Model-based Reinforcement Learning

Sanjiban Choudhury

Cornell Bowers CIS
Computer Science

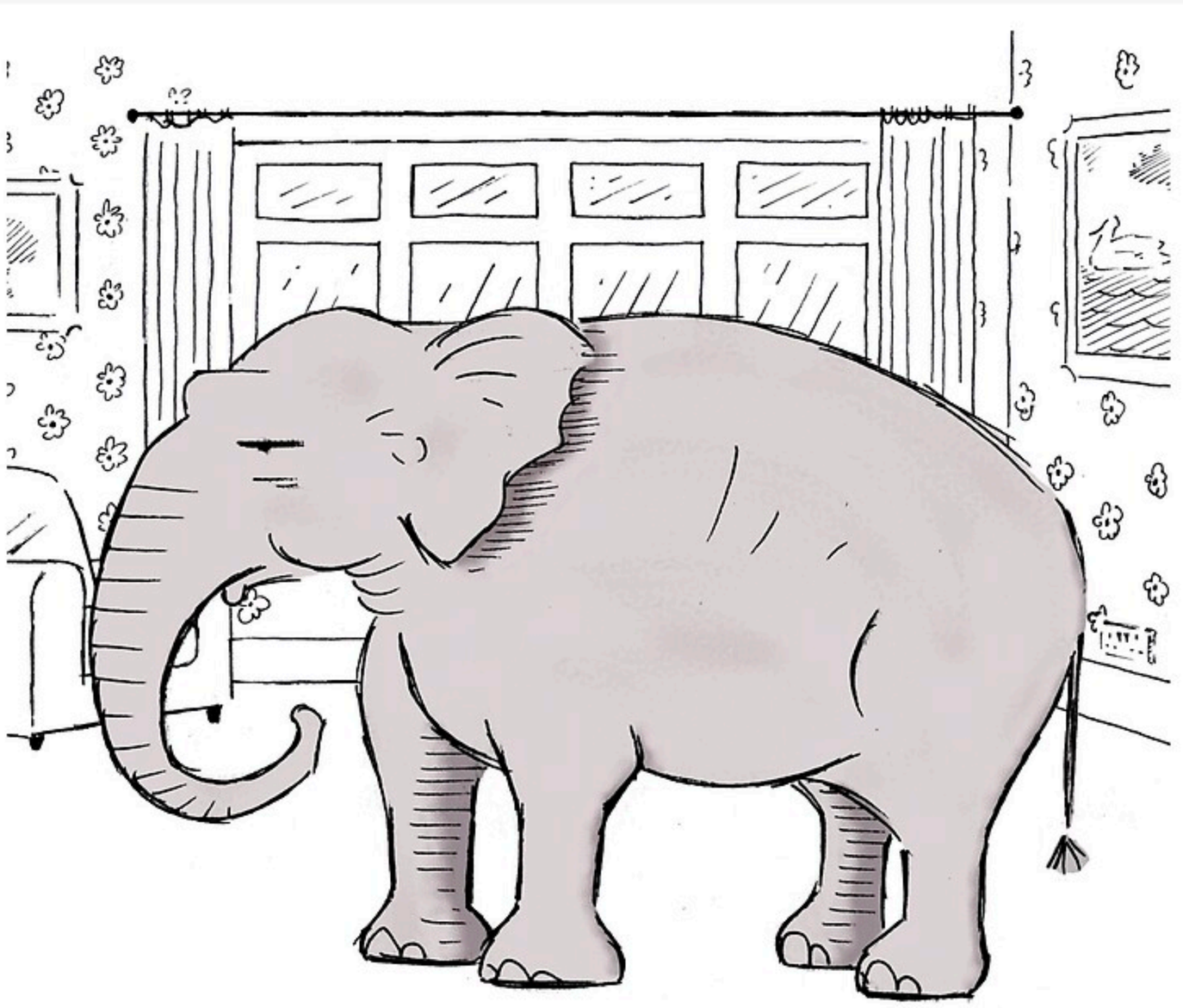
RL

=

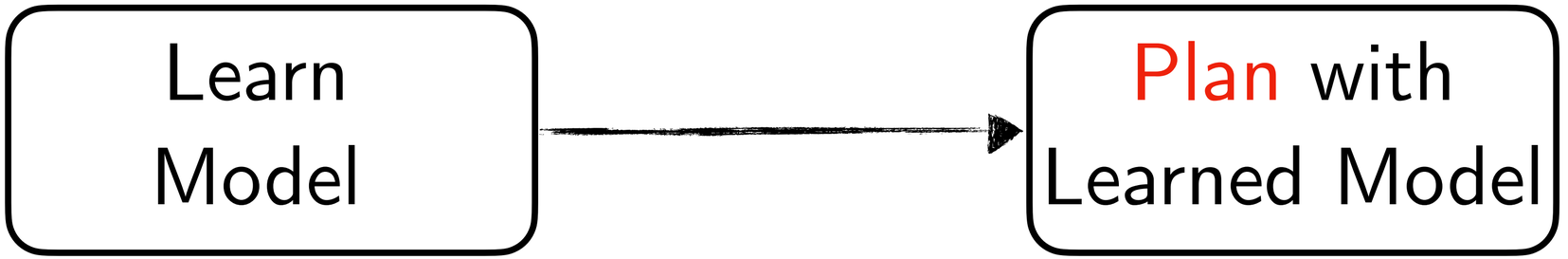
Learn model

+

Plan with model

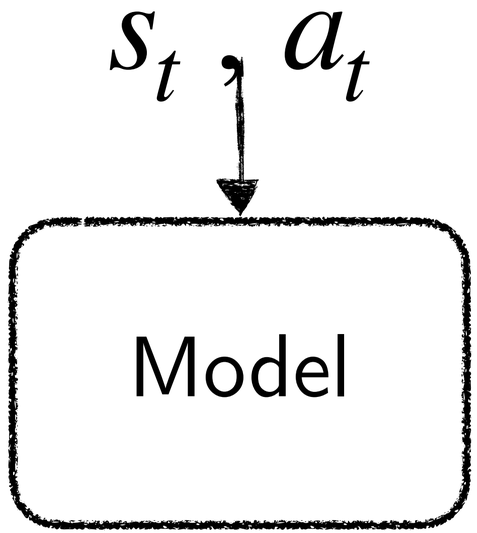


Model Based Reinforcement Learning

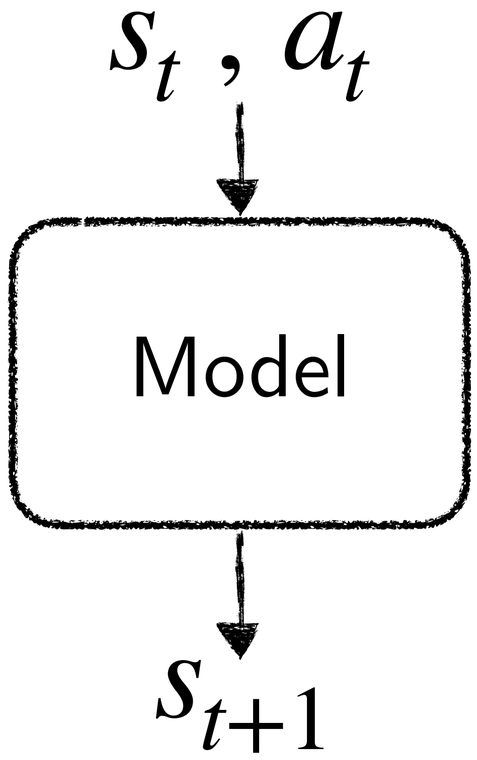


Models.

What is a model?



What is a model?



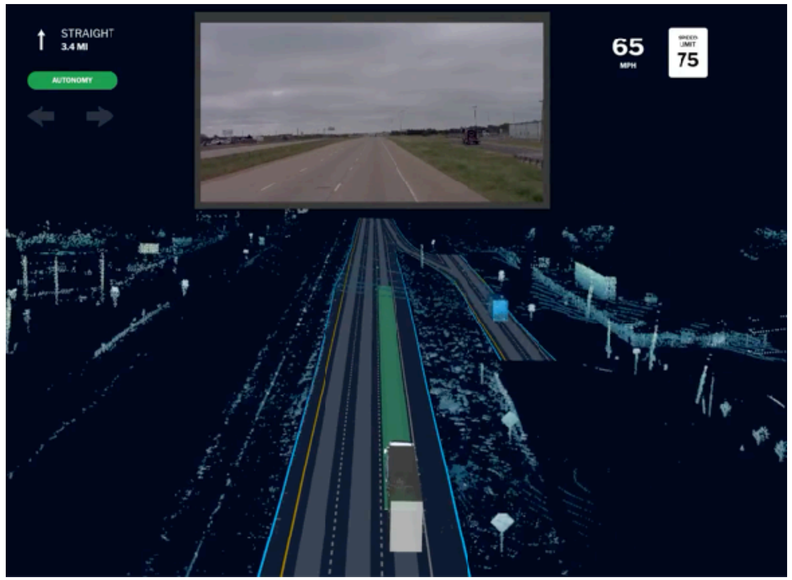
Why Model?

Models are *necessary*

Robots can't just try out random actions in the world!

Models are *necessary*

We invested heavily in simulators for helicopters and self-driving to verify behaviors before deployment



Models work in *theory*

Model-Based Reinforcement Learning with a Generative Model is Minimax Optimal

Alekh Agarwal
Microsoft
alekha@microsoft.com

Sham Kakade
University of Washington
sham@cs.washington.edu

Lin F. Yang
University of California, Los Angeles
linyang@ee.ucla.edu

April 7, 2020

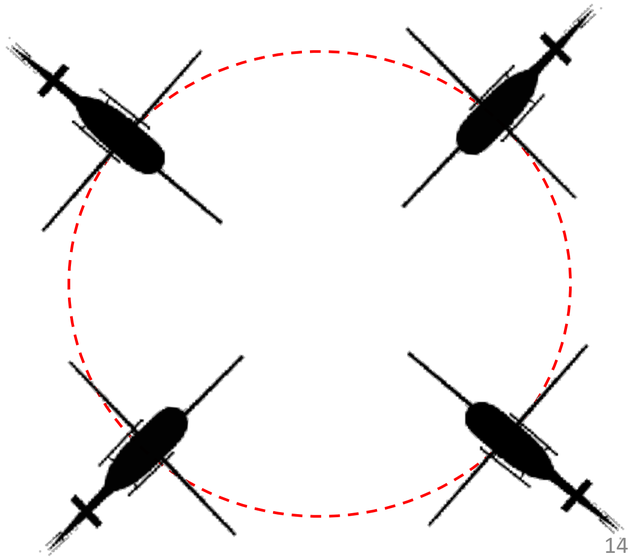
Models work in *practice*

Hafner et al. 2023

Learning Models.

Activity!

Example: Helicopter Aerobatics



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

Input / Output / Loss

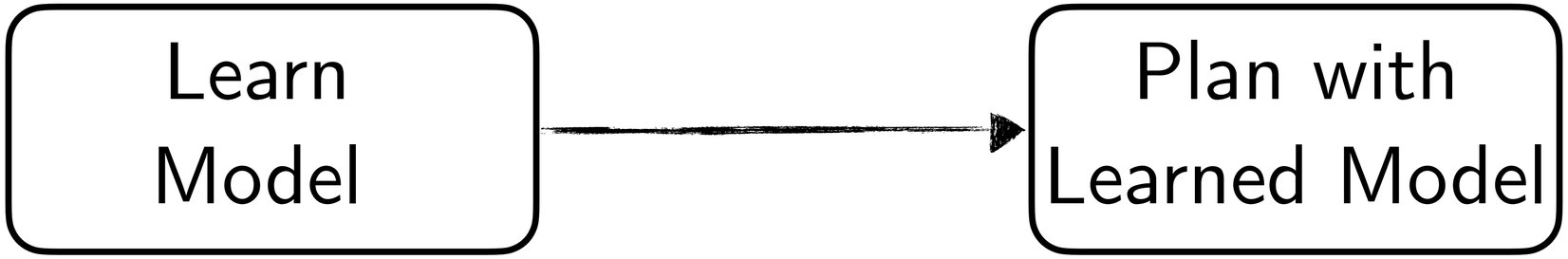
When poll is active respond at **PollEv.com/sc2582**

Send **sc2582** to **22333**

Think-Pair-Share

Think (30 sec): What model will you use for learning?
What planner would you use to execute a maneuver?

Pair: Find a partner



Share (45 sec): Partners exchange ideas

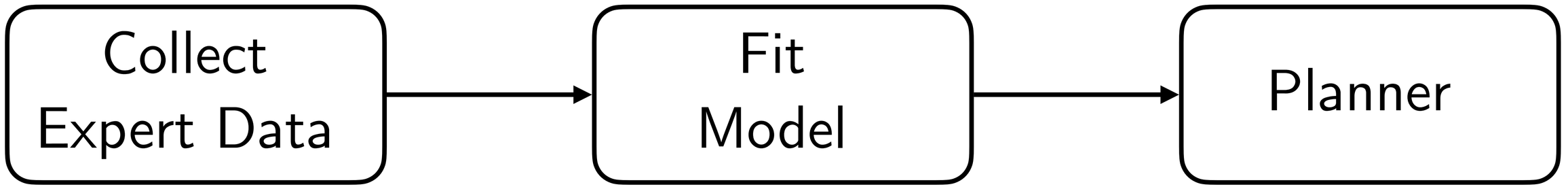
Question: How do you collect data for learning model?

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

Strategy

Train a model on state actions visited by the expert!

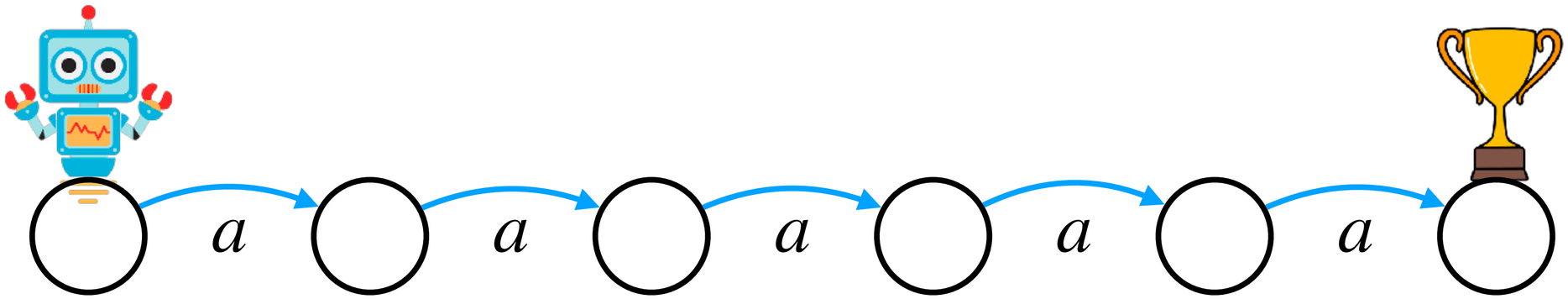
Model Based RL v1.0



*If I **perfectly** fit a model (i.e. training error zero),*
this should work, right?

World

$s' = M^*(s, a)$



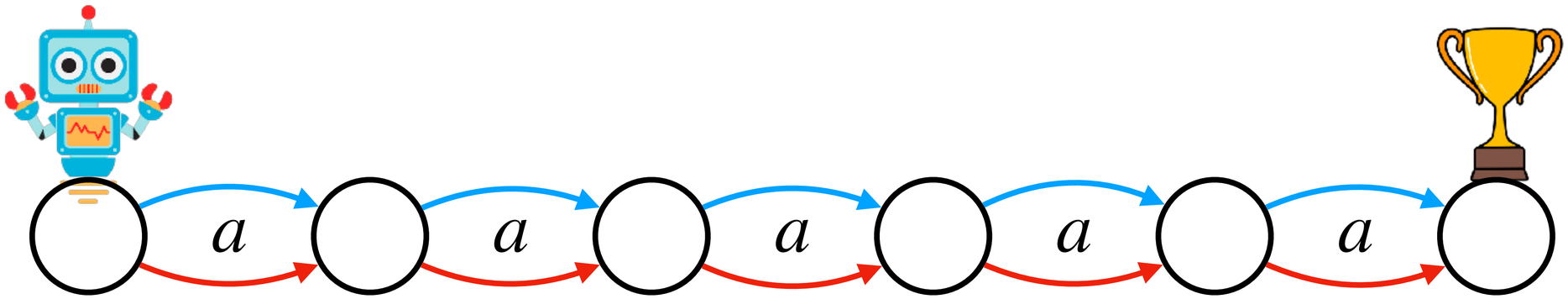
Experts picks action a to go to the goal

Model

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

World

$s' = M^*(s, a)$



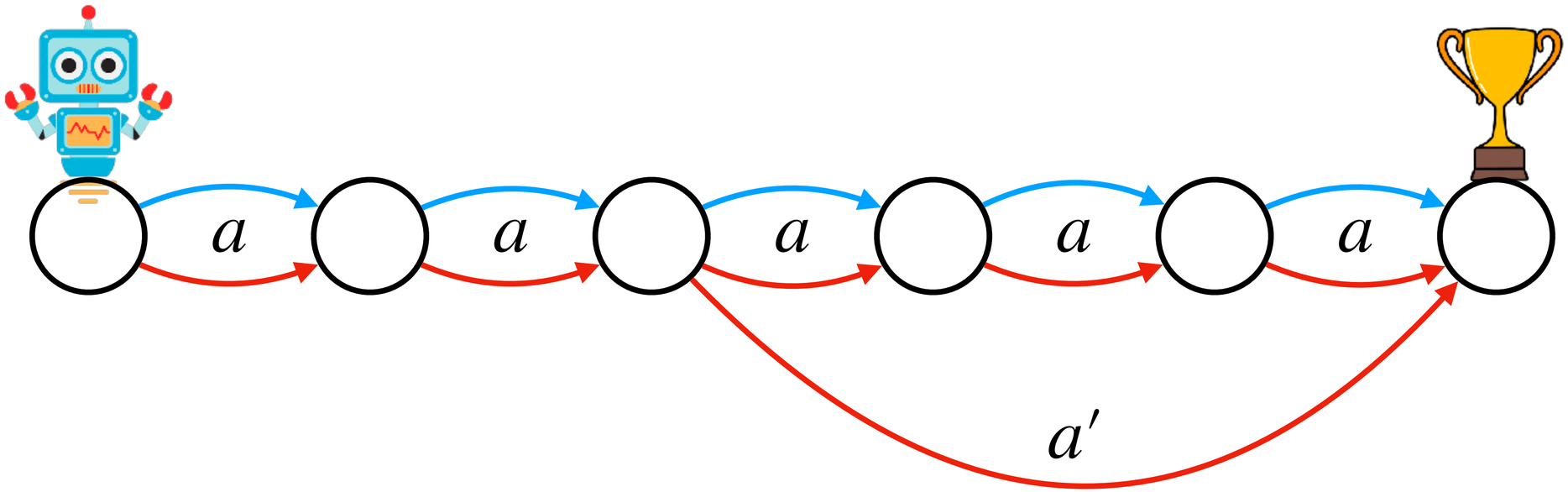
Model agrees with world, i.e. train error zero!

Model

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

World

$s' = M^*(s, a)$



What if the model is optimistic?

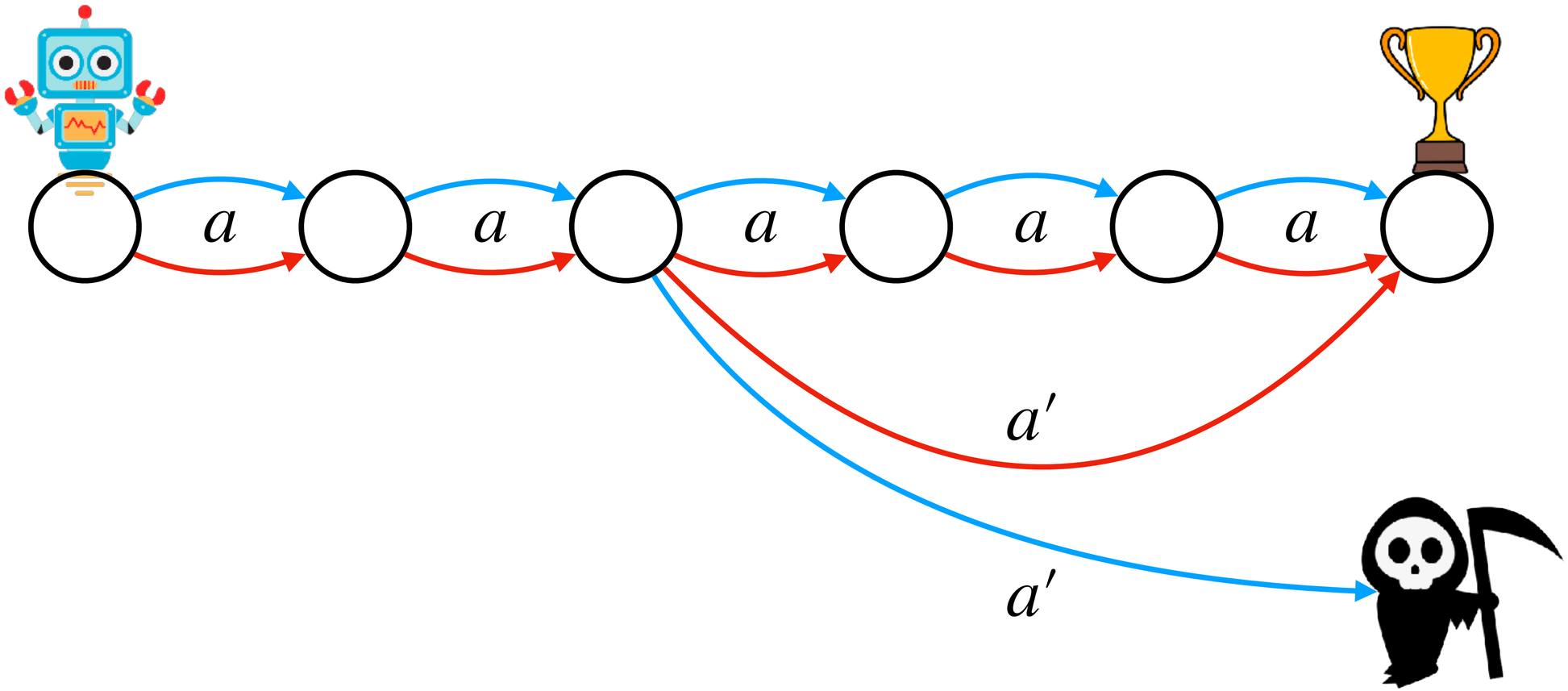
Predicts a short cut to the goal by taking action a'

Model

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

World

s'=$M^*(s,a)$



In reality the shortcut ends in death ...

Training on Expert Data

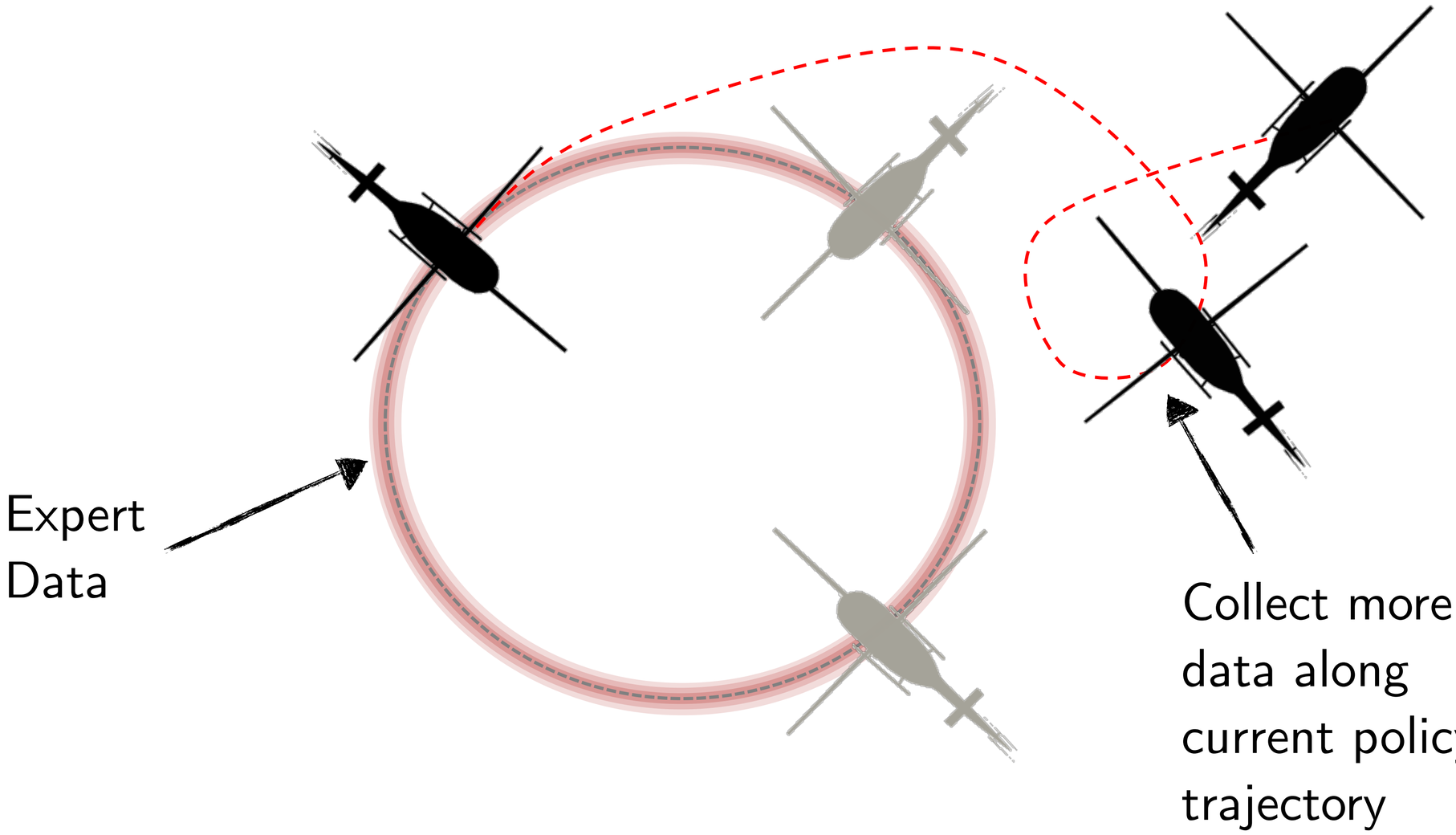
(From Ross and Bagnell, 2012)

Strategy

~~Train a model on state actions visited by the expert!~~

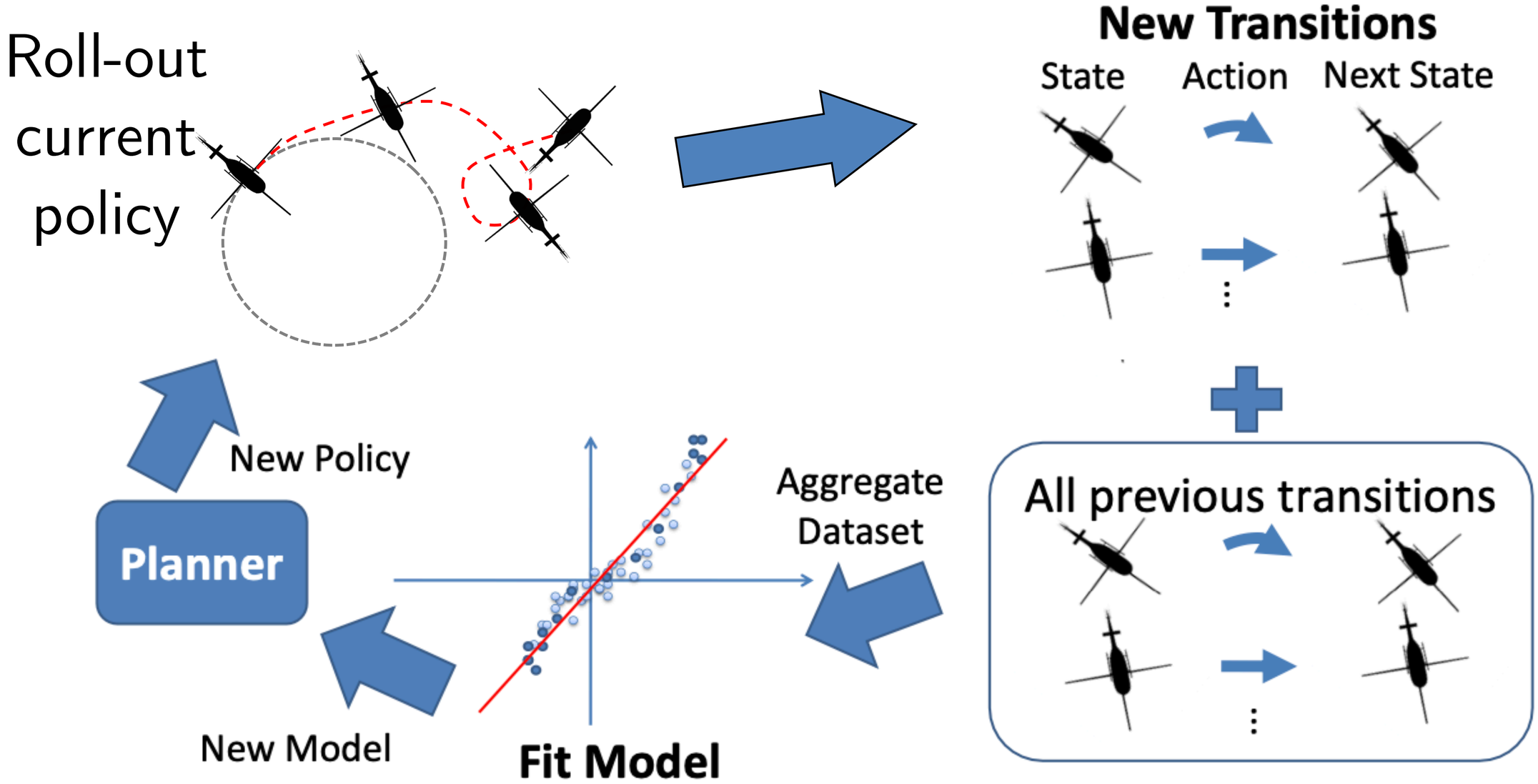
Train a model on state actions visited by the learner!

Improve model where policy goes



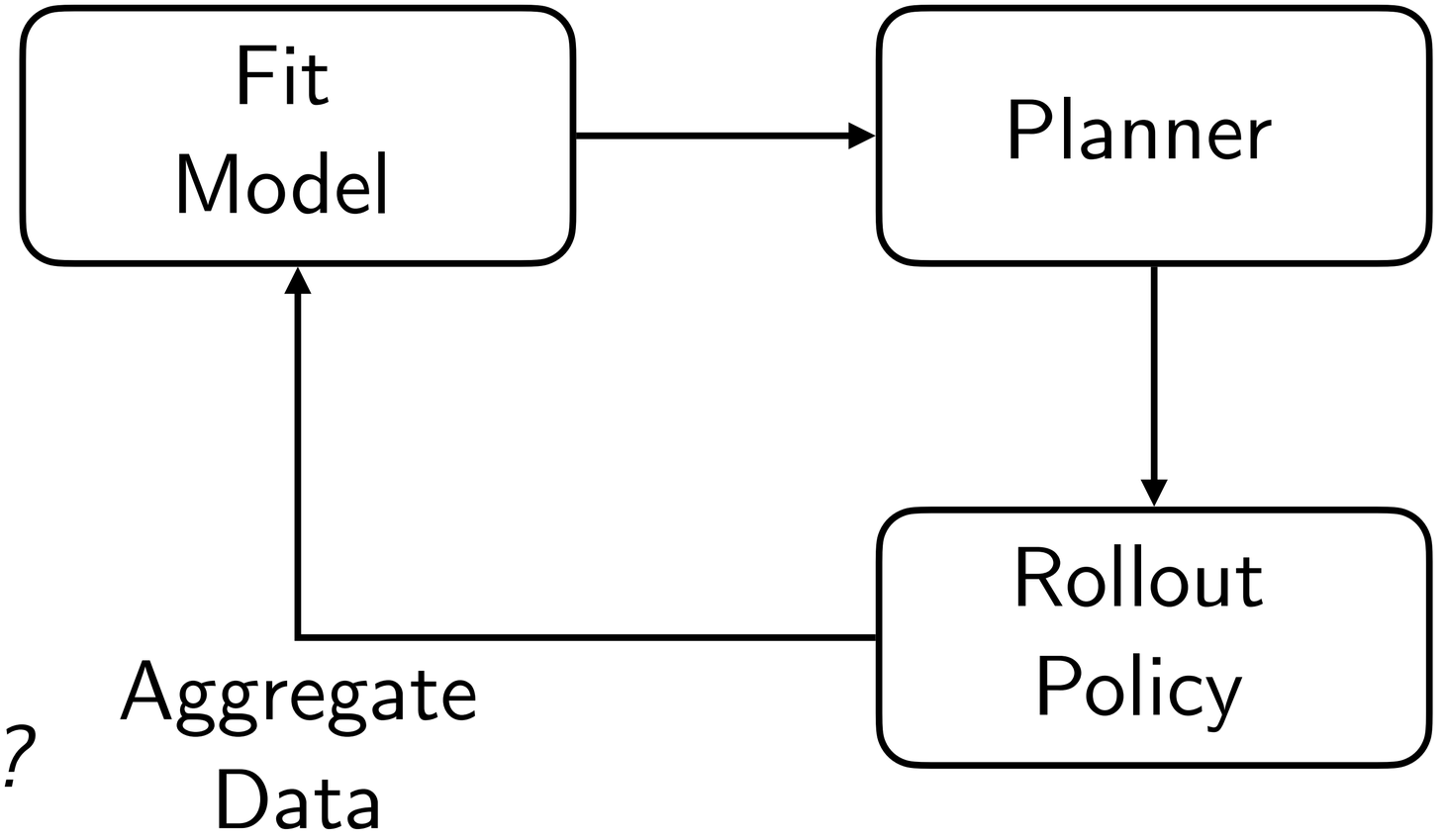
Don't we know an algorithm that does this?

DAGGER for Model-based RL!!



Model Based RL v2.0

*If I **perfectly** fit a model (i.e. training error zero), this should work, right?*

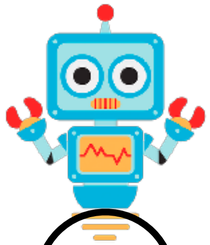


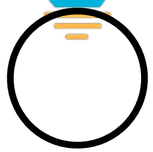
Model

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

World

$s' = M^*(s, a)$



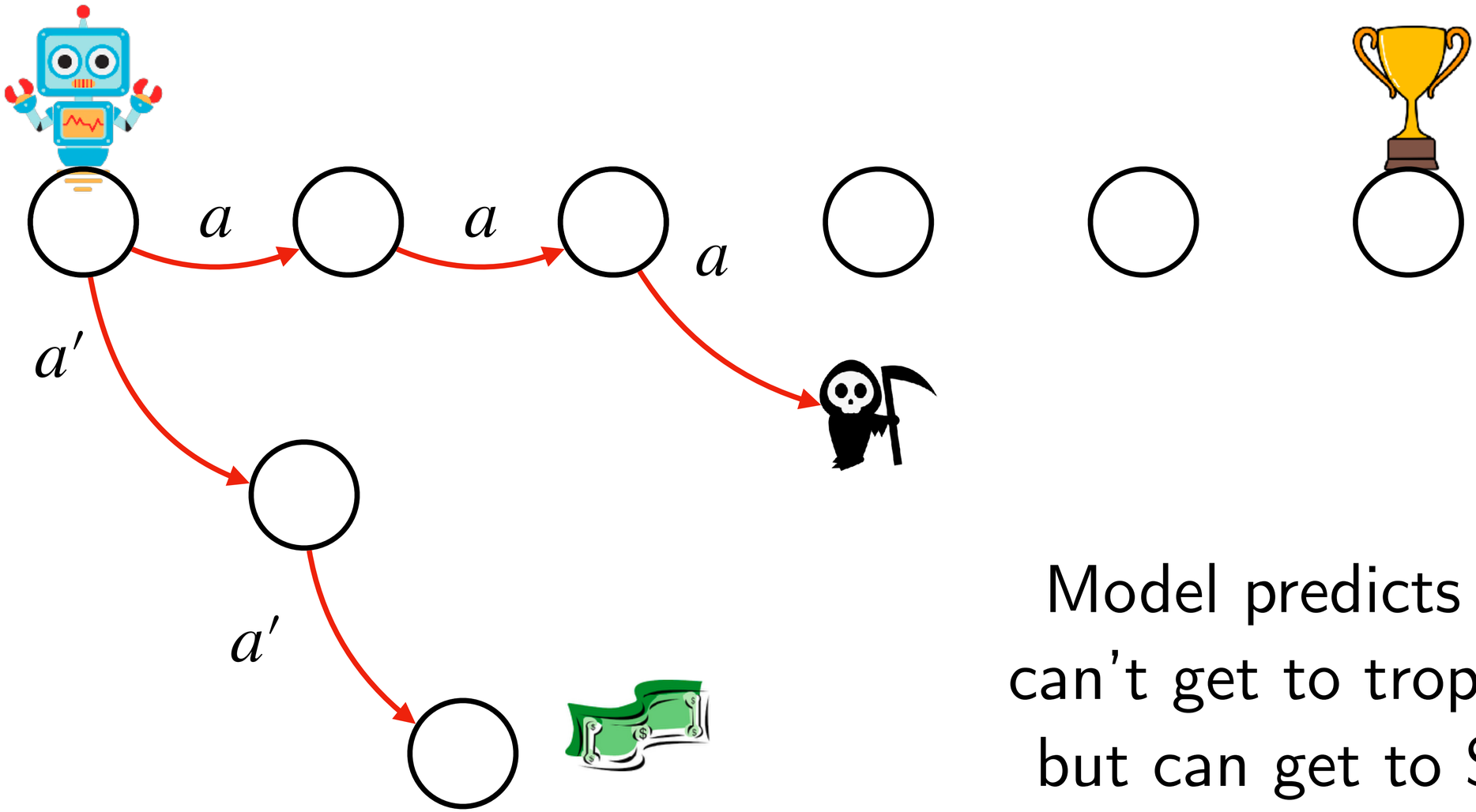


Model

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

World

$s' = M^*(s, a)$



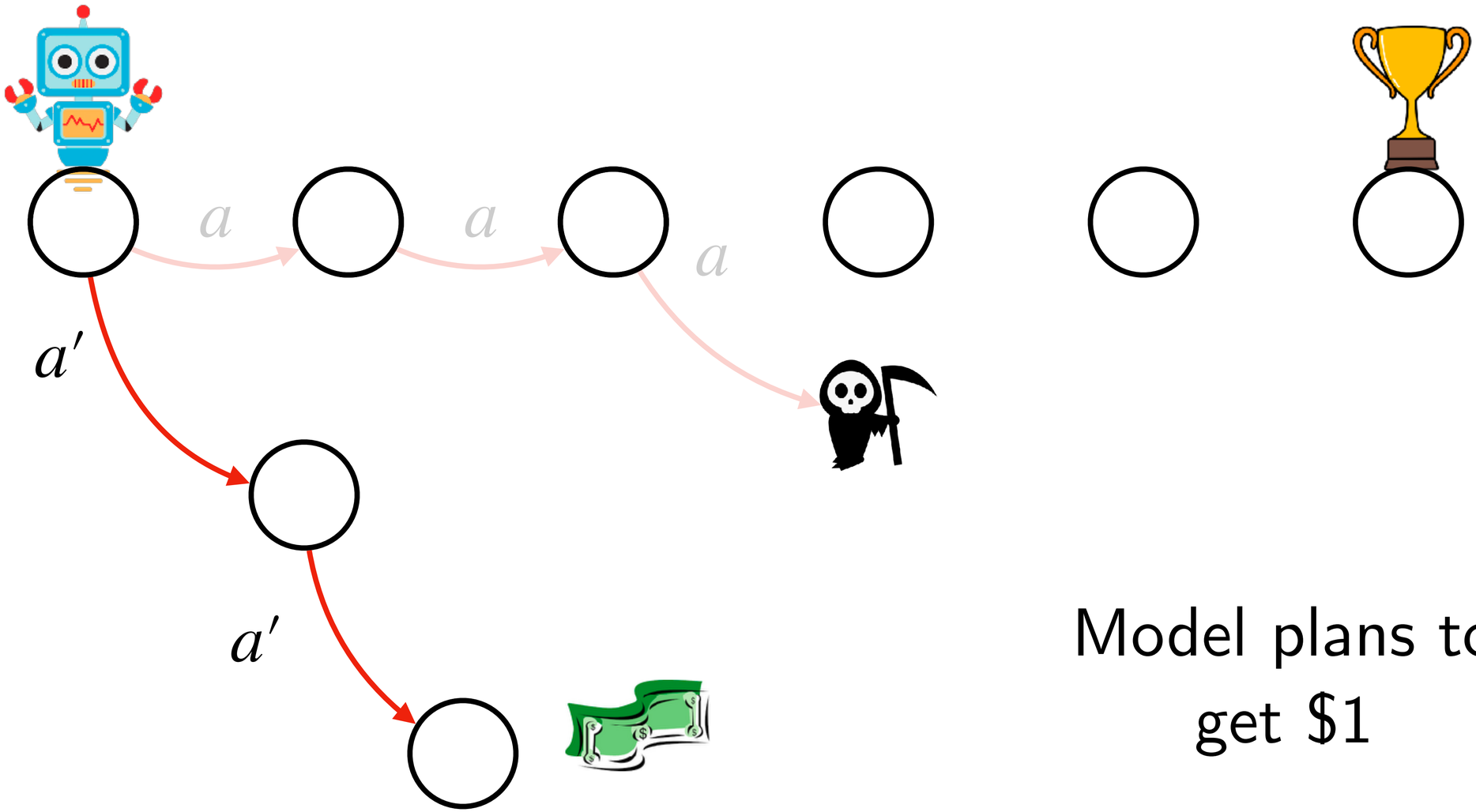
Model predicts it can't get to trophy, but can get to $1

Model

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

World

$s' = M^*(s, a)$



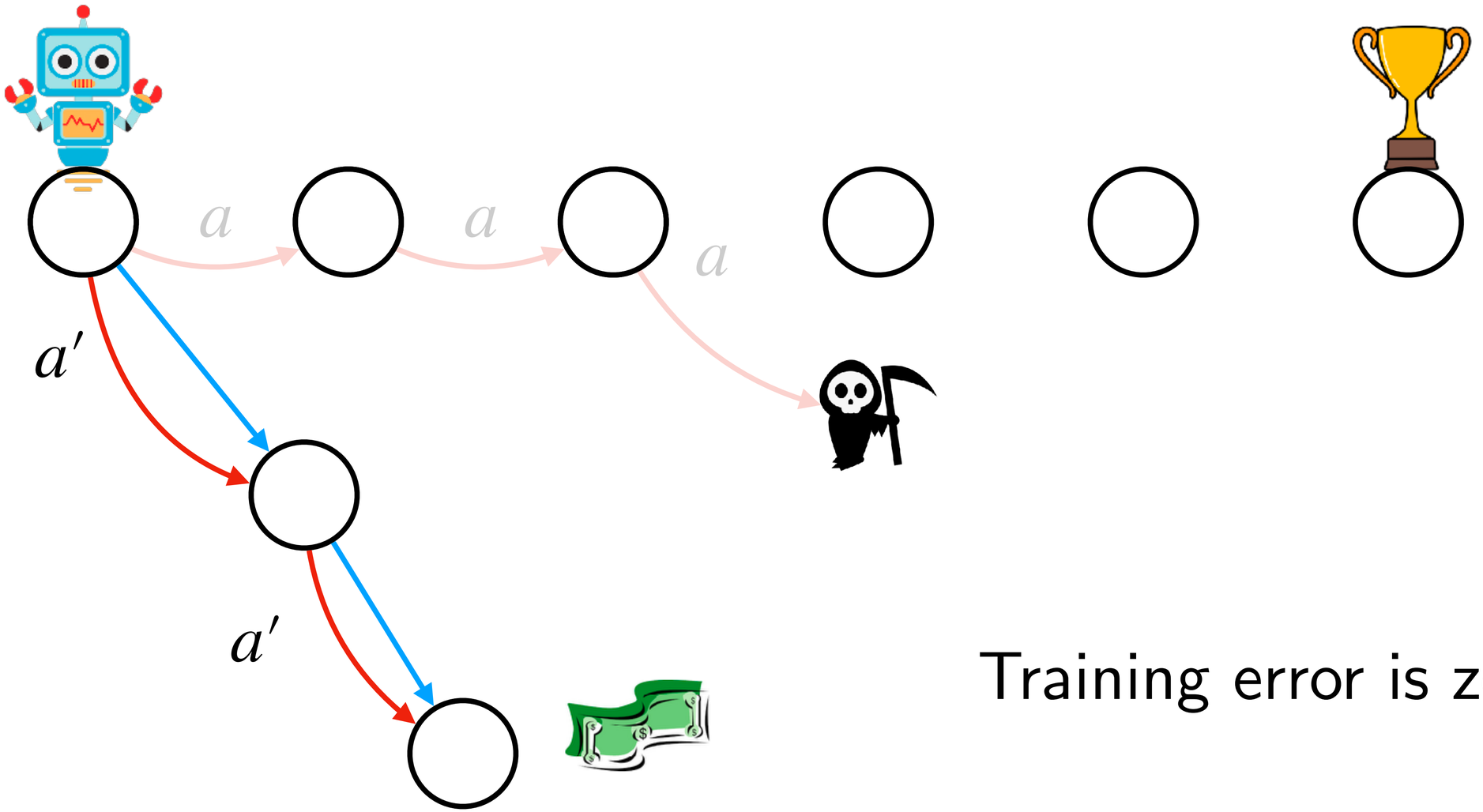
Model plans to get $1

Model

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

World

$s' = M^*(s, a)$



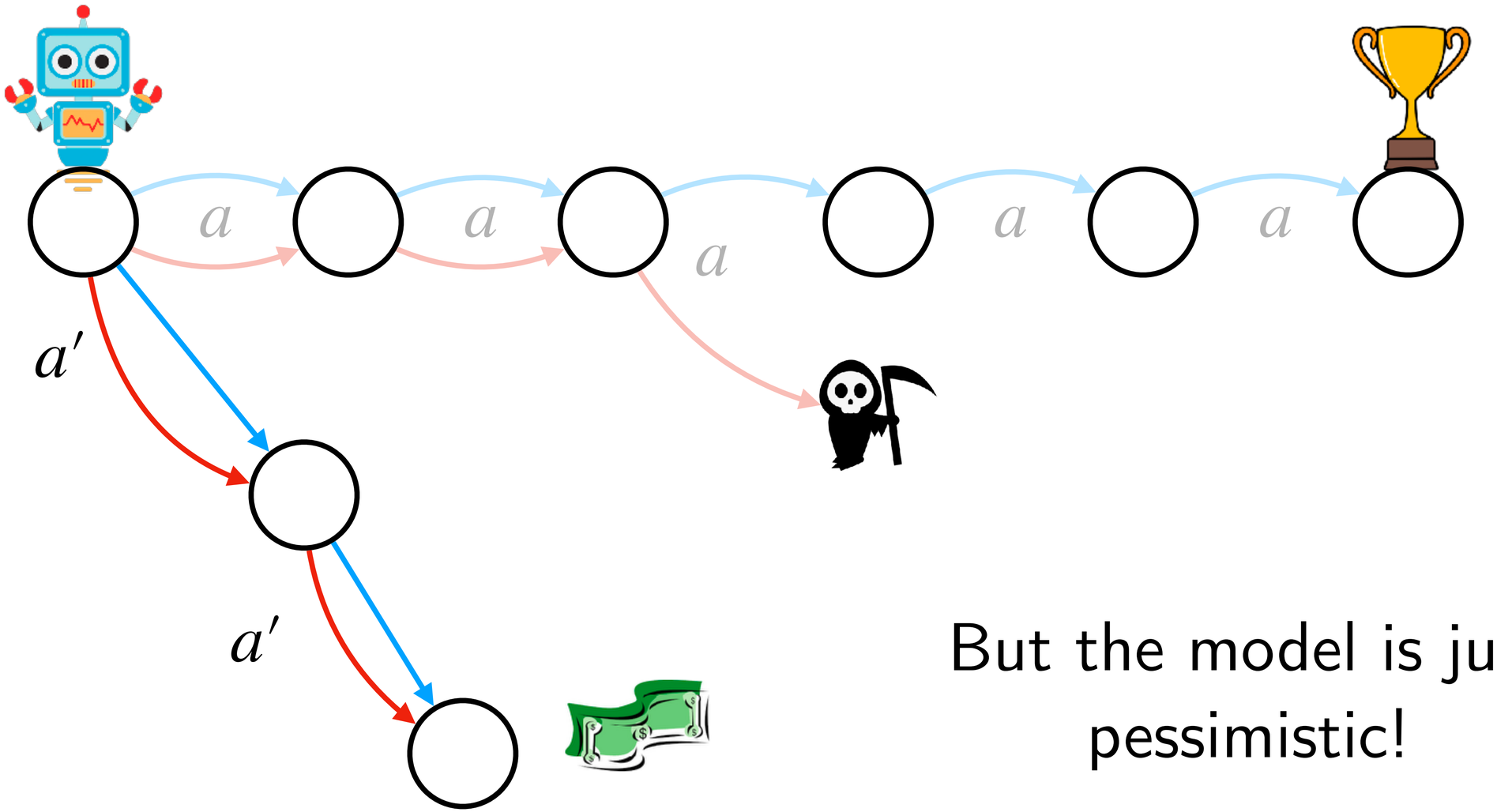
Training error is zero!

Model

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

World

$s' = M^*(s, a)$



But the model is just pessimistic!

Strategy

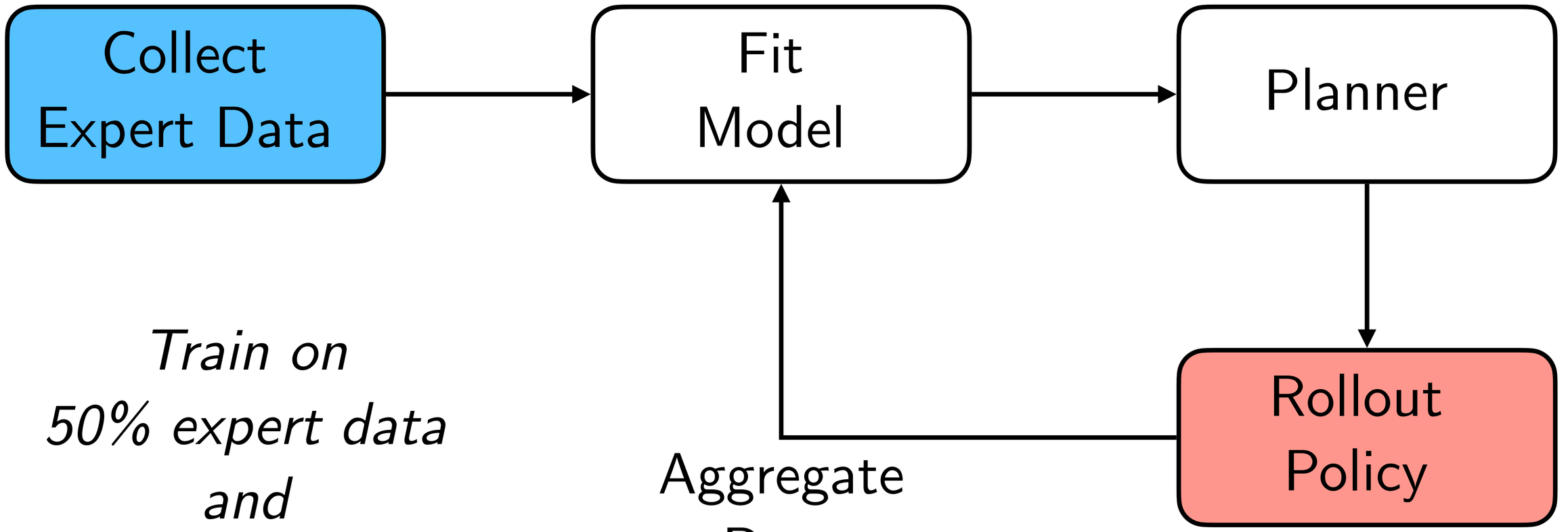
~~Train a model on state actions visited by the expert!~~

~~Train a model on state actions visited by the learner!~~

Train a model on state actions visited by
both the expert and the learner!

Model Learning with Planner in Loop

(Ross & Bagnell, 2012)



Train on
50% expert data
and
50% learner data

Model Learning with Planner in Loop

Collect data from an expert $\mathscr{D}_{\text{expert}} = \{(s, a, s')\}$

Initialize with a random policy π_1

Initialize empty data buffer $\mathscr{D}_{\text{learner}} \leftarrow \{\}$

For $i = 1, \ldots, N$

Execute policy π_i in the real world and collect data

$$\mathscr{D}_i = \{(s, a, s')\}$$

Aggregate data $\mathscr{D}_{\text{learner}} \leftarrow \mathscr{D}_{\text{learner}} \cup \mathscr{D}_i$

Train a new learner on 50% expert + 50% learner data

$$\pi_{i+1} \leftarrow \text{Train}(0.5 * \mathscr{D}_{\text{expert}} + 0.5 * \mathscr{D}_{\text{learner}})$$

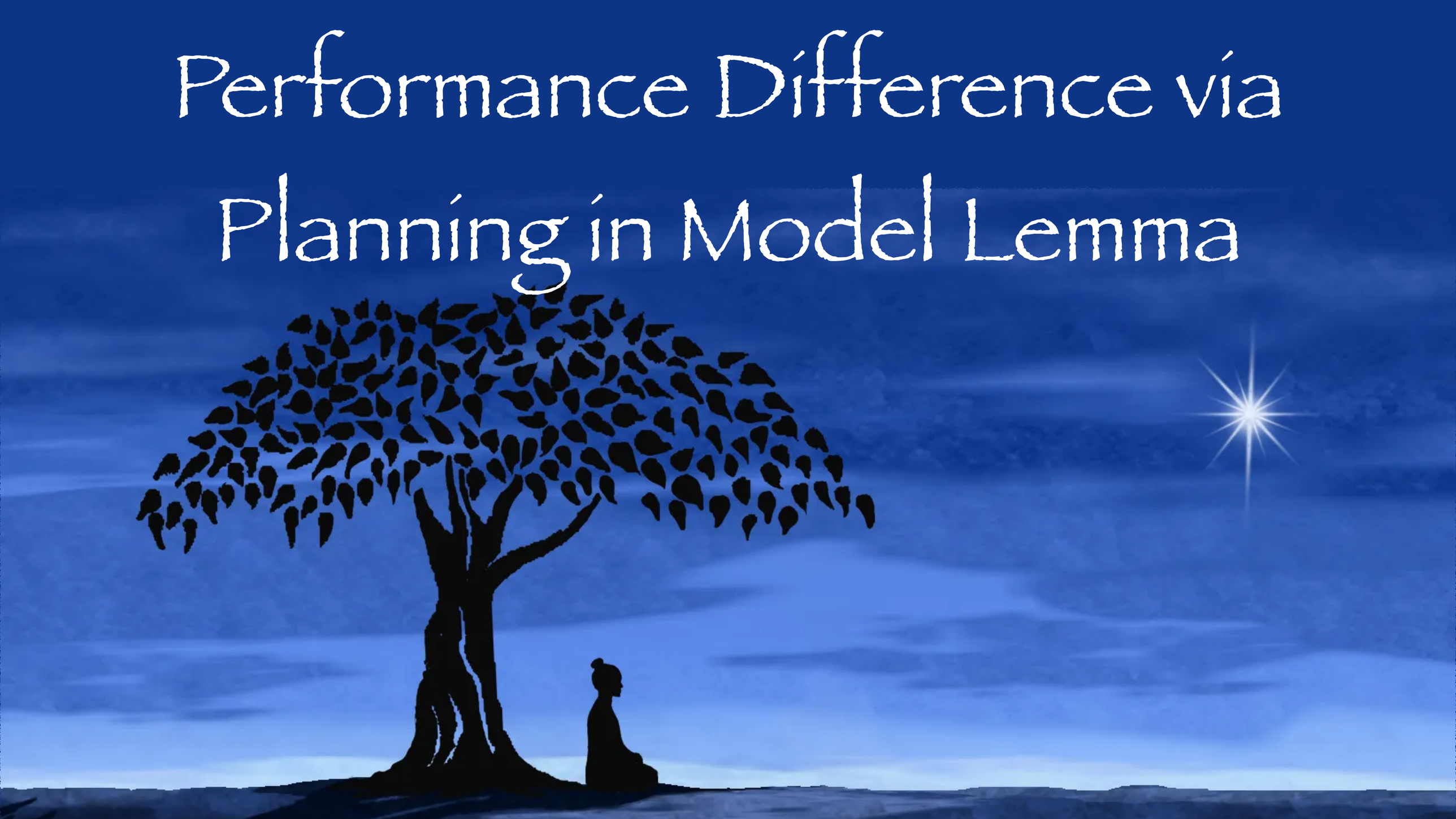
Select the best policy in $\pi_{1:N+1}$

Model learning on both expert and learner data works!

(From Ross & Bagnell, 2012)

Why is 50-50 the right thing to do?

Performance Difference via Planning in Model Lemma



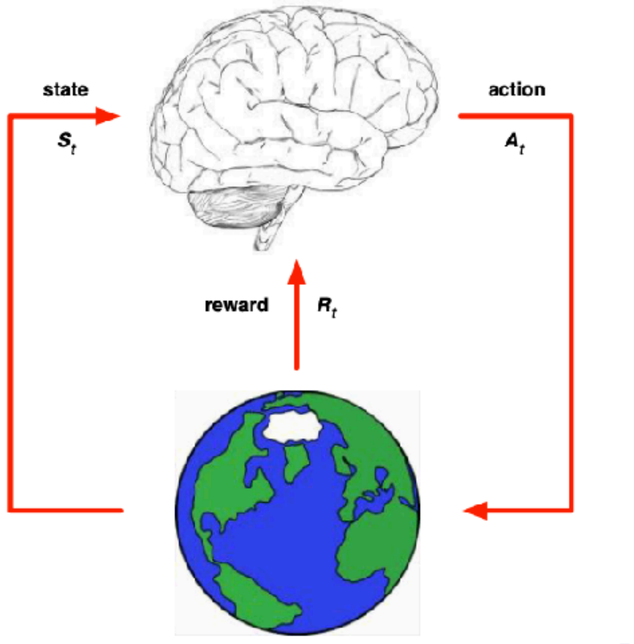
What makes for a good model?

A good model is one such that if

we plan with the model

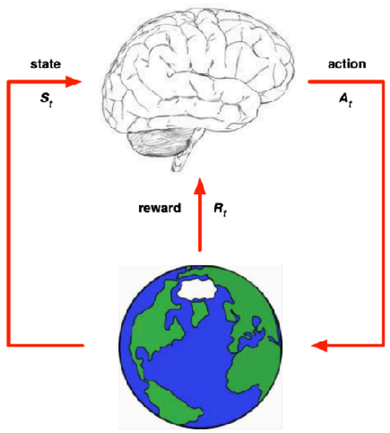
we get a good policy

Policy $\hat{\pi}$



Model $\hat{M}$

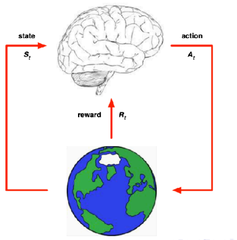
What makes for a good model?



$$V^{\hat{\pi}}(s_0) - V^{\pi^\star}(s_0)$$

A good model gives a good policy that has bounded performance difference

What makes for a good model?



$$V^{\hat{\pi}}_{M^*}(s_0) - V^{\pi^\star}_{M^*}(s_0)$$

A good model gives a good policy that has bounded performance difference in the real world

What makes for a good model?

$$\left[V^{\hat{\pi}}_{M*}(s_0) - V^{\hat{\pi}}_{\hat{M}}(s_0)\right] + \left[V^{\pi^\star}_{\hat{M}}(s_0) - V^{\pi^\star}_{M^\star}(s_0)\right]$$




Learner in real-world vs model

Expert in real-world vs model

$$+ \left[V^{\hat{\pi}}_{\hat{M}}(s_0) - V^{\pi^\star}_{\hat{M}}(s_0)\right]$$


Learner vs Expert in model

What makes for a good model?

Fit model on learner data!

+

Fit model on expert data!

+

Optimize policy in model!