Meta Reinforcement Learning
Adaptable Models & Policies

CS 224R

Reminders

Homework 3 due **Wednesday**

Project milestone due **next Wednesday**

Plan for Today

Meta-RL problem statement

Black-box meta-RL methods <- comes up in HW4

Optimization-based meta-RL methods

Next time: Learning to explore. <- part of HW4

Lecture goals:

- Understand the meta-RL problem statement & set-up
- Understand the basics of black-box meta RL algorithms
- Understand the basics & challenges of optimization-based meta RL algorithms

Problem Settings

Multi-Task Learning

Solve multiple tasks $\mathcal{T}_1, \cdots, \mathcal{T}_T$ at once.

$$\min_\theta \sum_{i=1}^{T} \mathcal{L}_i(\theta, \mathcal{D}_i)$$

Transfer Learning

Solve target task $\mathcal{T}_b$ after solving source task $\mathcal{T}_a$ by ***transferring*** knowledge learned from $\mathcal{T}_a$

The Meta-Learning Problem

Given data from $\mathcal{T}_1, \ldots, \mathcal{T}_n$, quickly solve new task $\mathcal{T}_{\text{test}}$

In all settings: tasks must share structure.

A reinforcement learning task:

action space   dynamics

$$\mathcal{T}_i \triangleq \{\mathcal{S}_i, \mathcal{A}_i, p_i(\mathbf{s}_1), p_i(\mathbf{s}'|\mathbf{s}, \mathbf{a}), r_i(\mathbf{s}, \mathbf{a})\}$$

state space   initial state distribution   reward

Meta-reinforcement learning = meta-learning with RL tasks

The Meta-Learning Problem

Supervised Learning:

Inputs: $\mathbf{x}$ Outputs: $\mathbf{y}$ Data: $\{(\mathbf{x}, \mathbf{y})_i\}$

$$\mathbf{y} = g_\phi(\mathbf{x})$$

Meta Supervised Learning:

Inputs: $\underbrace{\mathcal{D}^{\mathrm{tr}}}_{\{(\mathbf{x}, \mathbf{y})_{1:K}\}}$ $\mathbf{x}^{\mathrm{ts}}$ Outputs: $\mathbf{y}^{\mathrm{ts}}$ Data: $\{\mathcal{D}_i\}$

$$\mathbf{y}^{\mathrm{ts}} = f_\theta(\mathcal{D}^{\mathrm{tr}}, \mathbf{x}^{\mathrm{ts}})$$

$$\mathcal{D}_i : \{(\mathbf{x}, \mathbf{y})_j\}$$

Why is this view useful?

Reduces the meta-learning problem to the design & optimization of f.

Finn. *Learning to Learn with Gradients*. PhD Thesis. 2018

The Meta Reinforcement Learning Problem

Reinforcement Learning:

Inputs: $\cancel{\mathbf{x}}\ \mathbf{s}_t$ Outputs: $\cancel{\mathbf{y}}\ \mathbf{a}_t$

$$\cancel{\mathbf{y} = g_\phi(\mathbf{x})}$$

$$\mathbf{a}_t = \pi(\mathbf{s}_t; \theta)$$

Data: $\cancel{\{(\mathbf{x}, \mathbf{y})_i\}}$

$$\{(\mathbf{s}_t, \mathbf{a}_t, r_t, \mathbf{s}_{t+1})\}$$

Meta Reinforcement Learning:

Inputs: $\underbrace{\mathcal{D}^{\mathrm{tr}}}_{\text{k rollouts from } \pi}$ $\mathbf{s}_t$ Outputs: $\mathbf{a}_t$

$$\mathbf{a}_t = f_\theta(\mathcal{D}^{\mathrm{tr}}, \mathbf{s}_t)$$

Data: $\{\mathcal{D}_i\}$

dataset of datasets
collected for each task

Design & optimization of *f*   *and*   collecting appropriate data

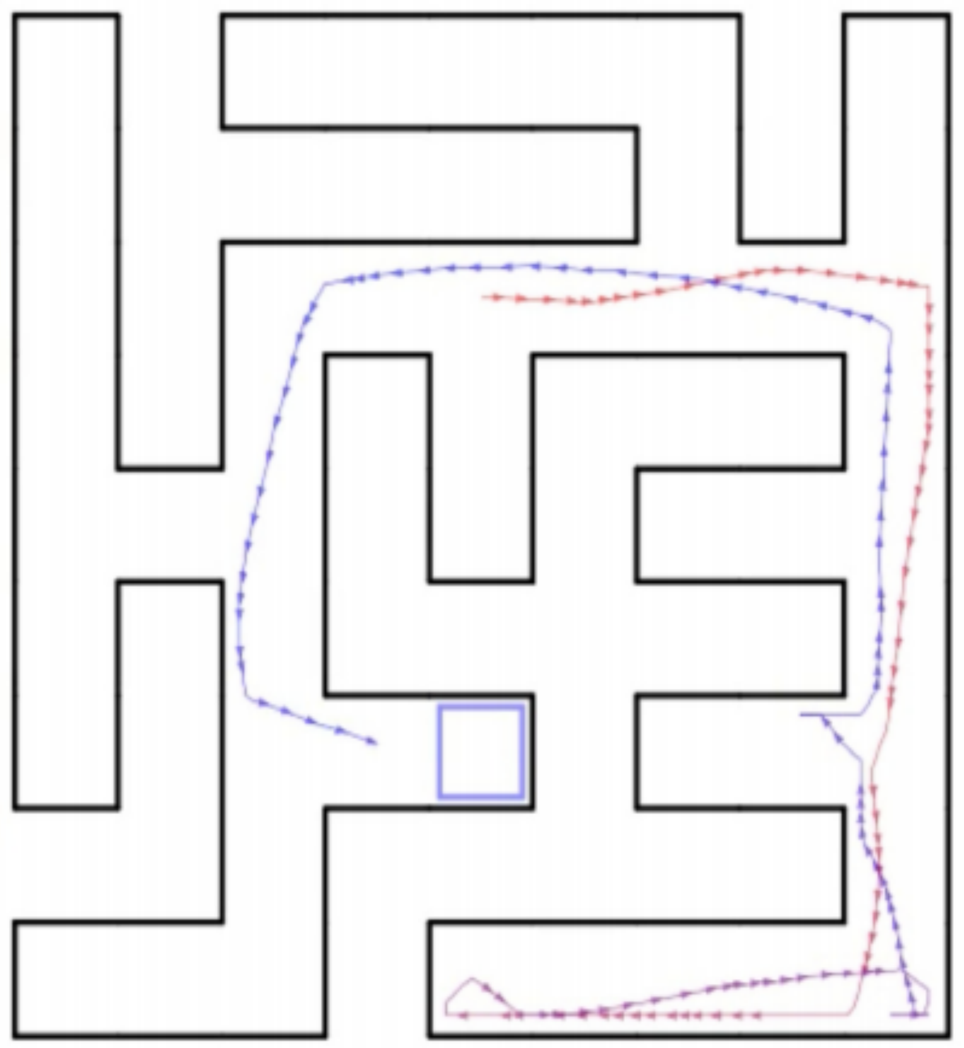
(learning to explore)

Finn. *Learning to Learn with Gradients*. PhD Thesis. 2018

Meta-RL Example: Maze Navigation

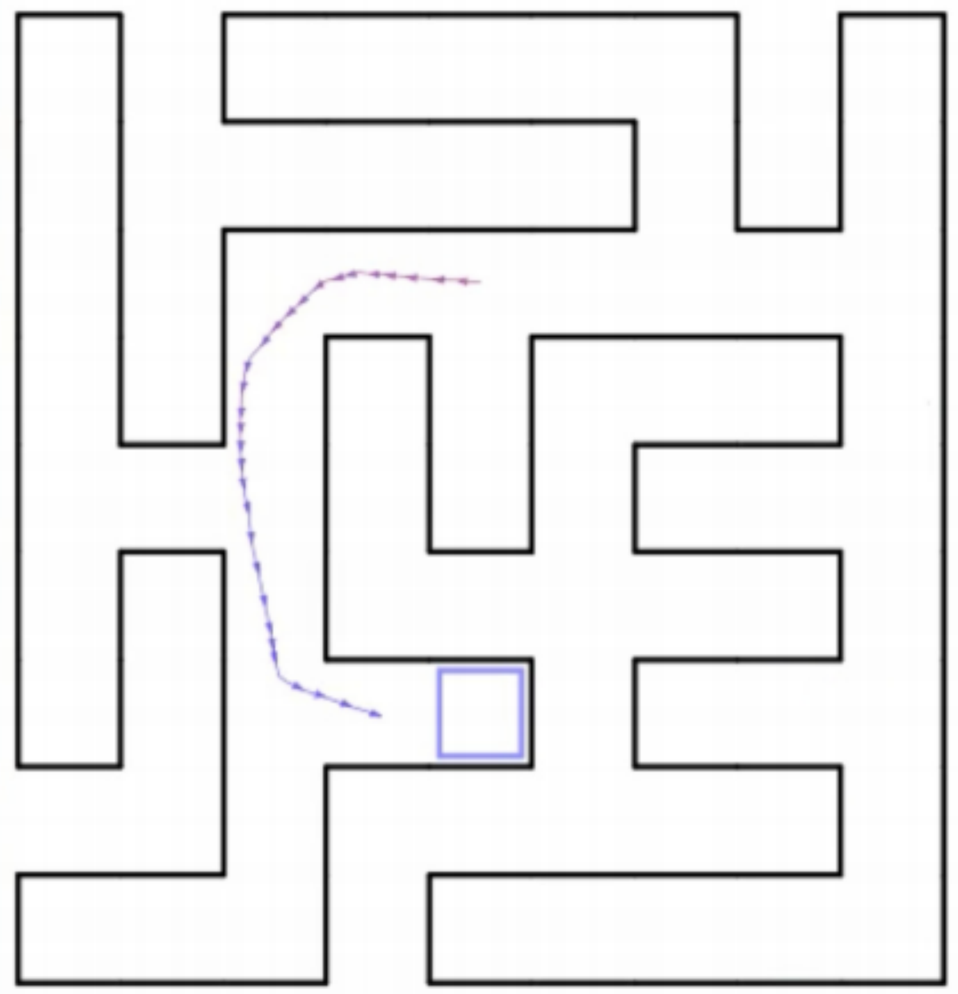
Goal:

Collect small amount of experience in new MDP



Collect $\mathscr{D}_{\mathrm{tr}} \sim \pi^{\mathrm{exp}}$

Learn policy that solves that MDP



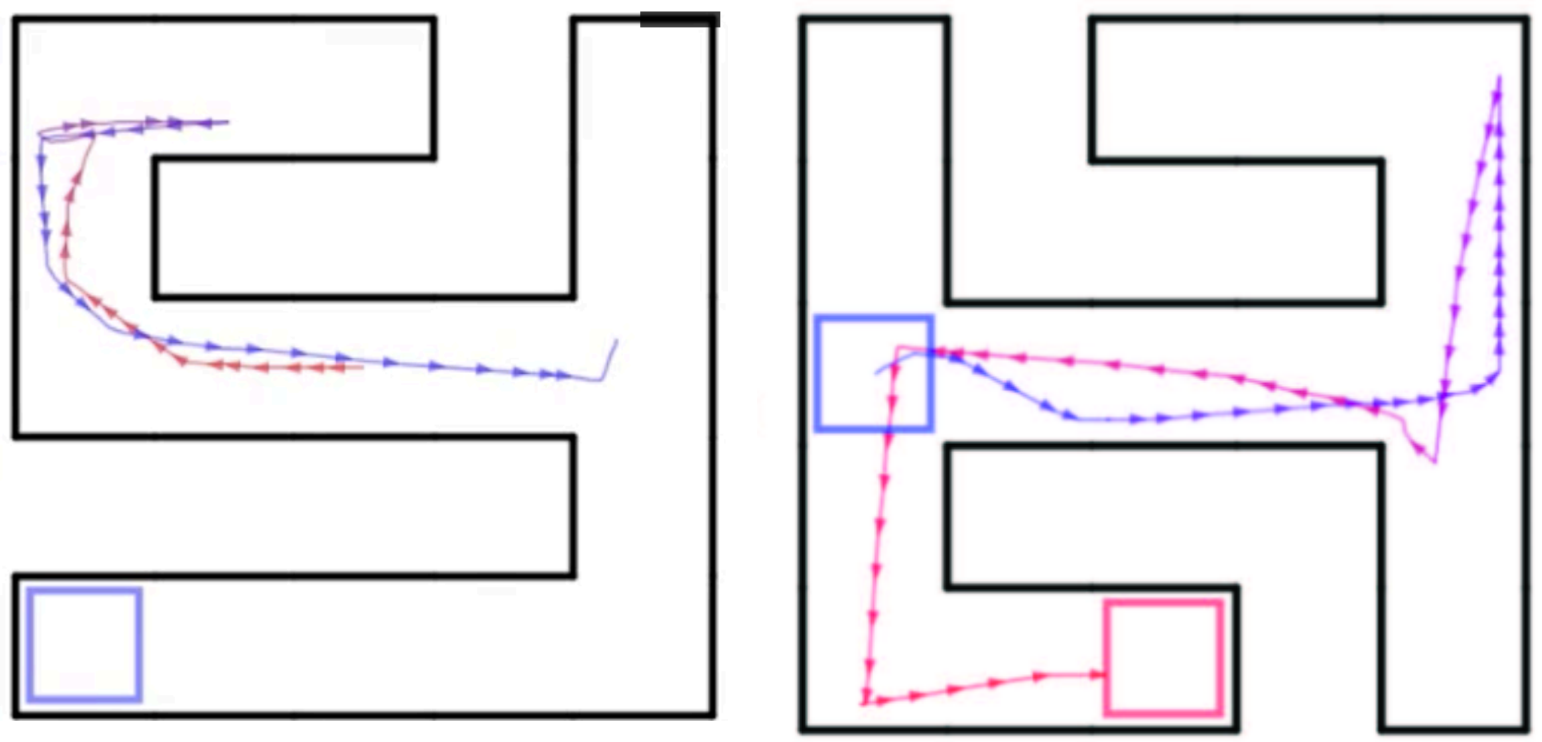
$\mathscr{D}_{\mathrm{tr}} \rightarrow \pi^{\mathrm{task}}$

diagram adapted from Duan et al. '17

Meta-RL Example: Maze Navigation

Meta-Train Time:

Learn how to efficiently explore & solve many MDPs:

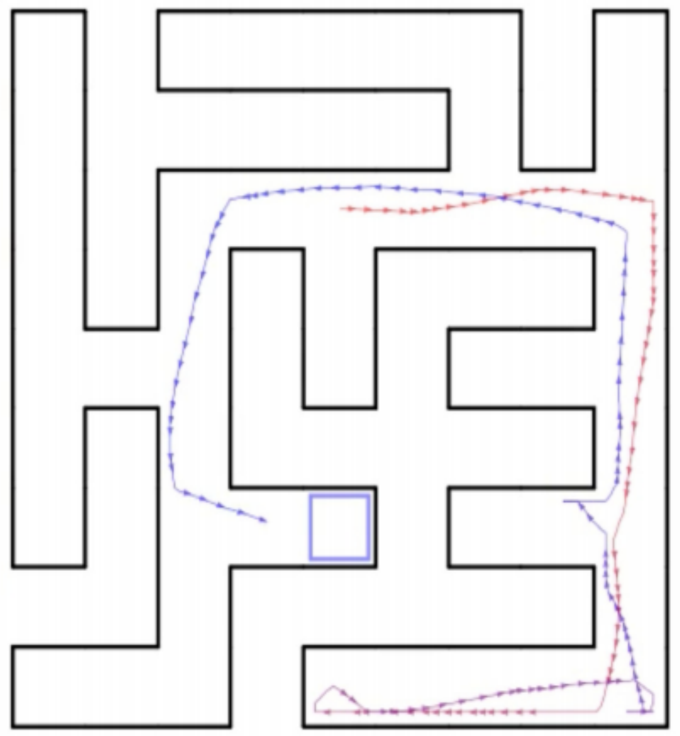


. . . meta-training tasks

Meta-train $\pi^{\text{exp}}, \pi^{\text{task}}$

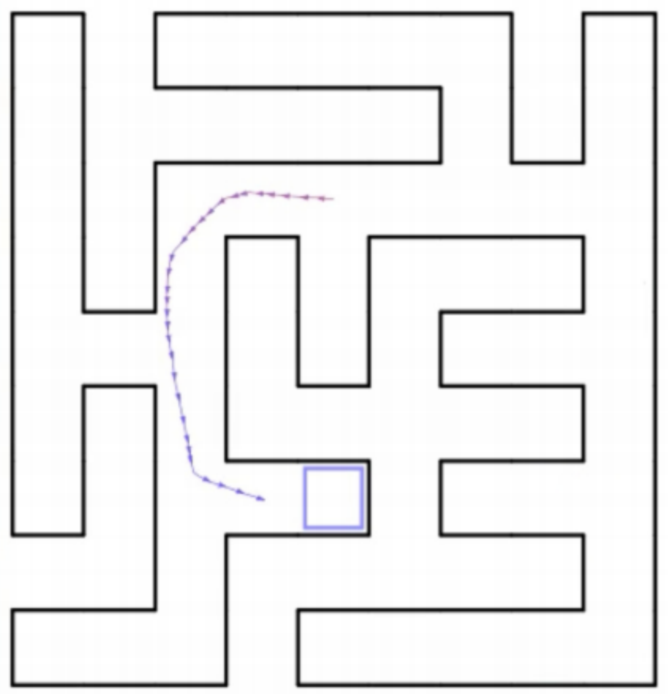
Meta-Test Time:

Collect small amount of experience in new MDP



Collect $\mathscr{D}_{\text{tr}} \sim \pi^{\text{exp}}$

Learn policy that solves that MDP



$\mathscr{D}_{\text{tr}} \to \pi^{\text{task}}$

Key assumption: Meta-training & meta-testing MDPs come from same distribution.

(so that we can expect generalization)

diagram adapted from Duan et al. ‘17

The Meta Reinforcement Learning Problem

Meta Reinforcement Learning:

Episodic Variant

Inputs: $\underbrace{\mathcal{D}_{\text{train}}}_{\text{k rollouts from } \pi}$ $\mathbf{s}_t$ Outputs: $\mathbf{a}_t$

$$\mathbf{a}_t = f_\theta(\mathcal{D}^{\text{tr}}, \mathbf{s}_t)$$

Online Variant

Inputs: $\underbrace{\mathcal{D}_{\text{train}}}_{1\ldots\text{k timesteps from } \pi}$ $\mathbf{s}_t$ Outputs: $\mathbf{a}_t$

$$\mathbf{a}_t = f_\theta(\mathcal{D}^{\text{tr}}, \mathbf{s}_t)$$

Note: exploration policy π and adaptation policy f_θ need not be the same.

Plan for Today

Meta-RL problem statement

Black-box meta-RL methods

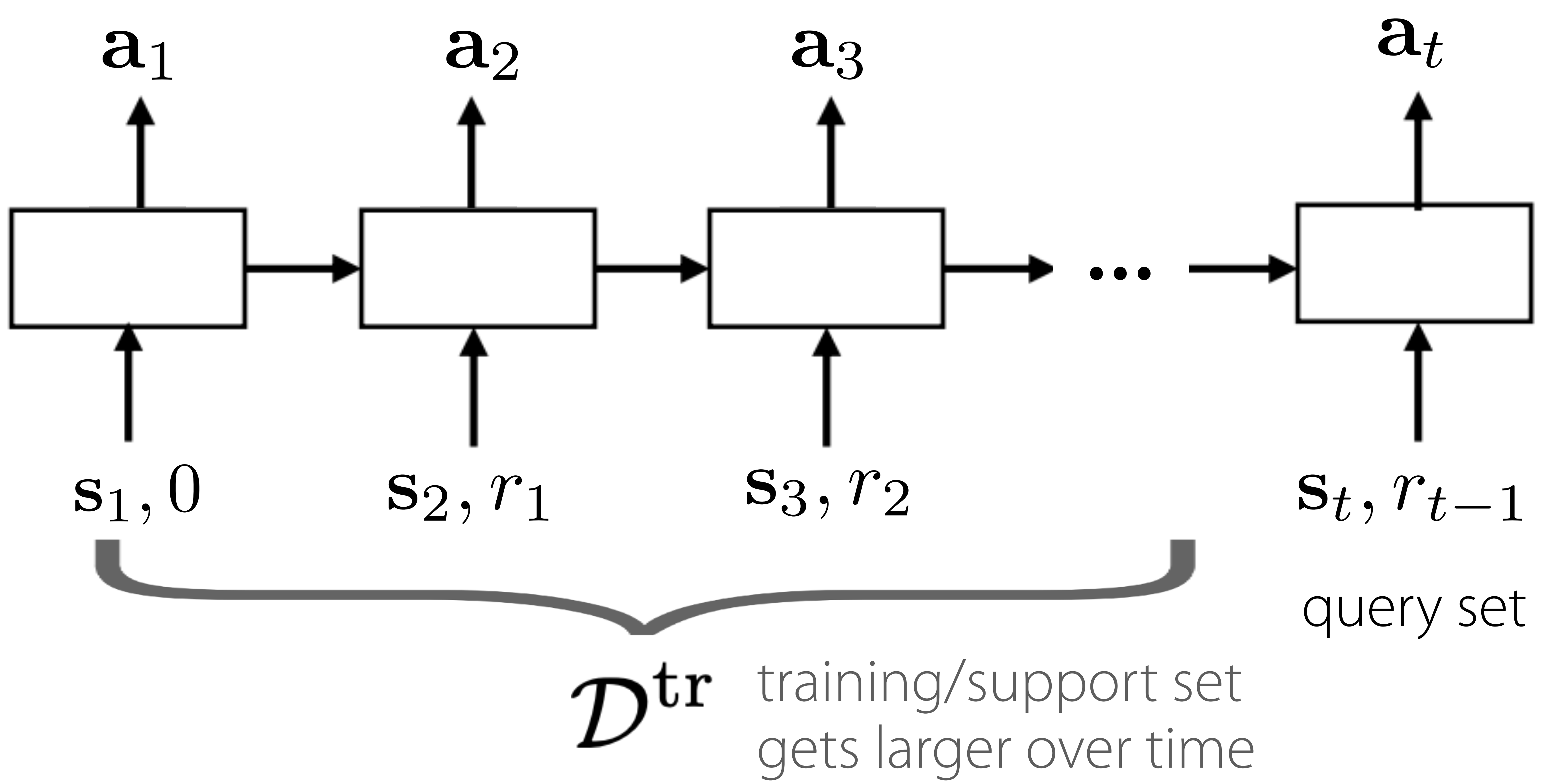
Optimization-based meta-RL methods

Black-Box Meta-RL: Overview

Black-box network

(LSTM, NTM, Conv, ...)

$$\mathbf{a}_t = f_\theta(\mathcal{D}^{\text{tr}}, \mathbf{s}_t)$$



Question: Why don't we need to pass in the actions $\mathbf{a}_{t-1}$ with the support set?

Question: How is this different from simply doing RL with a recurrent policy?

Reward is passed as input
(& trained across multiple MDPs)

Hidden state maintained
across episodes within a task!

Black-Box Meta-RL: Algorithm

1. Sample task $\mathcal{T}_i$
2. Roll-out policy $\pi(a \mid s, \mathcal{D}_i^{\mathrm{tr}})$ for N episodes

   (under dynamics $p_i(s' \mid s, a)$ and reward $r_i(s, a)$)
3. Store sequence in replay buffer for task $\mathcal{T}_i$.
4. Update policy to maximize discounted return for all tasks.

Black-Box Meta-RL: Algorithm

Meta-Training

1. Sample task $\mathcal{T}_i$
2. Roll-out policy $\pi(a|s, \mathcal{D}_i^{\mathrm{tr}})$ for N episodes

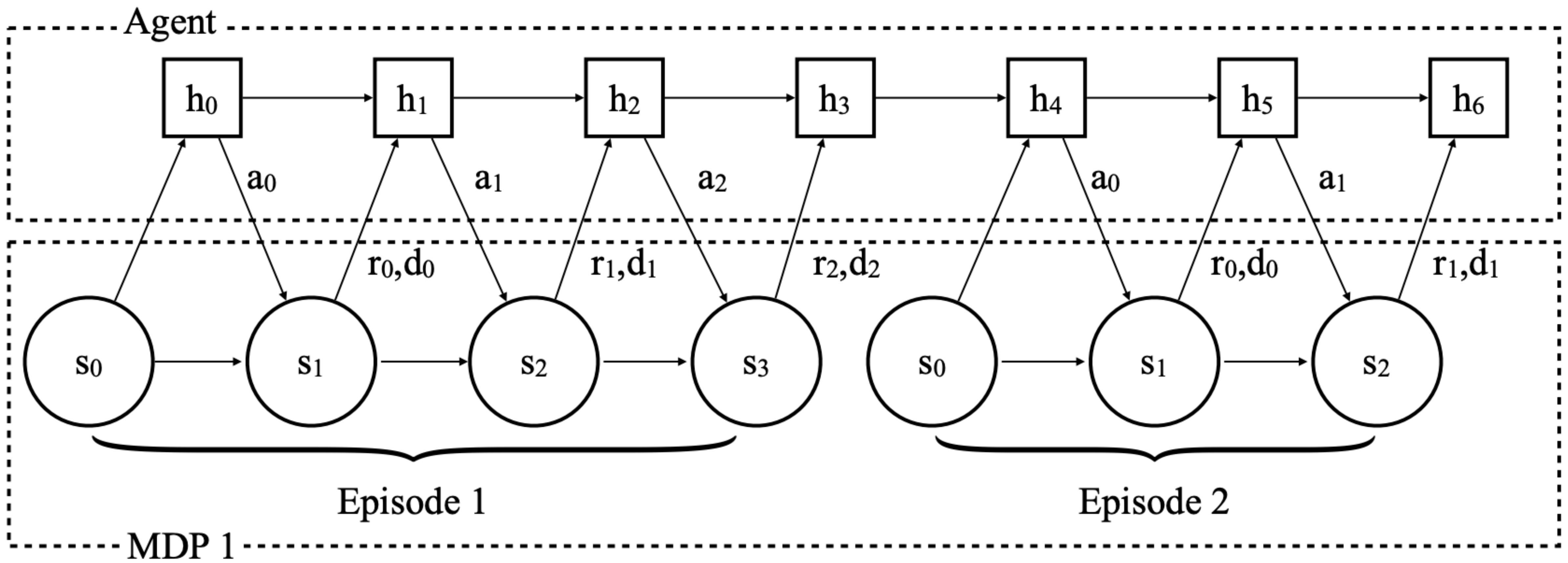
   (under dynamics $p_i(s'|s,a)$ and reward $r_i(s,a)$)
3. Store sequence in replay buffer for task $\mathcal{T}_i$.
4. Update policy to maximize discounted return for all tasks.

Meta-Test Time

1. Sample *new* task $\mathcal{T}_j$
2. Roll-out policy $\pi(a|s, \mathcal{D}_j^{\mathrm{tr}})$ for up to N episodes

Black-Box Meta-RL: Architectures & Optimizers

RNN architecture TRPO/A3C (on-policy)



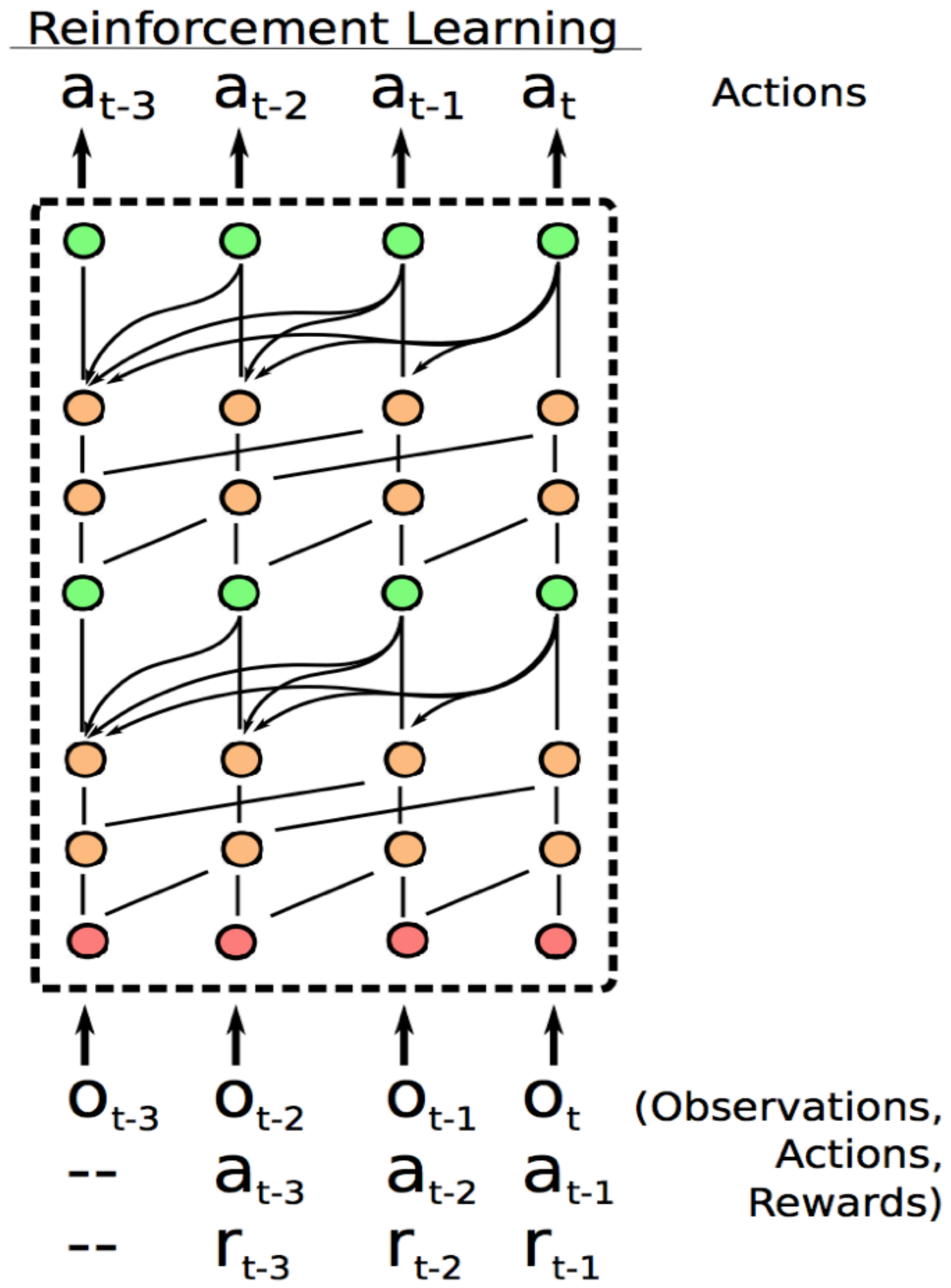
Duan, Schulman, Chen, Bartlett, Sutskever, Abbeel. *RL²: Fast Reinforcement Learning via Slow Reinforcement Learning*. 2017

Wang, Kurth-Nelson, Tirumala, Soyer, Leibo, Munos, Blundell, Kumaran, Botvinick. *Learning to Reinforcement Learn*. CogSci 2017

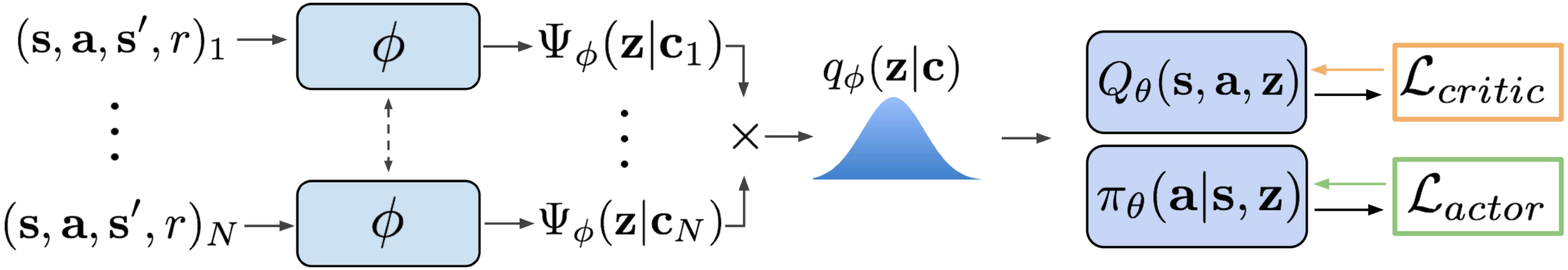
Attention + 1D conv

TRPO (on-policy)

Mishra, Rohaninejad, Chen, Abbeel. *A Simple Neural Attentive Meta-Learner*. ICLR 2018



Feedforward + average SAC (off-policy)



Rakelly, Zhou, Quillen, Finn, Levine. *Efficient Off-Policy Meta-Reinforcement Learning via Probabilistic Context Variables*. ICML 2019.

Meta-RL Example #1

From: Mishra, Rohaninejad, Chen, Abbeel. ***A Simple Neural Attentive Meta-Learner***. ICLR 2018

Experiment: Learning to visually navigate a maze
- train on 1000 small mazes
- test on held-out small mazes and large mazes

Meta-RL Example #1

From: Mishra, Rohaninejad, Chen, Abbeel. ***A Simple Neural Attentive Meta-Learner***. ICLR 2018

Experiment: Learning to visually navigate a maze

- train on 1000 small mazes
- test on held-out small mazes and large mazes

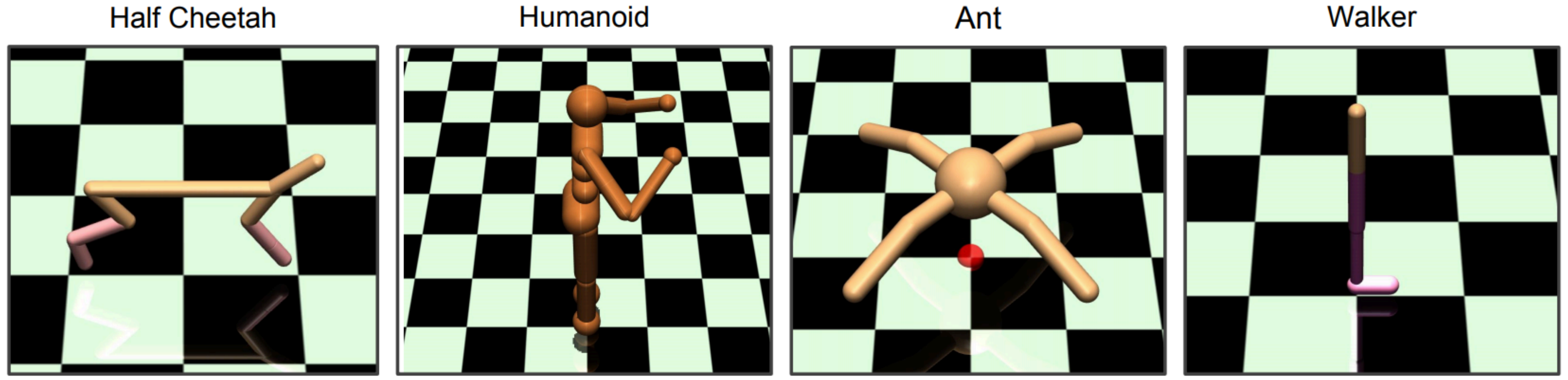
Method	Small Maze		Large Maze	
	Episode 1	Episode 2	Episode 1	Episode 2
Random	188.6 ± 3.5	187.7 ± 3.5	420.2 ± 1.2	420.8 ± 1.2
LSTM	52.4 ± 1.3	39.1 ± 0.9	180.1 ± 6.0	150.6 ± 5.9
SNAIL (ours)	$\mathbf{50.3 \pm 0.3}$	$\mathbf{34.8 \pm 0.2}$	$\mathbf{140.5 \pm 4.2}$	$\mathbf{105.9 \pm 2.4}$

Table 5: Average time to find the goal on each episode

Meta-RL Example #2

Rakelly, Zhou, Quillen, Finn, Levine. ***Efficient Off-Policy Meta-Reinforcement Learning via Probabilistic Context Variables.*** ICML 2019.

Experiment: Continuous control problems

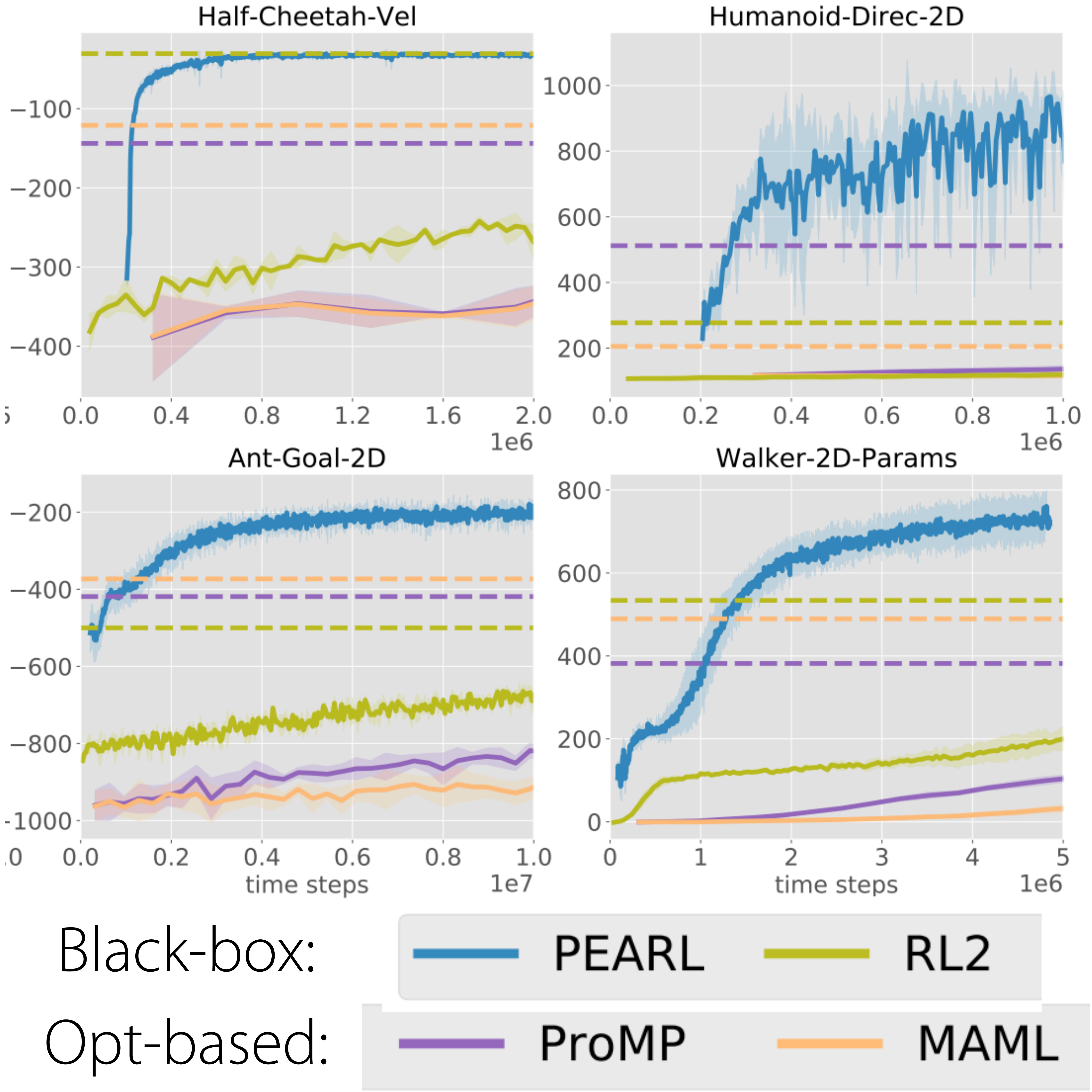


- different directions, velocities
- different physical dynamics

Meta-RL algos are very efficient at new tasks.

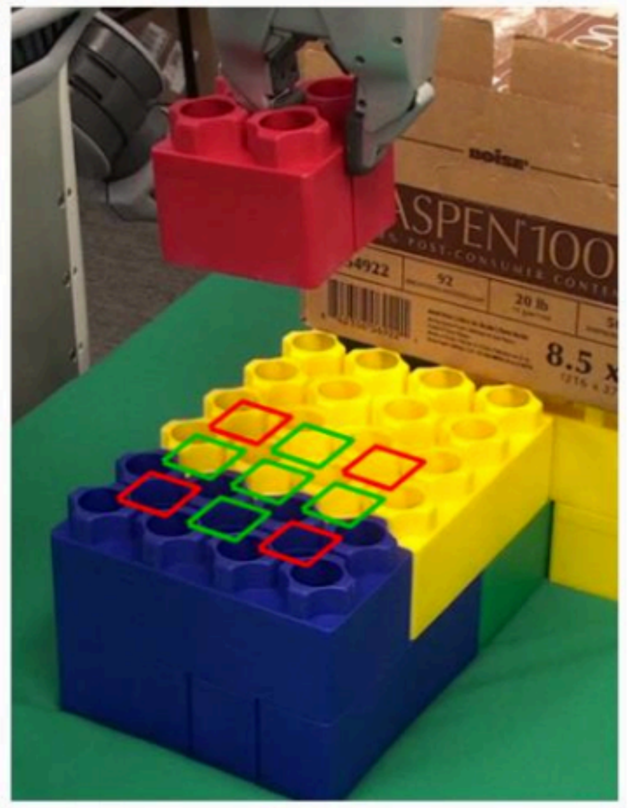
But, what about **meta-training efficiency**?

Question: Do you expect off-policy meta-RL to be more or less efficient than on-policy meta-RL?

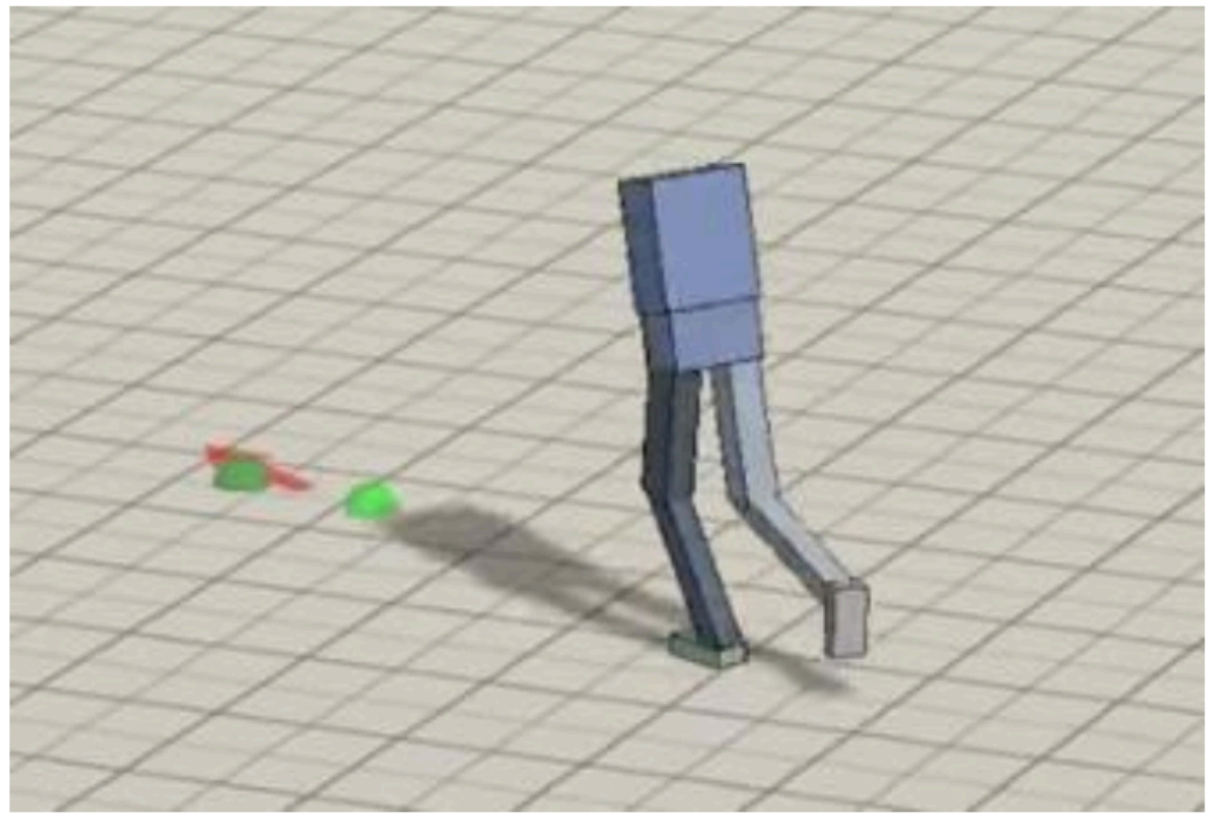


Digression: Connection to Multi-Task Policies

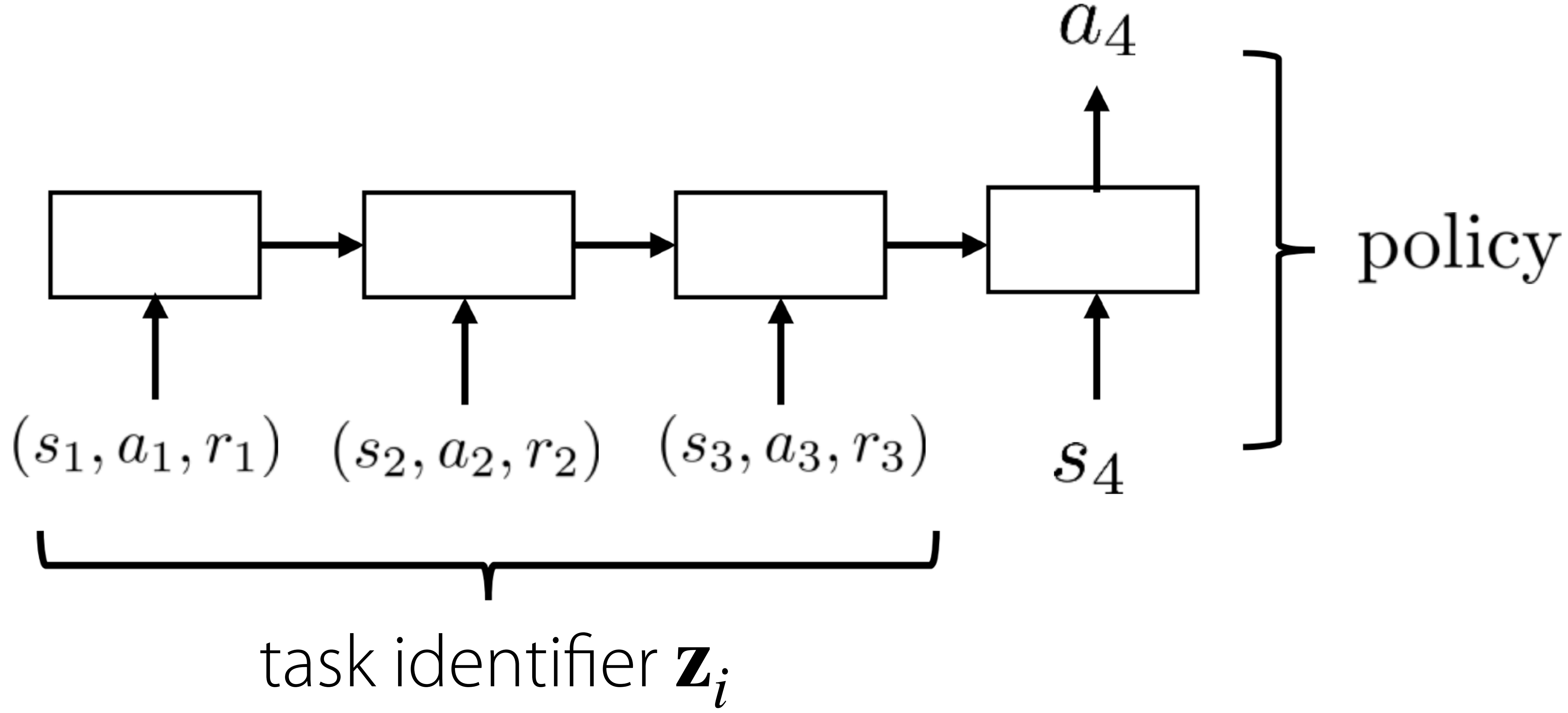
multi-task policy: $\pi_\theta(\mathbf{a} \mid \mathbf{s}, \mathbf{z}_i)$



$\mathbf{z}_i$: stack location



$\mathbf{z}_i$: walking direction



Multi-task policy with experience as task identifier.

What about **goal-conditioned policies** / **value functions**?

- rewards are a strict generalization of goals
- meta-RL objective is to ***adapt*** new tasks vs. ***generalize*** to new goals

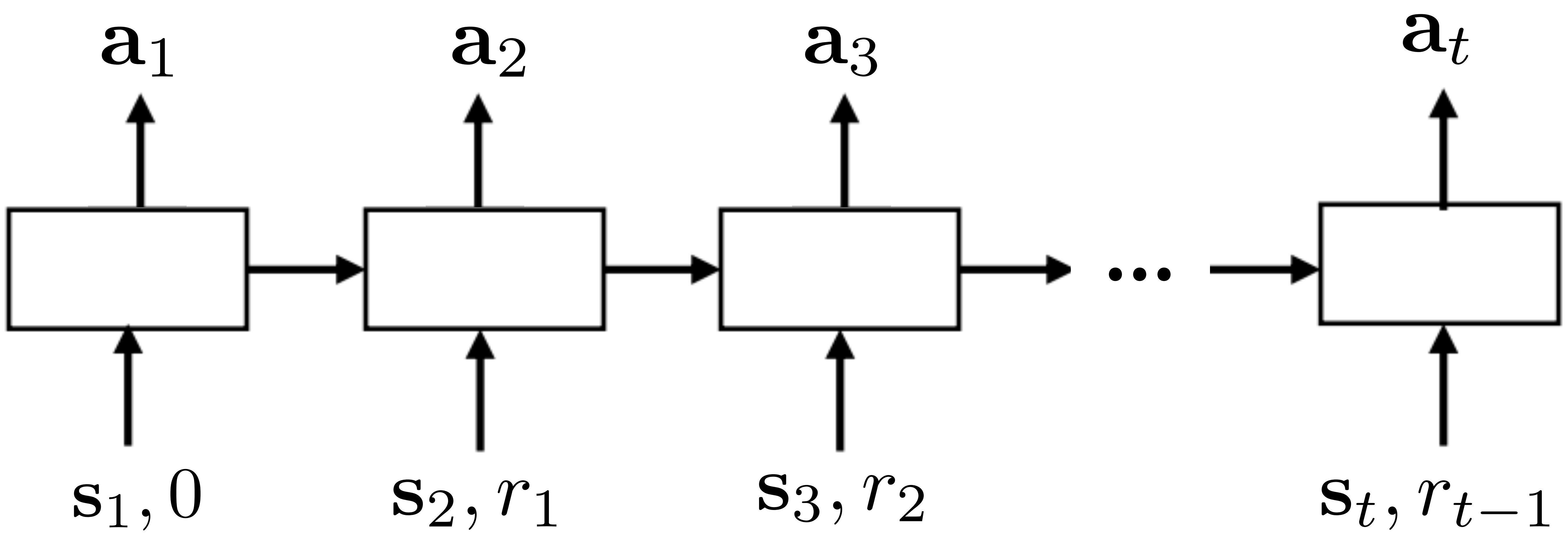
(**k-shot** vs. **0-shot**)

Black-Box Meta-RL Summary

Black-box network

(LSTM, NTM, Conv, …)

$$\mathbf{a}_t = f(\mathcal{D}_{\text{train}}, \mathbf{s}_t; \theta)$$



+ general & expressive
+ a variety of design choices in architecture

-- hard to optimize

~ inherits sample efficiency from outer RL optimizer

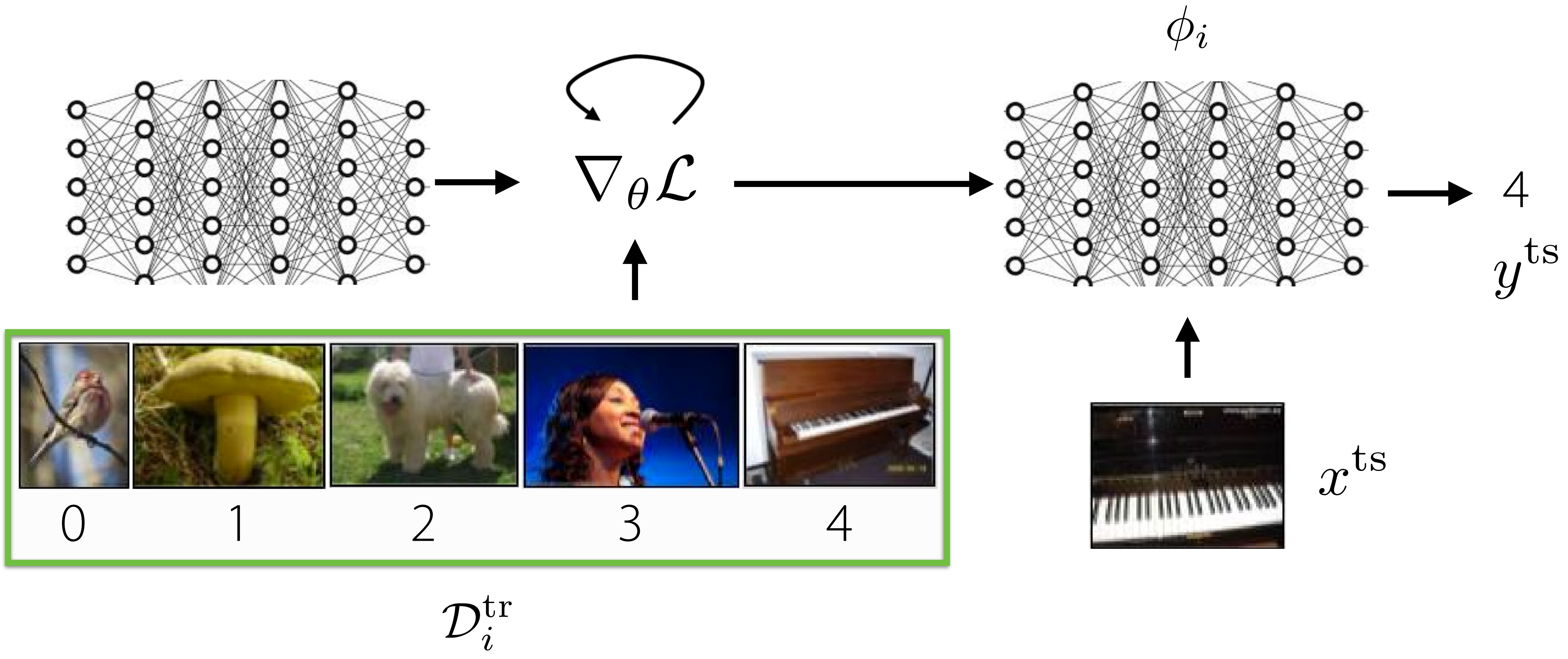
Plan for Today

Meta-RL problem statement

Black-box meta-RL methods

Optimization-based meta-RL methods

Optimization-Based Meta-Learning



Key idea: embed optimization inside the inner learning process

Fine-tuning

pre-trained parameters

Fine-tuning $\phi \leftarrow \theta - \alpha \nabla_{\theta} \mathcal{L}(\theta, \mathcal{D}^{\text{tr}})$

training data
for new task

(typically for many gradient steps)

Universal Language Model Fine-Tuning for Text Classification. Howard, Ruder. '18

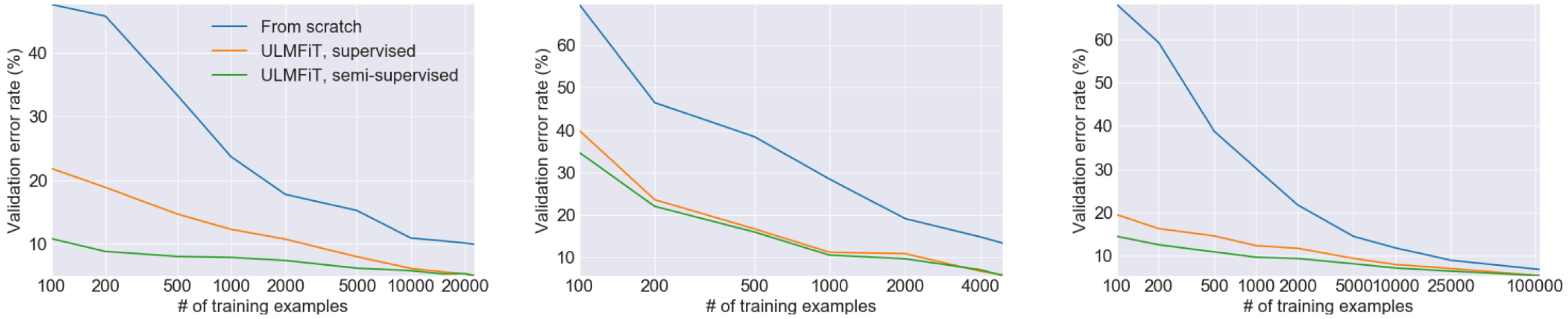


Figure 3: Validation error rates for supervised and semi-supervised ULMFiT vs. training from scratch with different numbers of training examples on IMDb, TREC-6, and AG (from left to right).

Fine-tuning less effective with **very small datasets**.

Optimization-Based Meta-Learning

pre-trained parameters

Fine-tuning

[test-time]

$$\phi \leftarrow \boldsymbol{\theta} - \alpha \nabla_\theta \mathcal{L}(\theta, \mathcal{D}^{\mathrm{tr}})$$

training data
for new task

Meta-learning $$\min_\theta \sum_{\text{task } i} \mathcal{L}(\theta - \alpha \nabla_\theta \mathcal{L}(\theta, \mathcal{D}_i^{\mathrm{tr}}), \mathcal{D}_i^{\mathrm{ts}})$$

Key idea: Over many tasks, learn parameter vector θ that transfers via fine-tuning

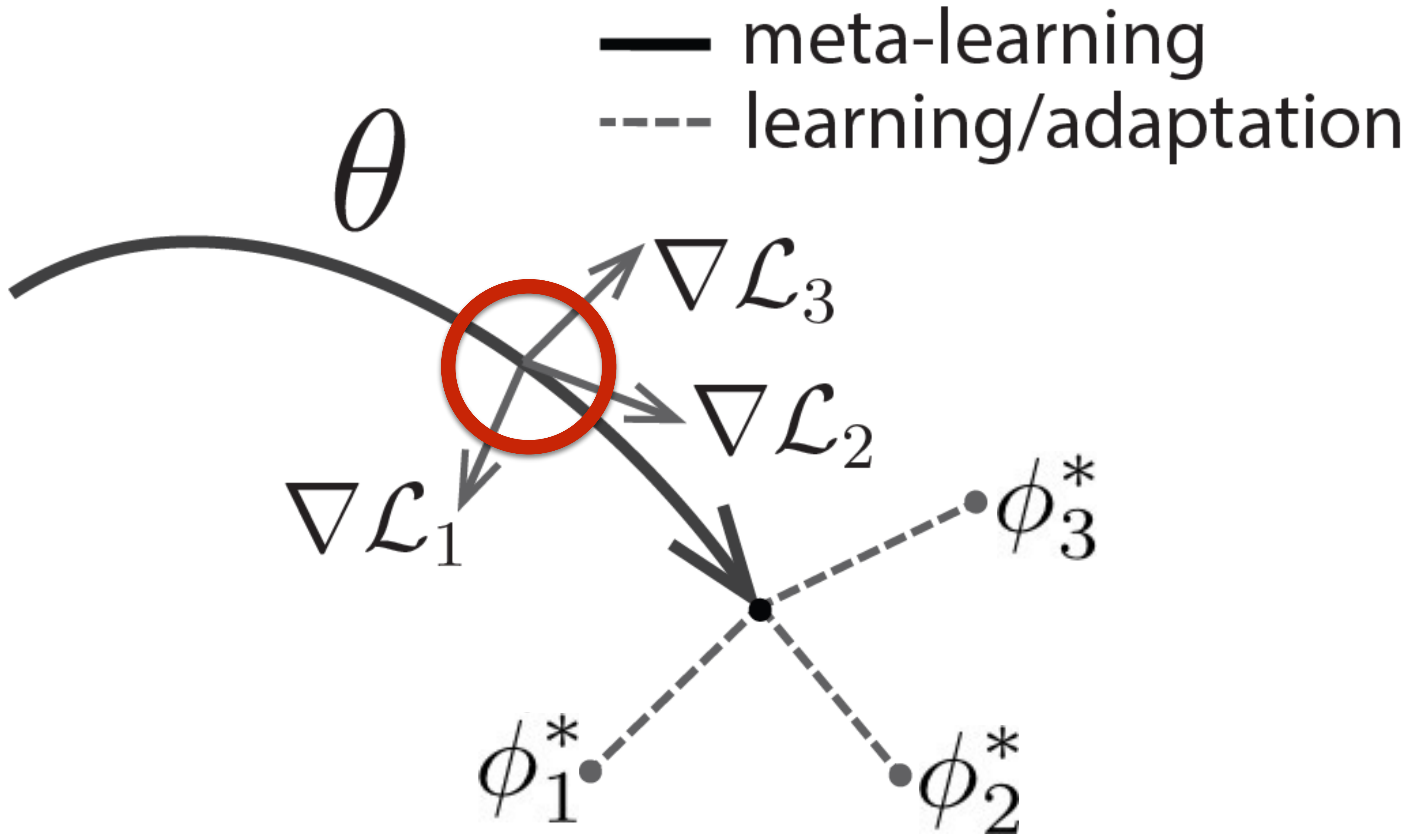
Finn, Abbeel, Levine. Model-Agnostic Meta-Learning. ICML 2017

Optimization-Based Meta-Learning

$$\min_{\theta} \sum_{\text{task } i} \mathcal{L}(\theta - \alpha \nabla_{\theta} \mathcal{L}(\theta, \mathcal{D}_i^{\text{tr}}), \mathcal{D}_i^{\text{ts}})$$

θ parameter vector being meta-learned

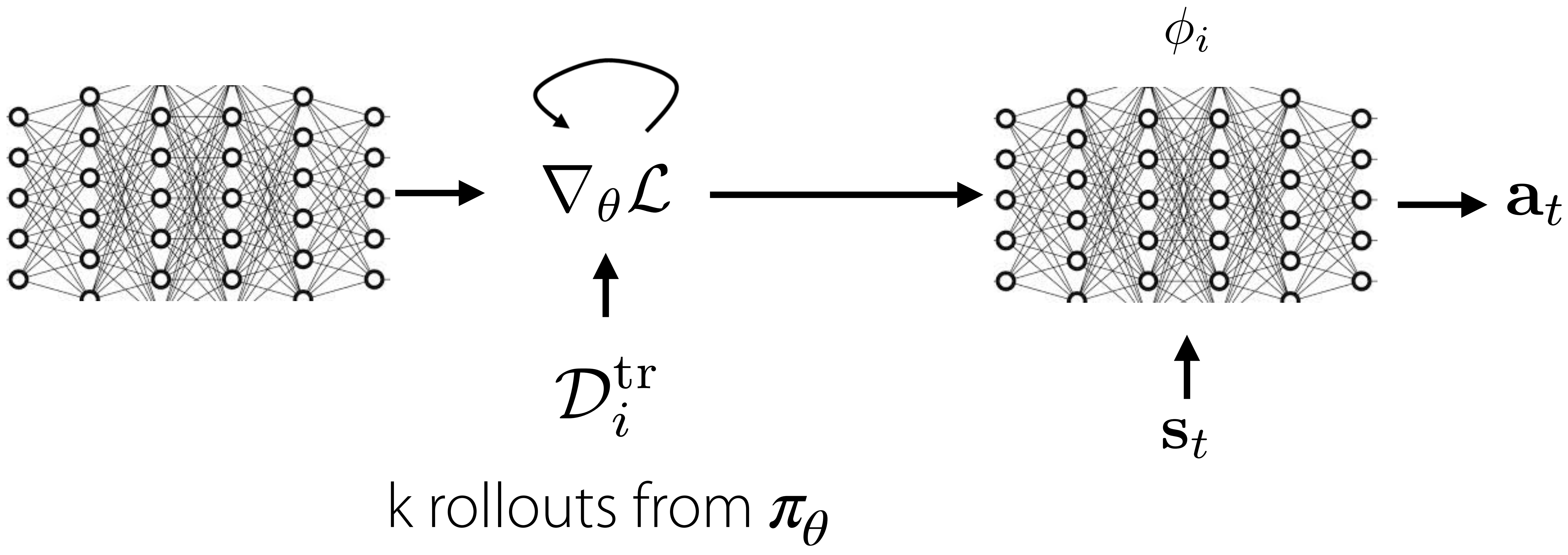
ϕ_i^* optimal parameter vector for task i



Model-Agnostic Meta-Learning

Finn, Abbeel, Levine. Model-Agnostic Meta-Learning. ICML 2017

Optimization-Based ~~Meta-Learning~~ Meta-RL



Key idea: embed optimization inside the inner learning process

Question: What should we use for the inner optimization and why?

Policy gradients?	Q-learning?	Model-based RL?
+ gradient-based!	- dynamic programming (requires many steps)	+ gradient-based (model learning=supervised)
+ on-policy (inefficient)		
- low information (esp w/ sparse rewards)	+ off-policy (data efficient)	+ off-policy (data efficient)

MAML with Policy Gradients

MAML: $$\min_\theta \sum_{\text{task } i} \mathcal{L}(\theta - \alpha \nabla_\theta \mathcal{L}(\theta, \mathcal{D}_i^{\text{tr}}), \mathcal{D}_i^{\text{ts}})$$

Policy Gradient: $$\nabla_\theta J_i(\theta) = E_{\tau \sim \pi_\theta, \mathcal{T}_i}\left[\left(\sum_t \nabla_\theta \log \pi_\theta(\mathbf{a}_t|\mathbf{s}_t)\right)\left(\sum_t r_i(\mathbf{s}_t, \mathbf{a}_t)\right)\right]$$

Meta-Training

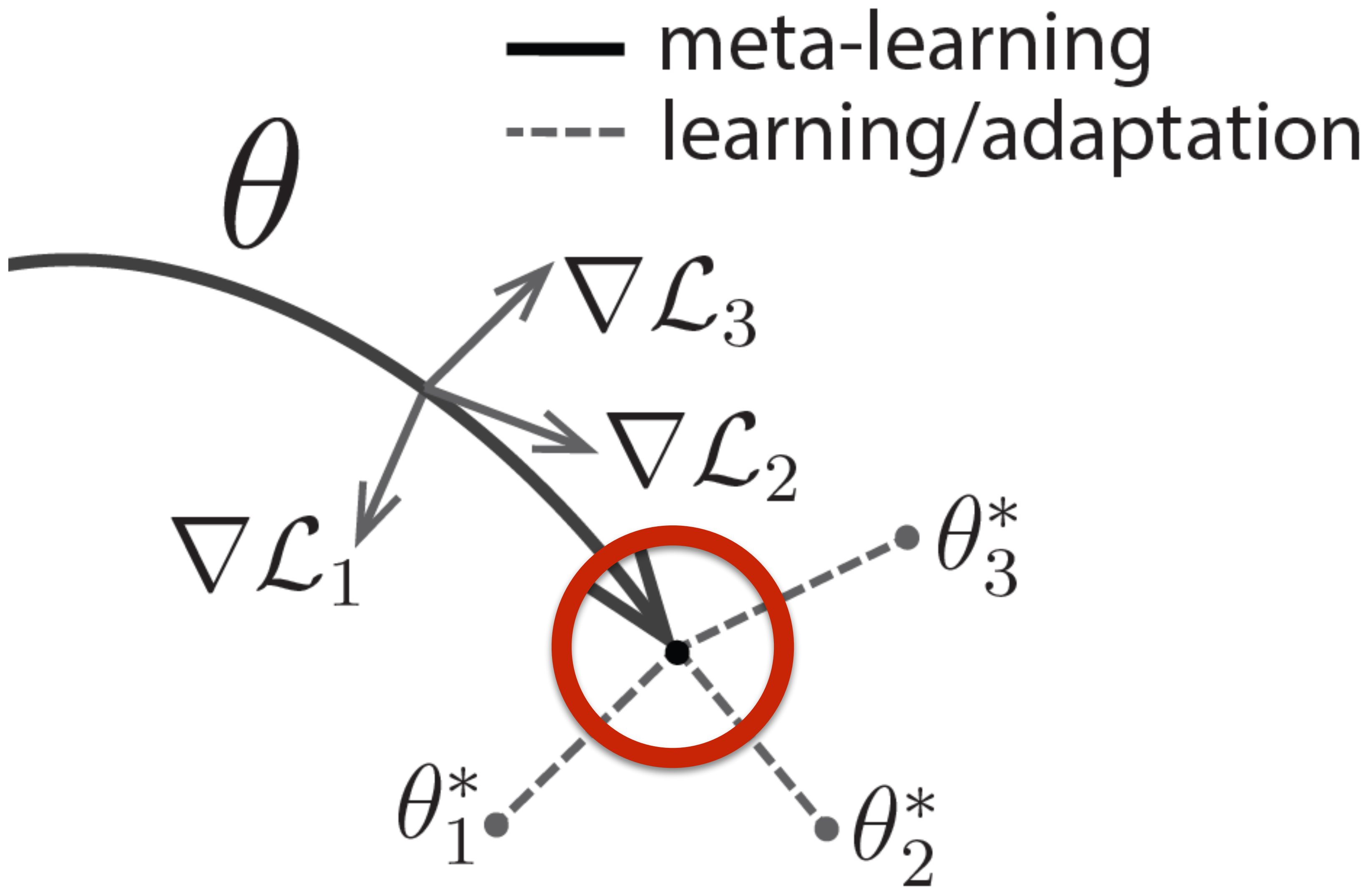
1. Sample task $\mathscr{T}_i$
2. Collect $\mathscr{D}_i^{\text{tr}}$ by rolling out π_θ
3. Inner loop adaptation: $\phi_i = \theta + \alpha \nabla_\theta J_i(\theta)$
4. Collect $\mathscr{D}_i^{\text{ts}}$ by rolling out π_{ϕ_i}
5. Outer loop update: $\theta \leftarrow \theta + \beta \sum_{\text{task } i} \nabla_\theta J_i(\phi_i)$

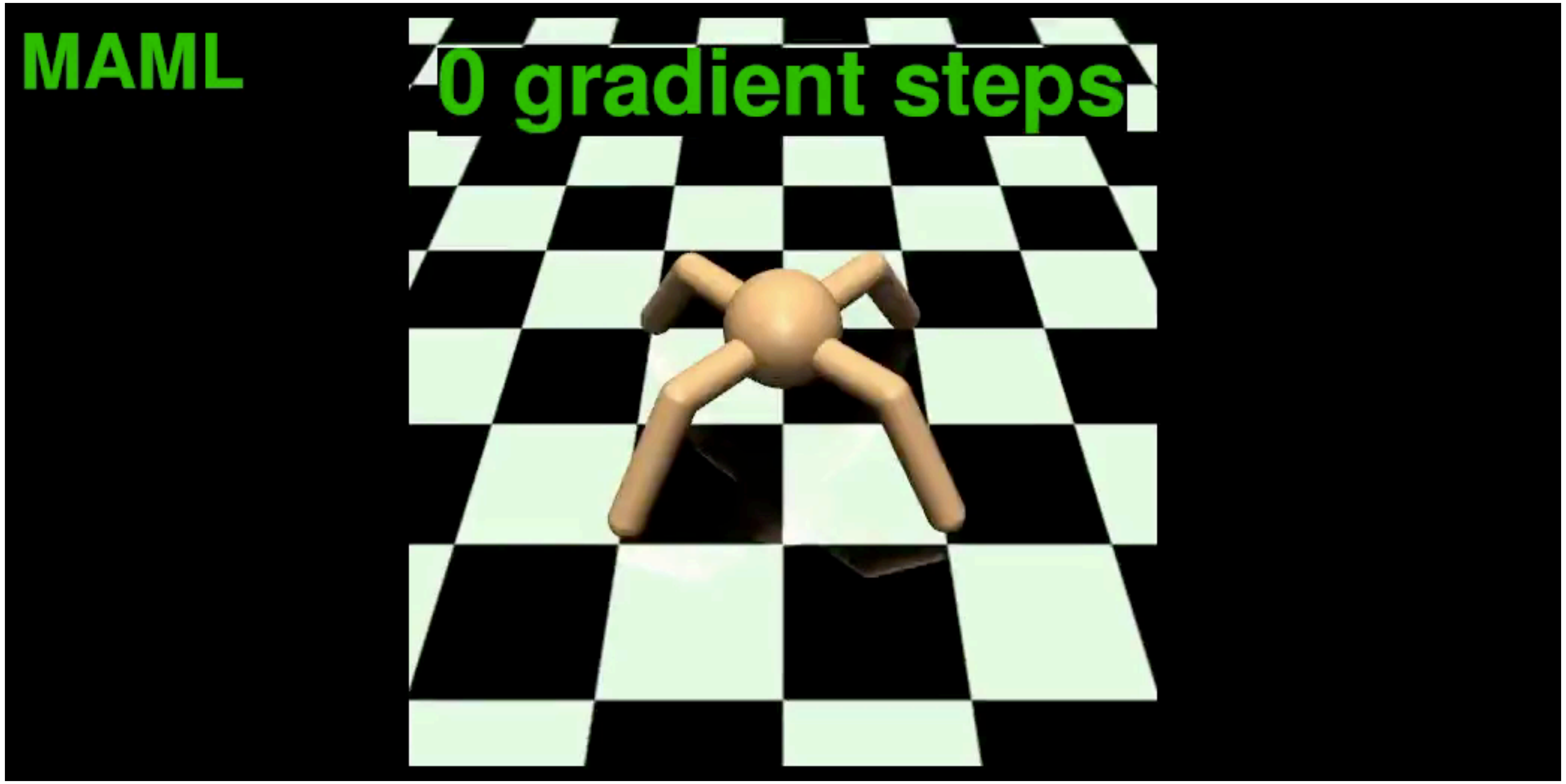
Meta-Test Time

1. Sample *new* task $\mathscr{T}_j$
2. Collect $\mathscr{D}_j^{\text{tr}}$ by rolling out π_θ
3. Adapt policy:

$$\phi_j = \theta + \alpha \nabla_\theta J_j(\theta)$$

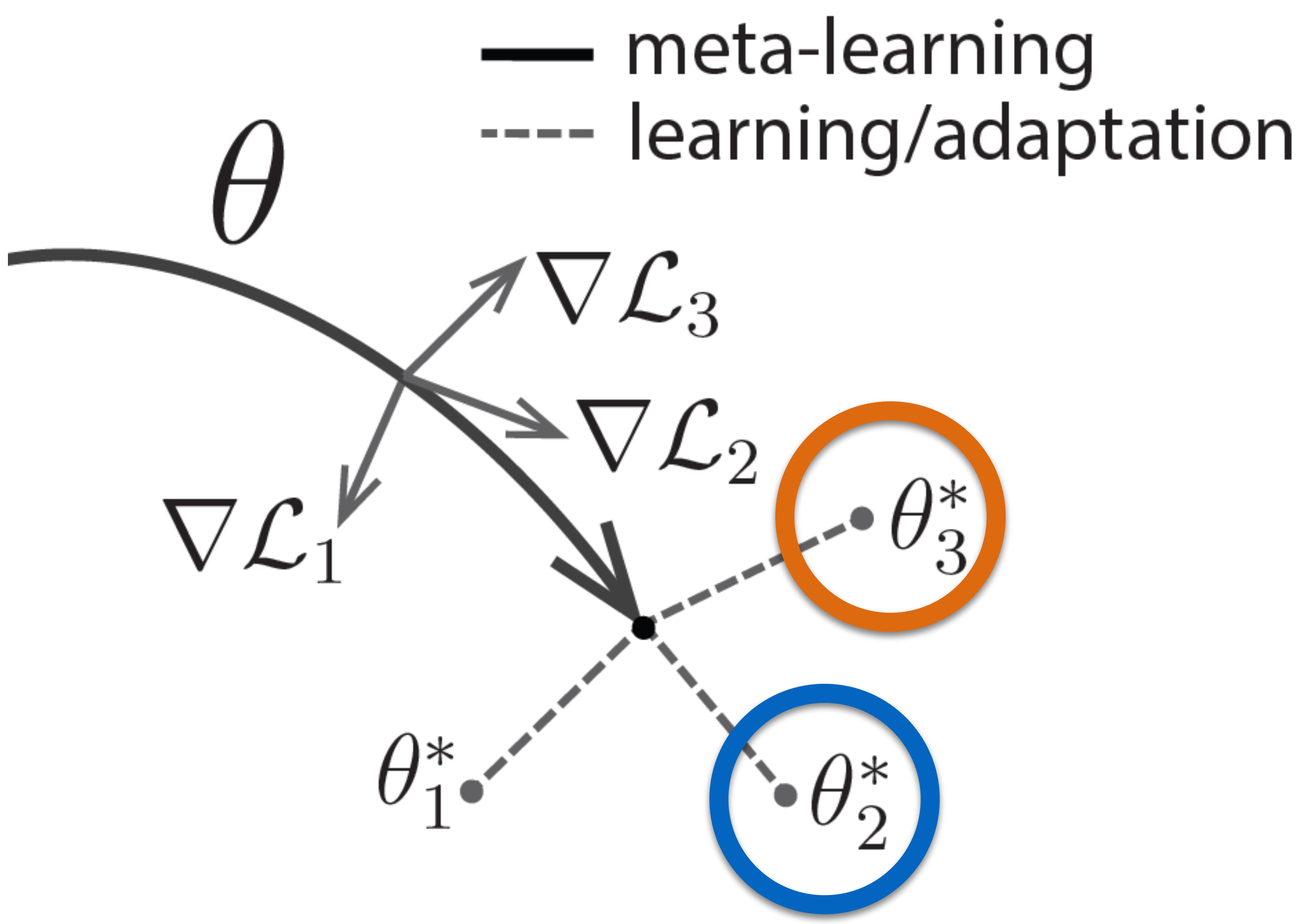
MAML with Policy Gradients

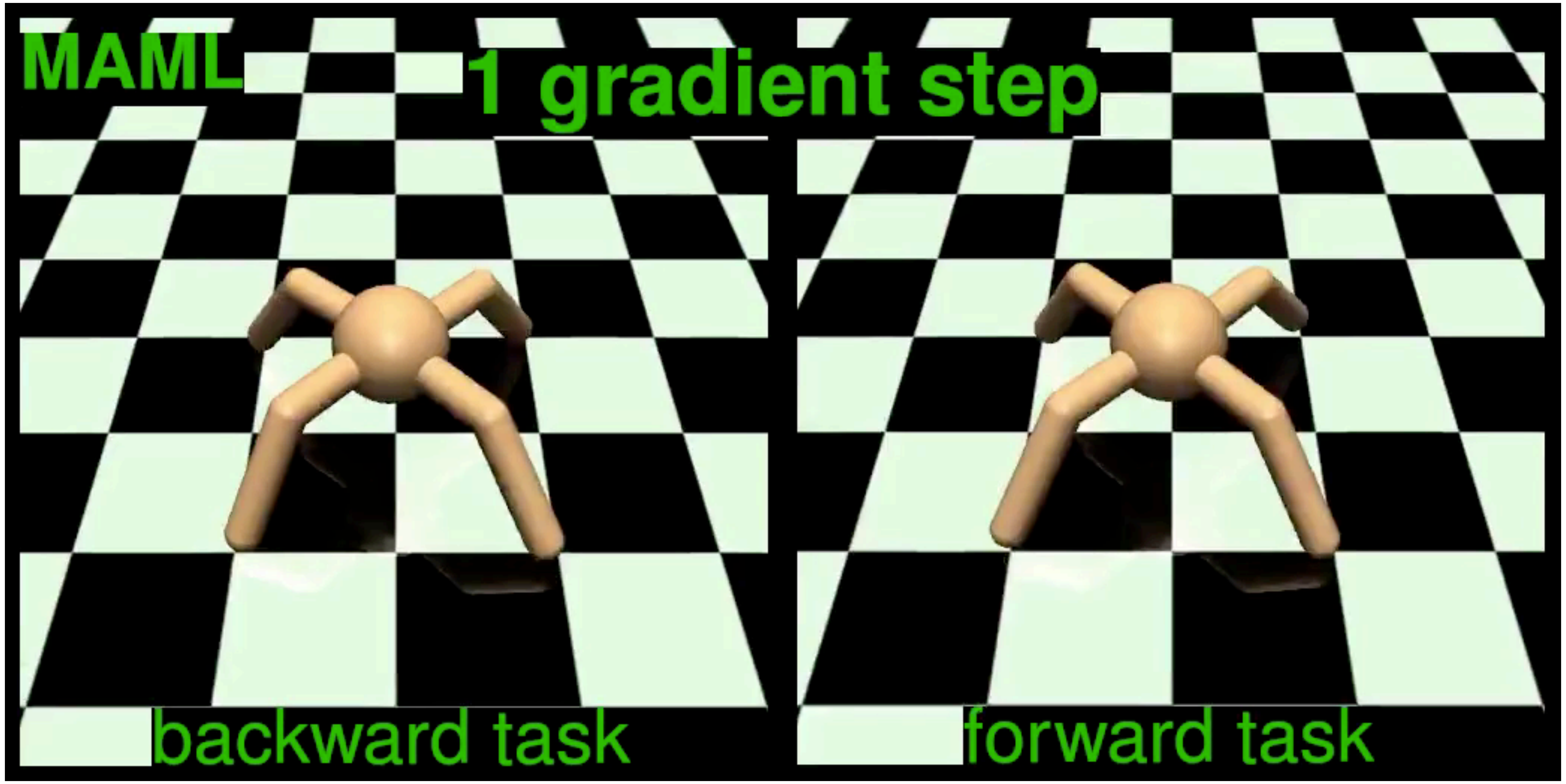




Finn, Abbeel, Levine. *Model-Agnostic Meta-Learning for Fast Adaptation of Deep Networks*. ICML 2017

MAML with Policy Gradients





Finn, Abbeel, Levine. *Model-Agnostic Meta-Learning for Fast Adaptation of Deep Networks*. ICML 2017

MAML with Model-Based RL

Online Variant Inputs: $\underbrace{\mathcal{D}_{\text{train}}}_{\text{k timesteps from } \pi}$ $\mathbf{s}_t$ Outputs: $\mathbf{a}_t$

Meta-test time:

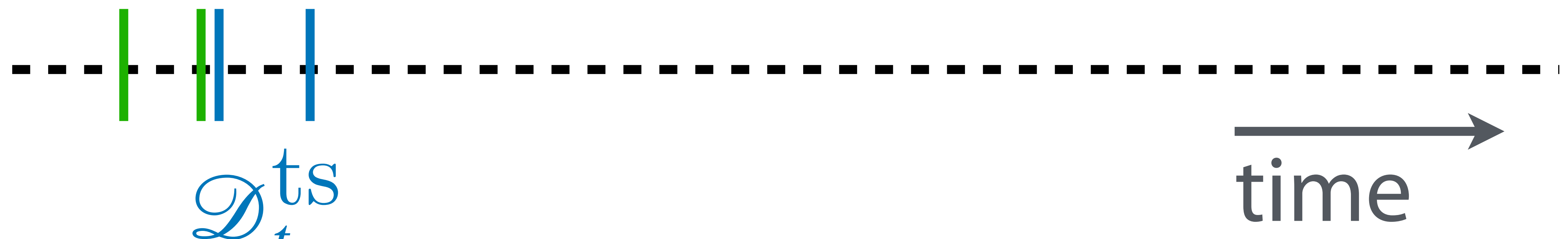
1. Adapt model $f_\theta \rightarrow f_{\phi_t}$ to last k time steps
2. Plan $a_t, \ldots, a_{t+h}$ using adapted model f_{ϕ_t}

Meta-training:

$\mathscr{D}_t^{\text{tr}}$

$\mathbf{s}_{t-k:t}, \mathbf{a}_{t-k:t}$

tasks: windows in time



$\mathscr{D}_t^{\text{ts}}$

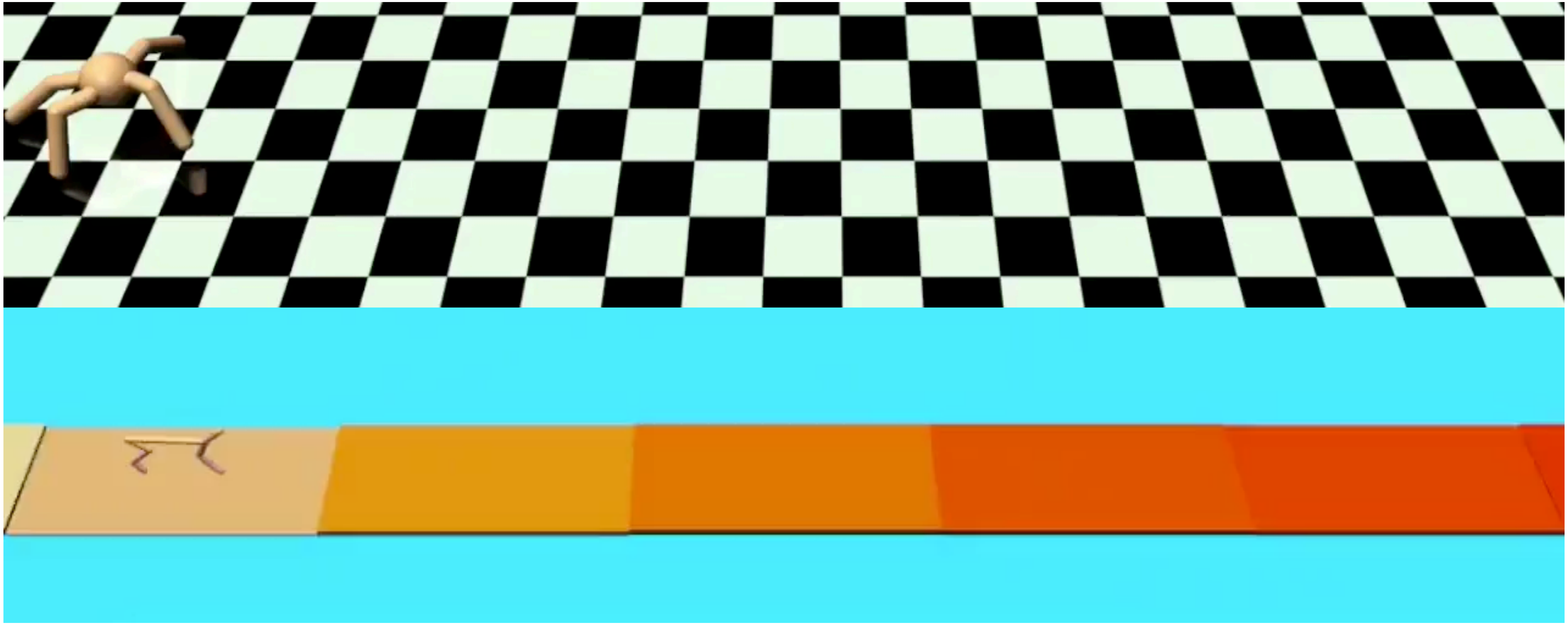
$\mathbf{s}_{t:t+h}, \mathbf{a}_{t:t+h}$

Nagabandi*, Clavera*, Liu, Fearing, Abbeel, Levine, Finn. *Learning to Adapt in Dynamic Environments through Meta-RL*. ICLR'19

Dynamic Environments without Adaptation

Model-Based RL Only

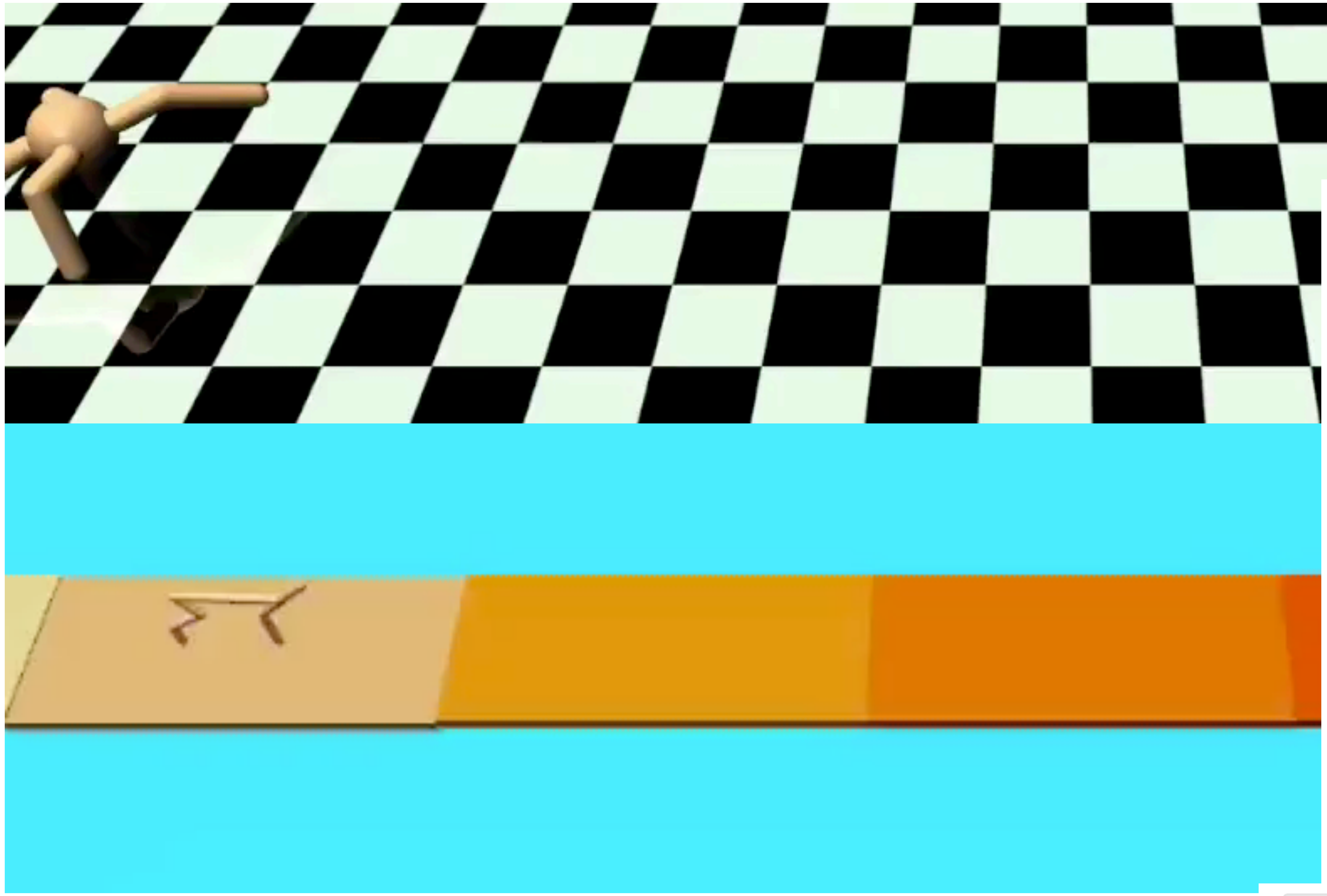
Tries to fit single model $f(s' | s, a)$ to varying $p_t(s' | s, a)$.

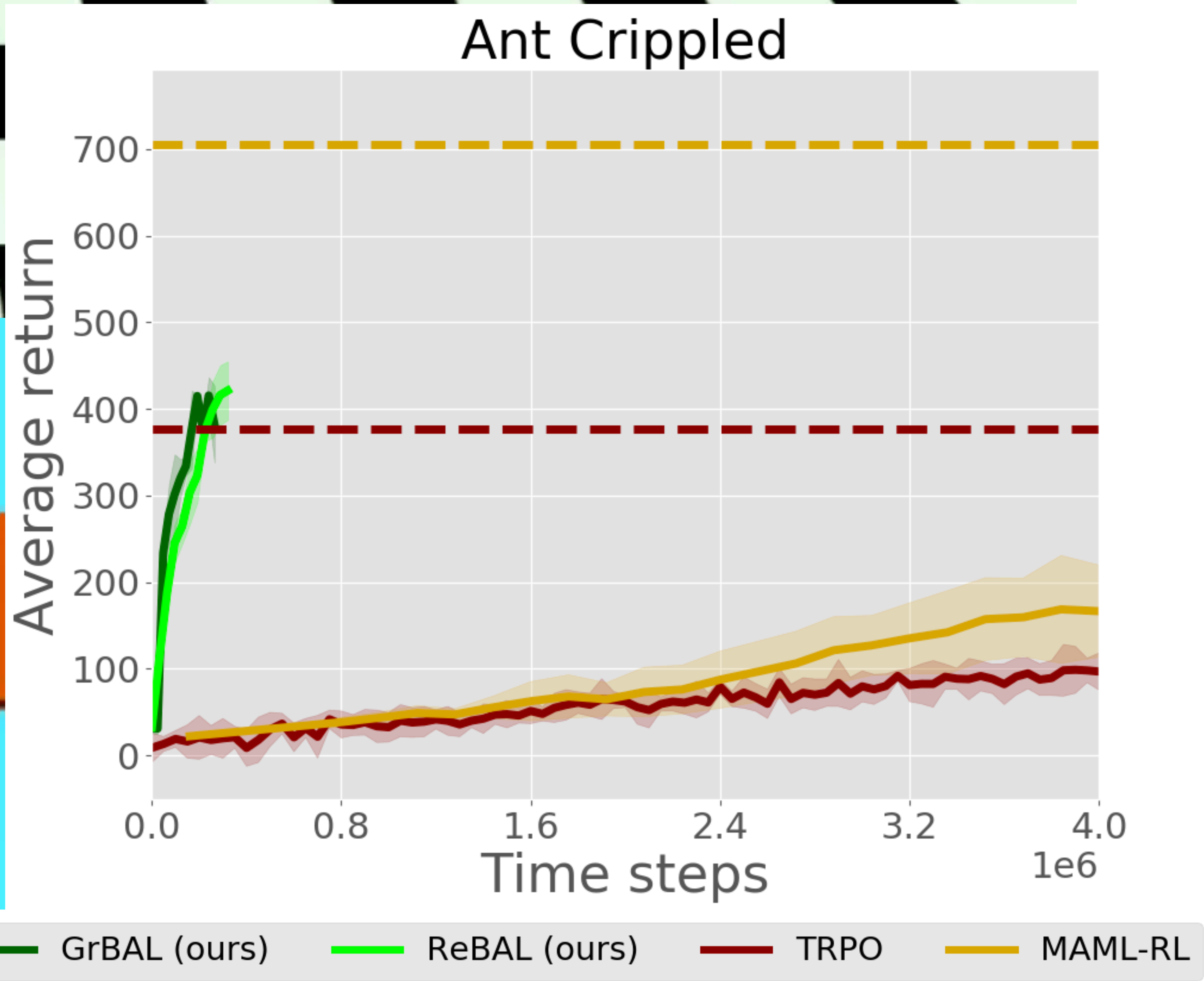


Nagabandi*, Clavera*, Liu, Fearing, Abbeel, Levine, Finn. *Learning to Adapt in Dynamic Environments through Meta-RL*. ICLR'19

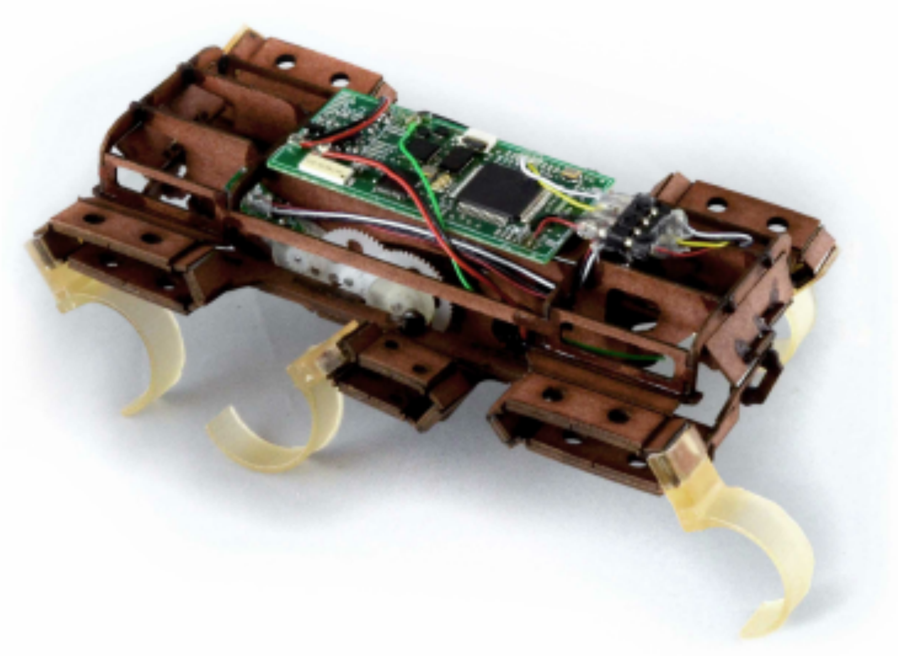
Dynamic Environments without Adaptation

MAML+Model-based RL



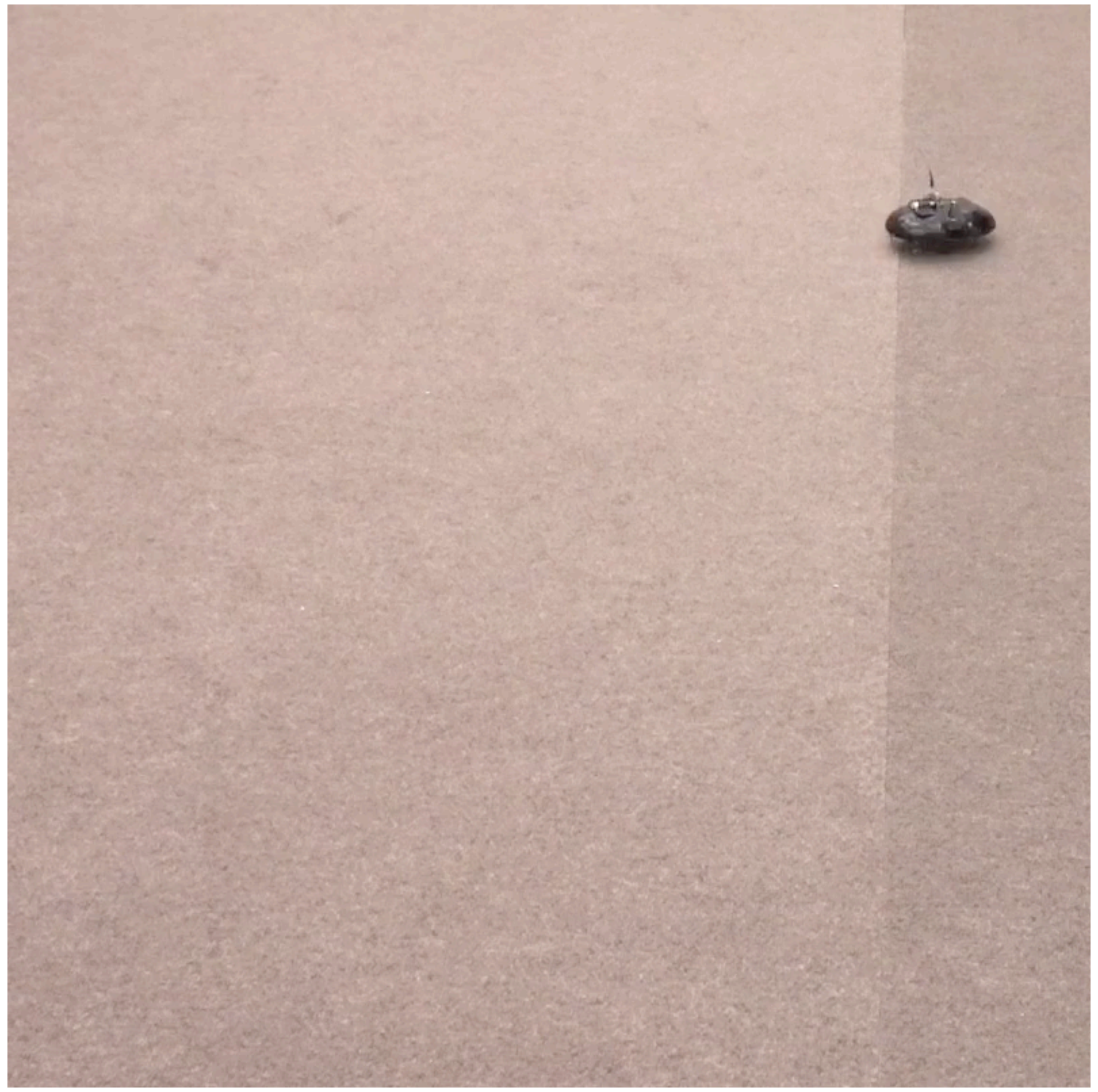


Nagabandi*, Clavera*, Liu, Fearing, Abbeel, Levine, Finn. *Learning to Adapt in Dynamic Environments through Meta-RL*. ICLR'19



VelociRoACH Robot

Meta-train on **variable terrains**

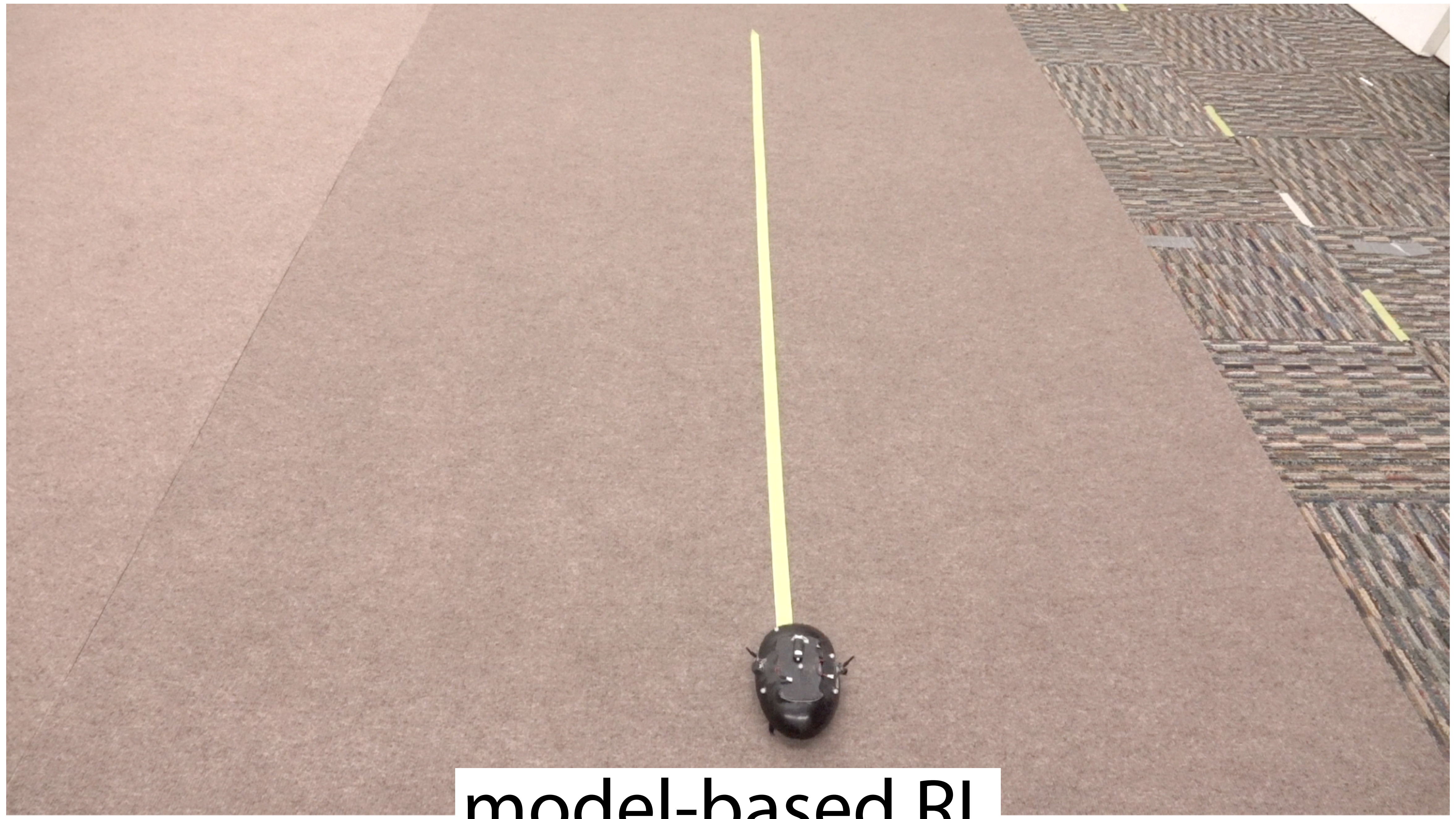


Meta-test with **slope, missing leg, payload, calibration errors**

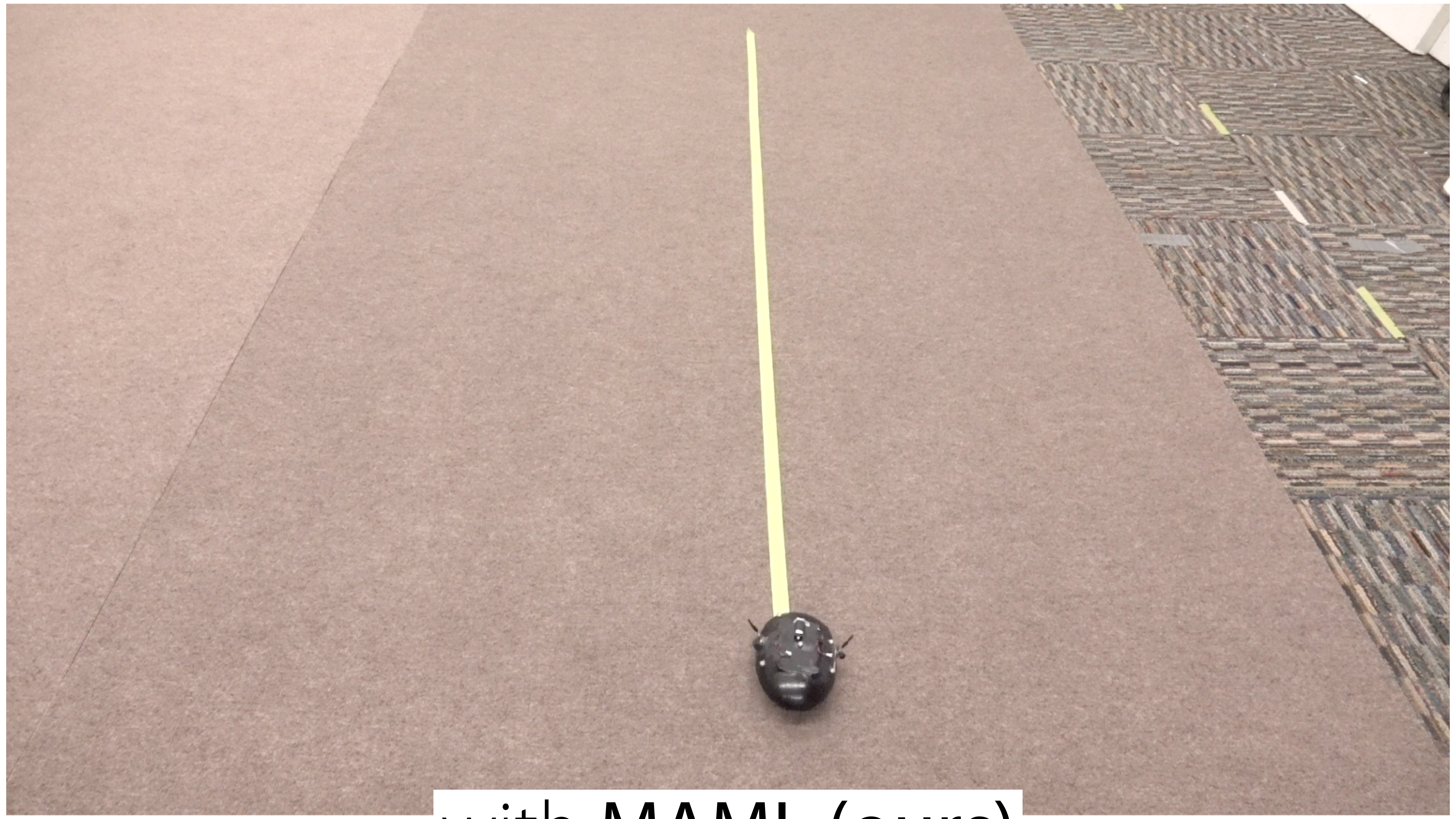
Nagabandi*, Clavera*, Liu, Fearing, Abbeel, Levine, Finn. *Learning to Adapt in Dynamic Environments through Meta-RL*. ICLR'19

VelociRoACH Robot

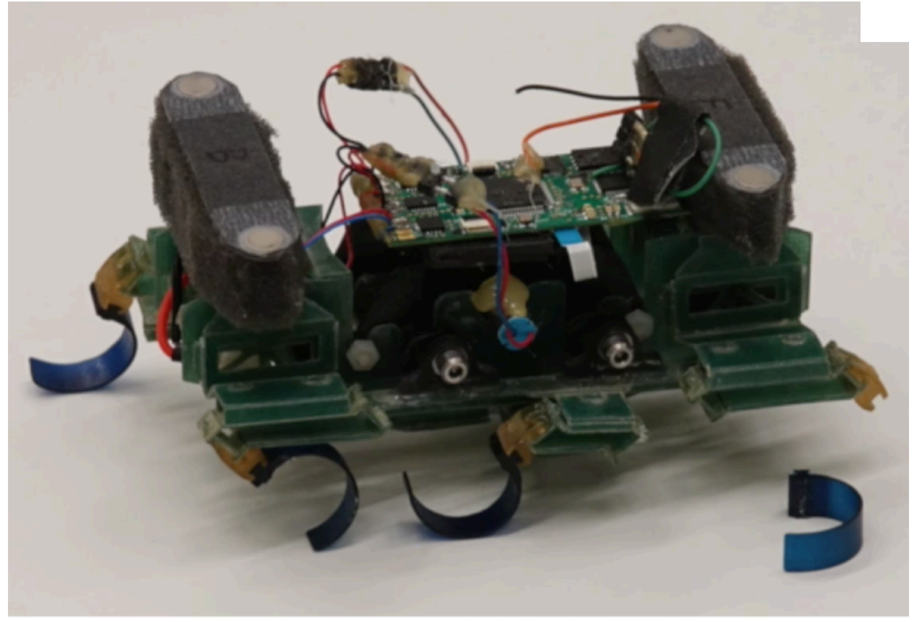
Meta-train on **variable terrains** **Meta-test** with **slope**, **missing leg**, **payload**, **calibration errors**



model-based RL
(no adaptation)

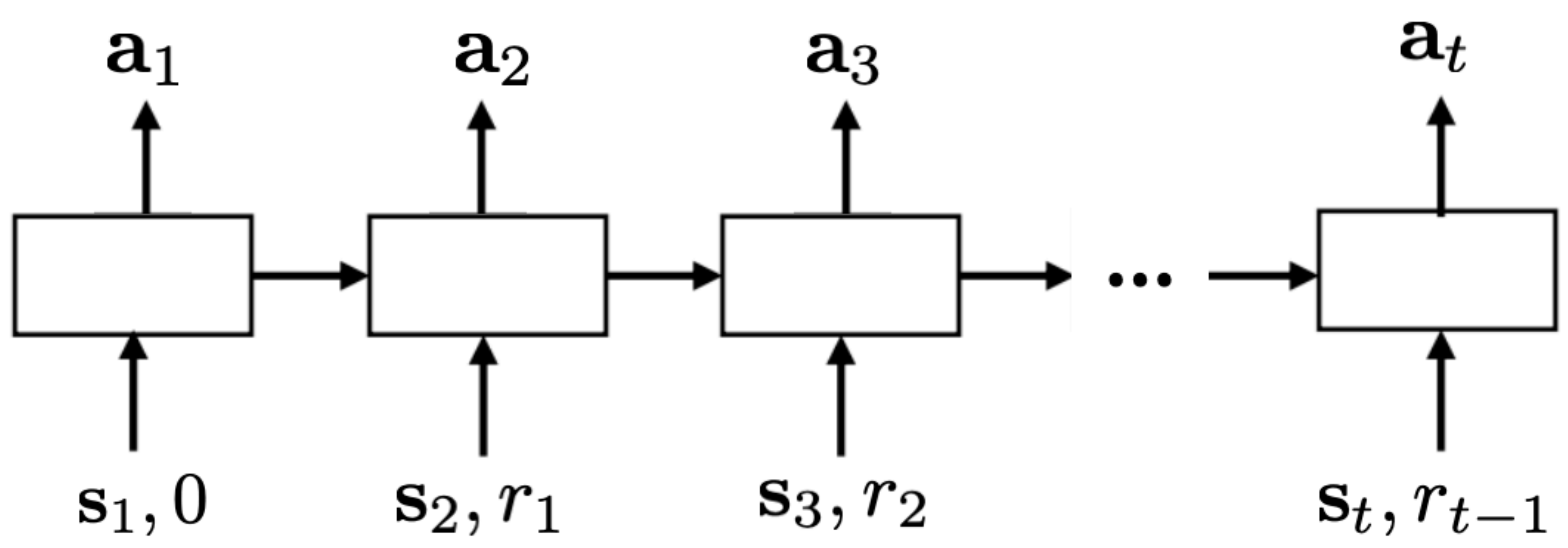


with **MAML (ours)**



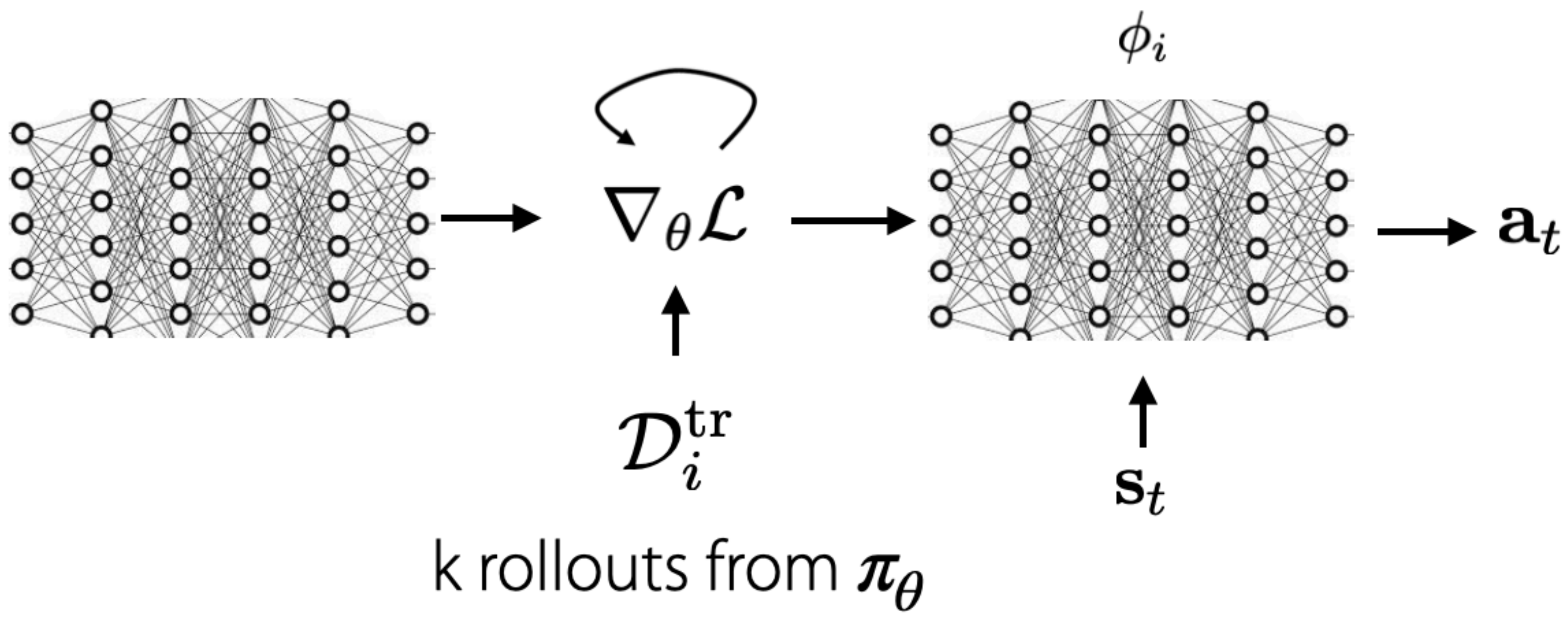
Nagabandi*, Clavera*, Liu, Fearing, Abbeel, Levine, Finn. *Learning to Adapt in Dynamic Environments through Meta-RL*. ICLR'19

Black-Box Meta-RL



- \+ general & expressive
- \+ a variety of design choices in architecture & objective
- -- hard to optimize

Optimization-Based Meta-RL



- \+ inductive bias of optimization built in
- \+ easy to combine with policy gradients, model-based methods
- -- policy gradients very noisy
- -- hard to combine with value-based RL methods

Both: inherit sample efficiency from outer RL optimizer

Plan for Today

Meta-RL problem statement

Black-box meta-RL methods

Optimization-based meta-RL methods

Lecture goals:

- Understand the meta-RL problem statement & set-up
- Understand the basics of black-box meta RL algorithms
- Understand the basics & challenges of optimization-based meta RL algorithms

Next time

Today: meta-RL basics

Wednesday: learning to explore via meta-RL

Reminders

Homework 3 due **Wednesday**

Project milestone due **next Wednesday**