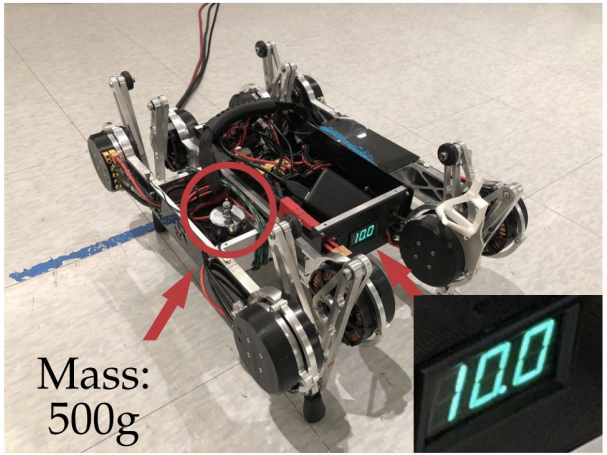
- **Train most skills in sim: “meta-policy”**
- **Fine-Tune + Adapt w/ a little real world data: “adapted-policy”**
- **Model-Free:** Only needs feed-forward policy mapping **state -> action**.



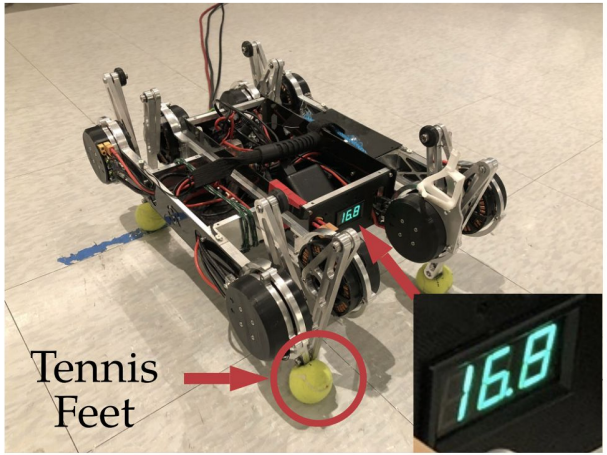
Minitaur Robot adapts to mass imbalances and voltage changes.

Task Setup

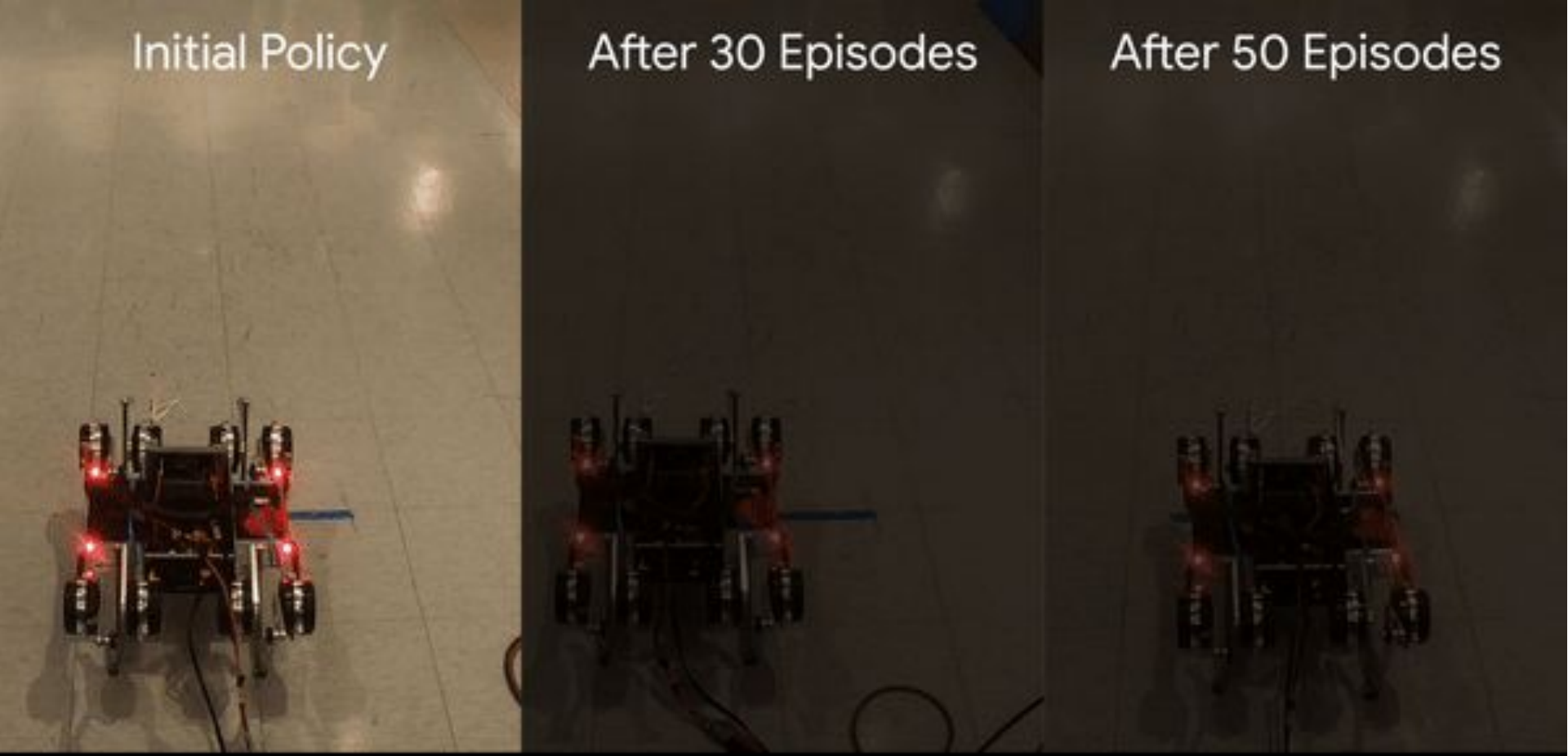
Mass-Voltage Task



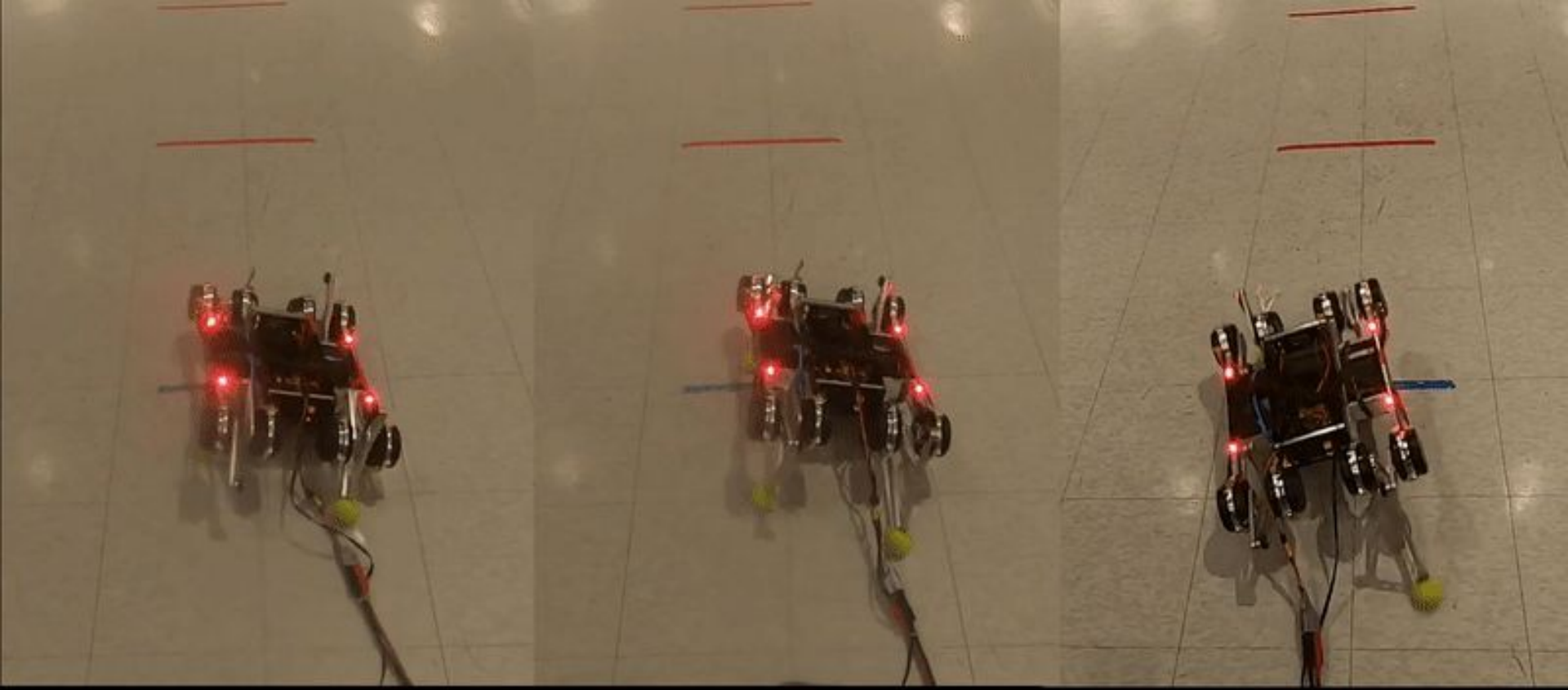
Friction Task



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



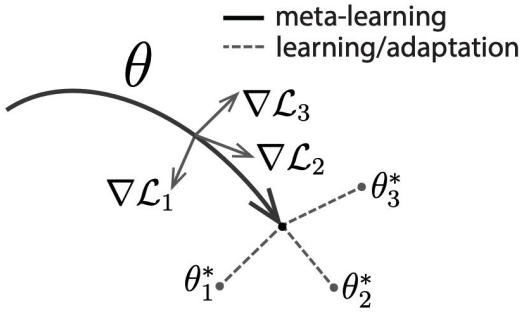
The initial policy shifts to the right.


Domain Randomization
PG-MAML
Our Method

How did we get here?

Formalism of Meta-Learning

- Original MAML problem: Find optimal meta-policy



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

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

Gradient-Based Meta Learning is Complicated!

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

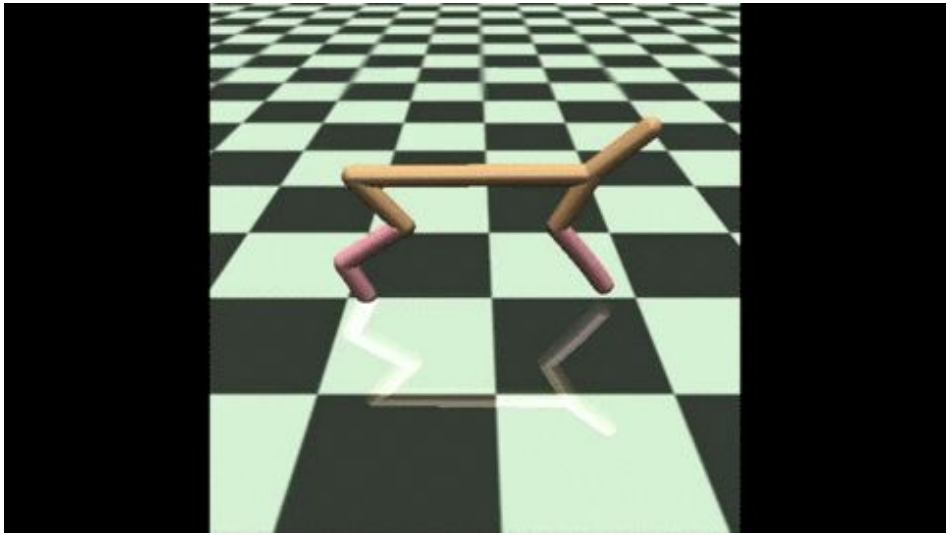
$$\nabla_\theta U = \mathbf{I} + \alpha\int\mathcal{P}_T(\tau|\theta)\nabla^2_\theta\log\pi_\theta(\tau)R(\tau)d\tau + \alpha\int\mathcal{P}_T(\tau|\theta)\nabla_\theta\log\pi_\theta(\tau)\nabla_\theta\log\pi_\theta(\tau)^T R(\tau)d\tau$$

- **Policy Gradient (PG)-MAML**
- **Challenge:** Estimation of the gradient is very complicated.
- **Limitations:** **Doesn't allow non-differentiable operators U**

Previous Results in MAML

Restricted to **reward function** changes, not **dynamics changes.**

Example: Forwards + Backwards HalfCheetah



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

Importance of Dynamics Adaptation

- In real world, we care more about **dynamics changes** for robust walking.

Minitaur RL Framework

Minitaur MDP: (Observation, Action, Reward)

- **Observation:** Roll + Pitch Angle, 8 Motor Angles, and sin/cos phase variable
- **Action:** Swing and Extension of each leg
- **Reward:** Velocity minus energy (torque * angular velocity), encourages straight walking

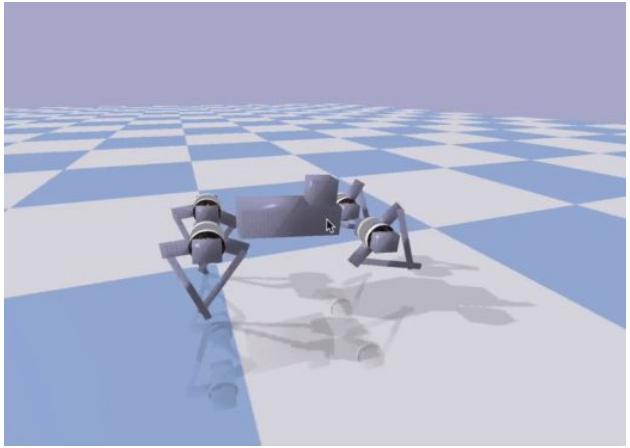
$$r(t) = \min(v, v_{\max})dt - 0.005 \sum_{i=1}^{8} \tau_i \omega_i dt$$

MAML 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



PG-MAML for Legged Robots - Challenges

- PG-MAML is **stochastic: Jerky random actions can be bad** for real robots.

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

- Real world is never deterministic. If $f(\theta)$ is objective, we always observe (non-Markovian) noise:

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

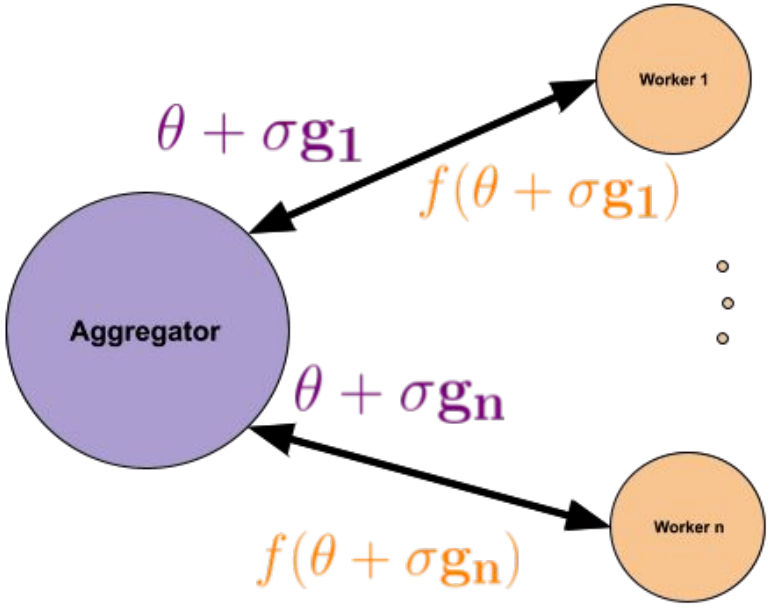
Alternative: Evolutionary/Blackbox Methods

Evolutionary Strategies (ES):

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

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

Gradient of the Gaussian Smoothing of the function



ESGrad (f, θ, n, σ)

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

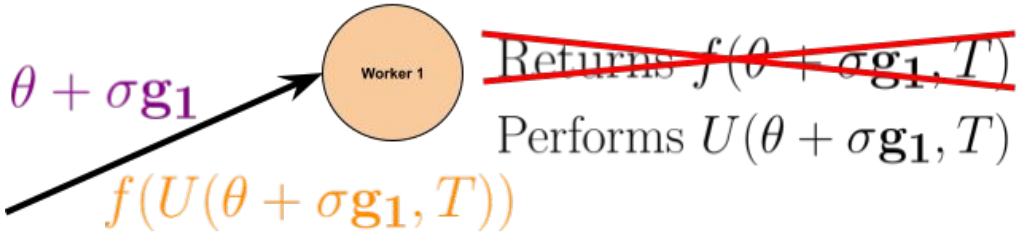
Sample n i.i.d $N(0, I)$ vectors $g_1, \ldots, g_n$;

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

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).

PG-MAML vs ES-MAML Conceptually

PG-MAML's Catch 22:

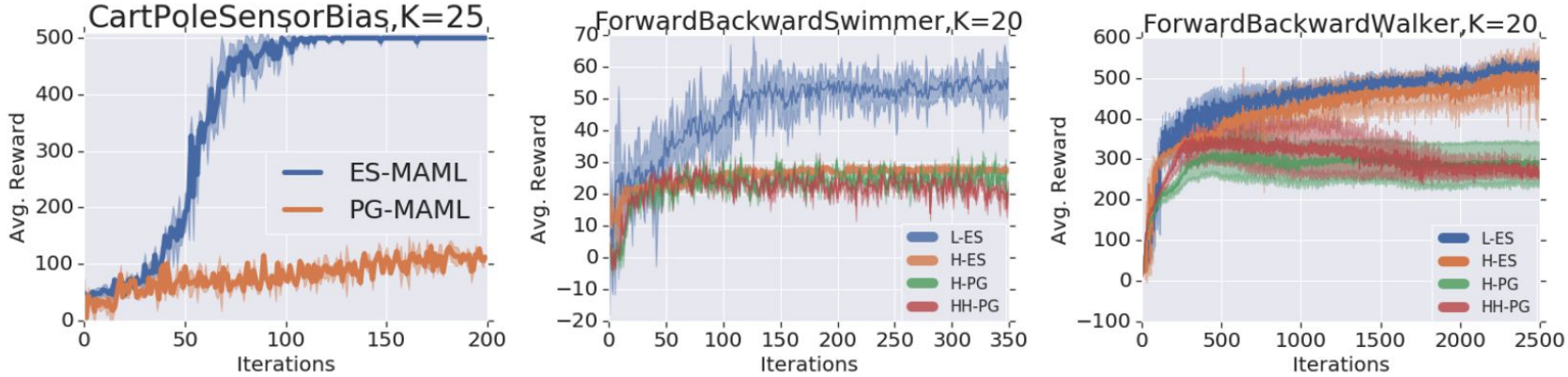
- Needs stochastic policies
  - Makes **random actions**
  - Noise problem becomes even worse.
- Action-based exploration
  - Relies on **random actions**
- Inner + Outer loop both **gradient-based**
- Adaptation improvement **not guaranteed.**

ES-MAML:

- Allows ***deterministic* policies**
  - Doesn't exacerbate noise problem.
- Parameter Space Exploration
  - Also **doesn't add randomness to policy.**
- Inner + Outer loop both **Zeroth-order optimization.**
- Hill-Climb Operator **enforces improvement.**

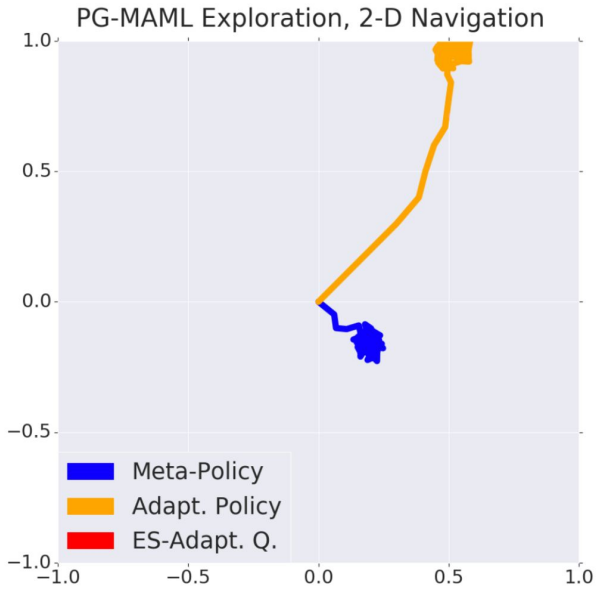
ES-MAML: Continuous Control Benefits

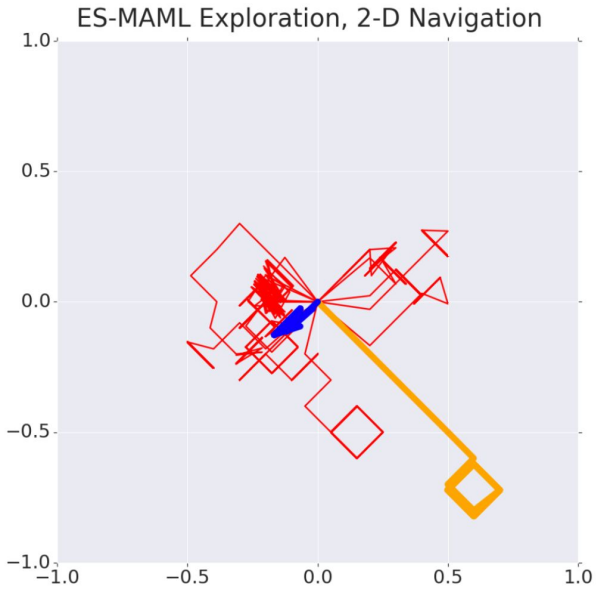
Figure 4: Stability comparisons of ES and PG on the Biased-Sensor CartPole and Swimmer, Walker2d environments. (L), (H), and (HH) denote linear, one- and two-hidden layer policies.



ES-MAML vs PG-MAML: Exploration Fundamentals

- PG-MAML makes small moves, triangulates goal location
- ES-MAML moves different directions, figures out goal from total reward

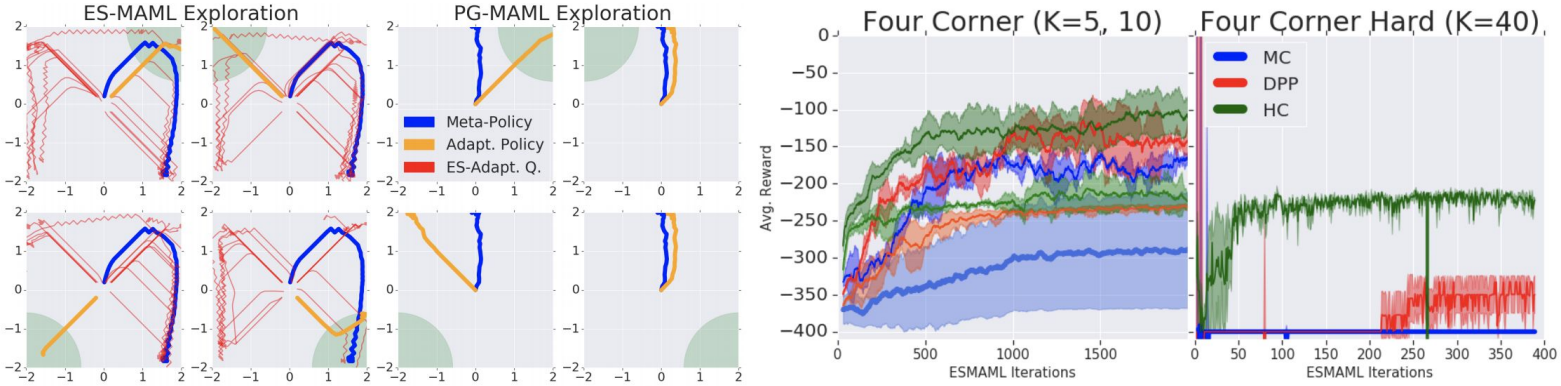




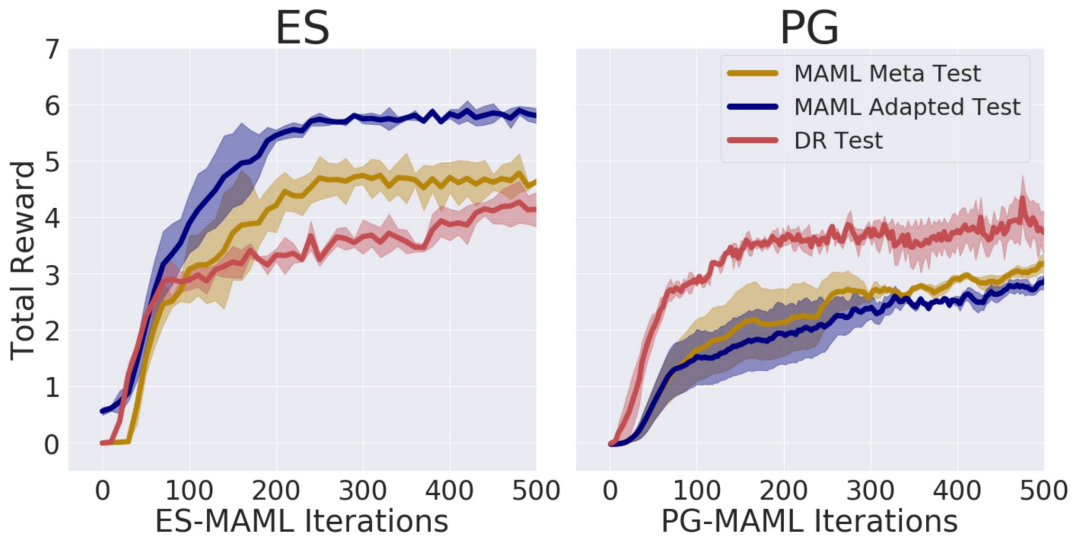
ES-MAML: Exploration Benefits - 4 Corners

- Exploration via **parameter space** solves hard tasks for PG-MAML.
- **Hill-Climbing (HC)** 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.



Minitaur Sim Results: ES-MAML vs PG-MAML

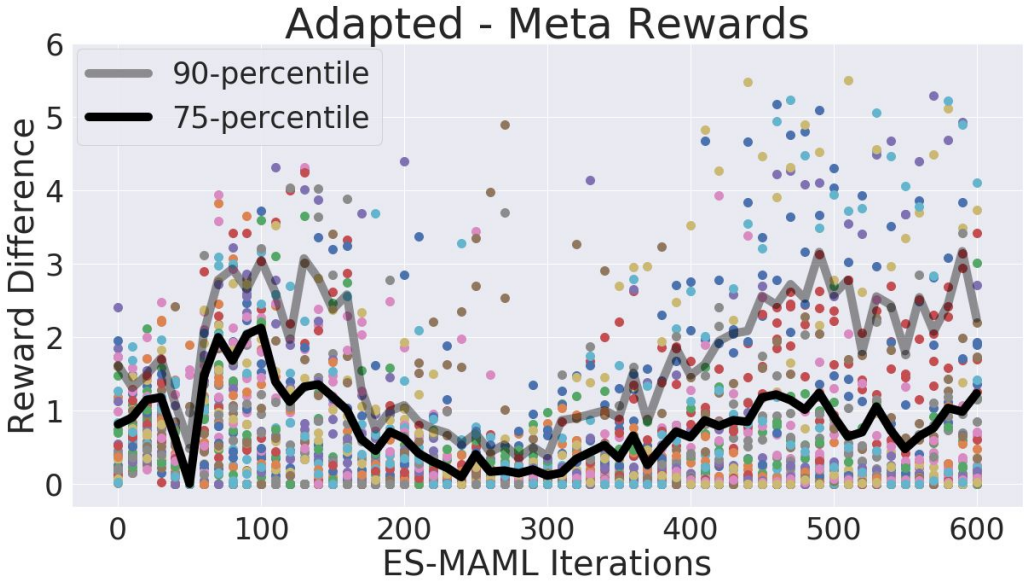


- **ES-MAML** > **PG-MAML and Domain Randomization (DR)**
- Hill-Climbing **enforces Adapted > Meta**, while PG-MAML **has no guarantees.**

Minitaur Sim: Distribution Across Tasks

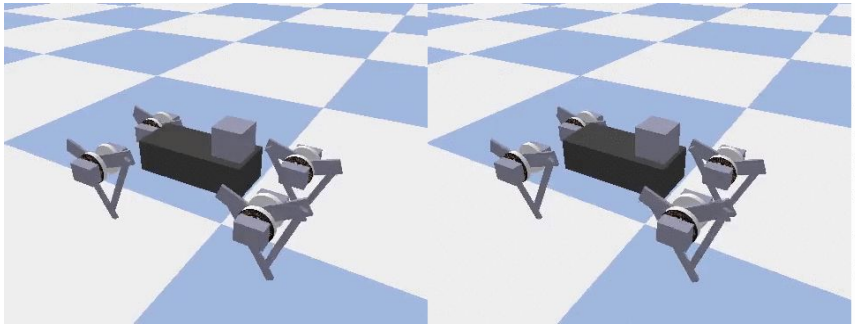
Is adaptation even needed for this benchmark?

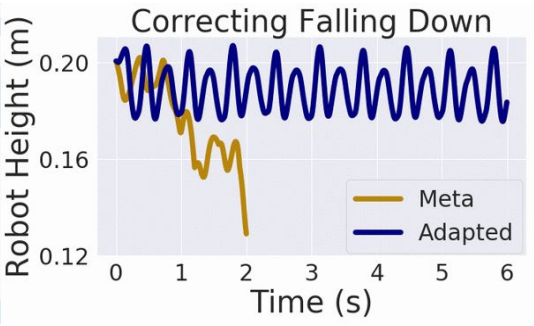
Yes! Multiple tasks need improvement by adaptation



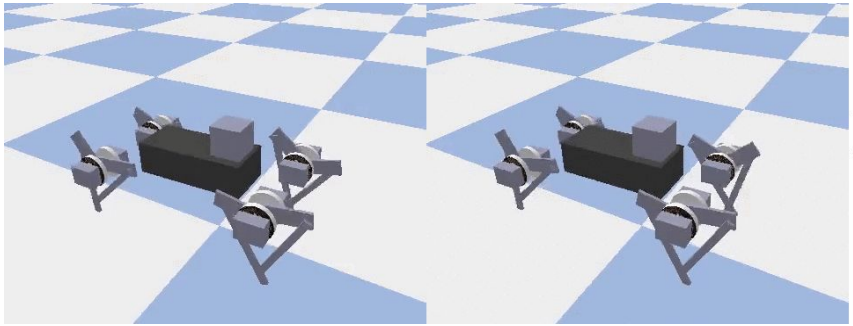
Simulation Results: Qualitative Changes

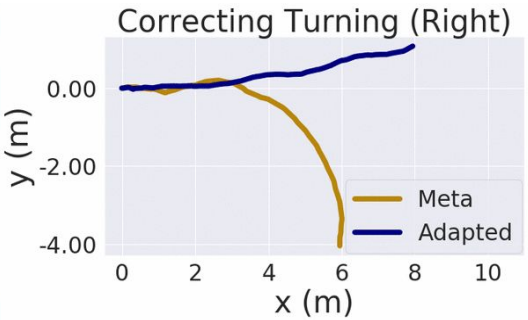
Correction from falling:





Correcting walking direction:



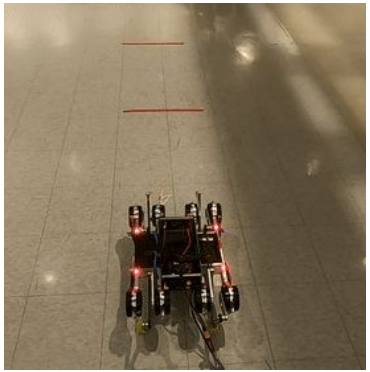
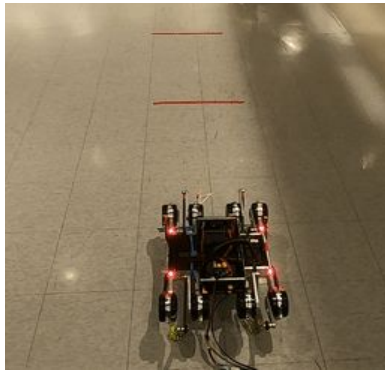
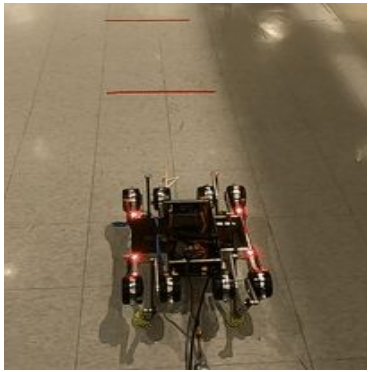
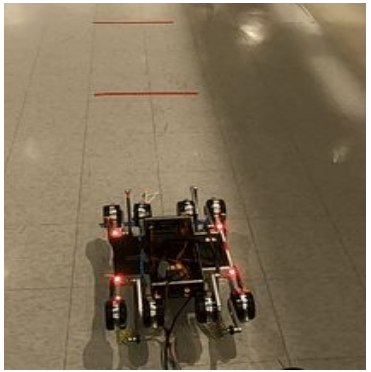


What about the noisy real world?

Adaptation in the noisy real world

When there is noise:

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

How do we modify hill-climbing?

Sequential Hill-Climbing

Sequential (Original):

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

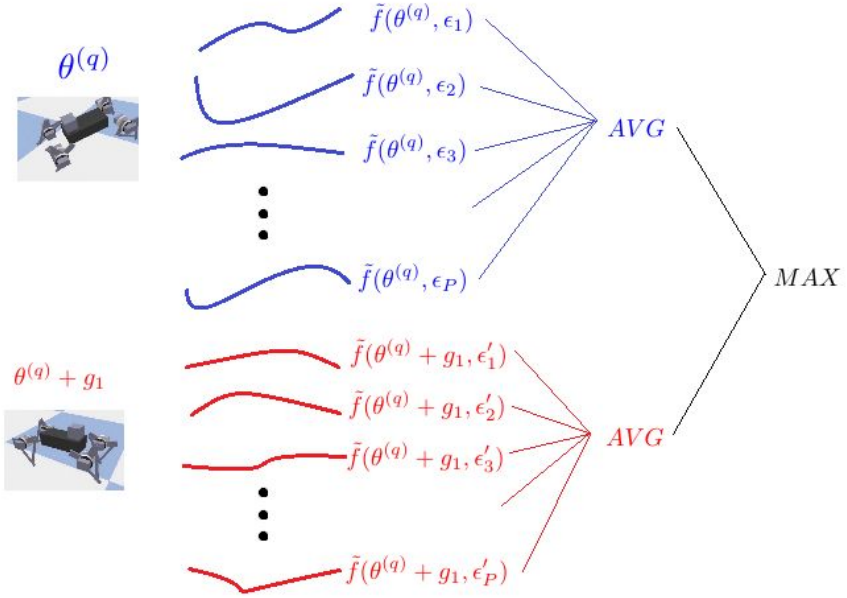
$$\theta^{(q+1)} = \underset{\theta \in \{\theta^{(q)}, \theta^{(q)} + \alpha \mathbf{g}\}}{\operatorname{argmax}} f(\theta)$$

$$\theta_{meta} \rightarrow \theta^{(1)} \rightarrow \ldots \rightarrow \theta^{(Q)}$$

Average Hill-Climbing

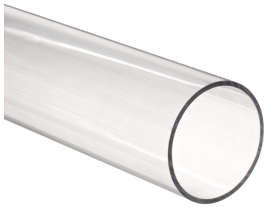
Average evaluation over *P* trials - Assumption of **expected objective**

- **Fails** when noise is:
  - **Not IID**. Ex: Robot motor overheats over time.
  - **Not zero mean**. Ex: Robot falls randomly
- **Low sample efficiency** - Multiple rollouts committed to single parameter
  - Need to know noise magnitude in advance



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

Understanding the Problem



- Allowed fixed number of noisy objective evaluations
  - Total Hill-Climb Trajectory = “Tube” of size Q*P
    - Q = “length”: number of proposed parameter changes
    - P = “thickness”: number of parallel evaluations
  - Also called *Multifidelity Optimization* in Hyperparameter Optimization
- Big Question: **How should we model noise?**
  - Roughly speaking, **we shouldn’t.**
    - Ends up being unrealistic + complicated
  - We should just assume it’s near **adversarial**.

Is the problem even solvable then?

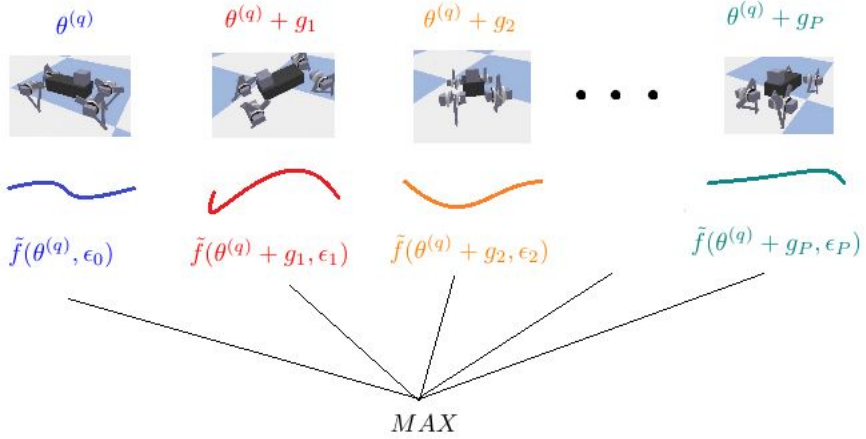
$$\widetilde{f}(\theta, \varepsilon) = f(\theta) + \varepsilon$$

Yes!

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!



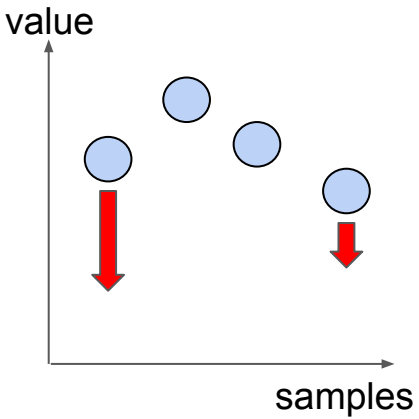
$$\theta^{(q+1)} = \underset{\theta \in \{\theta^{(q)}, \theta^{(q)} + \alpha \mathbf{g}_1, \ldots, \theta^{(q)} + \alpha \mathbf{g}_P\}}{\operatorname{argmax}} \tilde{f}(\theta, \varepsilon)$$

Intuitive Explanation

1. Suppose I sample *P* objectives
2. Nature negatively corrupts a fraction of these samples

Behavior of Operations:

- **Summation:** Even **one** sample can affect the outcome.
  - Easily affected by **magnitude of noise**
- **Argmax:** Affected only if argmax got chosen.
  - **Independent** of noise magnitude of neighbors.
  - Picking **second place** isn't bad either!



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

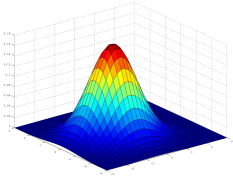
$f : \mathbb{R}^d \to \mathbb{R}$ *is a* (μ, ρ)*-strong concave function*

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

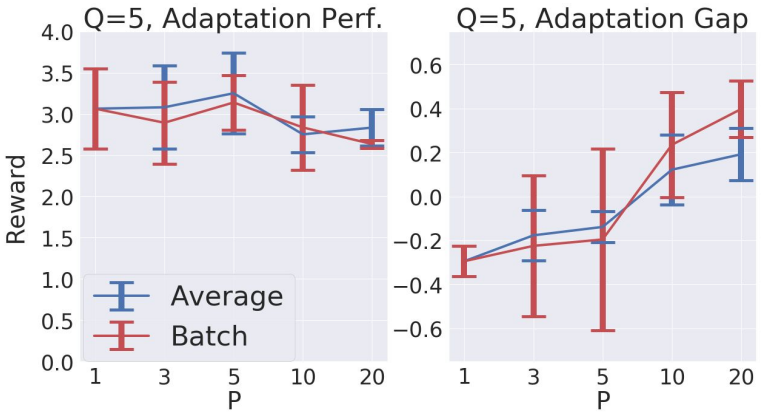
any constant satisfying:

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

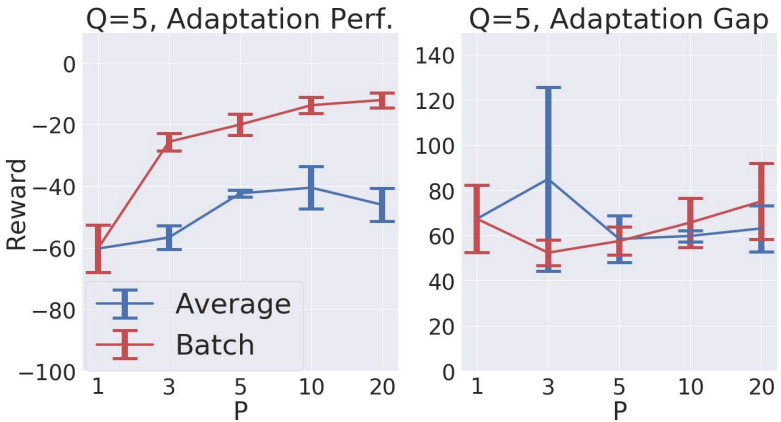
$$or\ |f(\theta^{\mathrm{opt}}) - f(\theta_i)| \leq D\phi\ for\ some\ \theta_i$$

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

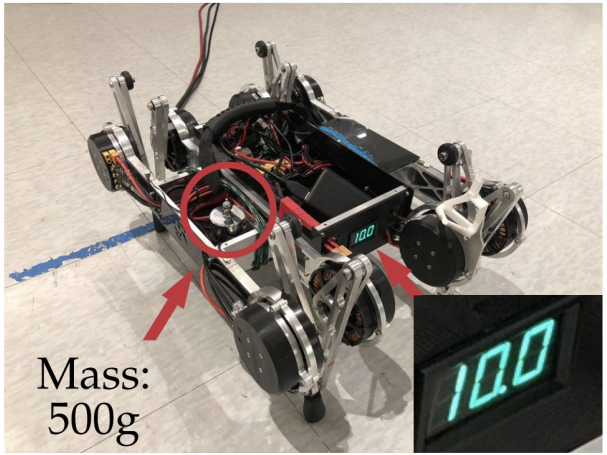


Noisy Nav-2D Environment

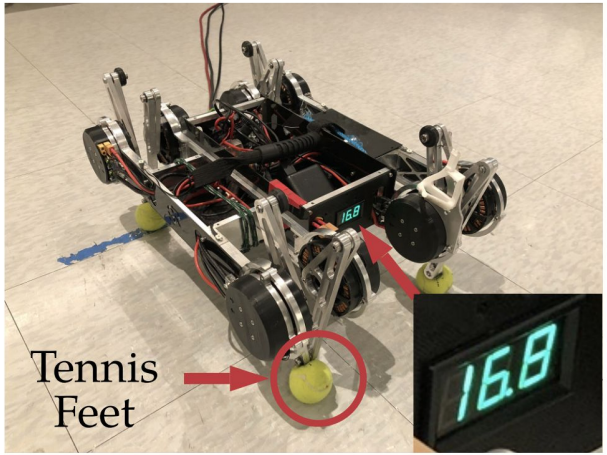
Real-Robot Experimental Ablations

Task Setup (Reminder)

Mass-Voltage Task

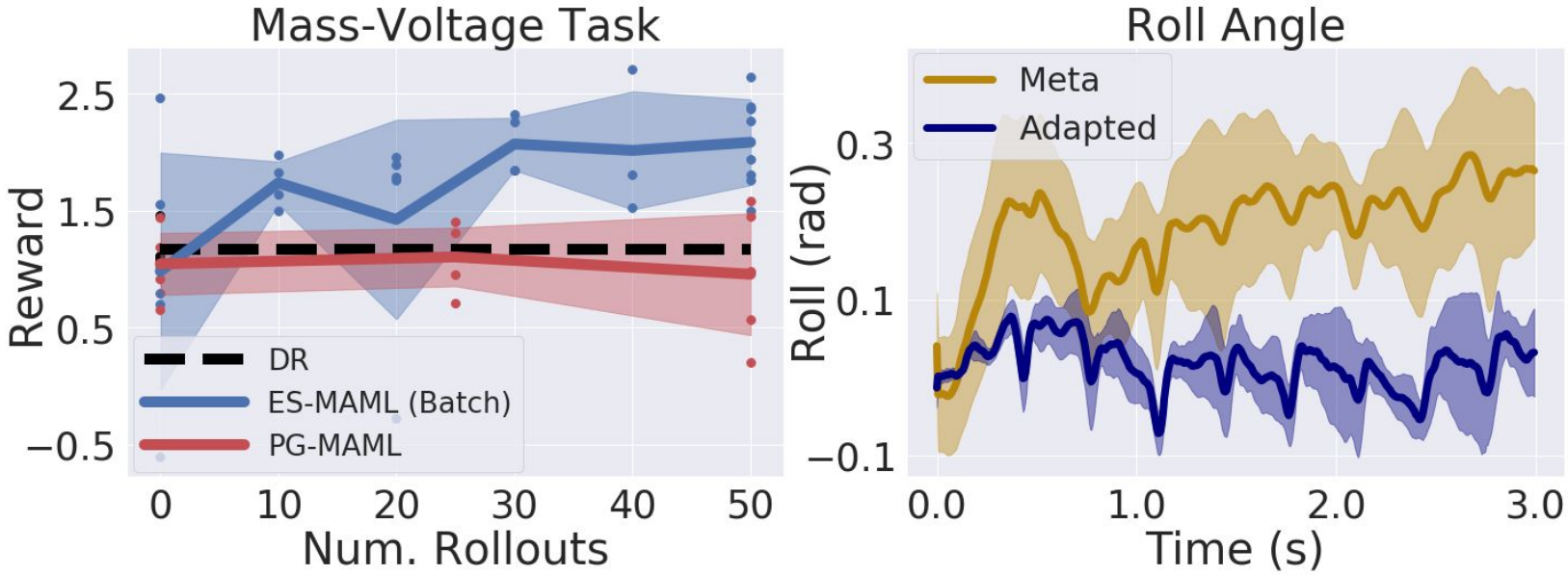


Friction Task



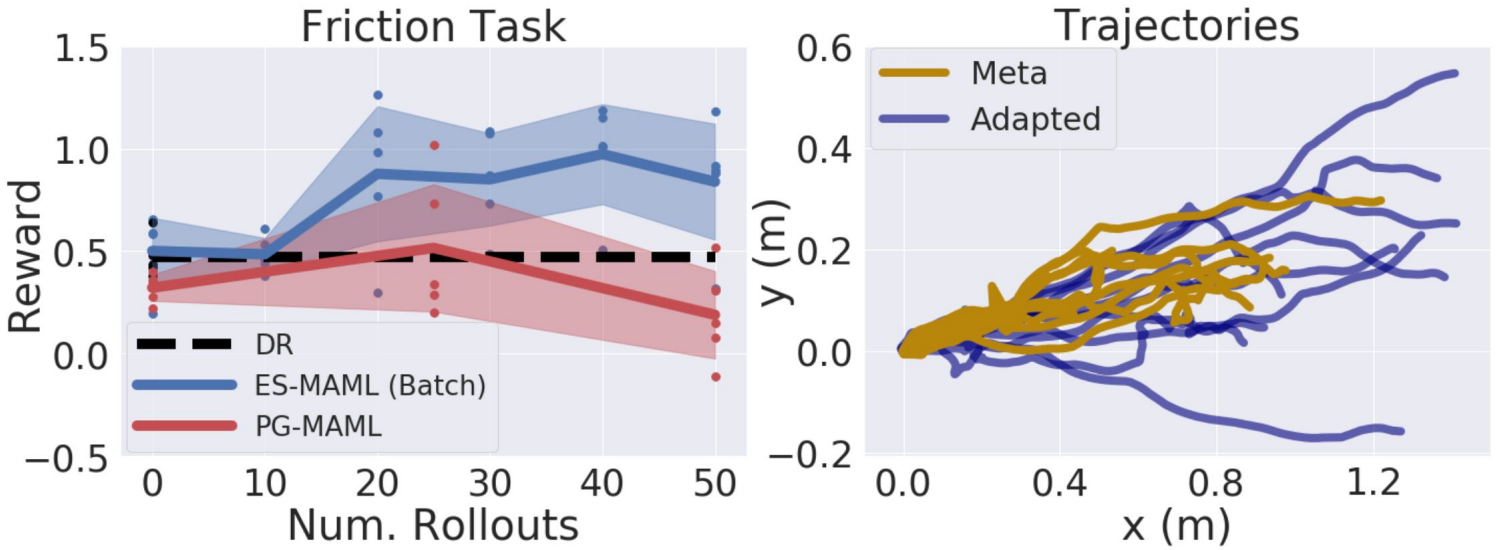
- Mass Voltage: 500g mass on side, voltage reduced to disrupt leg synchronization
- Friction: Tennis Balls on feet, to reduce gait via slipping.

Mass-Voltage Task



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

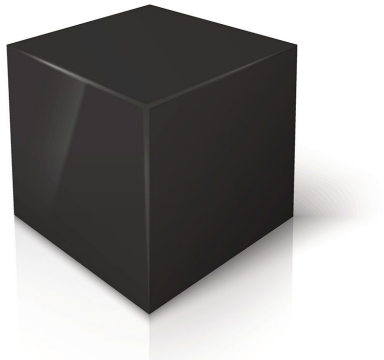
Friction Task



- ES-MAML outperforms PG-MAML and Domain Randomization (DR)
- ES-MAML produces longer trajectories.

Conclusion

- We have demonstrated one of the first **successful application of MAML on a challenging real robot task**.
- ES-MAML + Batch Hill-Climbing (our method) enables fast adaptation on robots.
  - Noise-resilient
  - Allows all the benefits of Zero-Order/Blackbox methods 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?
- Other applications of blackbox outer + inner loops
  - NAS, Genetic Programming, Hyperparameter Optimization, etc.

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!