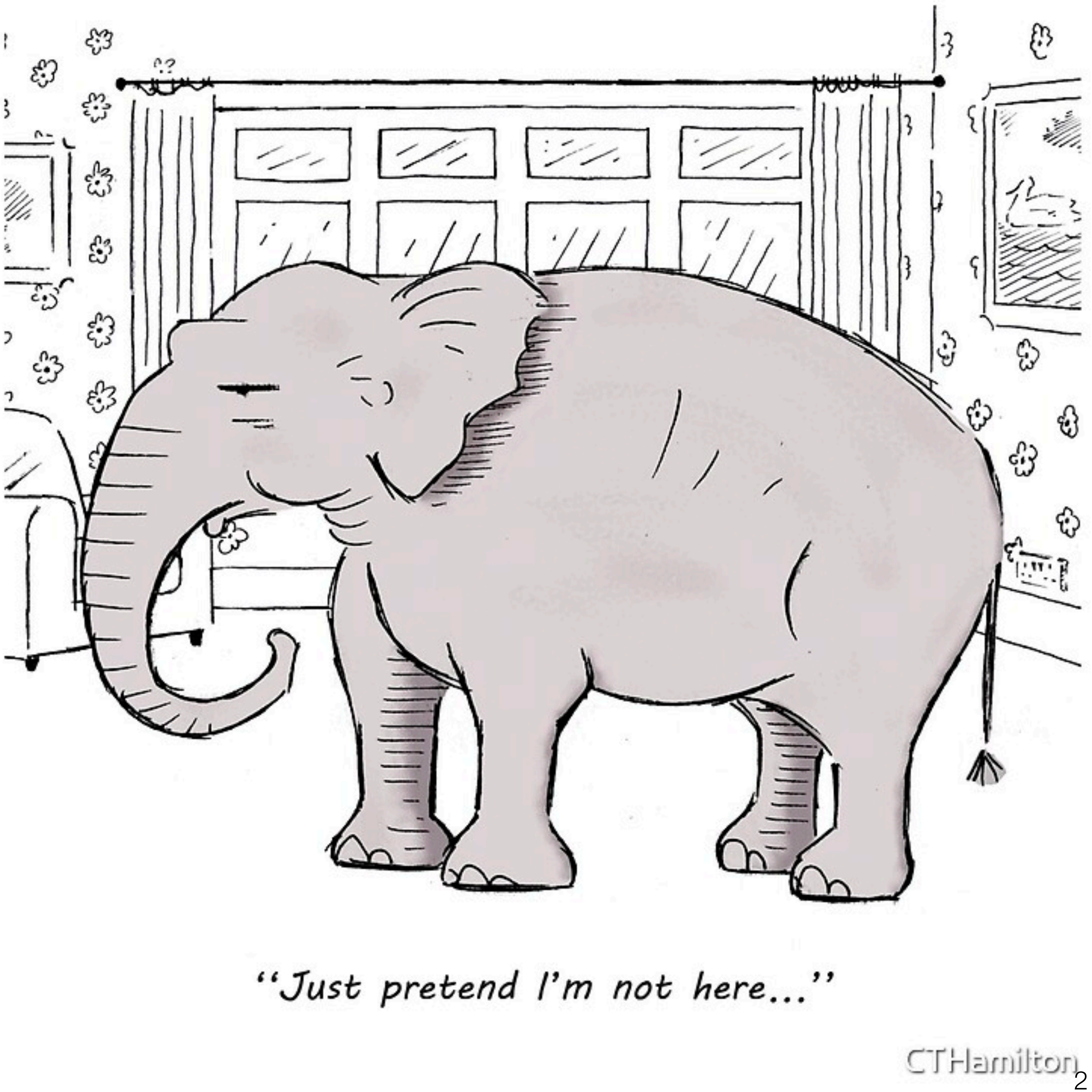
Planning with Inaccurate Models

Sanjiban Choudhury

Cornell Bowers C·IS
Computer Science

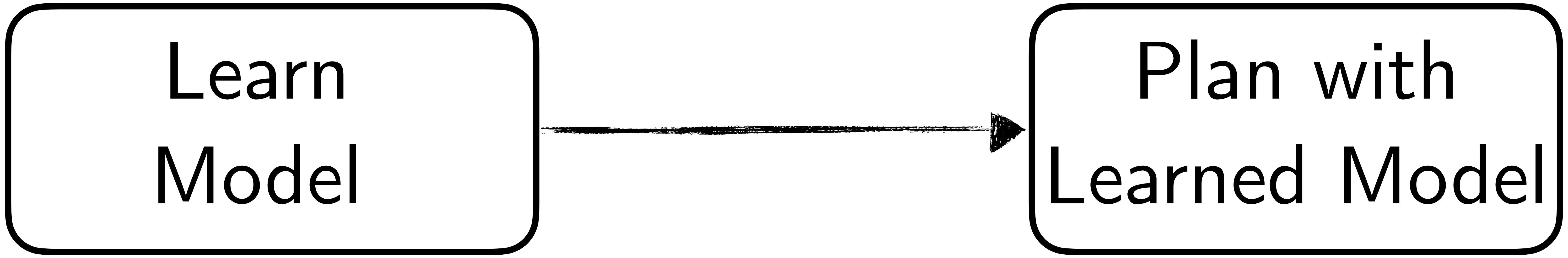
Elephant in the room: Why can't we just *learn a model*?



"Just pretend I'm not here..."

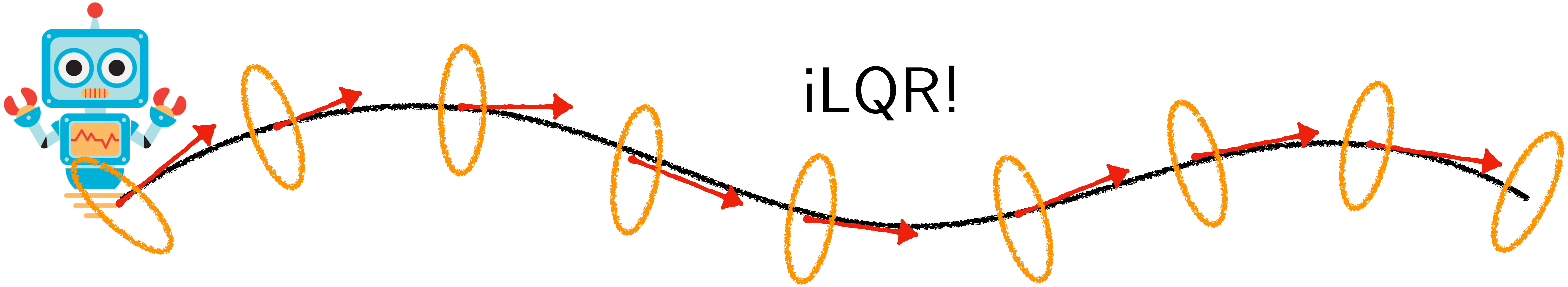
CTHamilton

Model Based Reinforcement Learning



Learn Model

Plan with Learned Model



iLQR!

2560, 2.5 second trajectories sampled with cost-weighted average @ 60 Hz

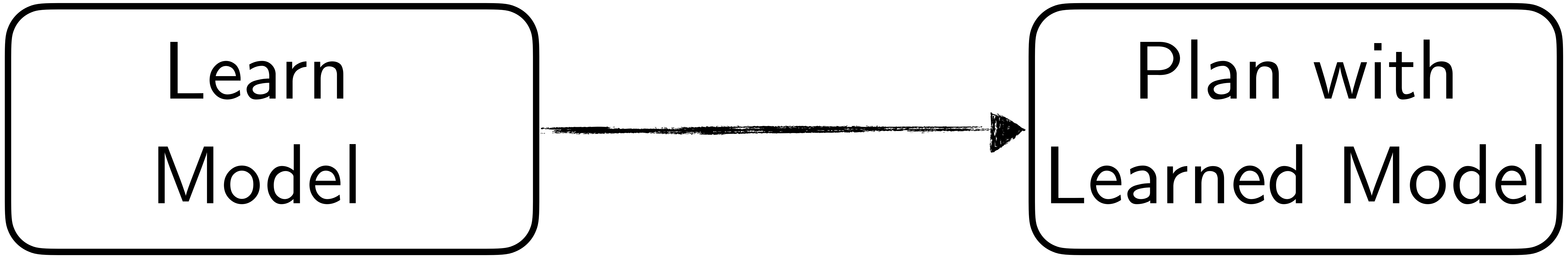
Georgia Tech Auto Rally (Byron Boots lab)

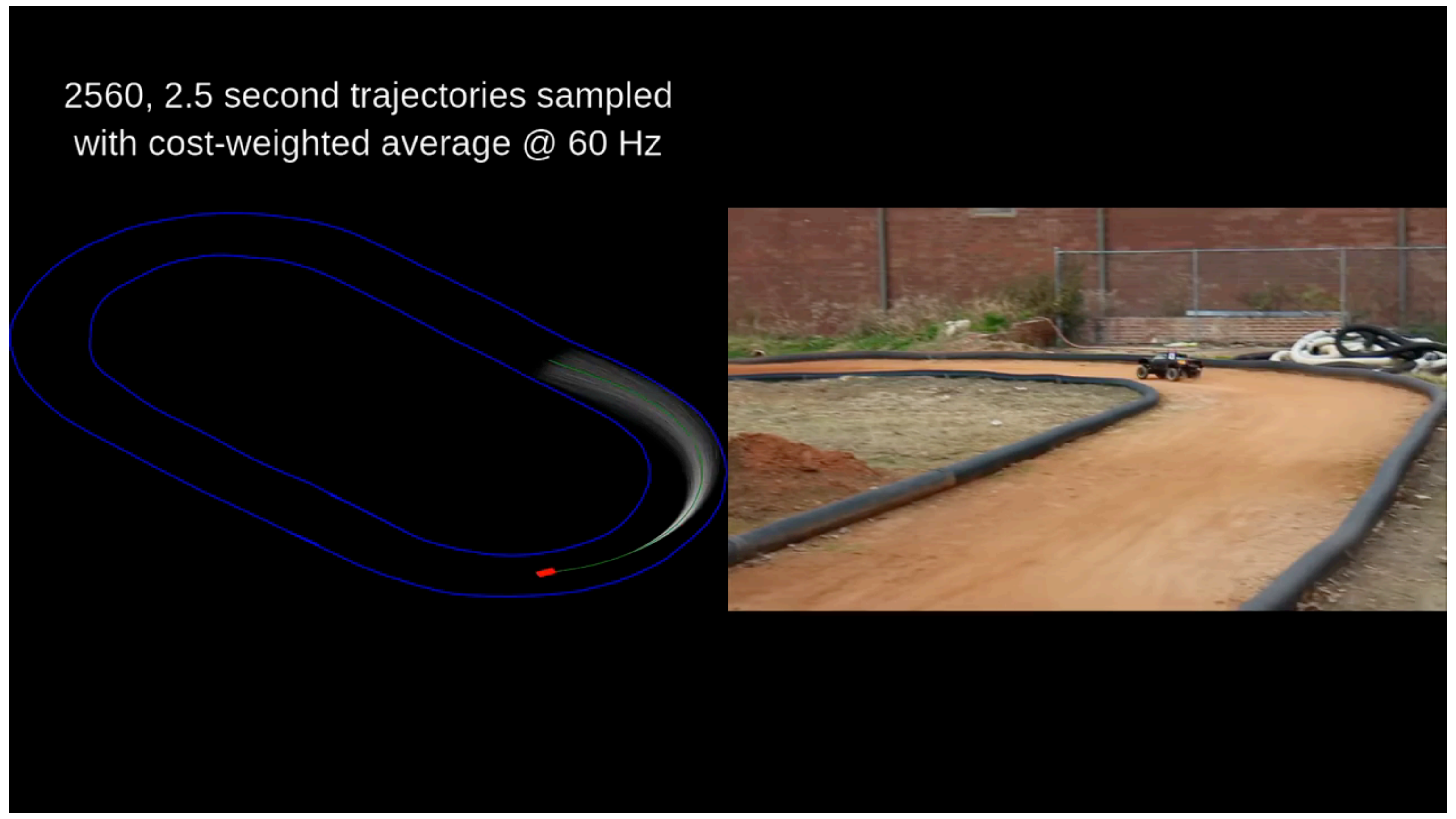
Activity!

Think-Pair-Share

Think (30 sec): What architecture would you use to learn a model for rally car? What loss function will you use? What planner?

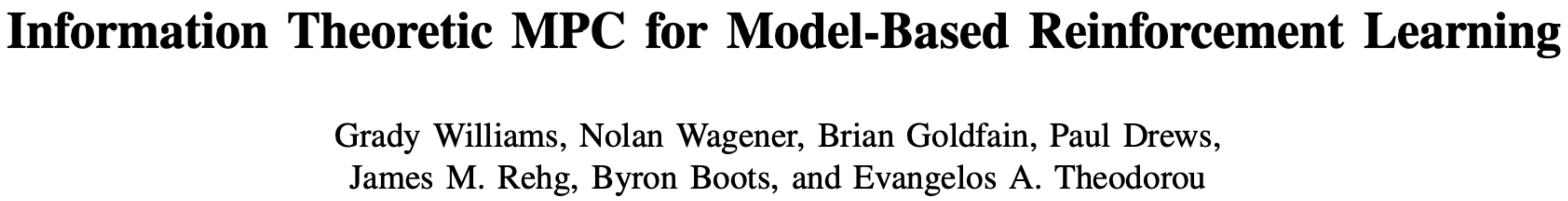
Pair: Find a partner





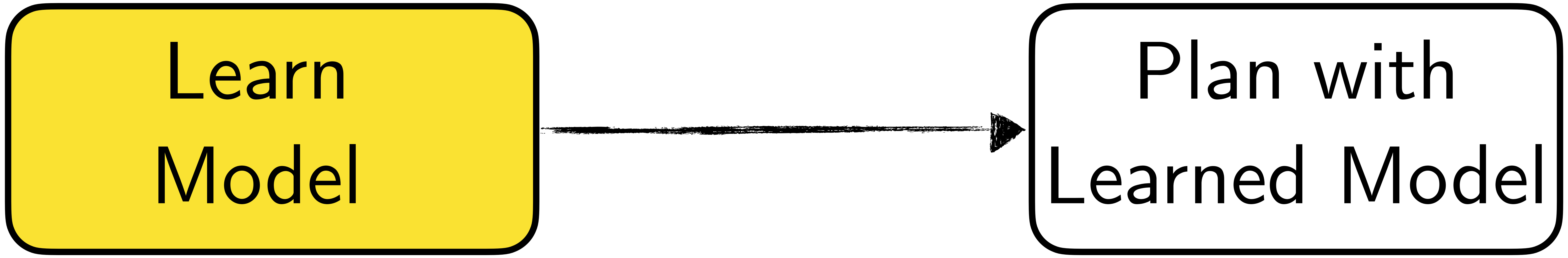
Share (45 sec): Partners exchange ideas

Part 1: System Identification



Information Theoretic MPC for Model-Based Reinforcement Learning

Grady Williams, Nolan Wagener, Brian Goldfain, Paul Drews,
James M. Rehg, Byron Boots, and Evangelos A. Theodorou



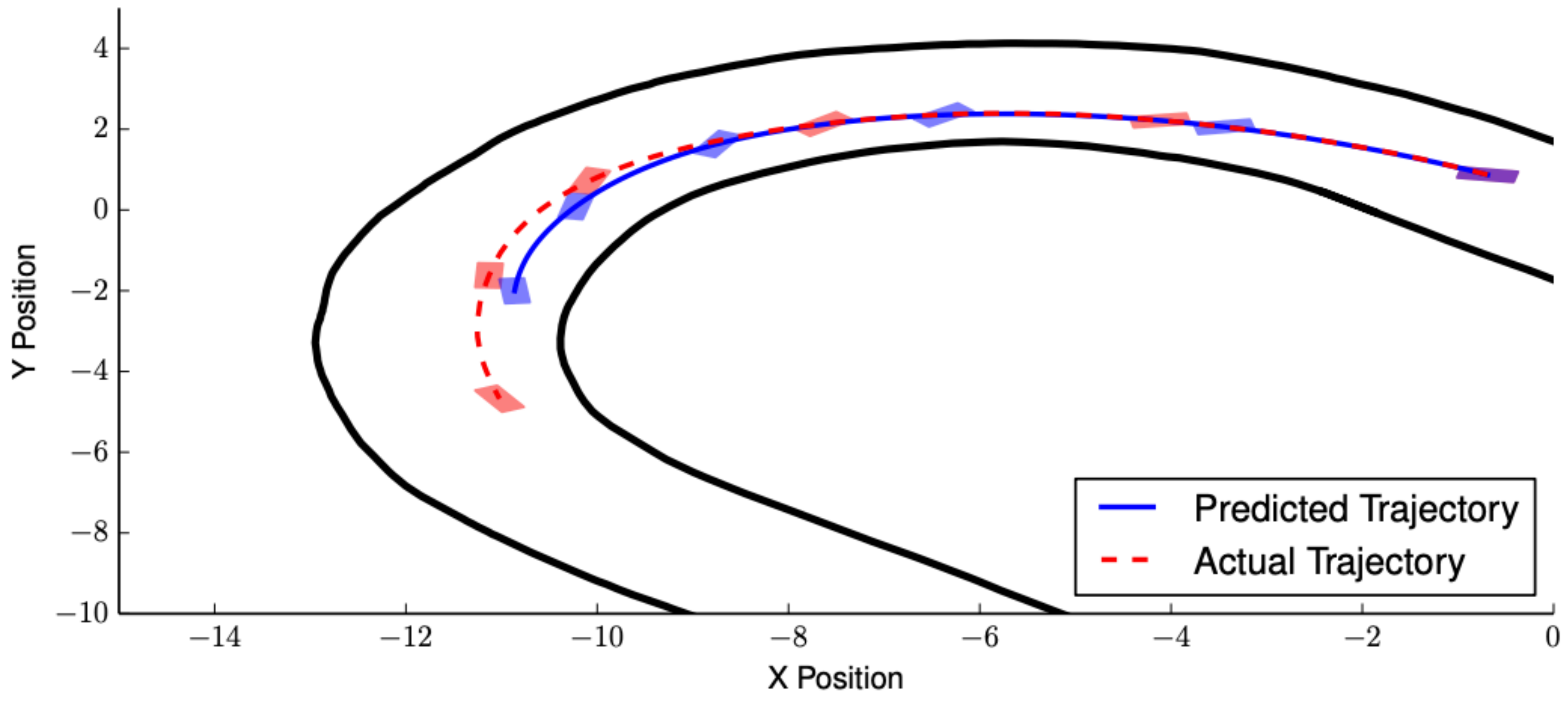
Learn Model

Plan with Learned Model

Collect data of rally car $(x_1, u_1, x_2, u_2, \ldots)$

$$\mathbf{x}_{t+1} = \mathbf{F}(\mathbf{x}_t, \mathbf{u}_t) = \begin{bmatrix} \mathbf{q}_t + \dot{\mathbf{q}}_t \Delta t \\ \dot{\mathbf{q}}_t + \mathbf{f}(\mathbf{x}_t, \mathbf{u}_t) \Delta t \end{bmatrix}$$

2 Layer MLP



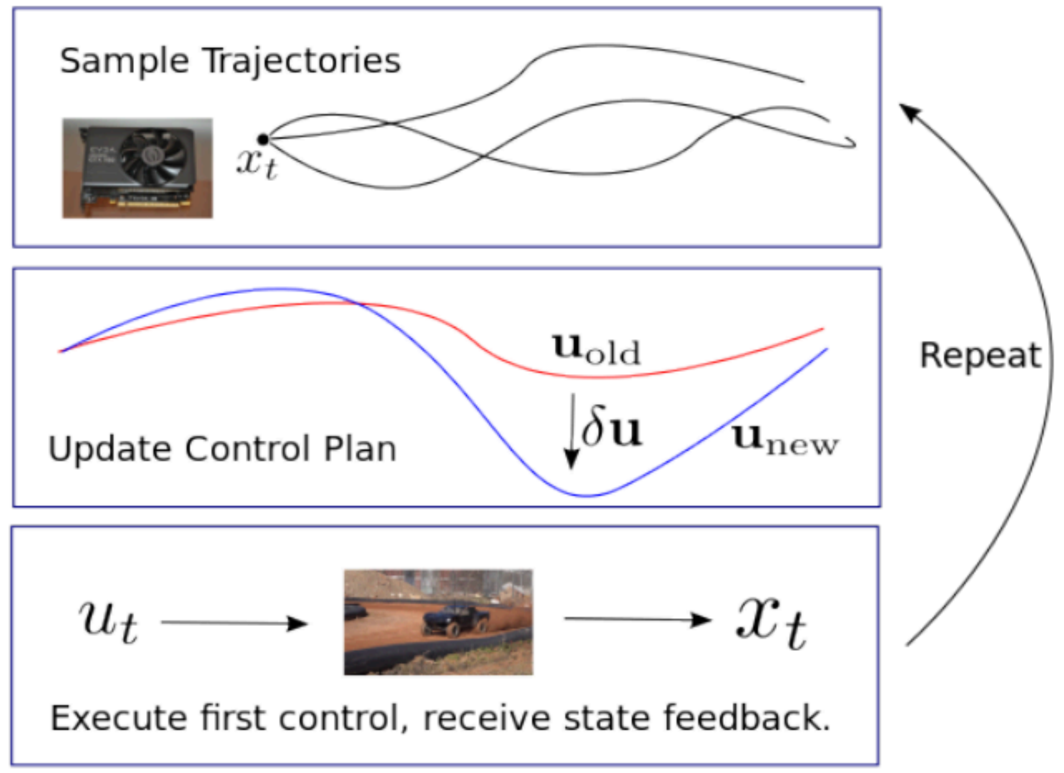
Part 2: Planning

Information Theoretic MPC for Model-Based Reinforcement Learning

Grady Williams, Nolan Wagener, Brian Goldfain, Paul Drews,
James M. Rehg, Byron Boots, and Evangelos A. Theodorou

Learn Model → Plan with Learned Model

1. Sample and evaluate trajectories
2. Compute control update
3. Execute first control in sequence, receive state feedback
4. Repeat, using the un-executed portion of the previous control sequence to warm-start the trajectory

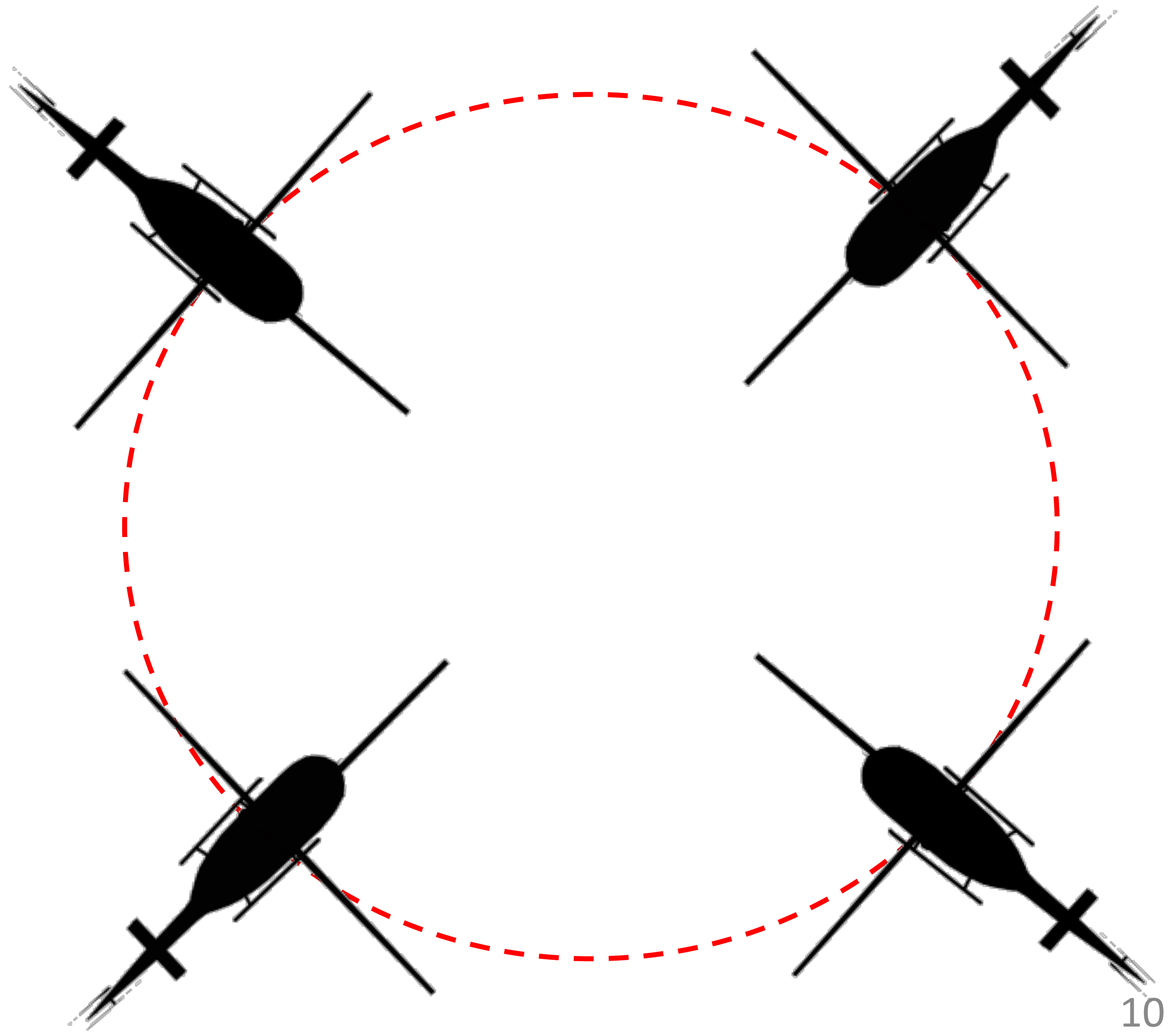


Cross Entropy like approach!

2560, 2.5 second trajectories sampled with cost-weighted average @ 60 Hz

Question: How do you collect data for learning model?

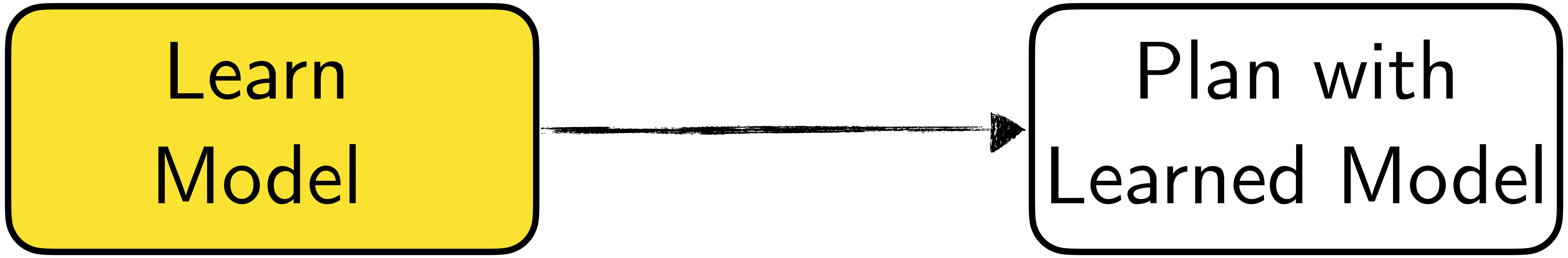
Another Example: Helicopter Aerobatics



A nose-in funnel!

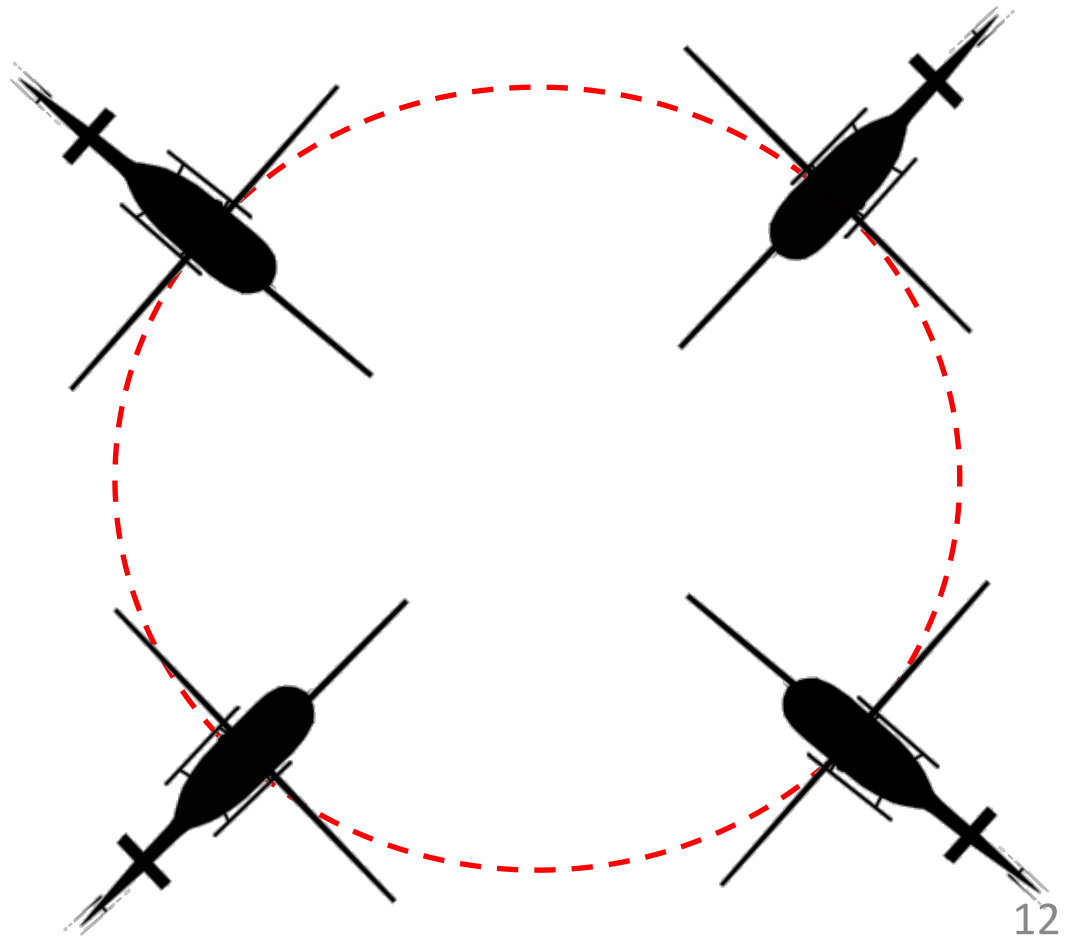
(Super cool work by Pieter Abeel et al. https://people.eecs.berkeley.edu/~pabbeel/autonomous_helicopter.html)

Part 1: System Identification



Learn Model

Plan with Learned Model

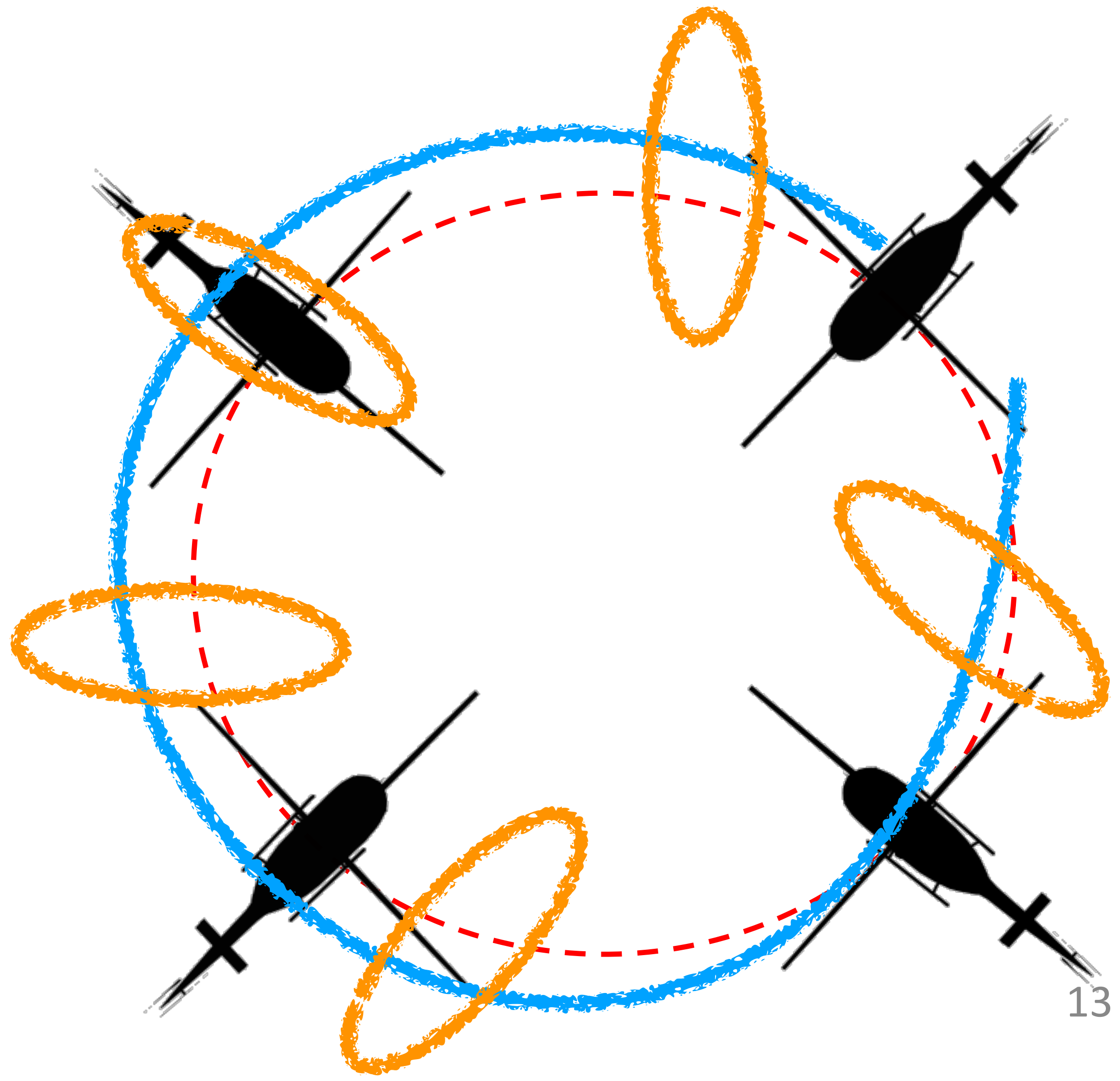


Learn a linear model around reference

$$\Delta x_{t+1} = A_t x_t + B_t u_t$$

Part 2: Planning

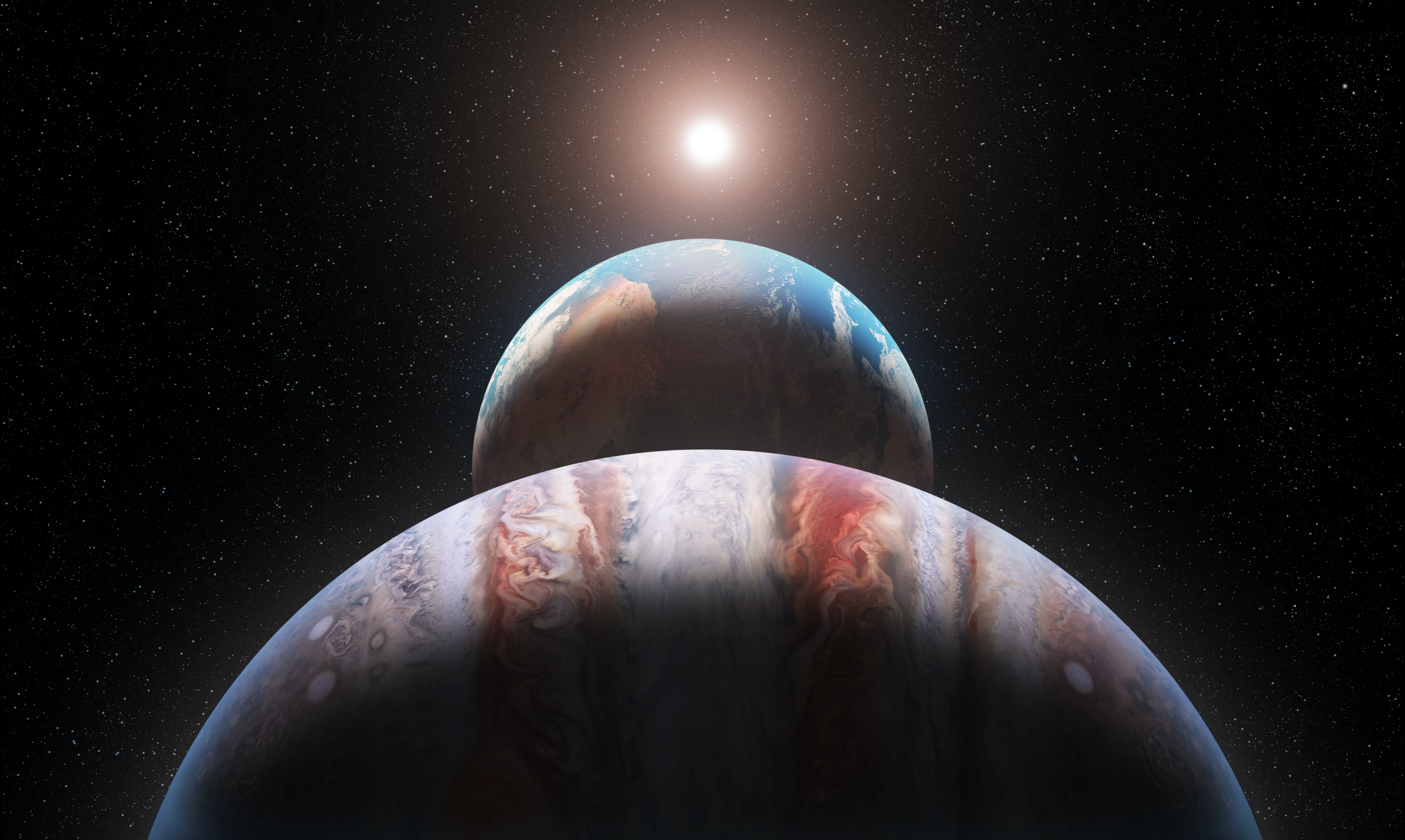
Learn Model → Plan with Learned Model



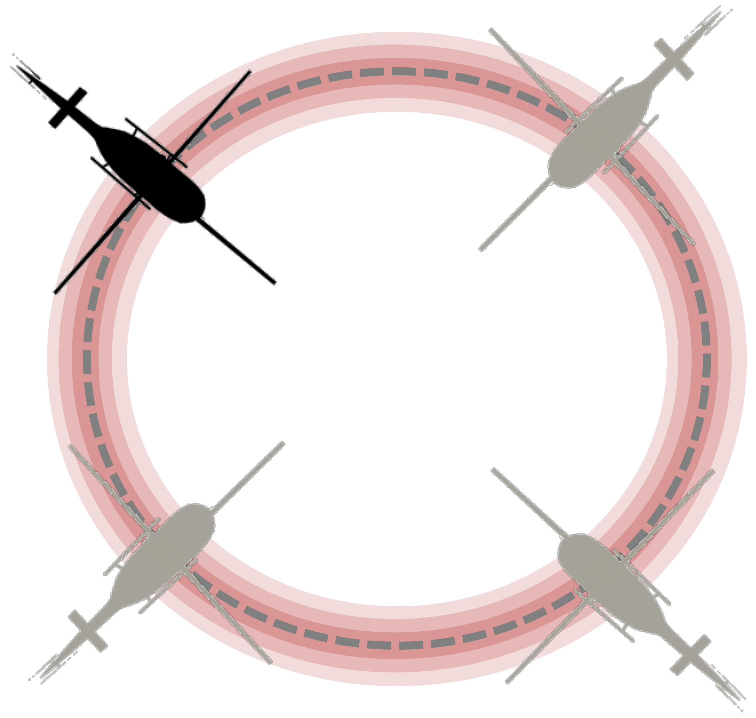
Use LQR with learnt models

How does a tiny error in my model affect performance?

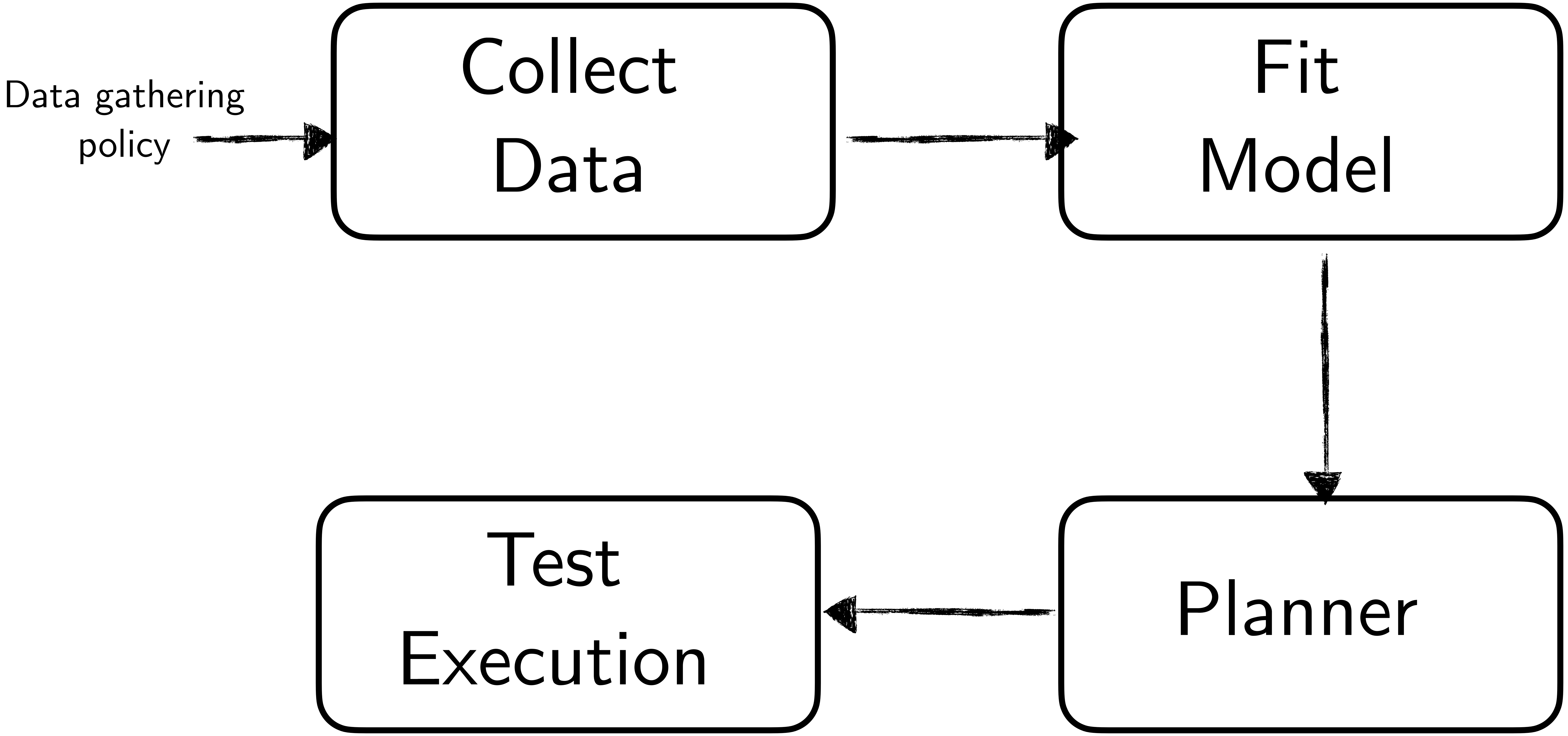
The Simulation Lemma



V0 of Model Based RL



Training Distribution



Data gathering policy

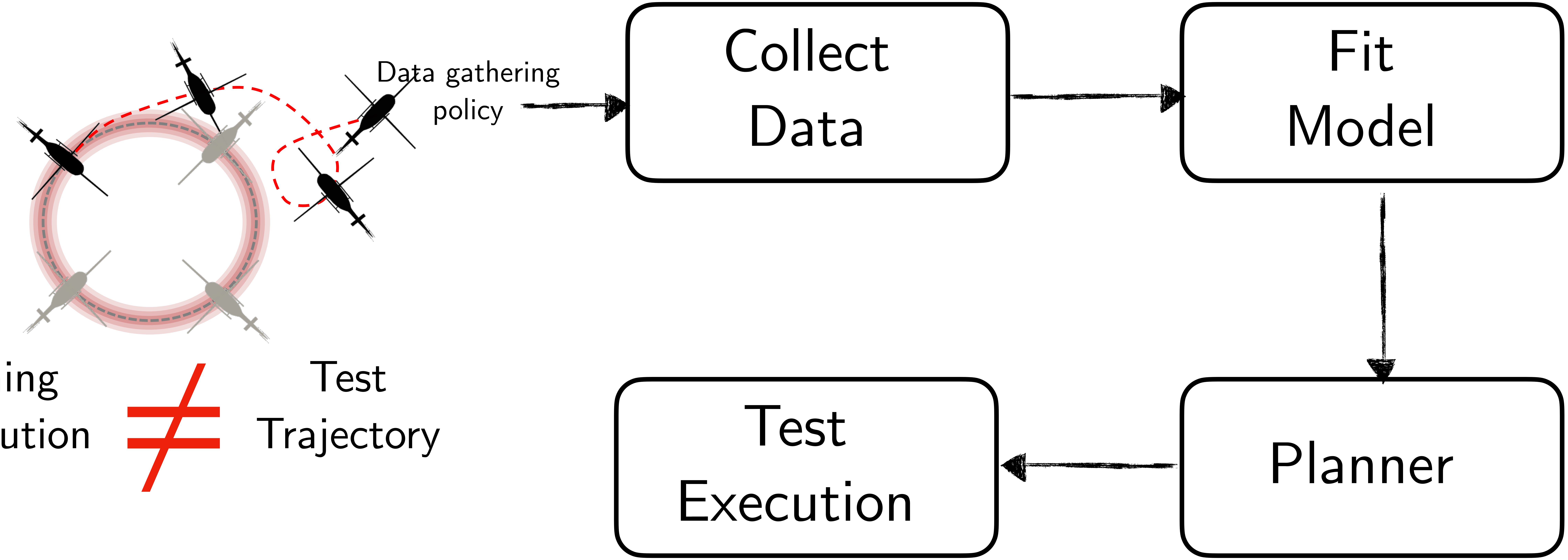
Collect Data

Fit Model

Planner

Test Execution

Problem: Train Test Mismatch!



Data gathering policy

Collect Data

Fit Model

Planner

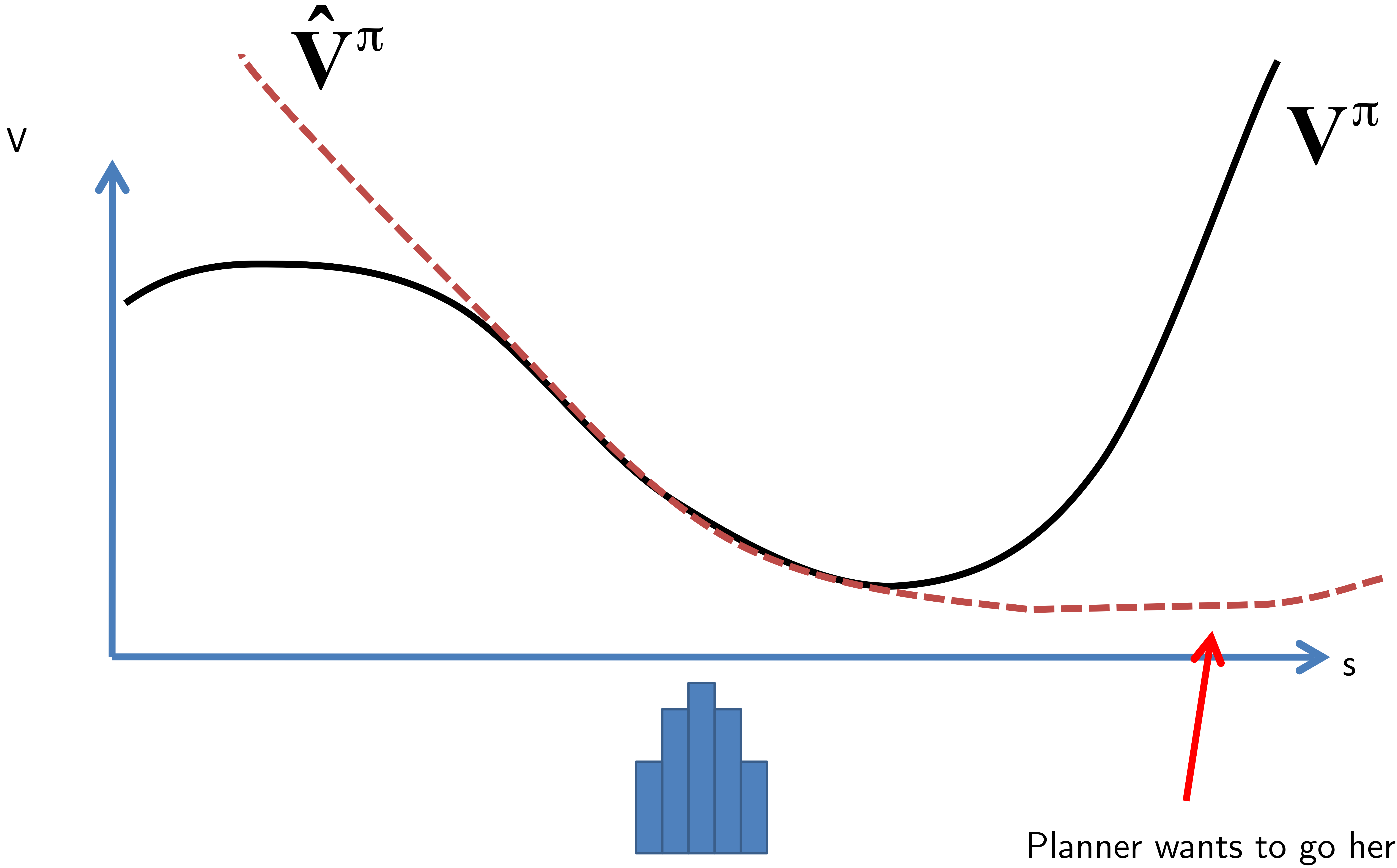
Test Execution

Training Distribution ≠ Test Trajectory

Super retro video of distribution mismatch!

Sure .. but what if we use really deep networks that drive down model learning loss really low?

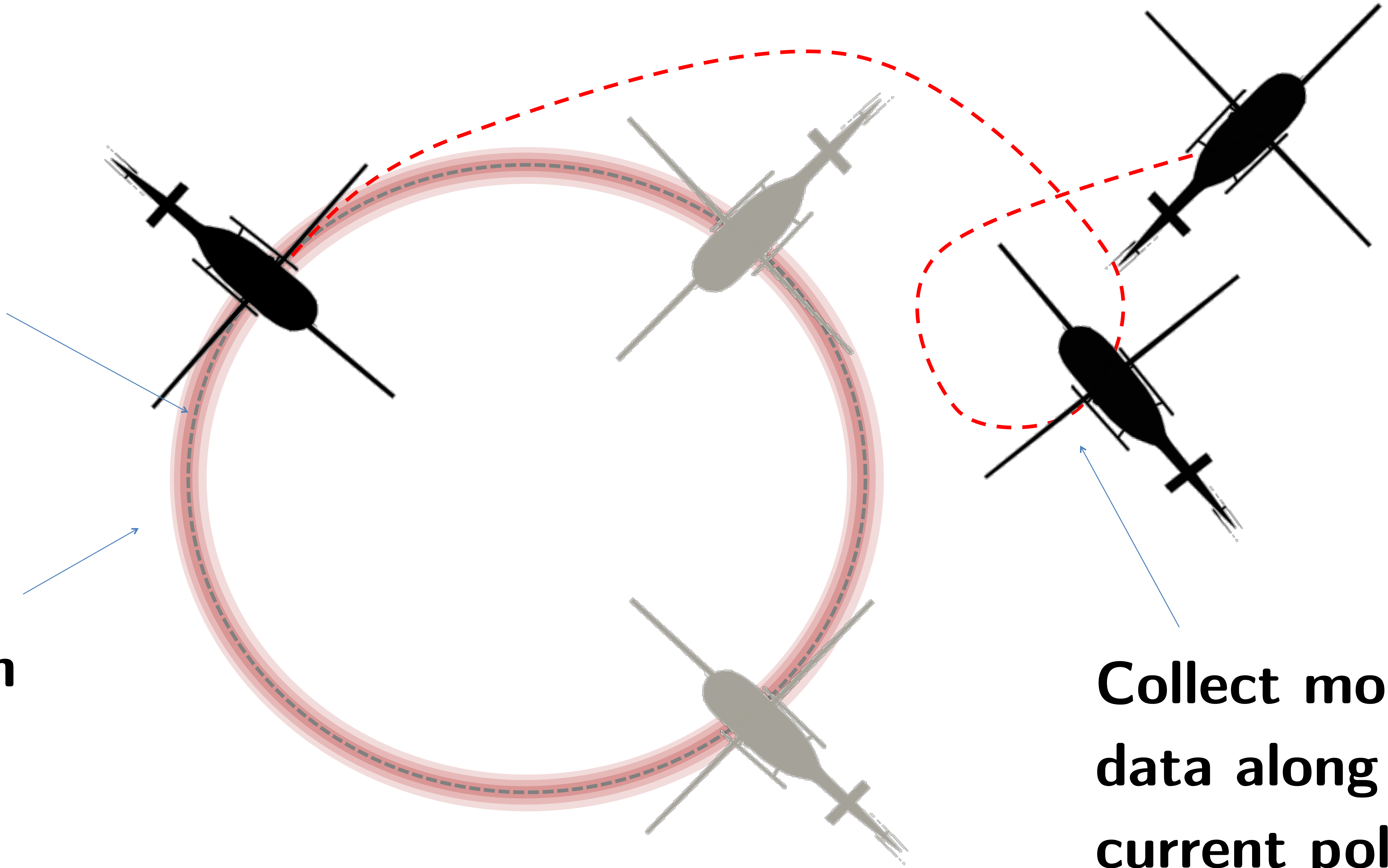
Mismatch amplified by planning



Data gathering policy

Planner wants to go here

Intuition: Improve model where policy goes

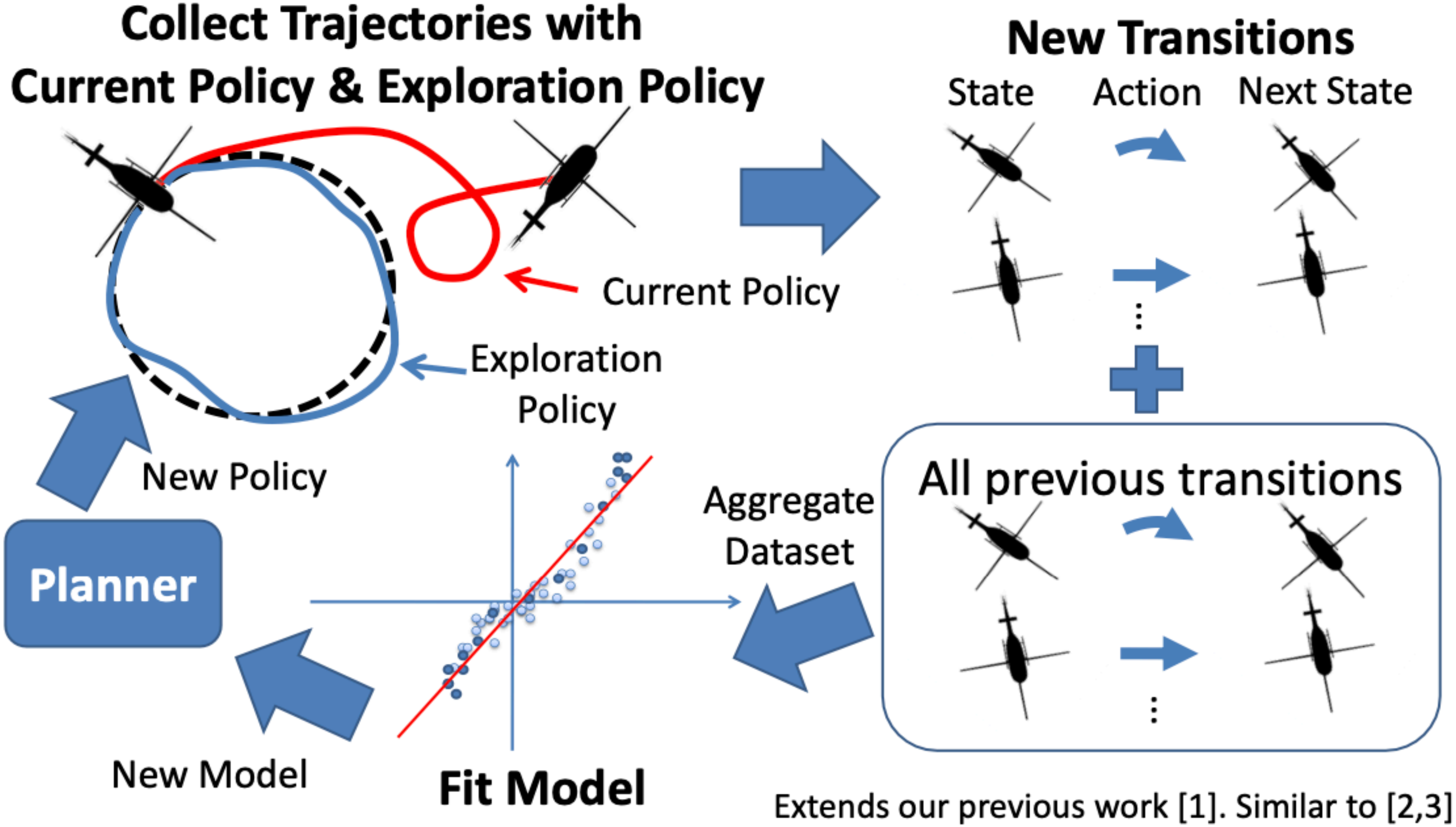


Desired Trajectory

Exploration Distribution

Collect more data along current policy's trajectory

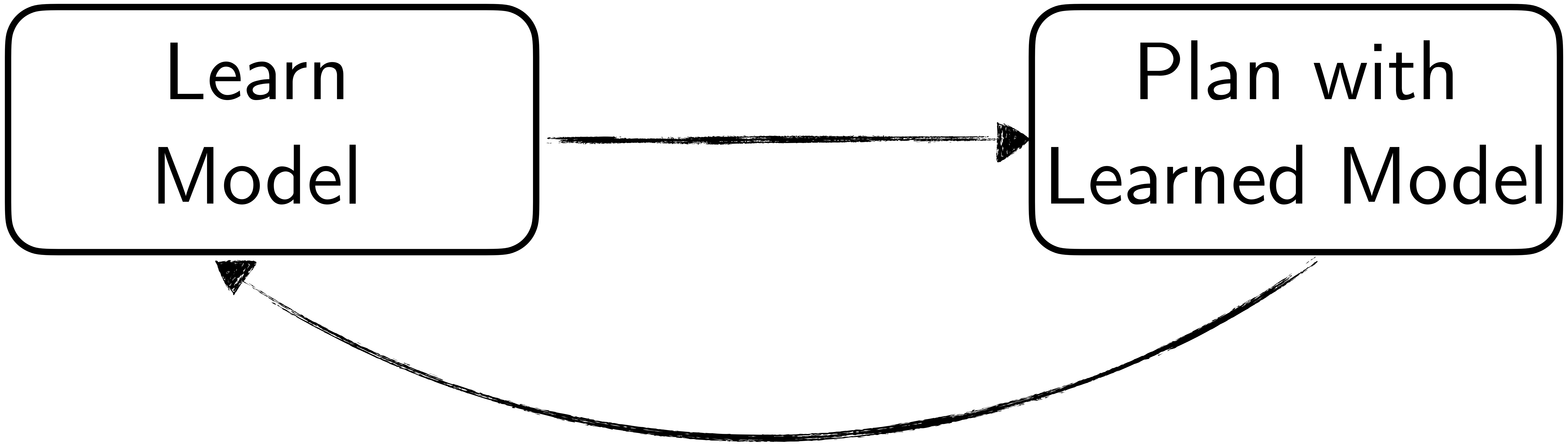
A simple algorithm: Dagger for System Identification



Extends our previous work [1]. Similar to [2,3]

Dagger works!

Planning is expensive …
Do we really have to run planning in an inner loop



Learn Model

Plan with Learned Model

No! Model need only help you correct

(a) Initial shot is off

(b) Aim at E_2 instead of E_1

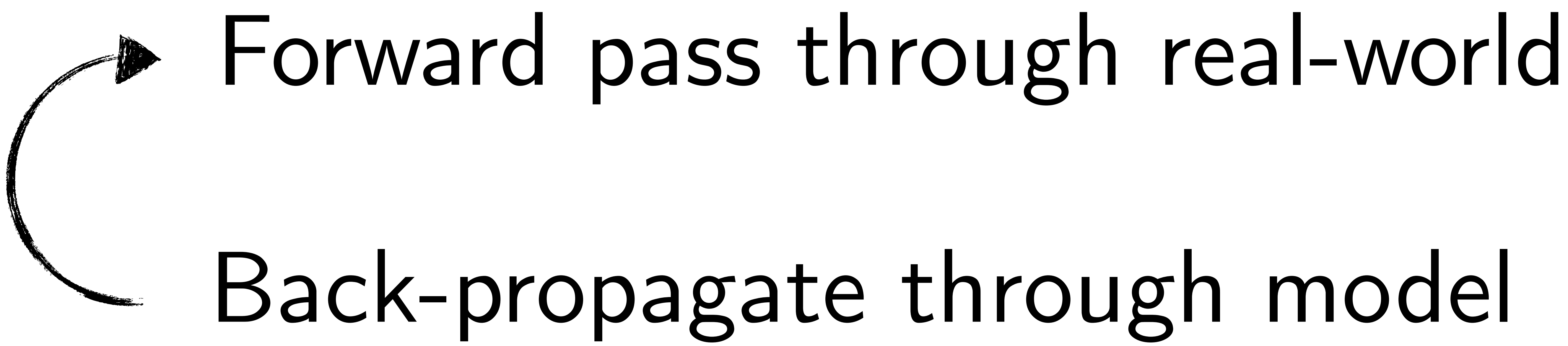
(c) Aim at E_3 and hit the bull's eye

Iterative Learning Control (ILC)

(a) Initial shot is off

(b) Aim at E_2 instead of E_1

(c) Aim at E_3 and hit the bull's eye



Forward pass through real-world

Back-propagate through model

The DREAMER Algorithm

DREAMER Overview

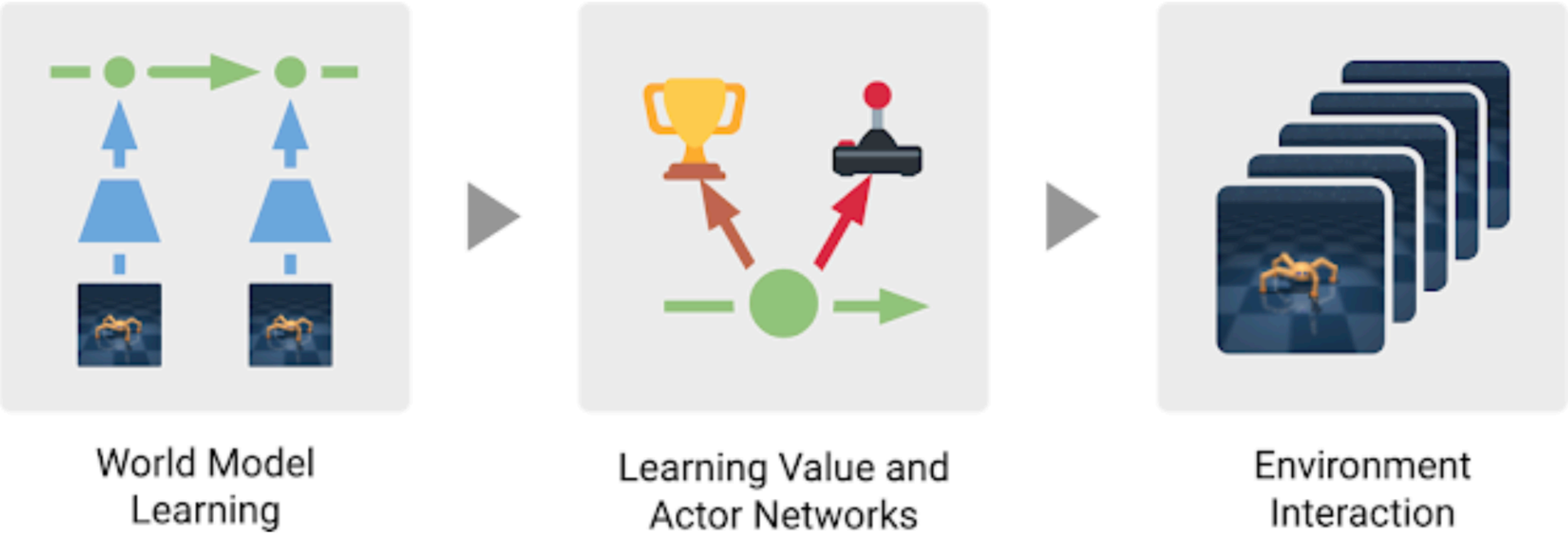
DREAM TO CONTROL: LEARNING BEHAVIORS BY LATENT IMAGINATION

Danijar Hafner *
University of Toronto
Google Brain

Timothy Lillicrap
DeepMind

Jimmy Ba
University of Toronto

Mohammad Norouzi
Google Brain



The three processes of the Dreamer agent. The world model is learned from past experience. From predictions of this model, the agent then learns a value network to predict future rewards and an actor network to select actions. The actor network is used to interact with the environment.

DREAMER: Learning World Model

DREAMER: Learning World Model

Dreamer: Learning Actor Critic from model

o_1

DREAMER: Results

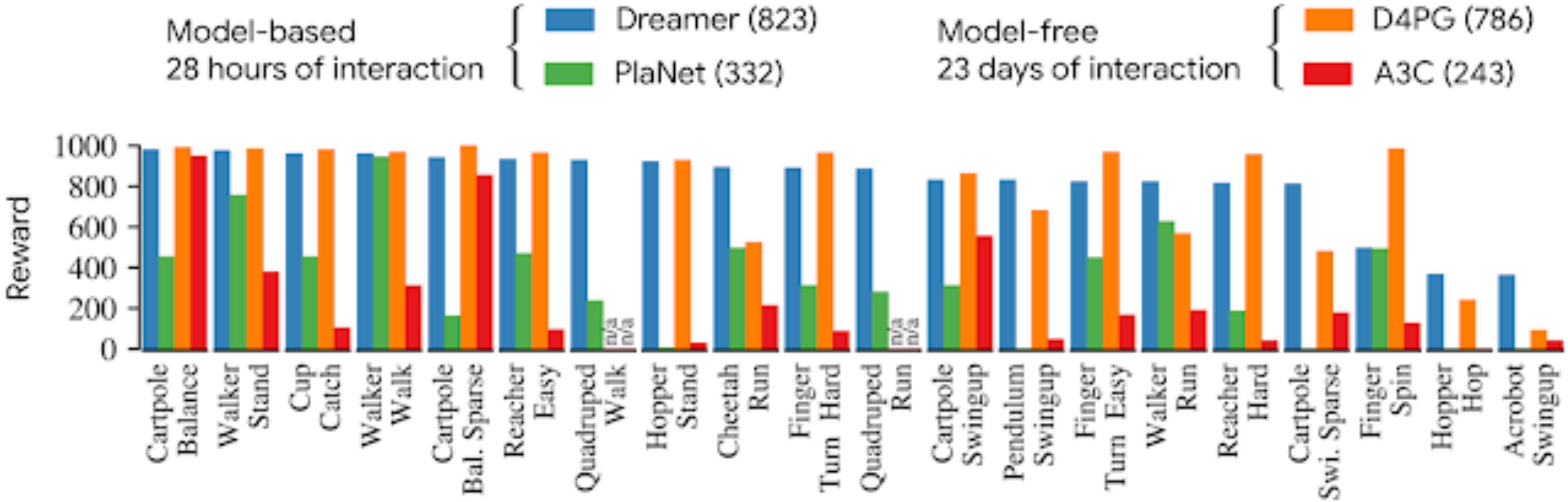
Sparse Cartpole

Acrobot Swingup

Hopper Hop

Walker Run

Quadruped Run



What makes a model good?

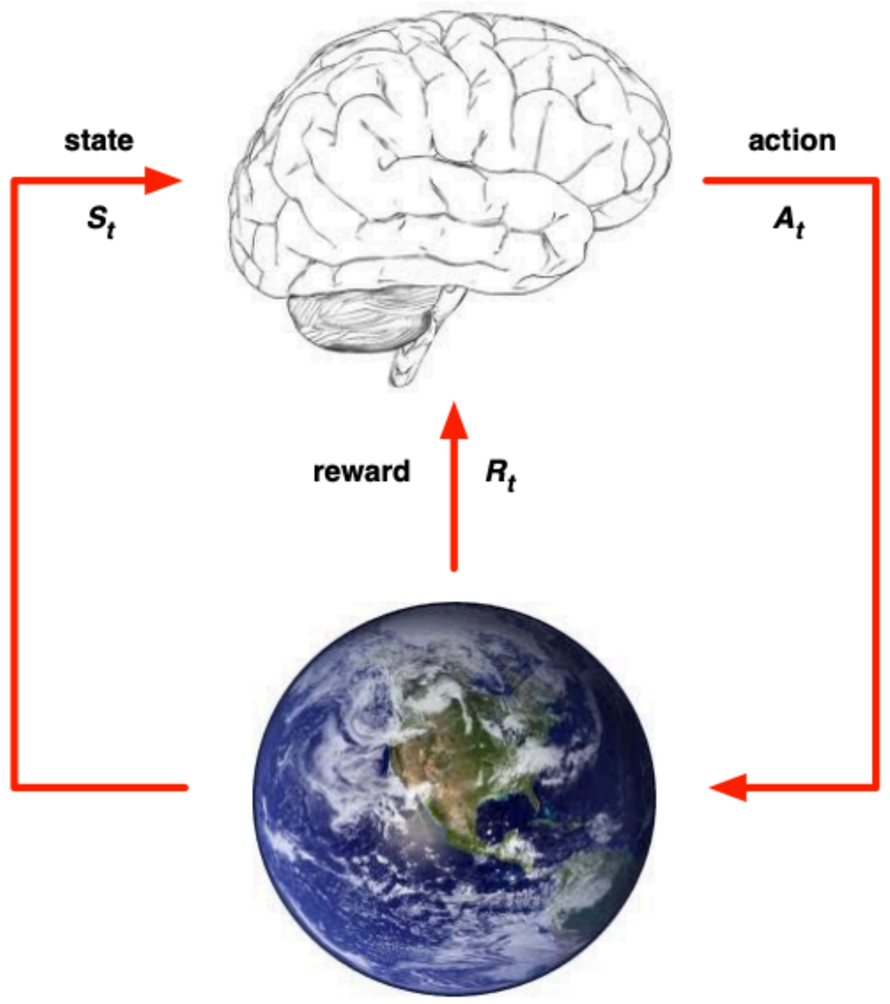
Model-Based OR Model Free?

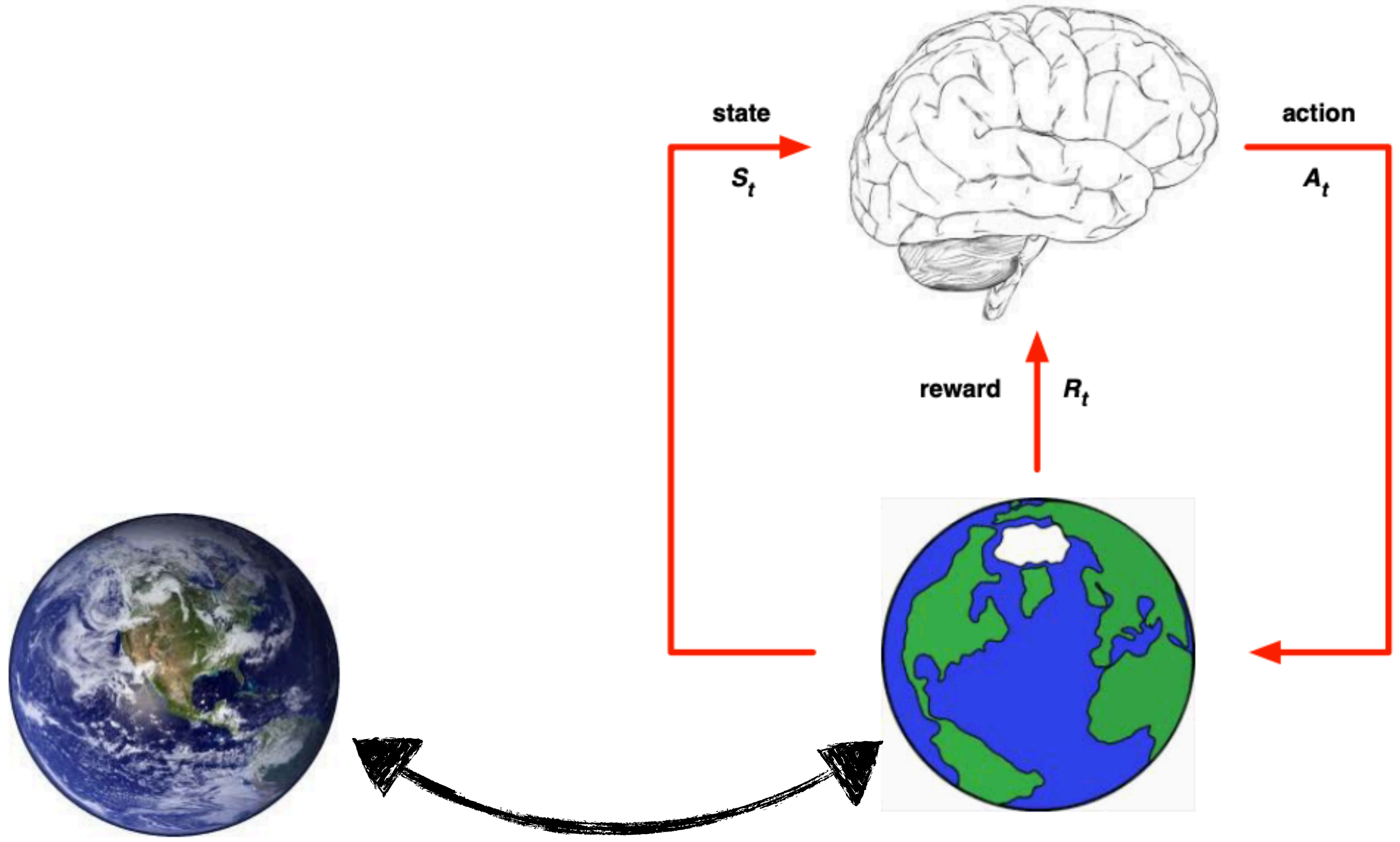
Model Free

Model Based

Directly learn π or $Q(s, a)$

Learn a model $P(s' | s, a)$, plan with model to find π





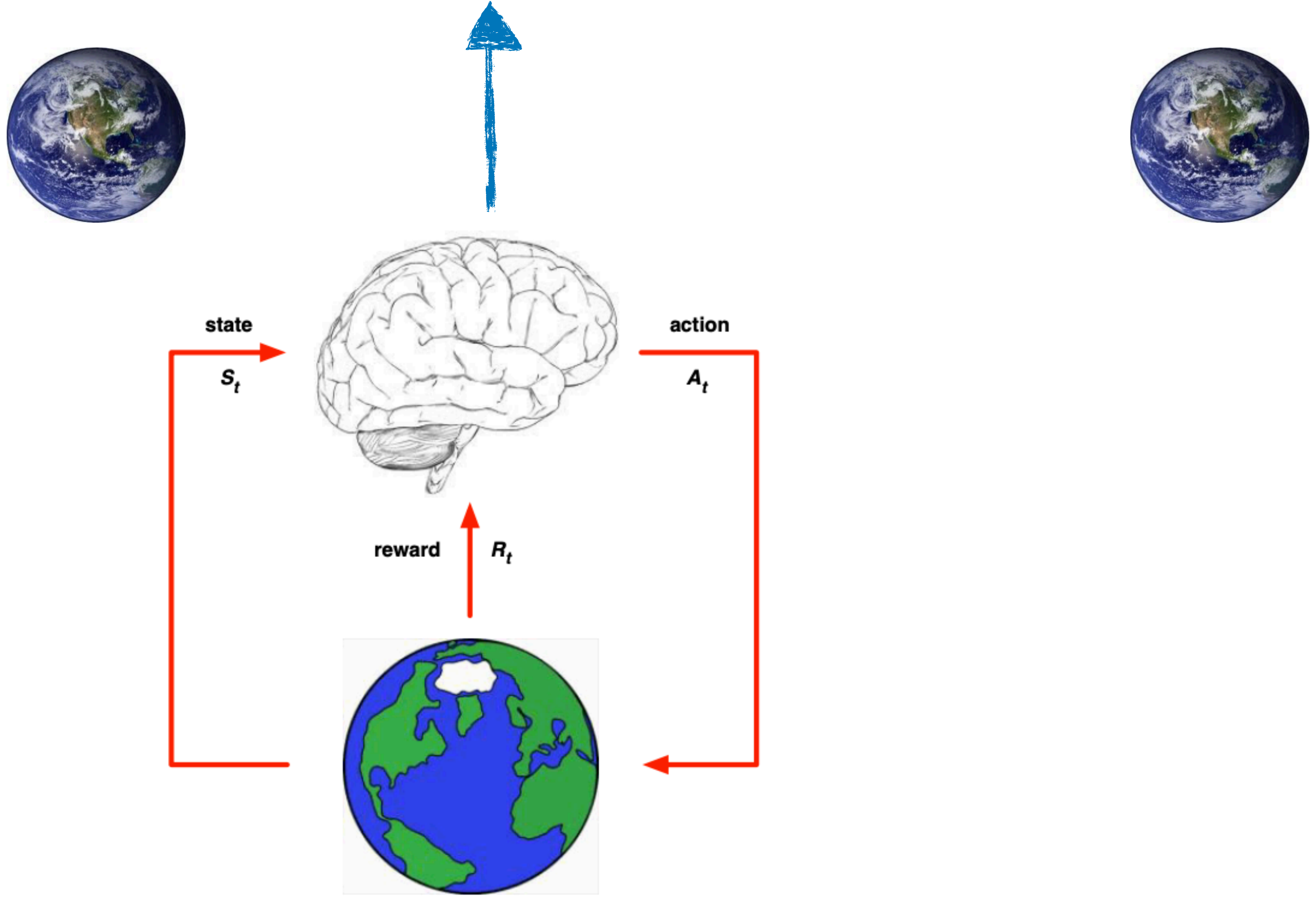
(From Lec 1.)

Lesson #2

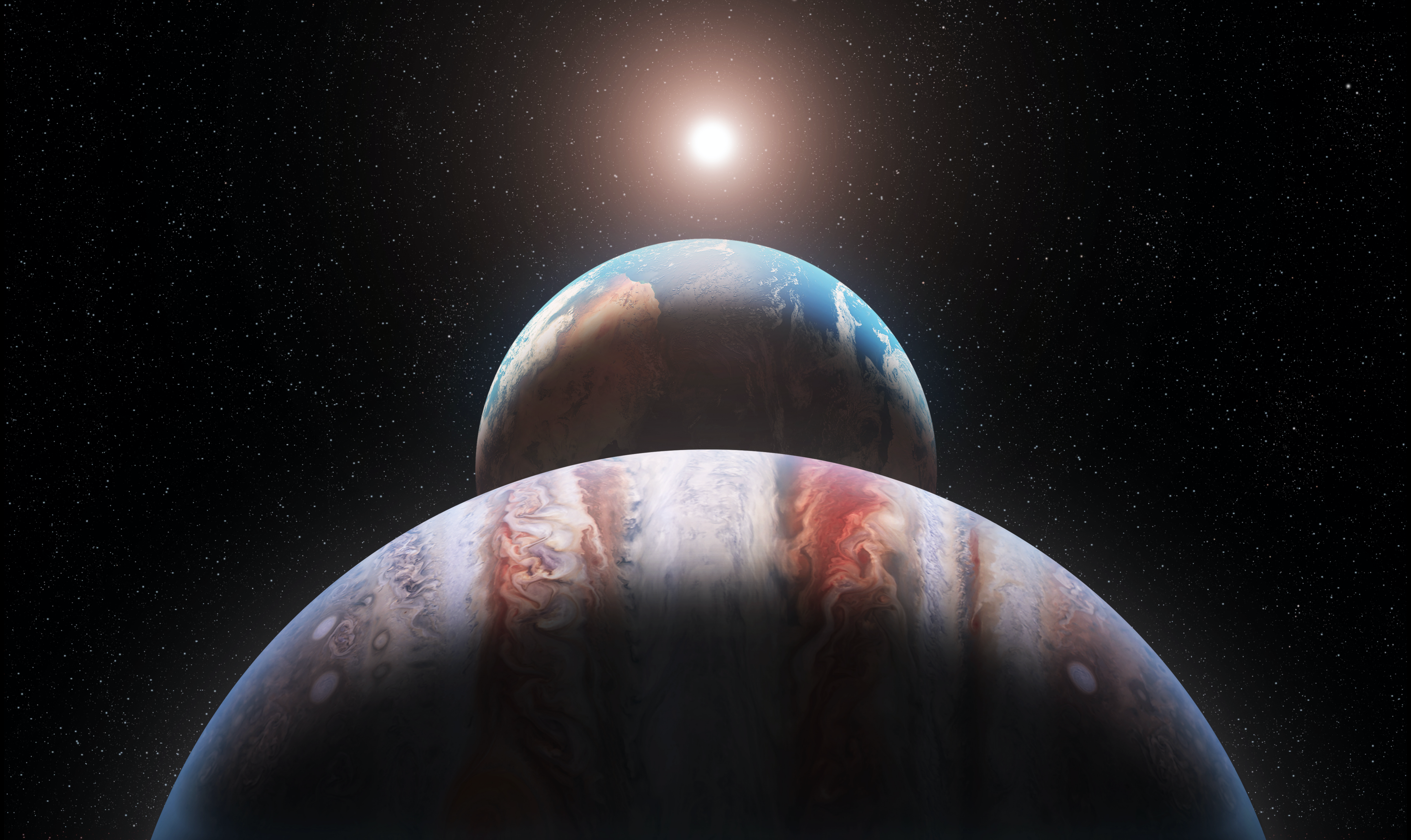
Models are useful fictions

What makes a model good?

$$J_{M^*}(\hat{\pi}) - J_{M^*}(\pi^*)$$



The *Double* Simulation Lemma



Model Learning: It's only a game!

$$\left| J_{M^*}(\hat{\pi}) - J_{M^*}(\pi^*) \right|$$

$$= \min_{\hat{M}} \max_{V} \sum_{t=1}^{T} \mathbb{E}_{s_t \sim d^{\hat{\pi}}_{M^*}} \left| \sum_{s_{t+1}} M^*(s_{t+1}|s_t,a_t) V(s_{t+1}) - \hat{M}(s_{t+1}|s_t,a_t) V(s_{t+1}) \right\|$$

😇 😈

$$+$$

$$\sum_{t=1}^{T} \mathbb{E}_{s_t \sim d^{\pi^*}_{M^*}} \left| \sum_{s_{t+1}} M^*(s_{t+1}|s_t,a_t) V(s_{t+1}) - \hat{M}(s_{t+1}|s_t,a_t) V(s_{t+1}) \right\|$$

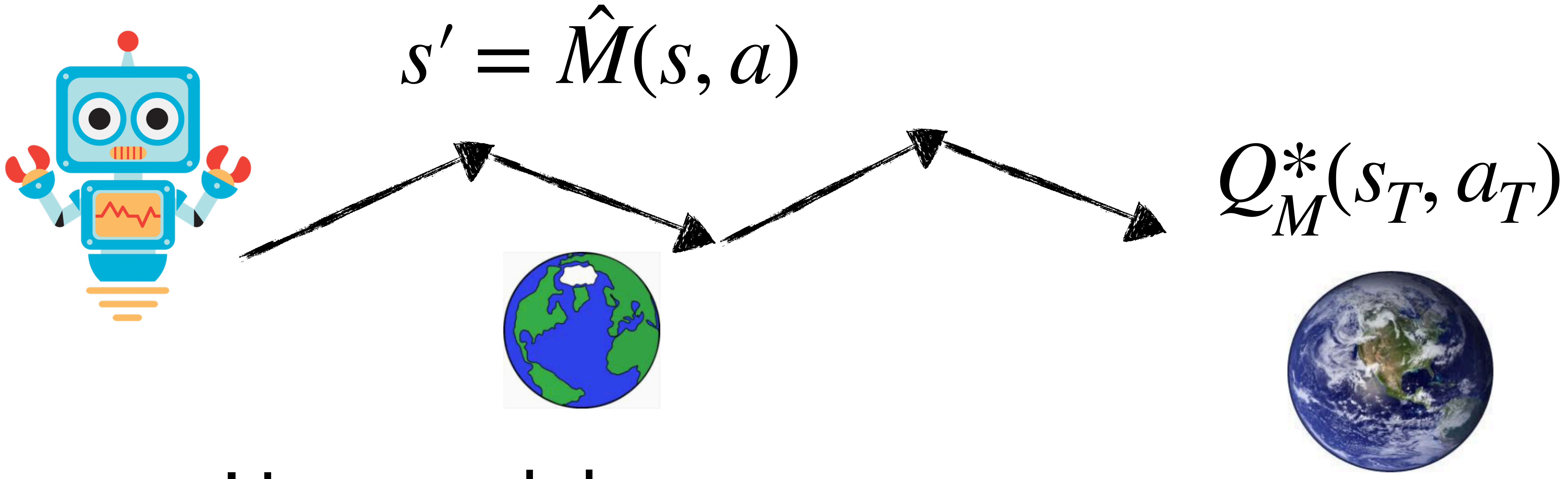
where $\hat{\pi} = \text{PLANNER}(\hat{M})$

If models are just a means to an end (performance difference) .. can we get the best of model-based and model-free?

Model-Based ~~OR~~ *AND* Model Free

Model Based

Model Free



$s' = \hat{M}(s, a)$

$Q^*_M(s_T, a_T)$

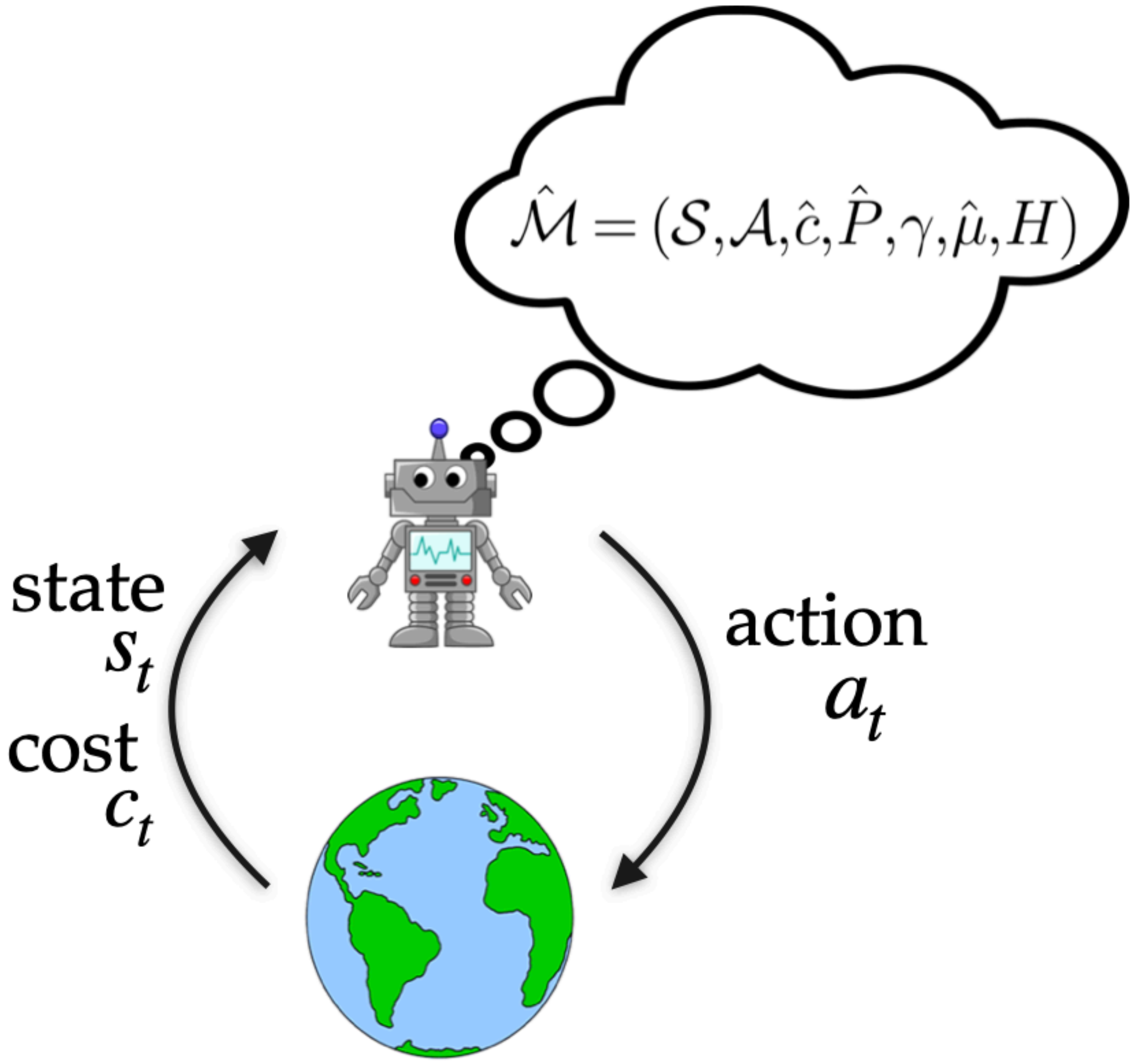
Use model to plan

Use model-free terminal value function

Blending MPC and Value Function

BLENDING MPC & VALUE FUNCTION APPROXIMATION FOR EFFICIENT REINFORCEMENT LEARNING

Mohak Bhardwaj[1] **Sanjiban Choudhury**[2] **Byron Boots**[1]
[1] University of Washington [2] Aurora Innovation Inc.



Step 1: Construct local Q-function estimate

$$Q_H^\phi(s,a) = \mathbb{E}_{\mu_{\hat{\mathcal{M}}}^{\pi_\phi}}\left[\sum_{i=0}^{H-1} \gamma^i \hat{c}(s_i,a_i) + \gamma^H \underbrace{\hat{Q}(s_H,a_H)}_{\text{terminal Q-estimate}} \,\middle|\, s_0 = s, a_0 = a\right]$$

Step 2: Optimize policy parameters

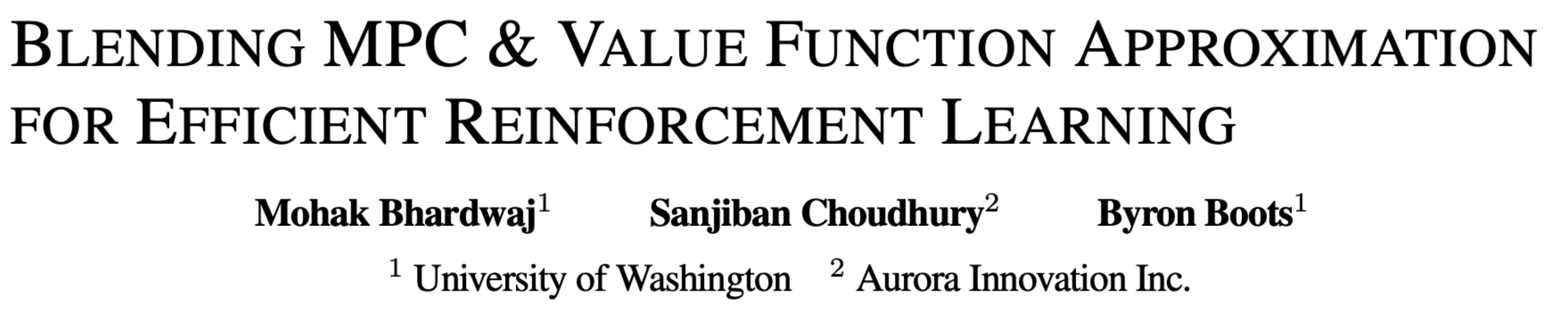
$$\phi_t^* = \operatorname*{argmin}_{\phi} Q_H^\phi(s_t, \pi_\phi(s_t))$$

Step 3: Sample action to execute

$$a_t \sim \pi_{\phi_t^*}(s_t)$$

MPC as Q-function Approximation

Blending MPC and Value Function



BLENDING MPC & VALUE FUNCTION APPROXIMATION FOR EFFICIENT REINFORCEMENT LEARNING

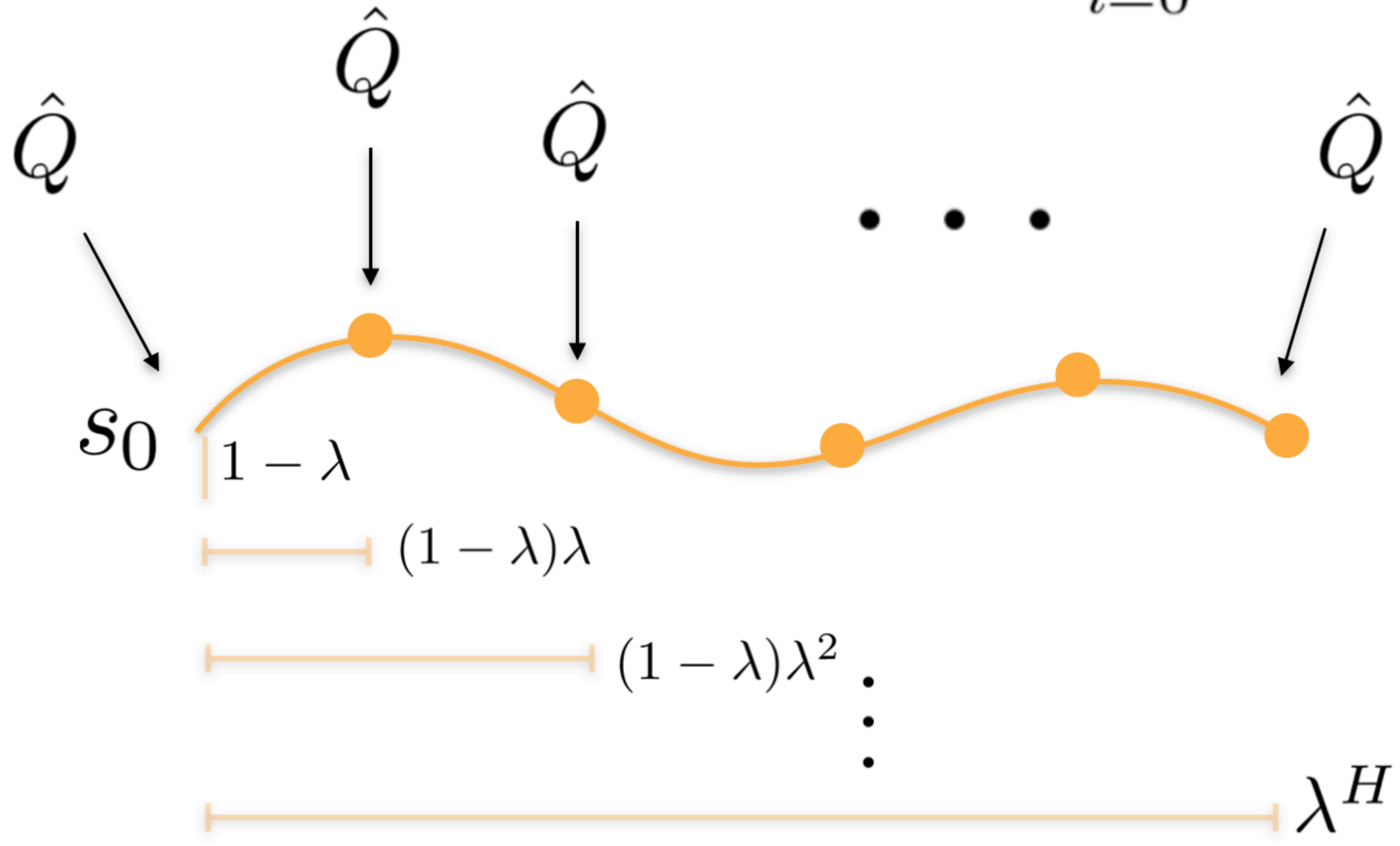
Mohak Bhardwaj[1] **Sanjiban Choudhury**[2] **Byron Boots**[1]
[1] University of Washington [2] Aurora Innovation Inc.

At every step along horizon (recursive)

$$\underbrace{Q^{\lambda}(s_i,a_i)}_{\text{current blended estimate}} = (1-\lambda)\underbrace{\hat{Q}(s_i,a_i)}_{\text{model-free}} + \lambda\Big(\underbrace{\hat{c}(s_i,a_i)}_{\text{model-based}} + \gamma\,\underbrace{Q^{\lambda}(s_{i+1},a_{i+1})}_{\text{future blended estimate}}\Big)$$

Un-rolling forward

$$Q_H^{\lambda}(s,a) = (1-\lambda)\sum_{i=0}^{H-1}\lambda^i Q_i^{\phi}(s,a) + \lambda^H Q_H^{\phi}(s,a)$$



Similar to forward-view $TD(\lambda)$ but trades-off model and value function bias against variance

Refer paper for connection to cost-shaping and Generalized Advantage Estimation [4]

Add model free value estimate at *every* timestep