

Data Efficient Reinforcement Learning for Autonomous Robots

Josiah Hanna

Department of Computer Science
The University of Texas at Austin



TEXAS

The University of Texas at Austin

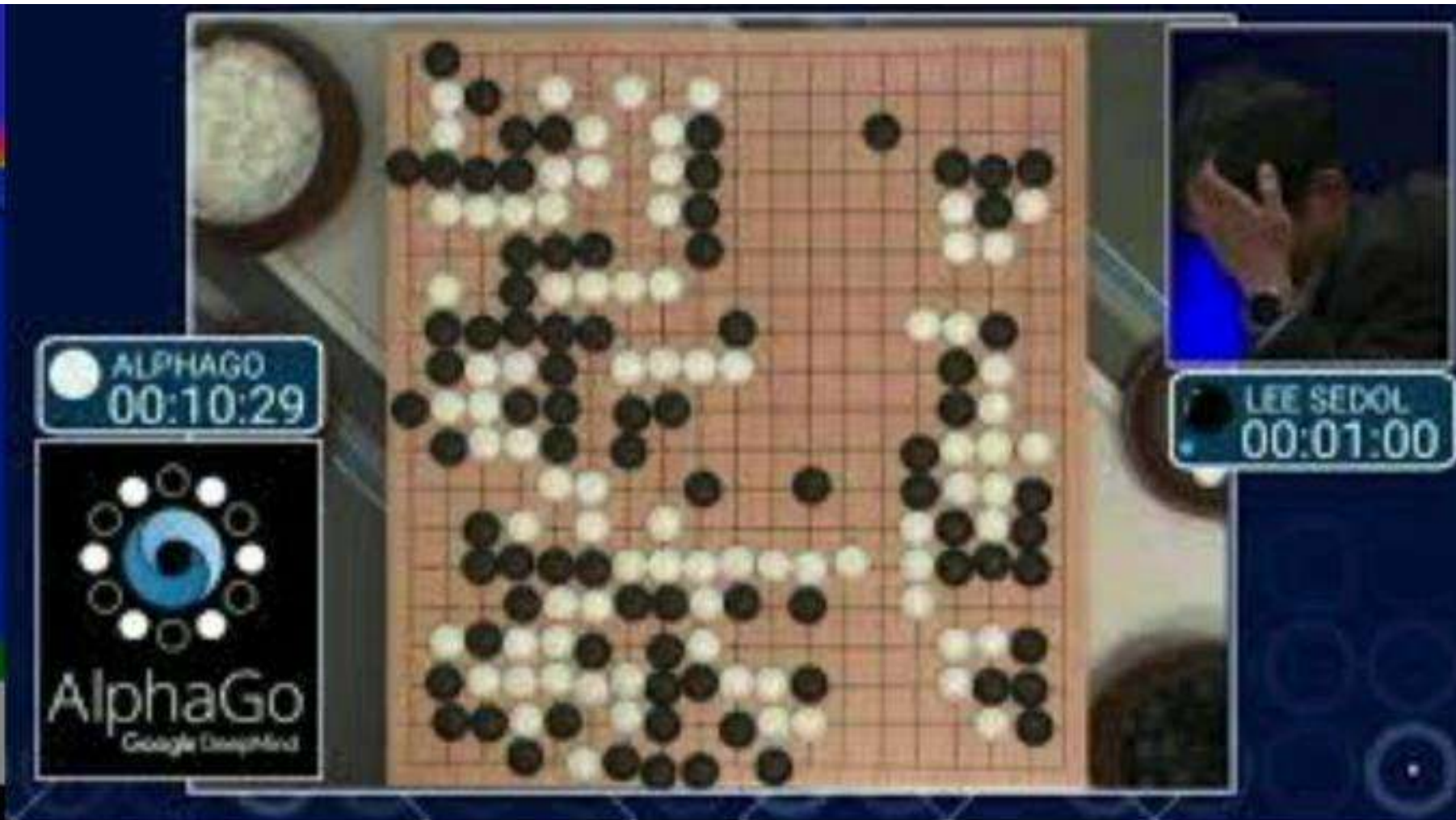




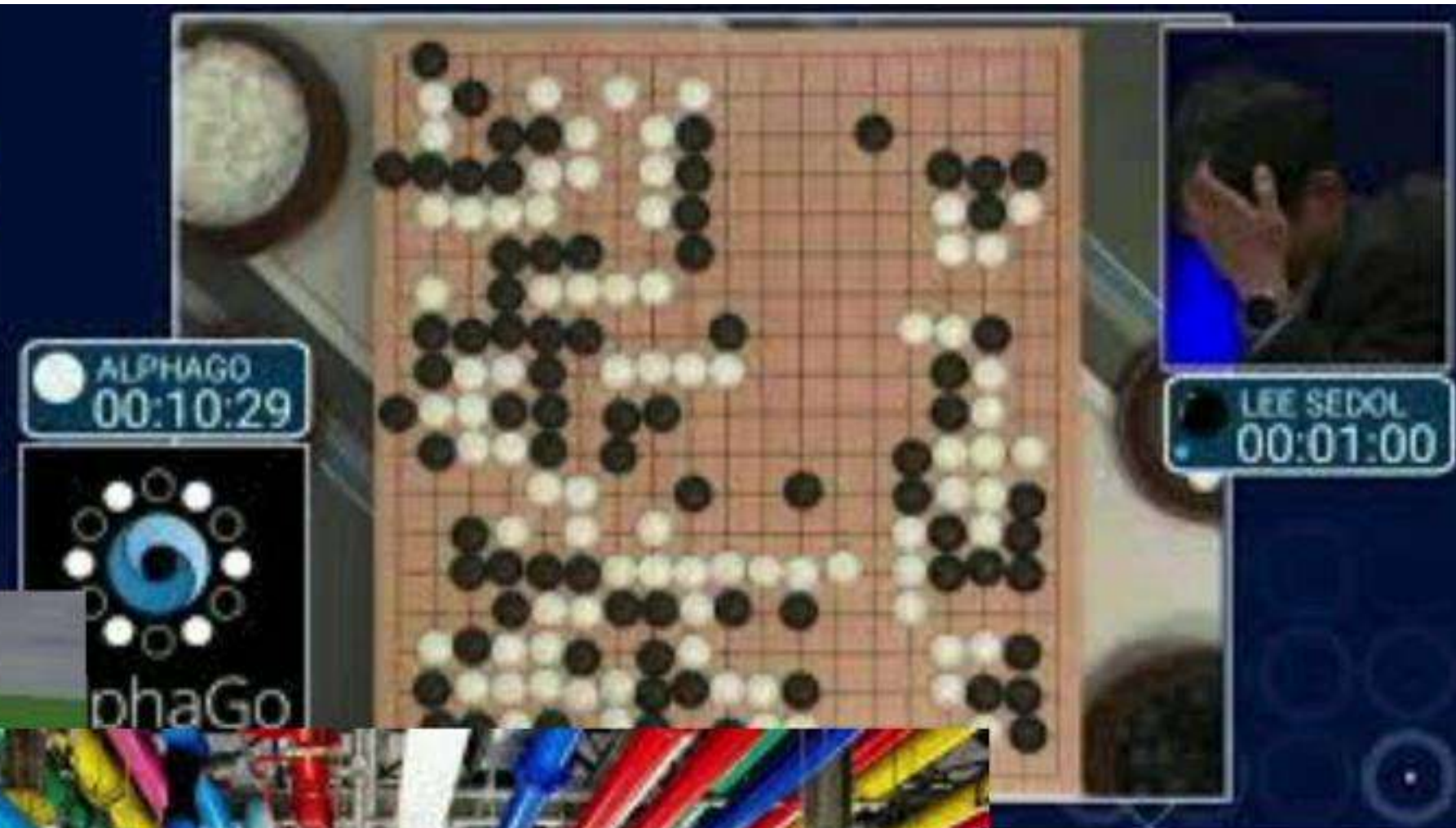
Computers and robots that learn how
to take actions to achieve goals.

Learn a **policy** that maps any given situation to an action.











50 millions
actions taken

ALPHAGO
00:10:29



LEE SEDOL
00:01:00



50 millions
actions taken

21 days, millions
of games



A screenshot from the game Endless Space. The background is a dark blue space with a yellow and orange nebula at the top. A yellow spaceship is visible in the upper left. A black rectangular box with white text is centered on the screen. The text reads "50 millions" on the first line and "actions taken" on the second line. To the right of the box, there are several small, light blue spaceship icons. At the bottom, there are more spaceship icons, including a larger yellow one in the center and several smaller light blue ones to its right.

**50 millions
actions taken**



21 days, millions of games



**1.5 years of
compute**



Can reinforcement learning be data efficient enough for real world applications?

How can an agent efficiently learn to predict the effects of its actions?

Can reinforcement learning be data efficient enough for real world applications?

How can an agent efficiently learn to predict the effects of its actions?



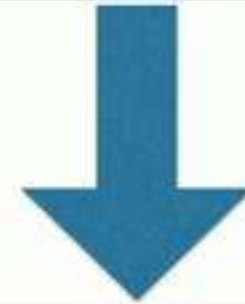
Estimate task performance for a fixed policy.

Can reinforcement learning be data efficient enough for real world applications?

How can an agent efficiently learn to predict the effects of its actions?



Estimate task performance for a fixed policy.



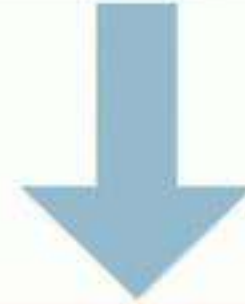
Predicting how actions change the state of the world.

Can reinforcement learning be data efficient enough for real world applications?

How can an agent efficiently learn to predict the effects of its actions?



Estimate task performance for a fixed policy.



Predicting how actions change the state of the world.

Can reinforcement learning be data efficient enough for real world applications?

The Reinforcement Learning World

State:

Action:

Reward:

Policy:

The Reinforcement Learning World



State:

Action:

Reward:

Policy:

The Reinforcement Learning World



State: Position of car or other cars

Action:

Reward:

Policy:

The Reinforcement Learning World



State: Position of car or other cars

Action: Steer wheel / brake / accelerate

Reward:

Policy:

The Reinforcement Learning World



State: Position of car or other cars

Action: Steer wheel / brake / accelerate

Reward: Reach destination

Policy:

The Reinforcement Learning World



State: Position of car or other cars

Action: Steer wheel / brake / accelerate

Reward: Reach destination

Policy: State to action

The Reinforcement Learning World



State: Position of car or other cars

Action: Steer wheel / brake / accelerate

Reward: Reach destination

Policy: State to action

$$S_0, A_0, R_0$$

The Reinforcement Learning World



State: Position of car or other cars

Action: Steer wheel / brake / accelerate

Reward: Reach destination

Policy: State to action

$$S_0, A_0, R_0, S_1$$

The Reinforcement Learning World



State: Position of car or other cars

Action: Steer wheel / brake / accelerate

Reward: Reach destination

Policy: State to action

$$S_0, A_0, R_0, S_1, \dots, S_L, A_L, R_L$$

The Reinforcement Learning World



State: Position of car or other cars

Action: Steer wheel / brake / accelerate

Reward: Reach destination

Policy: State to action

$$\underbrace{S_0, A_0, R_0, S_1, \dots, S_L, A_L, R_L}$$

Trajectory

The Reinforcement Learning World



State: Position of car or other cars

Action: Steer wheel / brake / accelerate

Reward: Reach destination

Policy: State to action

Policy Improvement: Find policy that maximizes expected cumulative reward.

The Reinforcement Learning World



State: Position of car or other cars

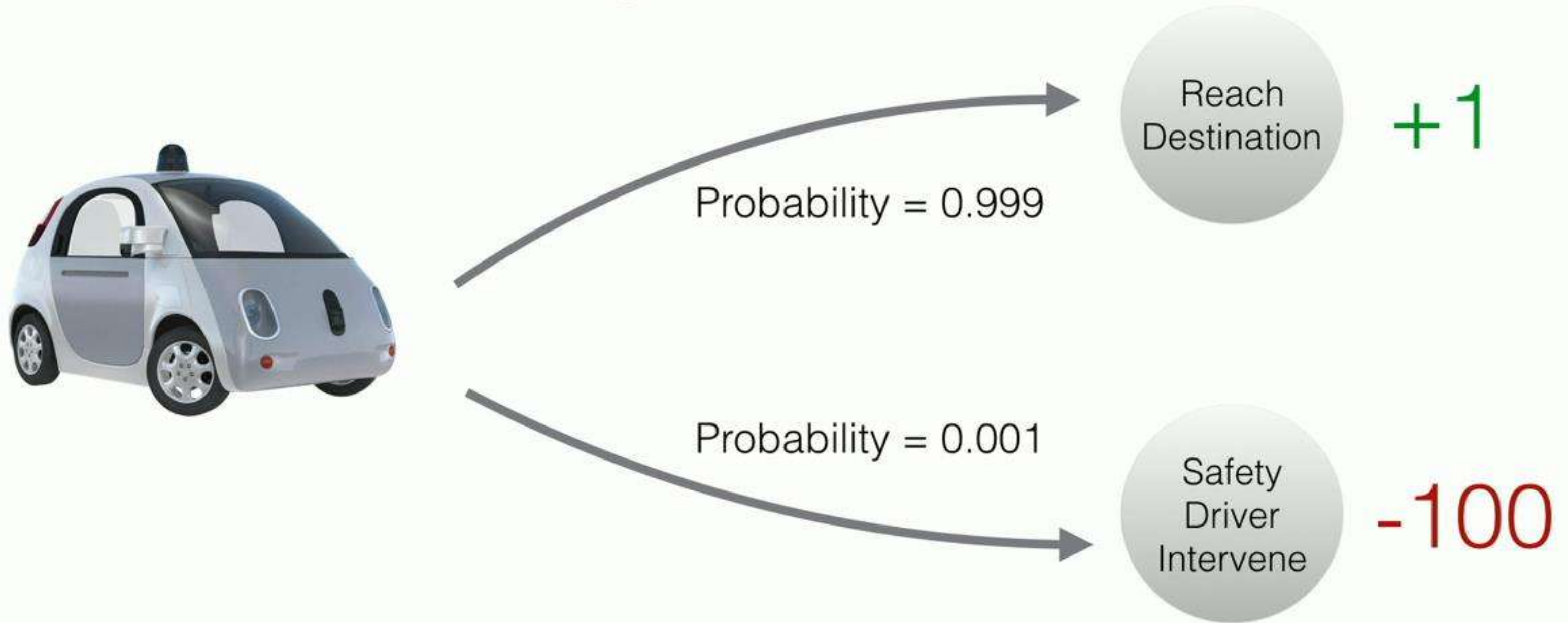
Action: Steer wheel / brake / accelerate

Reward: Reach destination

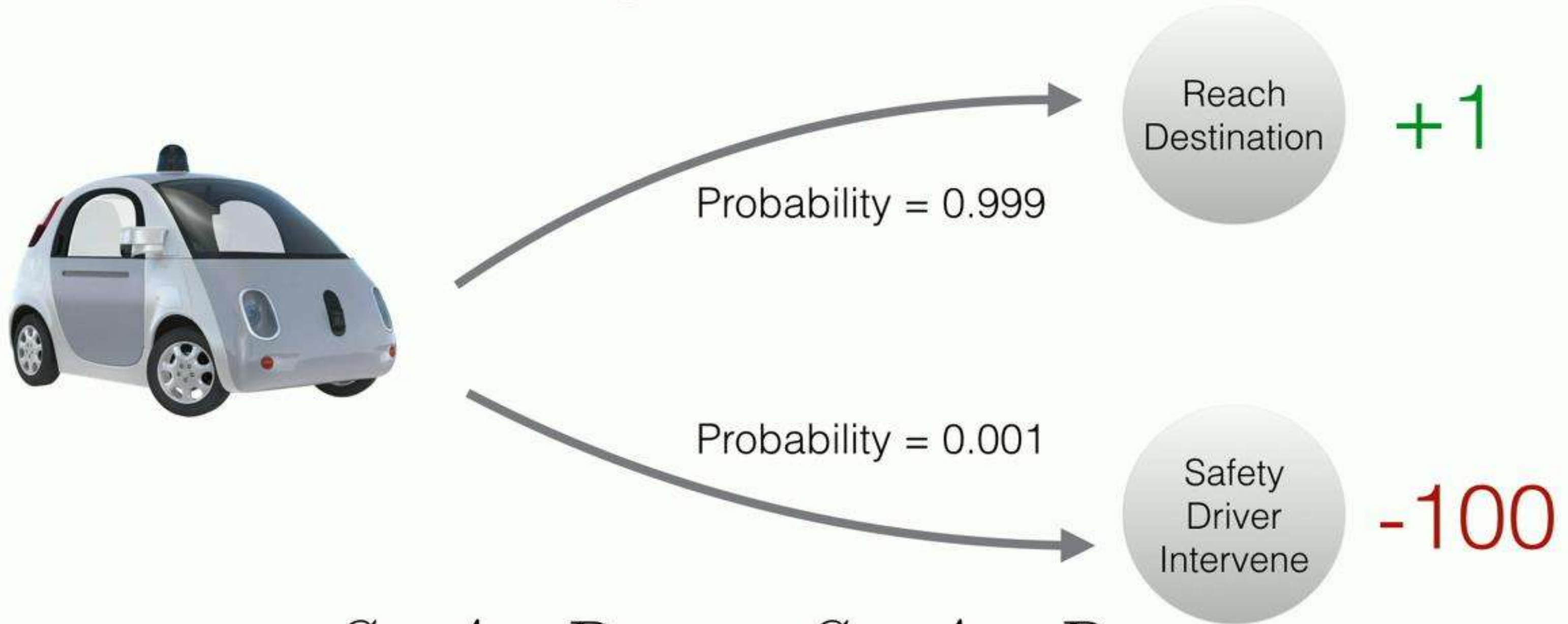
Policy: State to action

Policy Evaluation: Given a **fixed** policy, determine the expected cumulative reward of that policy.

Evaluating a fixed policy

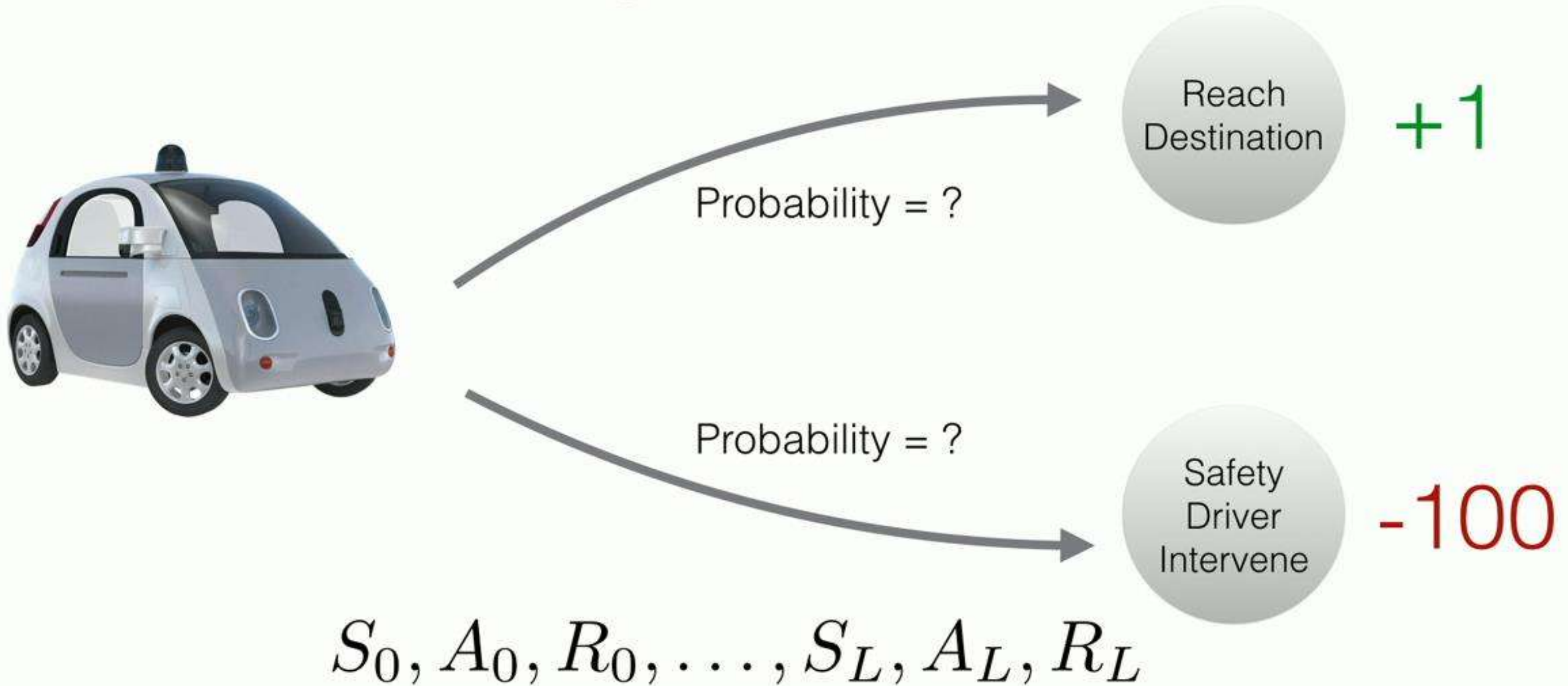


Evaluating a fixed policy



$S_0, A_0, R_0, \dots, S_L, A_L, R_L$

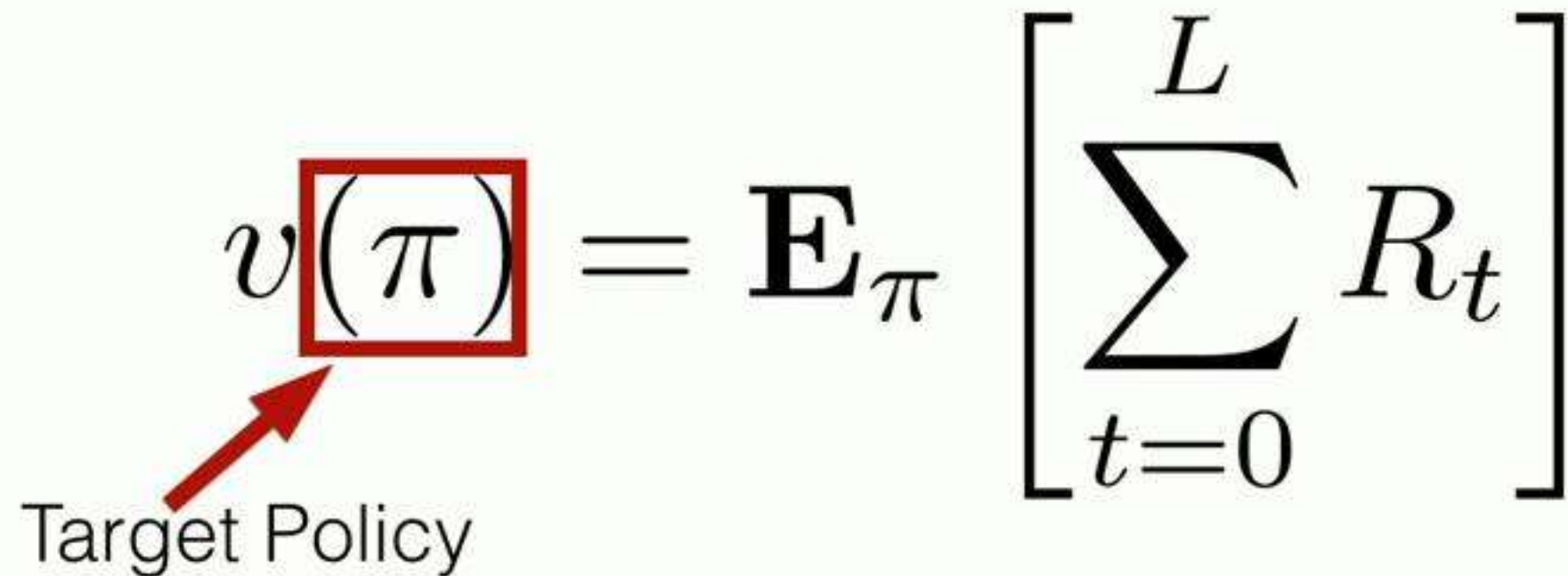
Evaluating a fixed policy



Evaluating a fixed policy

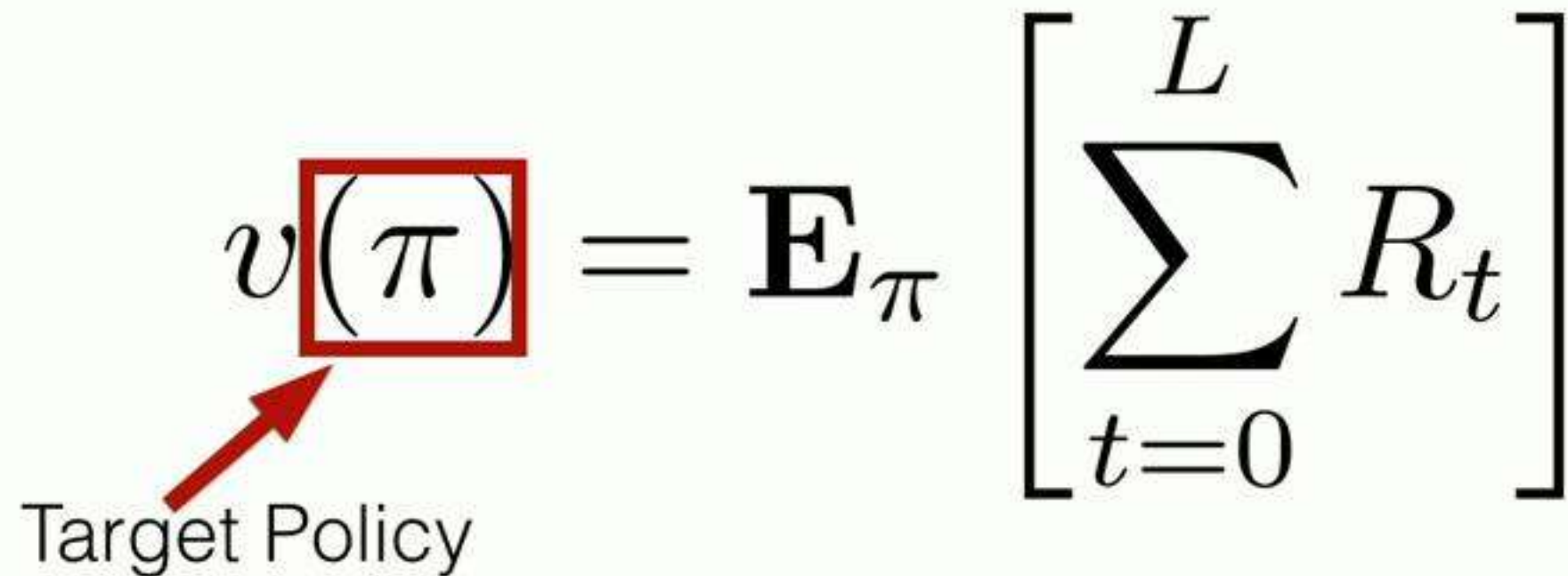
$$v(\pi) = \mathbf{E}_{\pi} \left[\sum_{t=0}^L R_t \right]$$

Evaluating a fixed policy


$$v(\pi) = \mathbf{E}_{\pi} \left[\sum_{t=0}^L R_t \right]$$

Target Policy

Evaluating a fixed policy


$$v(\pi) = \mathbf{E}_{\pi} \left[\sum_{t=0}^L R_t \right]$$

Target Policy

$$\pi : \mathcal{S} \times \mathcal{A} \rightarrow [0, 1]$$

Evaluating a fixed policy

$$v(\pi) = \mathbf{E}_{\pi} \left[\sum_{t=0}^L R_t \right]$$

Target Policy

Expected Total Reward

$$\pi : \mathcal{S} \times \mathcal{A} \rightarrow [0, 1]$$

Evaluating a fixed policy



Evaluating a fixed policy



**Testing
Time**

Evaluating a fixed policy



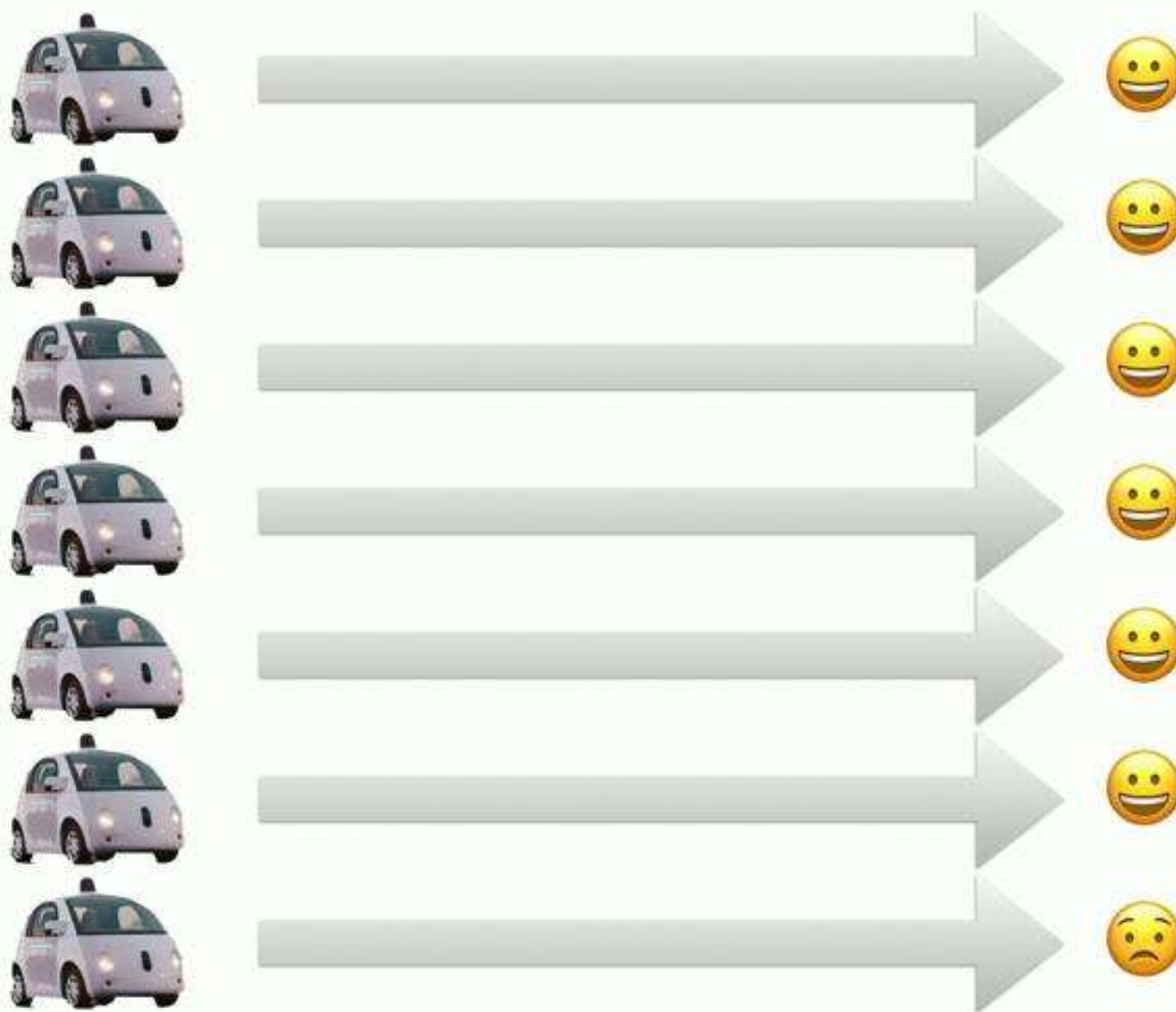
**Testing
Time**

Deployment Time

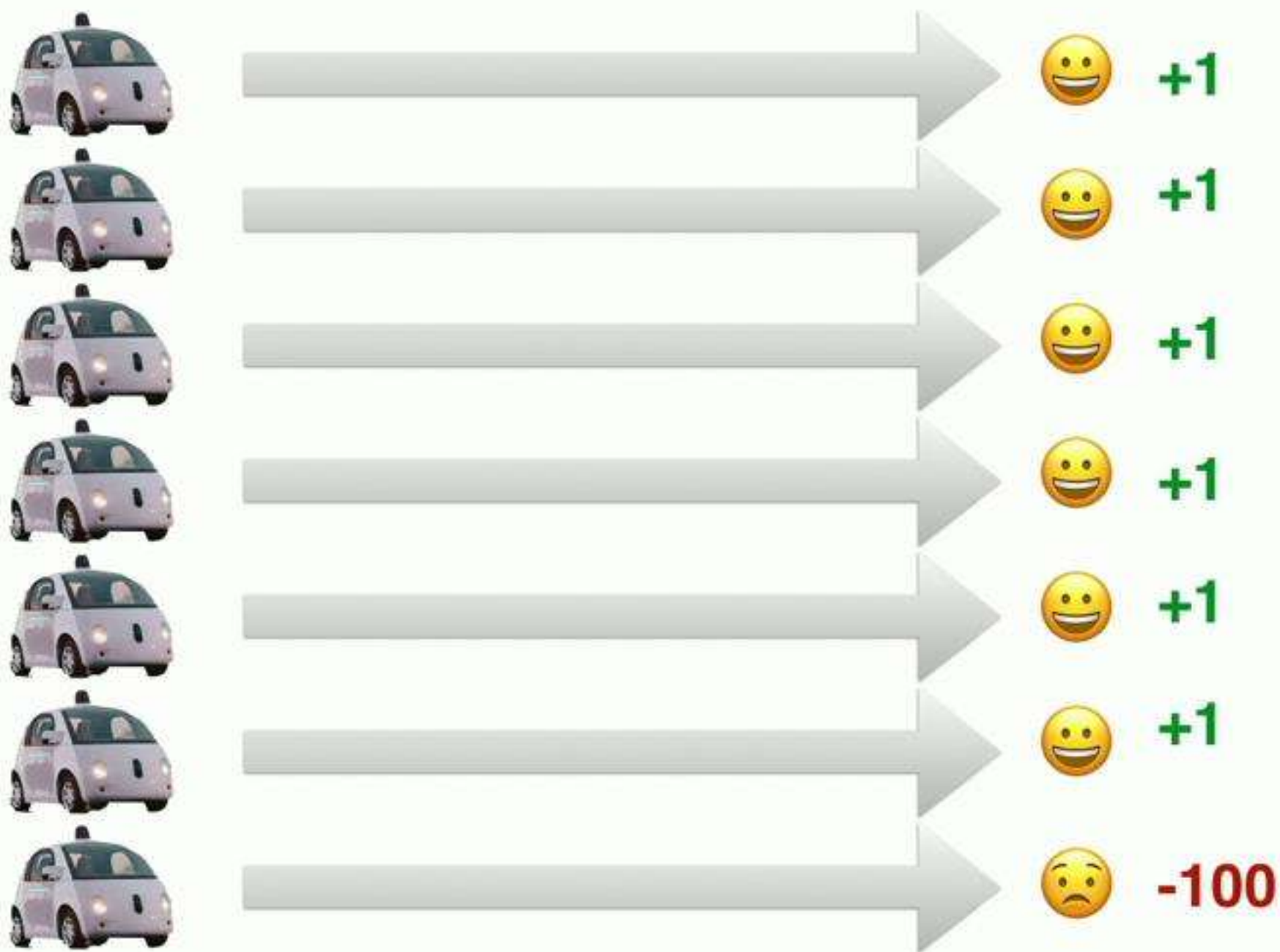
Evaluating a fixed policy



Evaluating a fixed policy



Evaluating a fixed policy



Passively evaluating a fixed policy

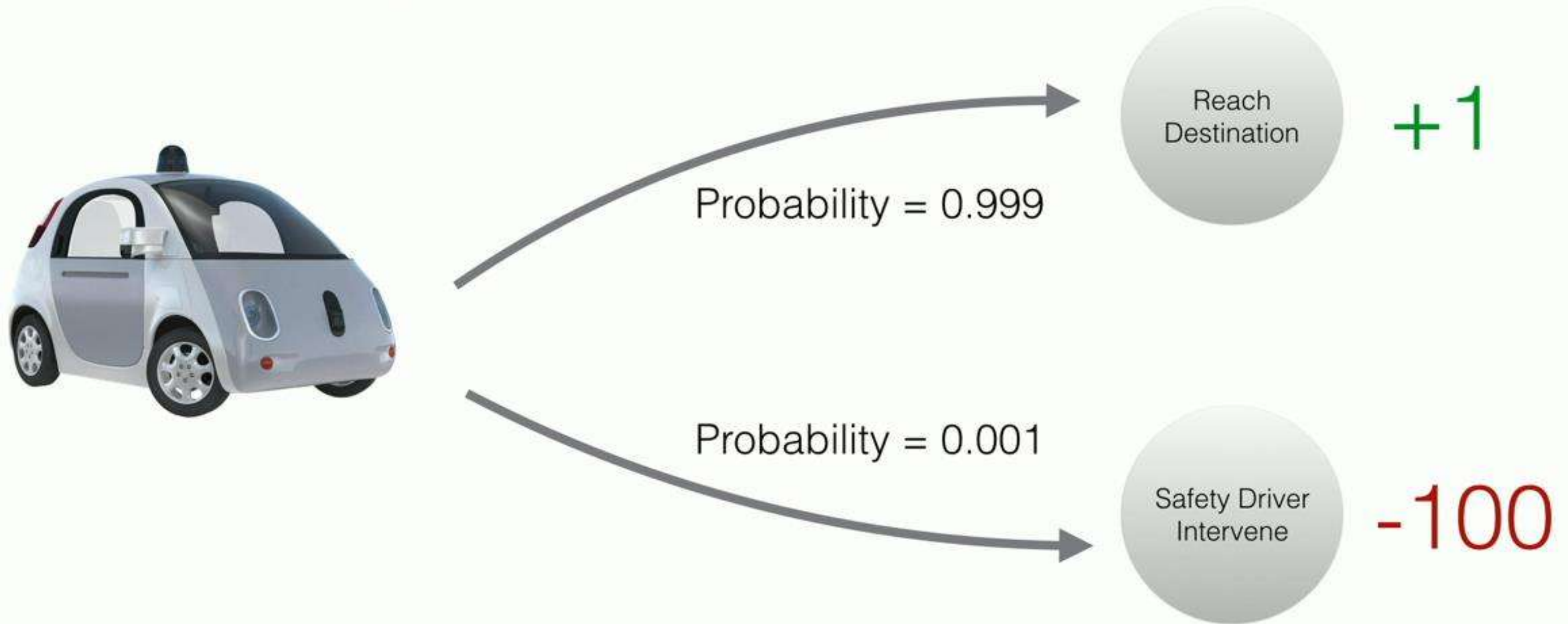
1. Repeatedly run the target policy.

$$S_0, A_0, R_0, \dots, S_L, A_L, R_L$$

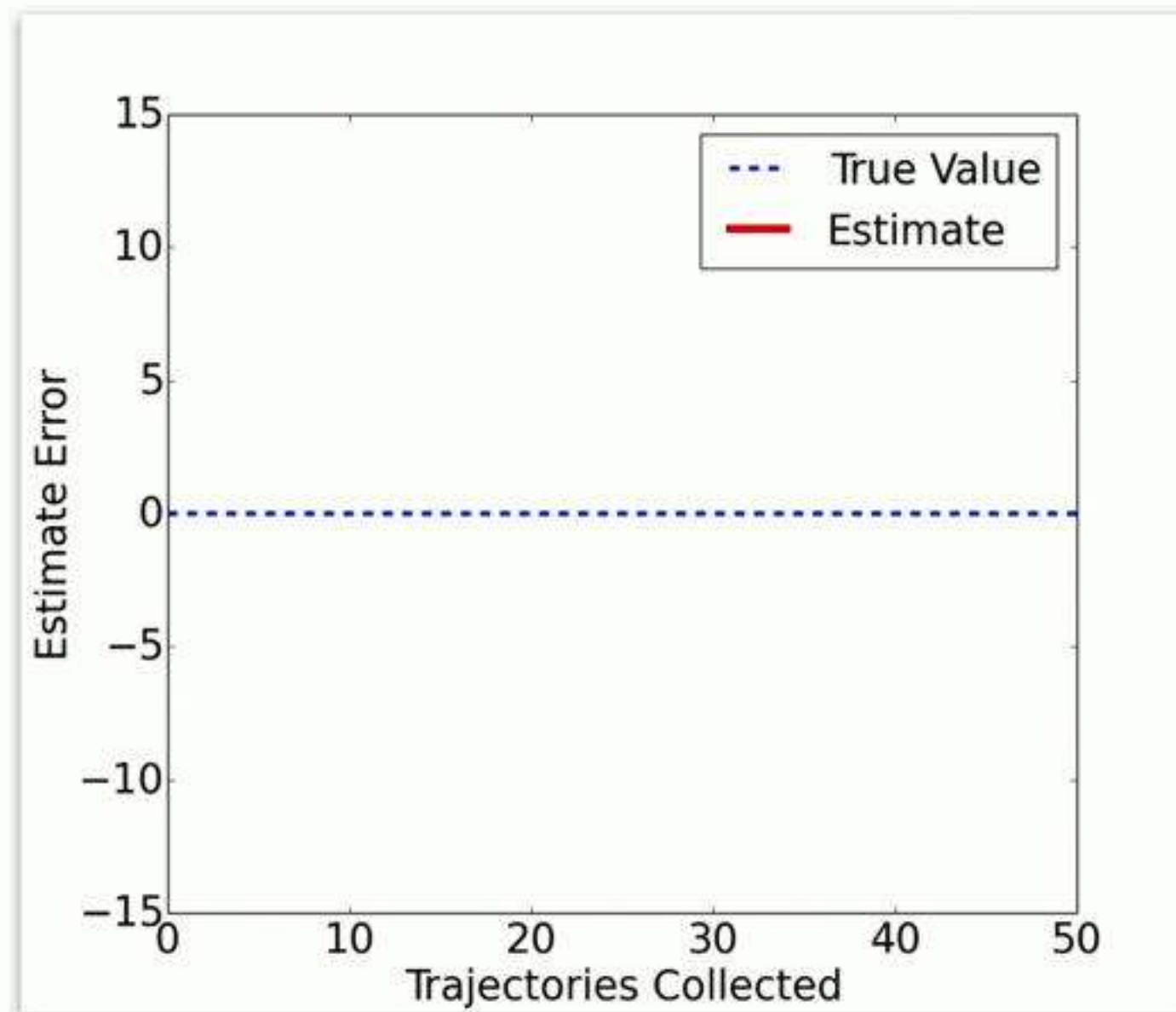
2. Average the total reward seen each trajectory.

$$\hat{v} = \frac{1}{m} \sum_{j=1}^m \sum_{t=0}^L R_t^{(j)}$$

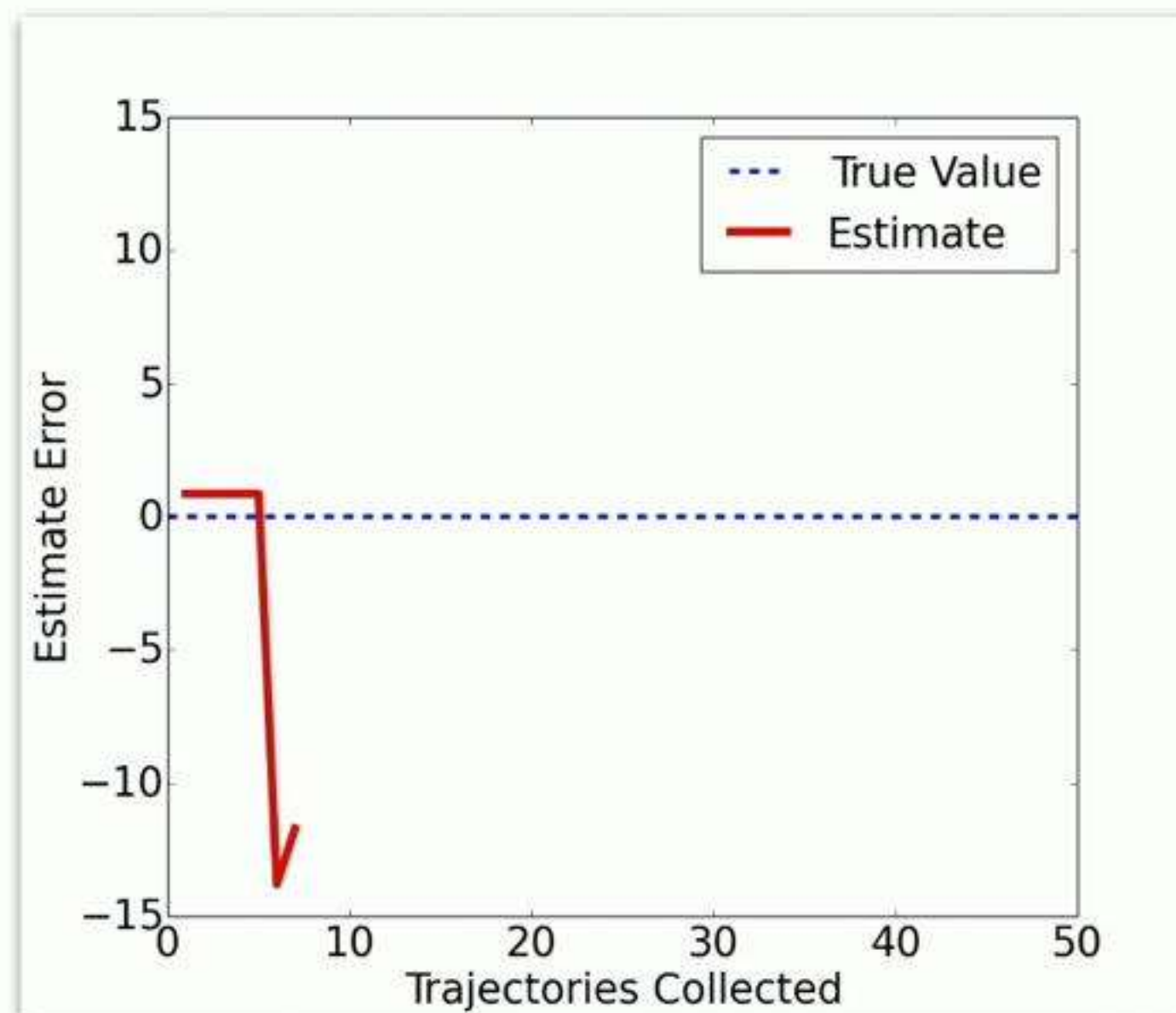
Can we improve on passive evaluation?



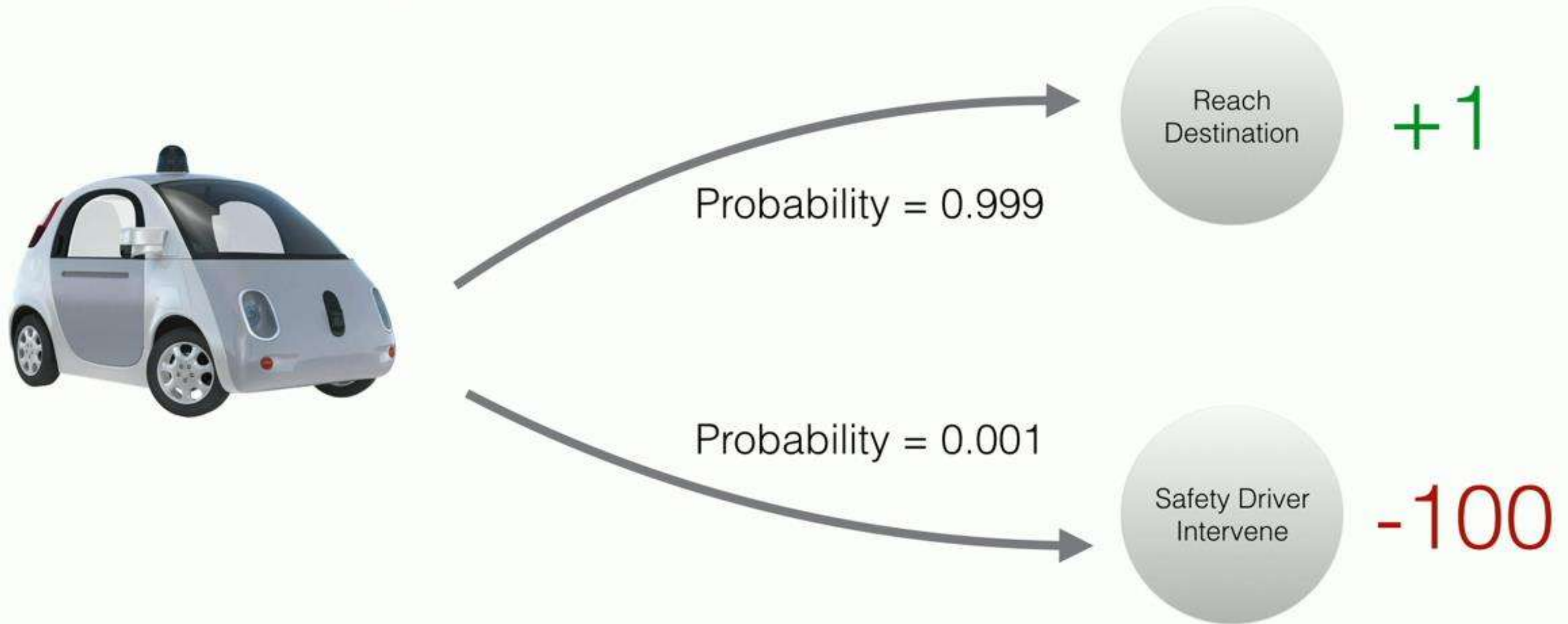
Can we improve on passive evaluation?



Can we improve on passive evaluation?



Can we improve on passive evaluation?



Evaluating a fixed policy



**Testing
Time**

Deployment Time

Can we improve on passive evaluation?

Challenges:

- ❑ Changing the data collection policy changes the data distribution.

Can we improve on passive evaluation?

Challenges:

- ❑ Changing the data collection policy changes the data distribution.
- ❑ We do not know the right data collection policy.

Can we improve on passive evaluation?

Challenges:

- ❑ Changing the data collection policy changes the data distribution.
- ❑ We do not know the right data collection policy.

Can we improve on passive evaluation?

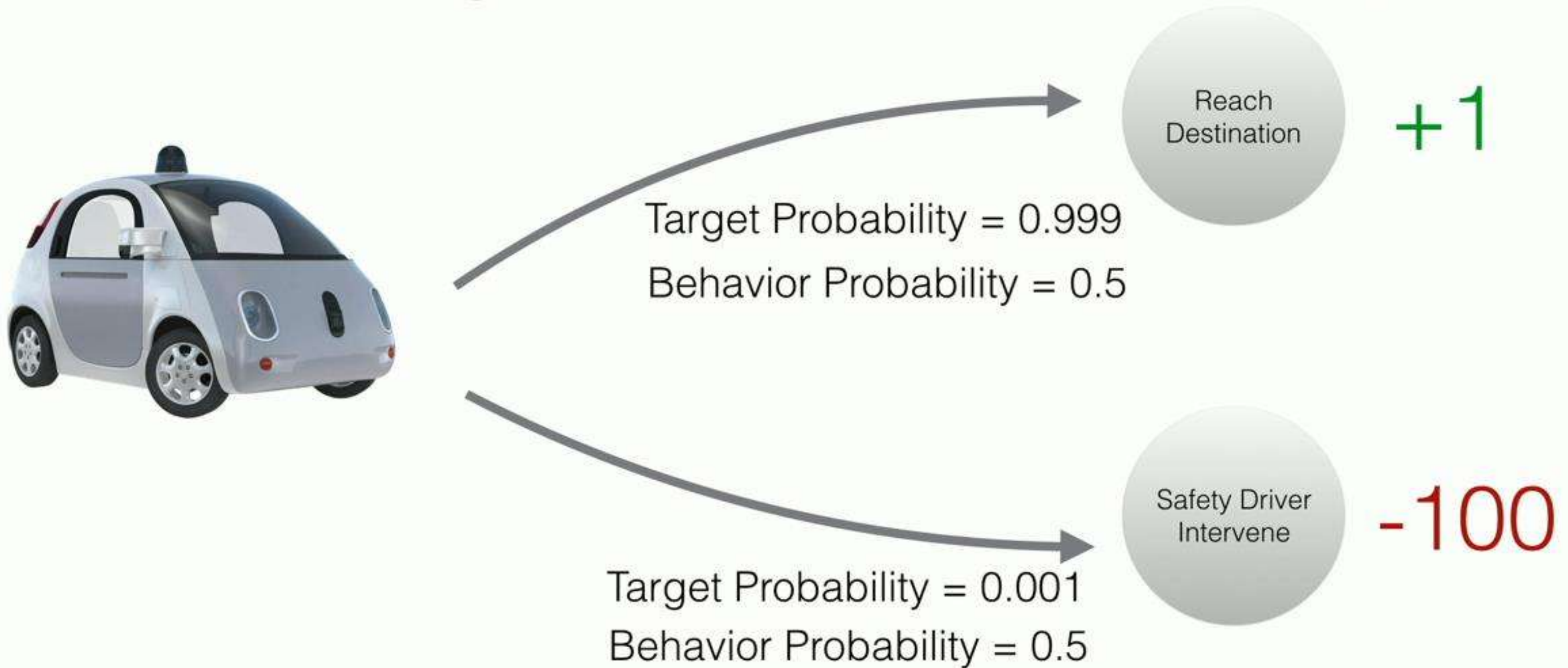
Challenges:

- ❑ Changing the data collection policy changes the data distribution.
- ❑ We do not know the right data collection policy.

Target Policy: Policy we want to evaluate.

Behavior Policy: Policy we collect data with.

Collecting data with any policy



Importance Sampling

1. Repeatedly run the behavior policy.
2. Add up all of the reward received along each trajectory.


Importance Sampling

1. Repeatedly run the behavior policy.
2. Add up all of the reward received along each trajectory.
- 3. Re-weight the reward total.**

Importance Sampling

1. Repeatedly run the behavior policy.
2. Add up all of the reward received along each trajectory.

3. Re-weight the reward total.

$$\left(\prod_{t=0}^L \frac{\pi(A_t|S_t)}{\pi_b(A_t|S_t)} \right) \times \boxed{\left(\sum_{t=0}^L R_t \right)}$$


Total Reward

Importance Sampling

1. Repeatedly run the behavior policy.
2. Add up all of the reward received along each trajectory.

3. Re-weight the reward total.

$$\left(\prod_{t=0}^L \frac{\pi(A_t|S_t)}{\pi_b(A_t|S_t)} \right) \times \left(\sum_{t=0}^L R_t \right)$$

Relative Likelihood **Total Reward**

Importance Sampling

1. Repeatedly run the behavior policy.
2. Add up all of the reward received along each trajectory.

3. Re-weight the reward total.

$$\left(\prod_{t=0}^L \frac{\pi(A_t|S_t)}{\pi_b(A_t|S_t)} \right) \times \left(\sum_{t=0}^L R_t \right)$$

Relative Likelihood **Total Reward**

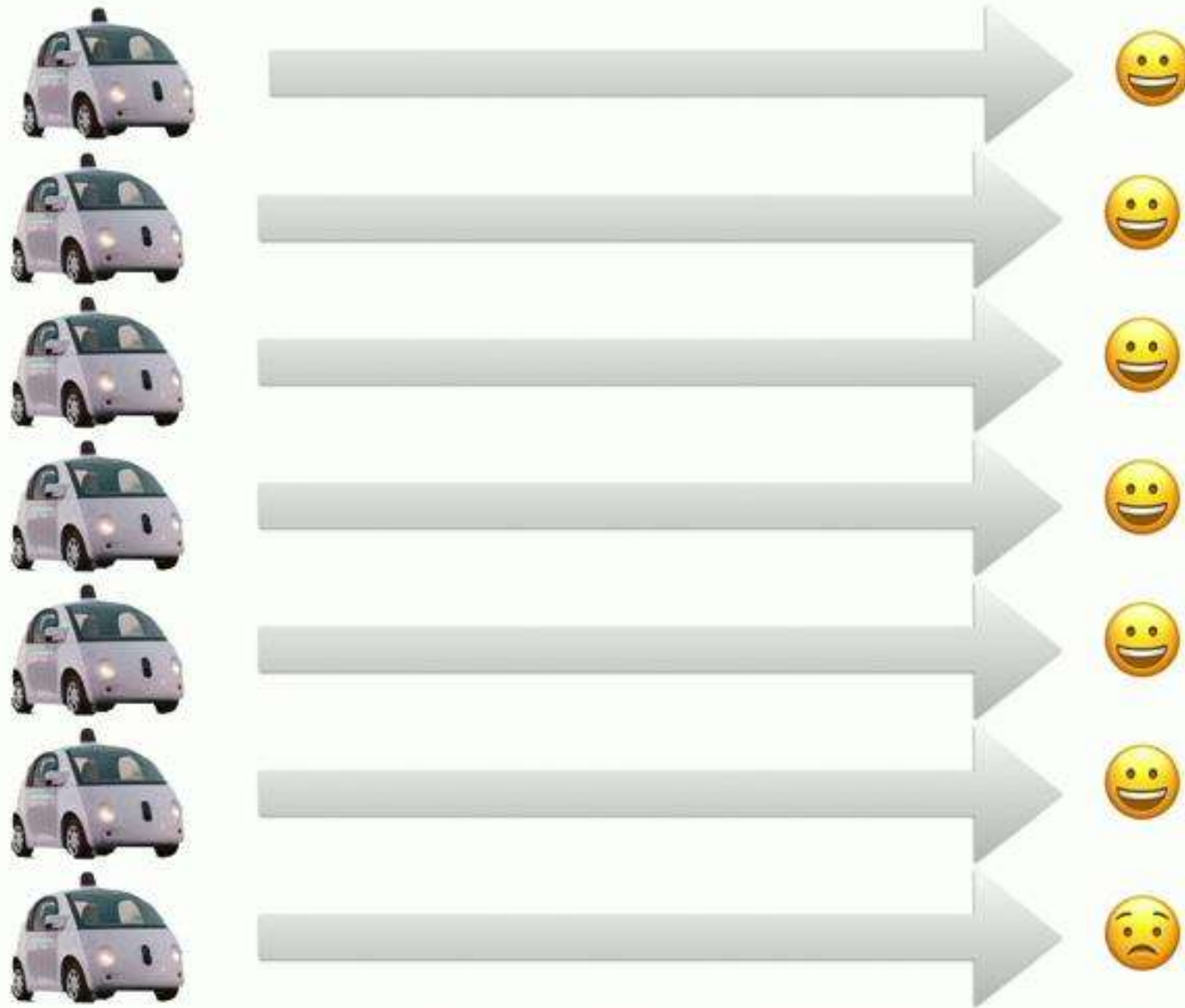
4. Average the re-weighted rewards.

Can we improve on passive evaluation?

Challenges:

- ☒ Changing the behavior policy changes the data distribution.
- ☐ We do not know the right behavior policy.

Can we perfectly evaluate with a single test driver?



Can we perfectly evaluate with a single test driver?



Can we perfectly evaluate with a single test driver?



$$\hat{v} = \left(\prod_{t=0}^L \frac{\pi(A_t|S_t)}{\pi_b(A_t|S_t)} \right) \times \left(\sum_{t=0}^L R_t \right)$$

Can we perfectly evaluate with a single test driver?



$$\hat{v} = \left(\prod_{t=0}^L \frac{\pi(A_t|S_t)}{\pi_b(A_t|S_t)} \right) \times \left(\sum_{t=0}^L R_t \right)$$



Single Trajectory Estimate

Can we perfectly evaluate with a single test driver?



$$v(\pi) \stackrel{?}{=} \left(\prod_{t=0}^L \frac{\pi(A_t|S_t)}{\pi_b(A_t|S_t)} \right) \times \left(\sum_{t=0}^L R_t \right)$$



Single Trajectory Estimate

Can we perfectly evaluate with a single test driver?



$$\boxed{v(\pi)} \stackrel{?}{=} \boxed{\left(\prod_{t=0}^L \frac{\pi(A_t|S_t)}{\pi_b(A_t|S_t)} \right) \times \left(\sum_{t=0}^L R_t \right)}$$

True Value

Single Trajectory Estimate

Learning the right behavior policy


Learning the right behavior policy

$$\text{MSE}(\hat{v}) = \mathbf{E}_{\pi_b} \left[(\hat{v}(\pi, \dots, \pi_b) - v(\pi))^2 \right]$$

Learning the right behavior policy



$$\text{MSE}(\hat{v}) = \mathbf{E}_{\pi_b} \left[(\hat{v}(\pi, \dots, \pi_b) - \boxed{v(\pi)})^2 \right]$$

True Value



Learning the right behavior policy

$$\text{MSE}(\hat{v}) = \mathbf{E}_{\pi_b} \left[\boxed{\hat{v}(\pi, \dots, \pi_b)} - \boxed{v(\pi)} \right]^2$$

Estimate  **True Value** 

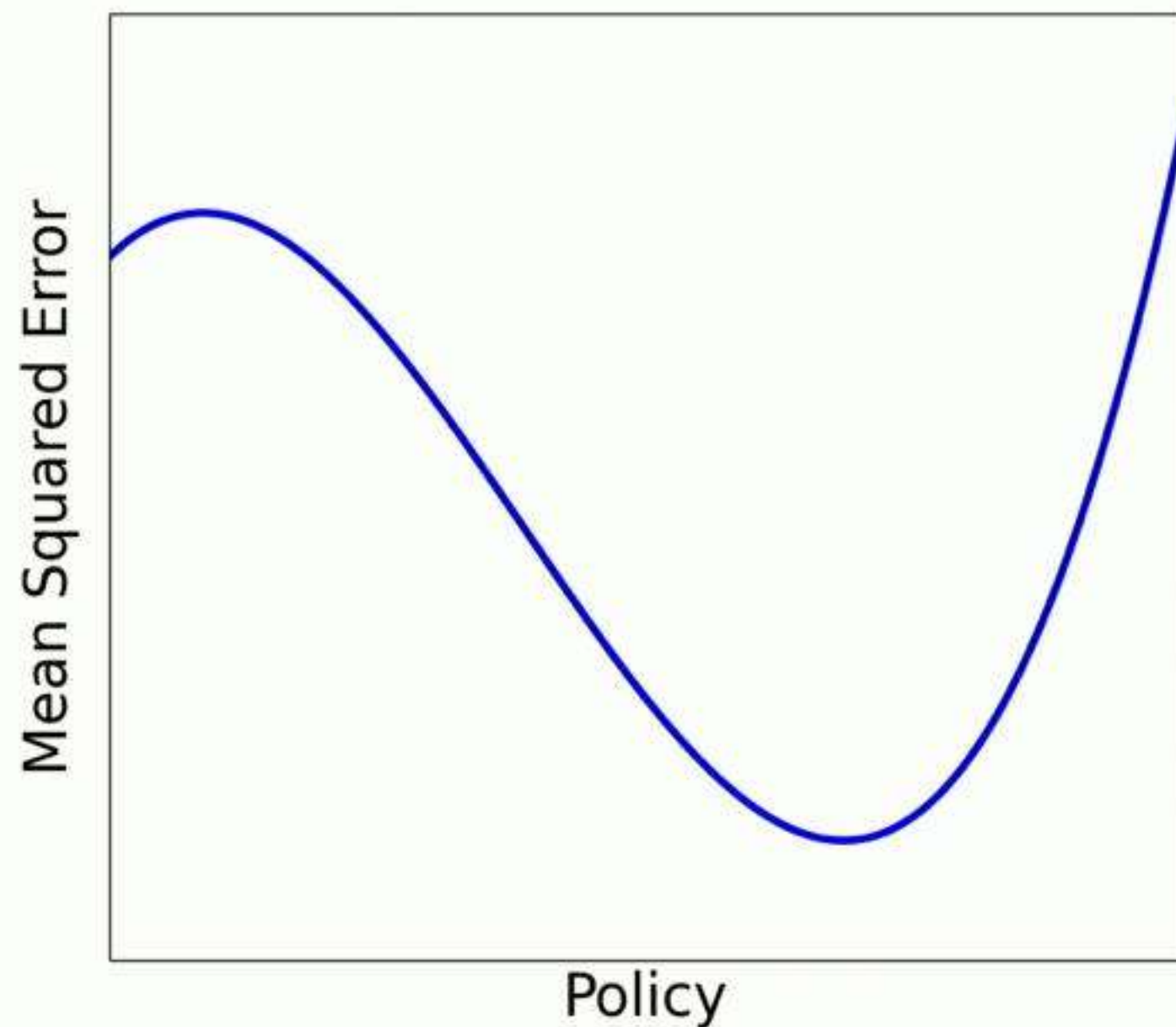
Learning the right behavior policy

$$\text{MSE}(\hat{v}) = \mathbf{E}_{\pi_b} \left[(\hat{v}(\pi, \dots, \pi_b) - v(\pi))^2 \right]$$

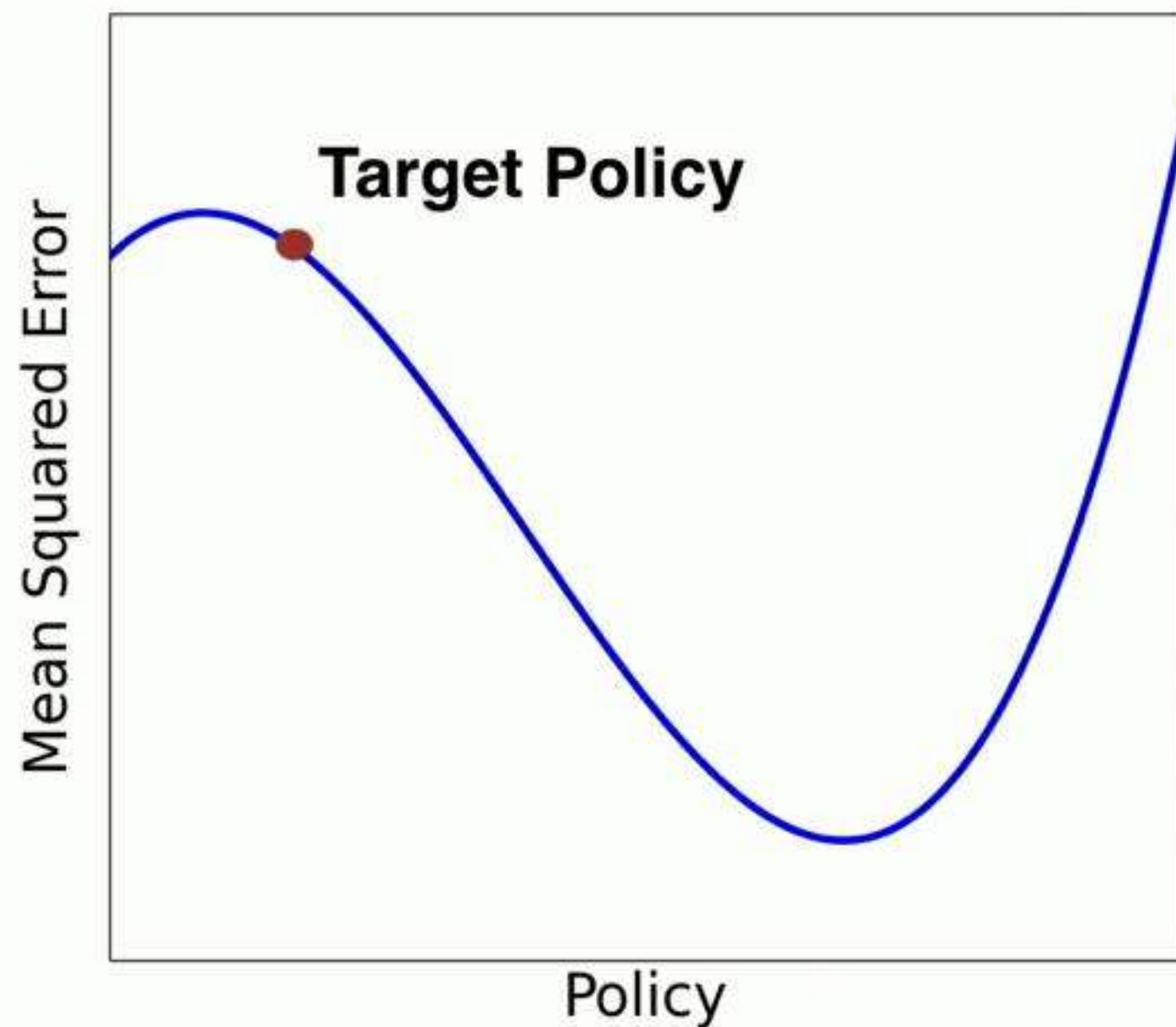
MSE **cannot** be estimated

Gradient of the MSE **can** be estimated
(with respect to behavior policy)

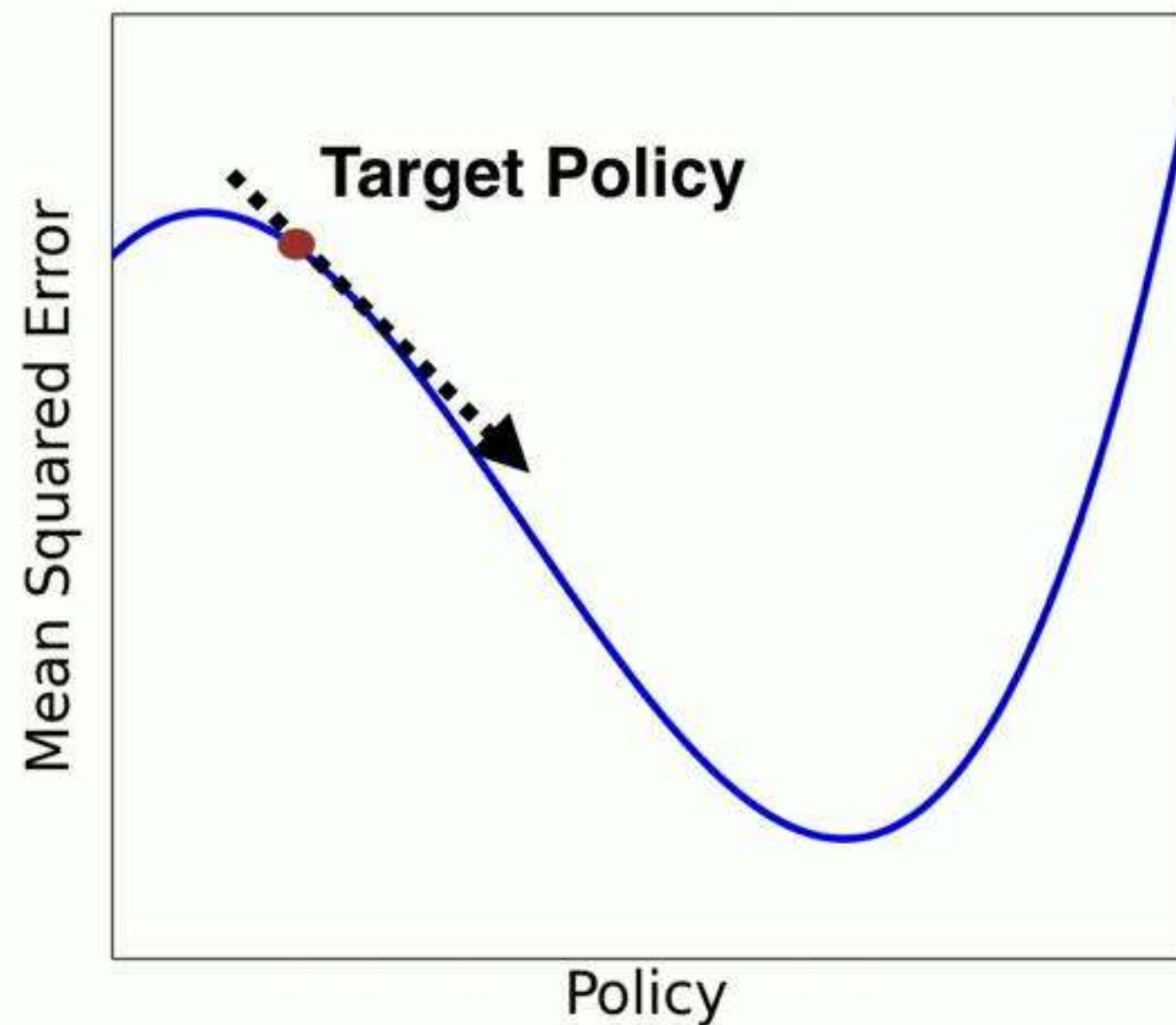
Behavior Policy Search



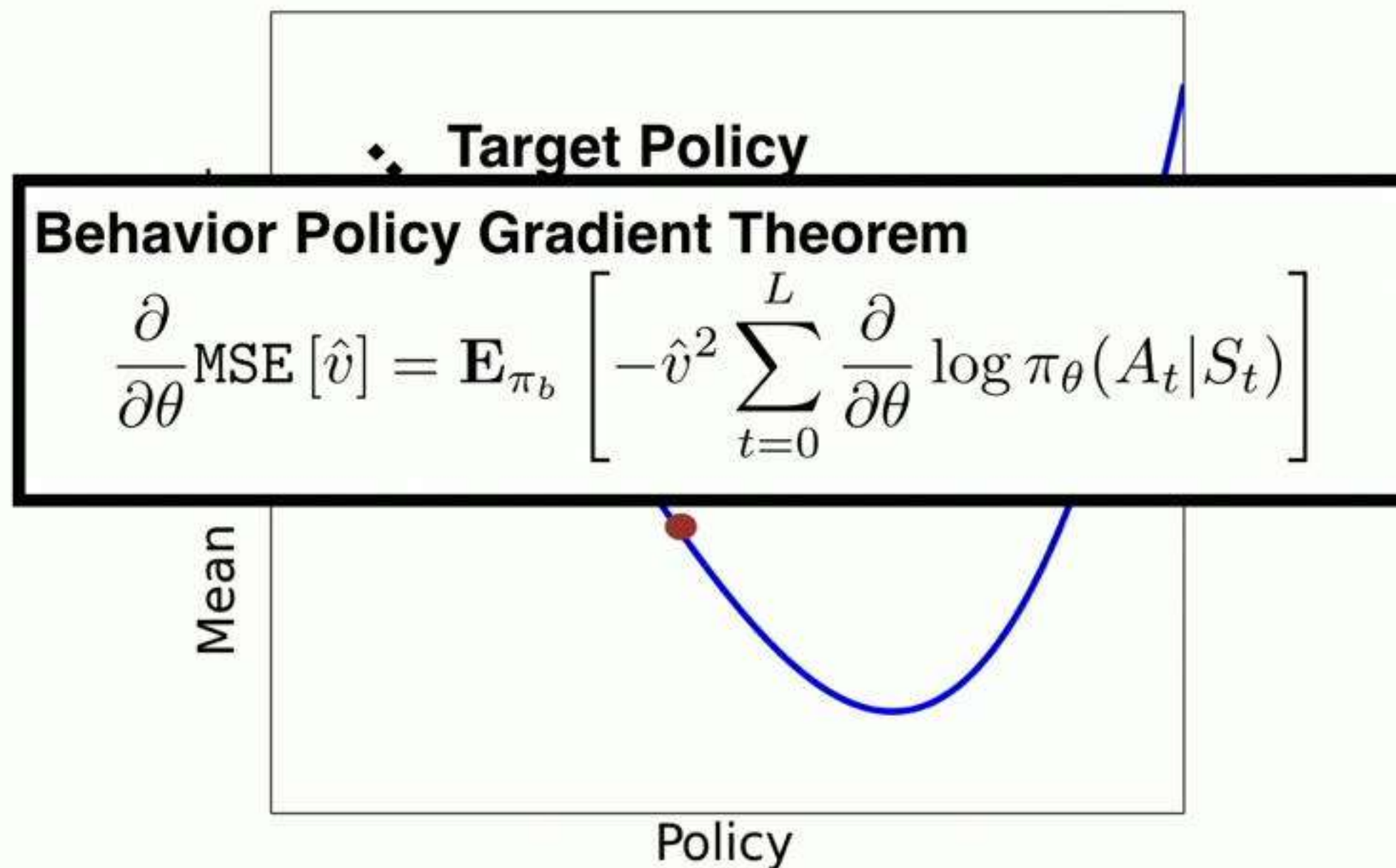
Behavior Policy Search



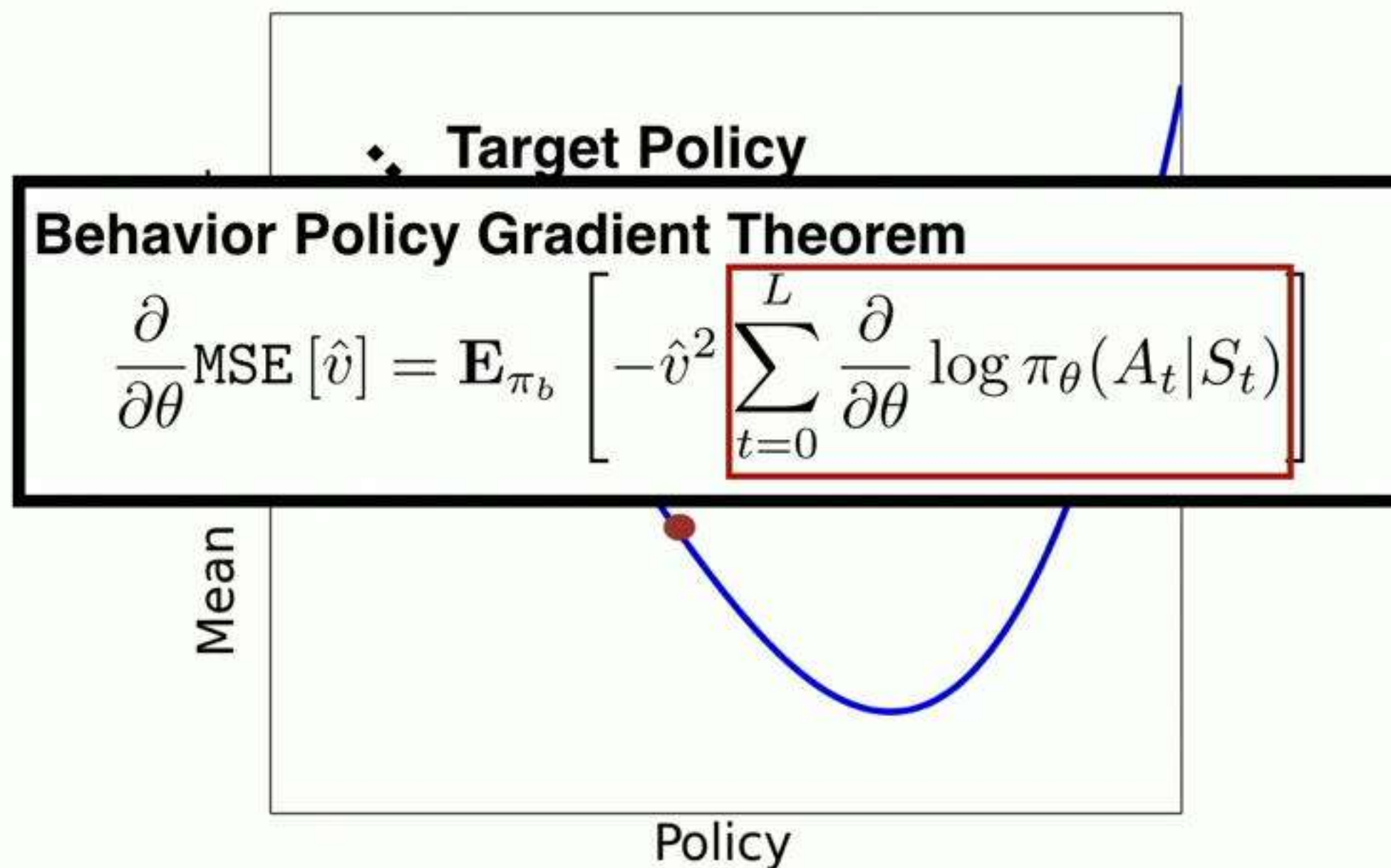
Behavior Policy Search



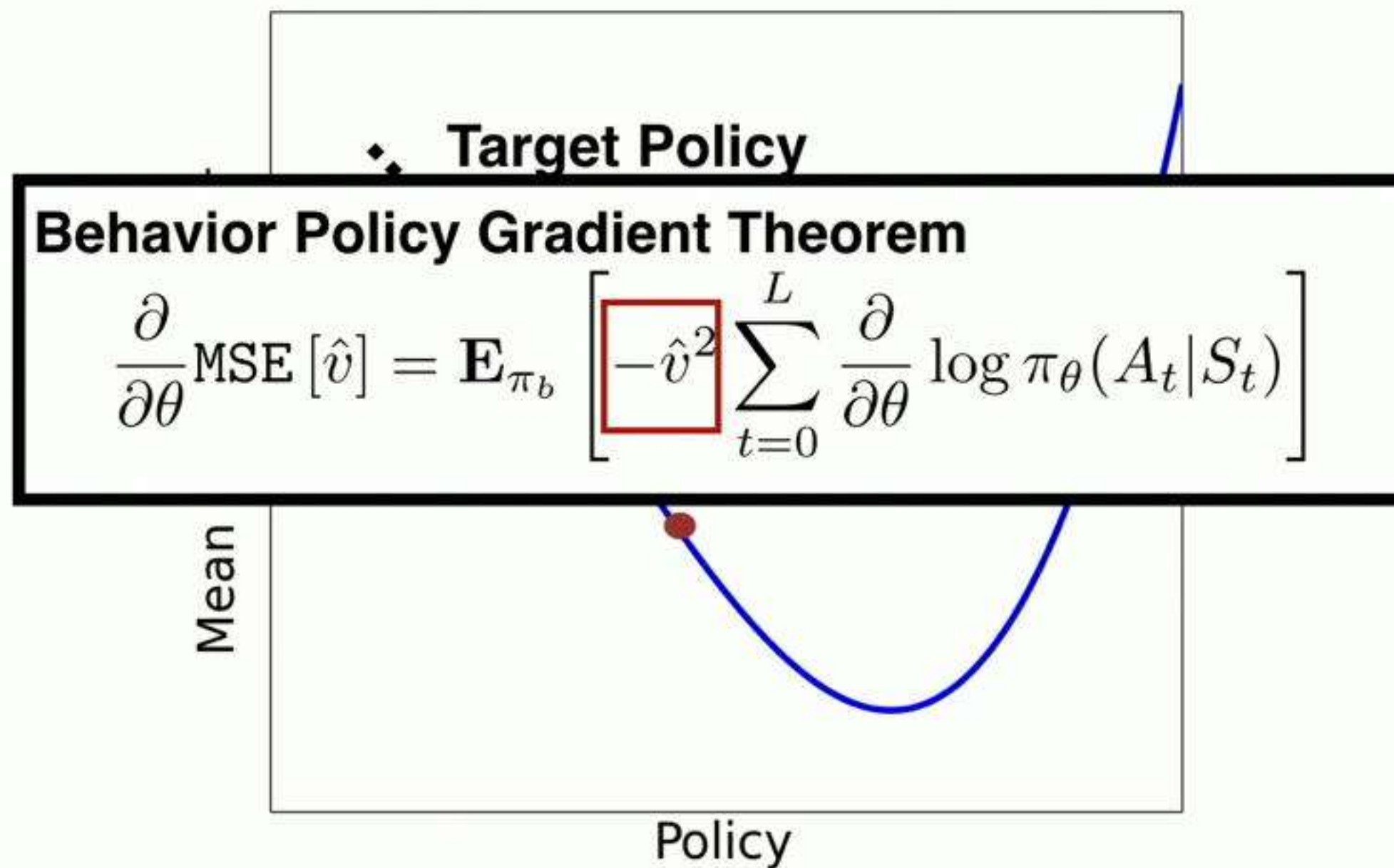
Behavior Policy Search



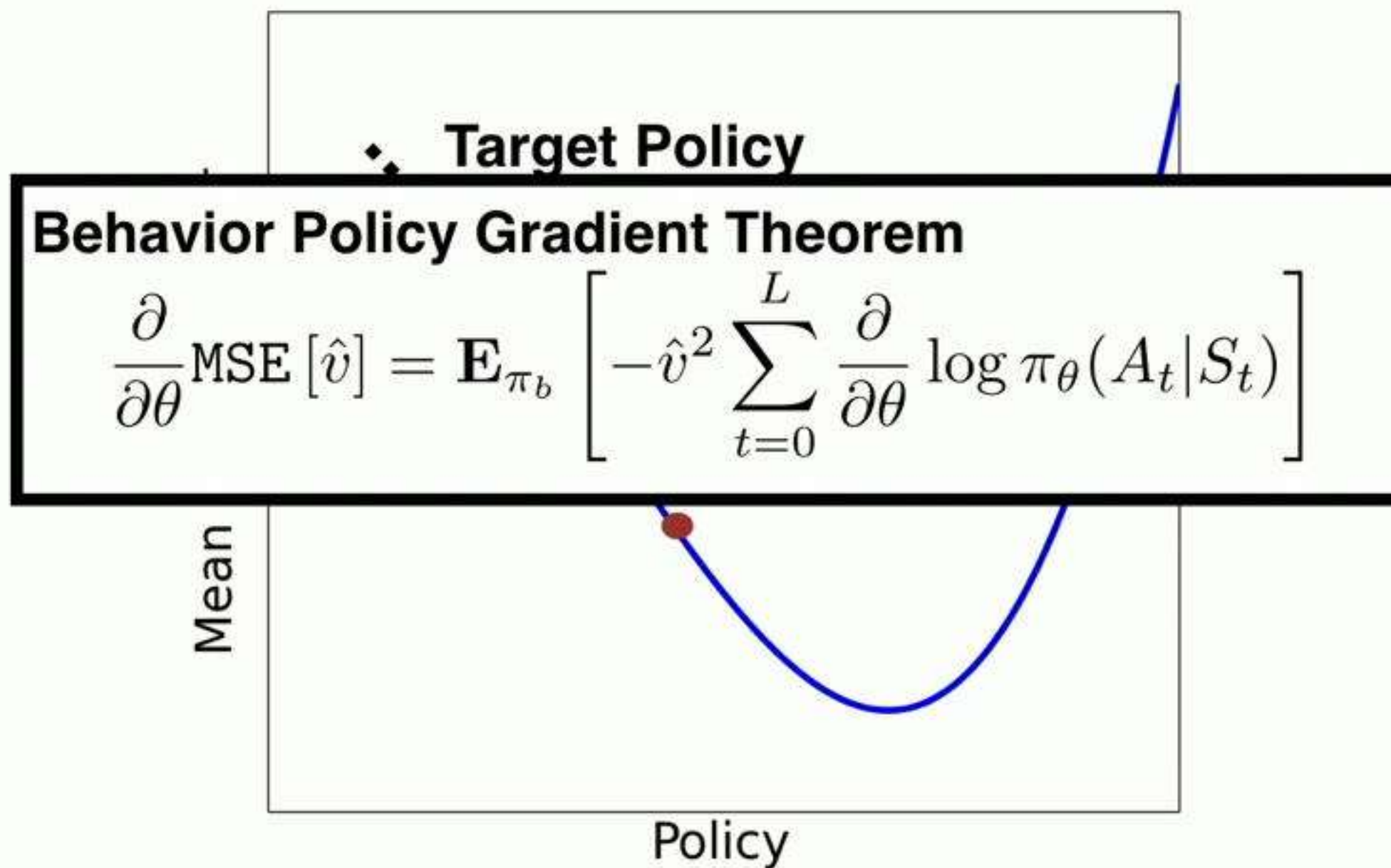
Behavior Policy Search



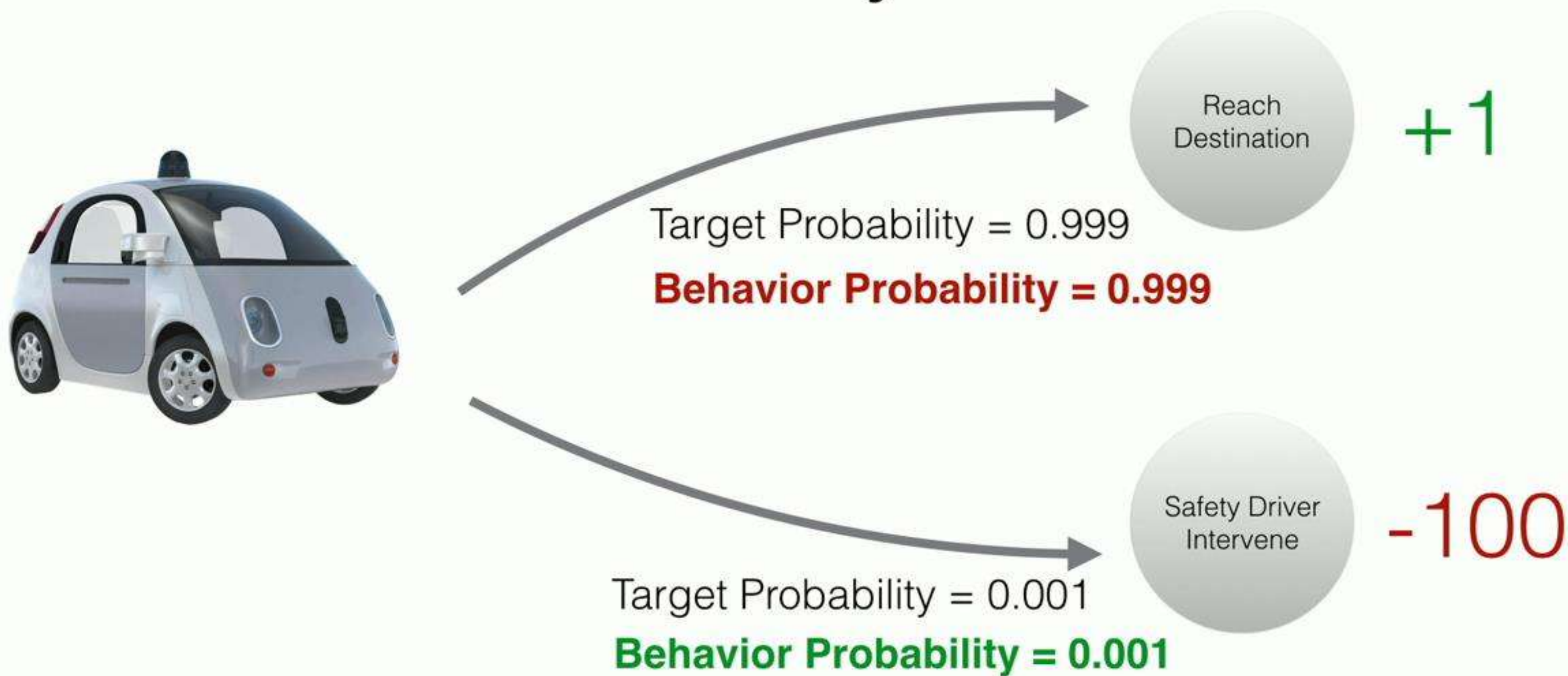
Behavior Policy Search



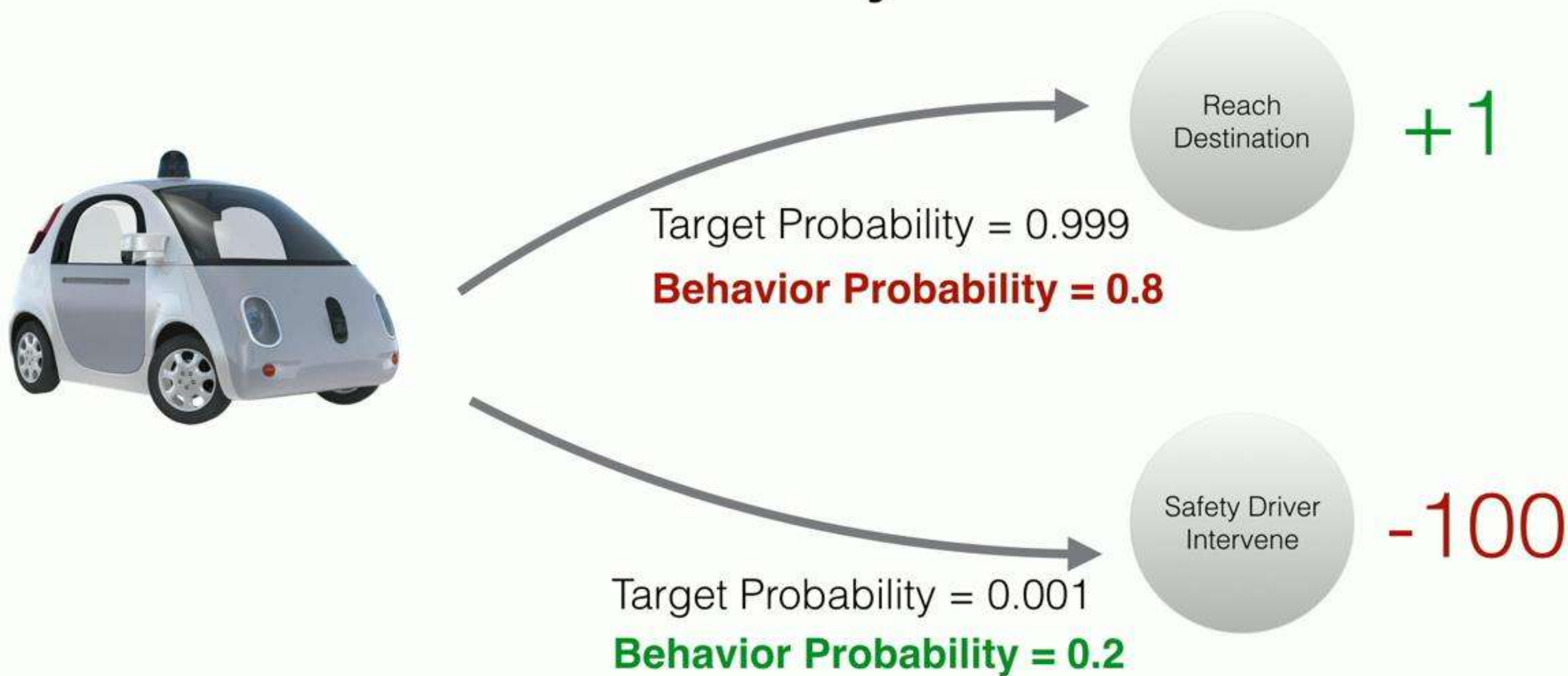
Behavior Policy Search



Behavior Policy Search



Behavior Policy Search



Behavior Policy Search

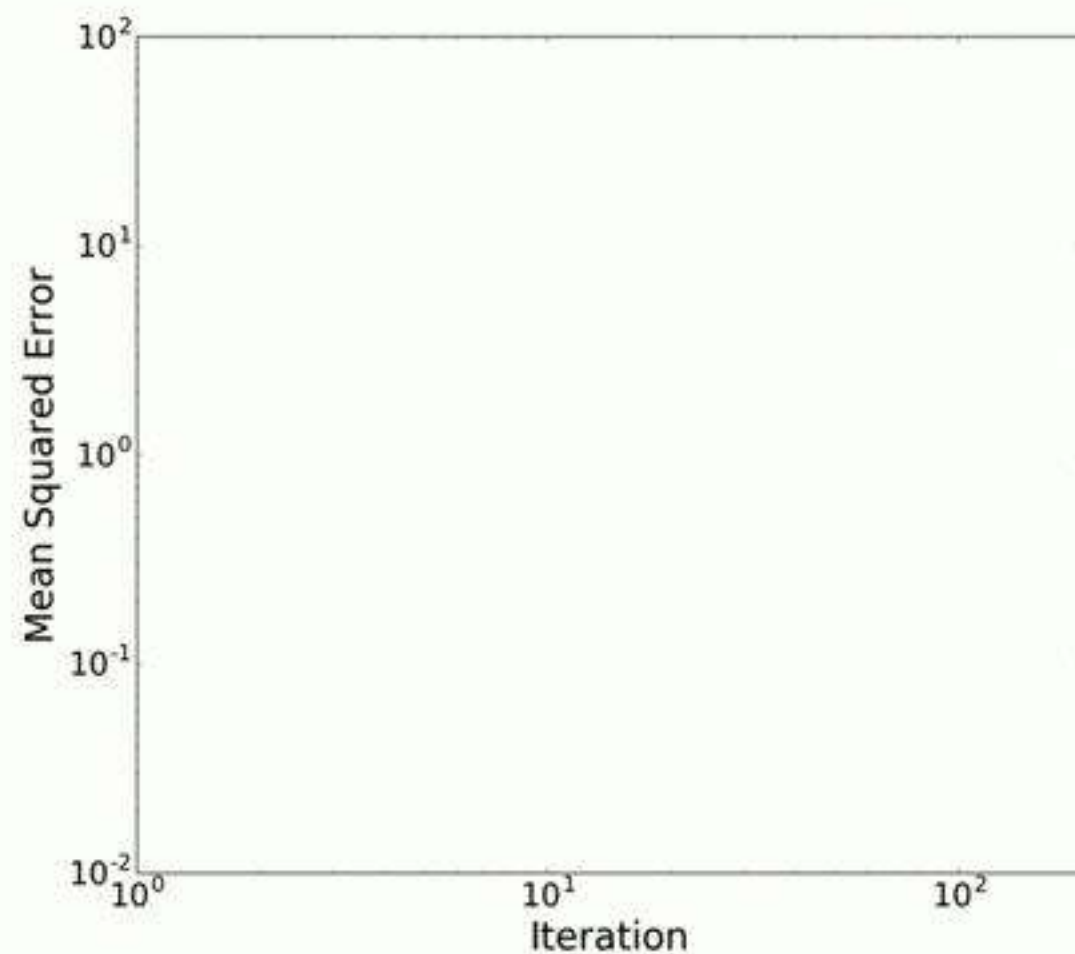
1. Collect **a small number** of trajectories with target policy.
2. Improve behavior policy.
3. Collect more trajectories with **improved** behavior policy.
4. Estimate the target policy's value **using all observed data.**
5. (Repeat if desired)

Can we improve on passive evaluation?

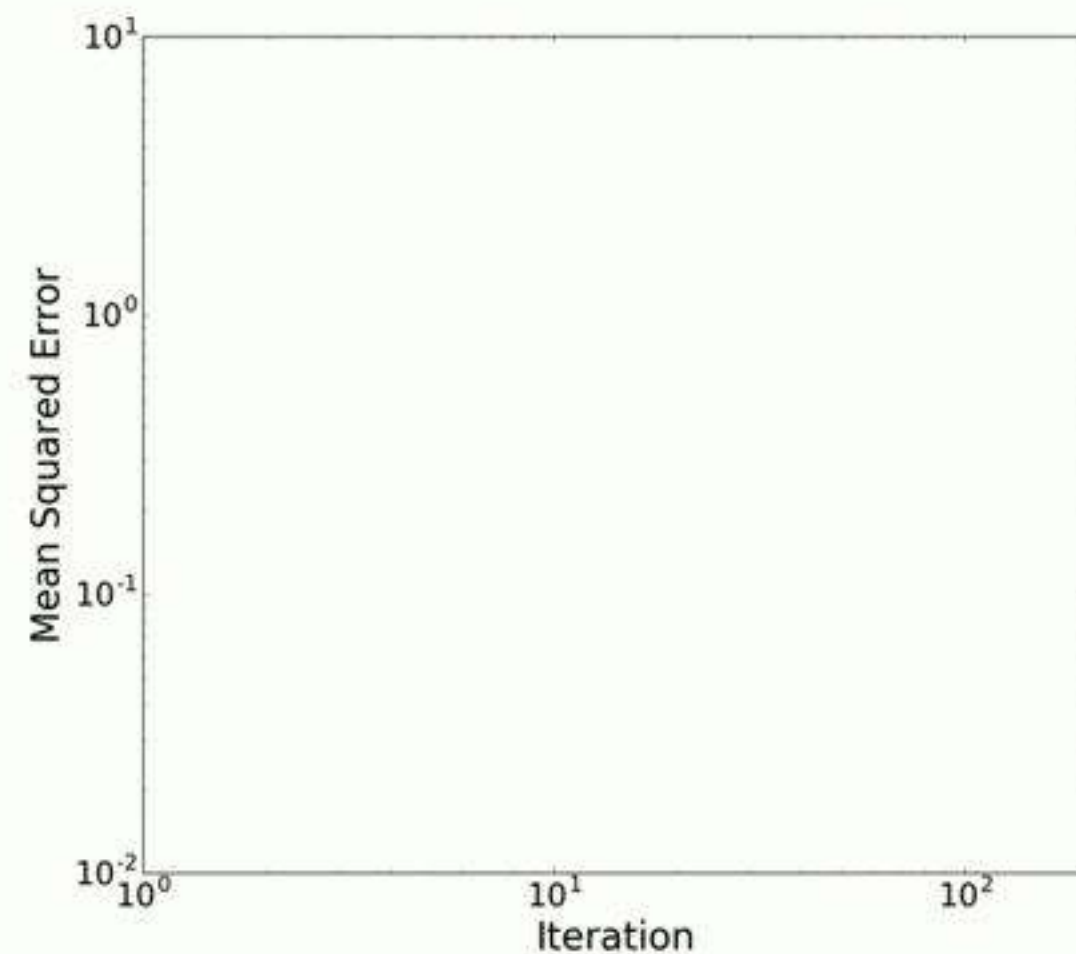
Challenges:

- ☒ Changing the behavior policy changes the data distribution.
- ☐ We do not know the right behavior policy.

Empirical Results

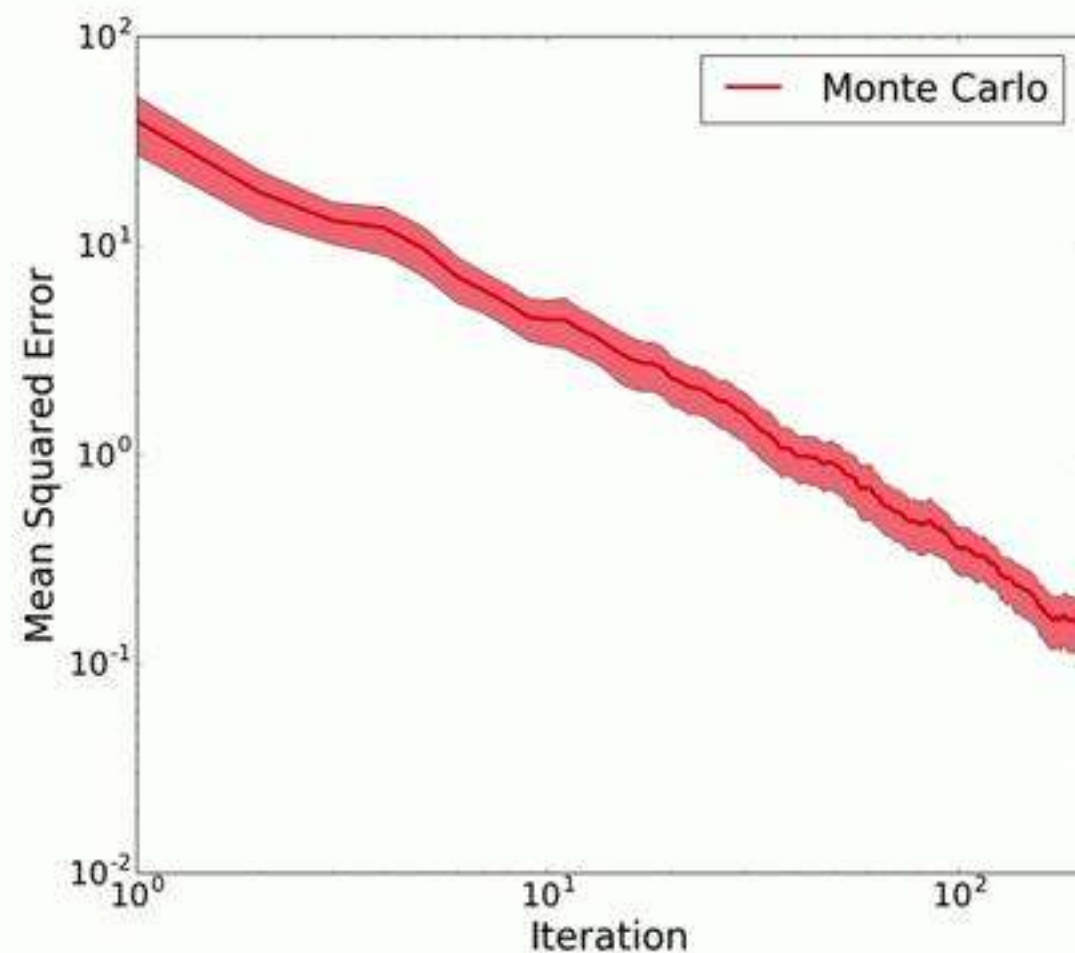


Cart-pole Swing-up

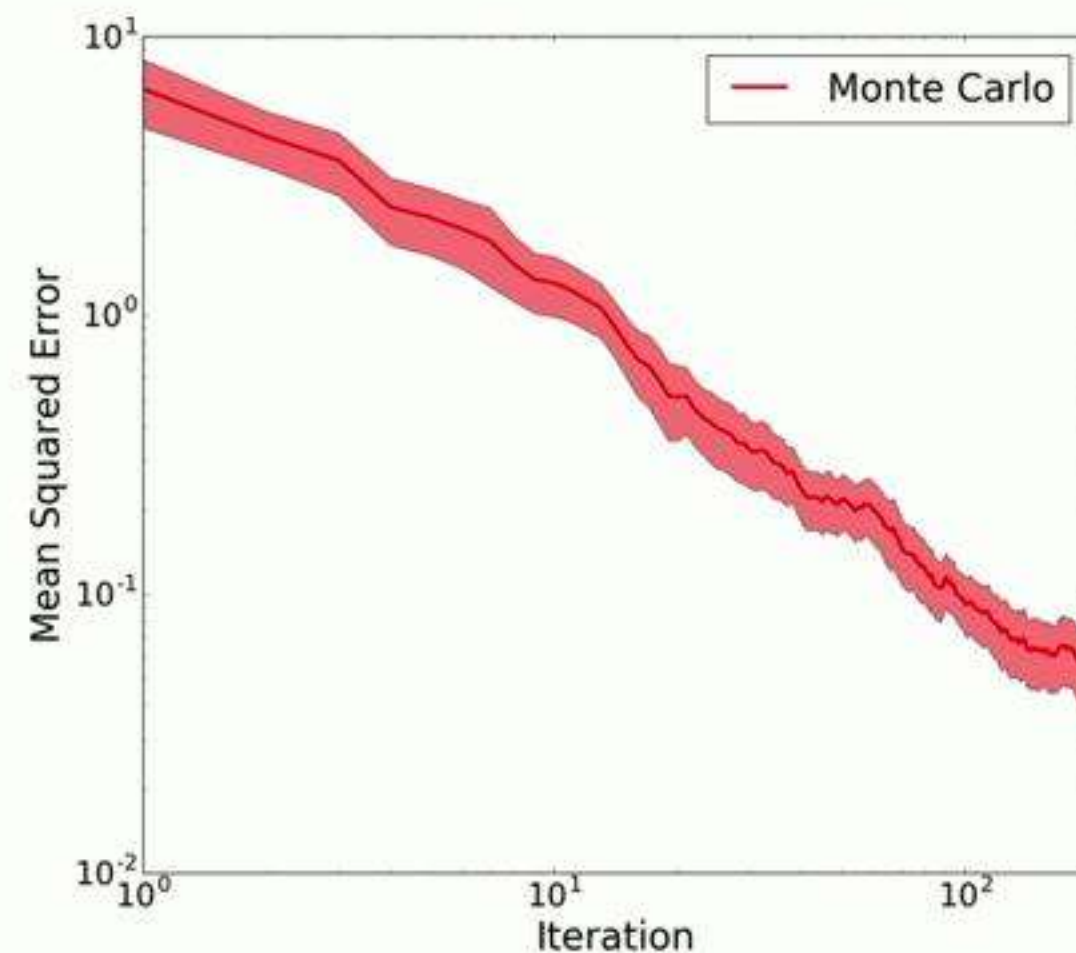


Acro-“bot”

Empirical Results

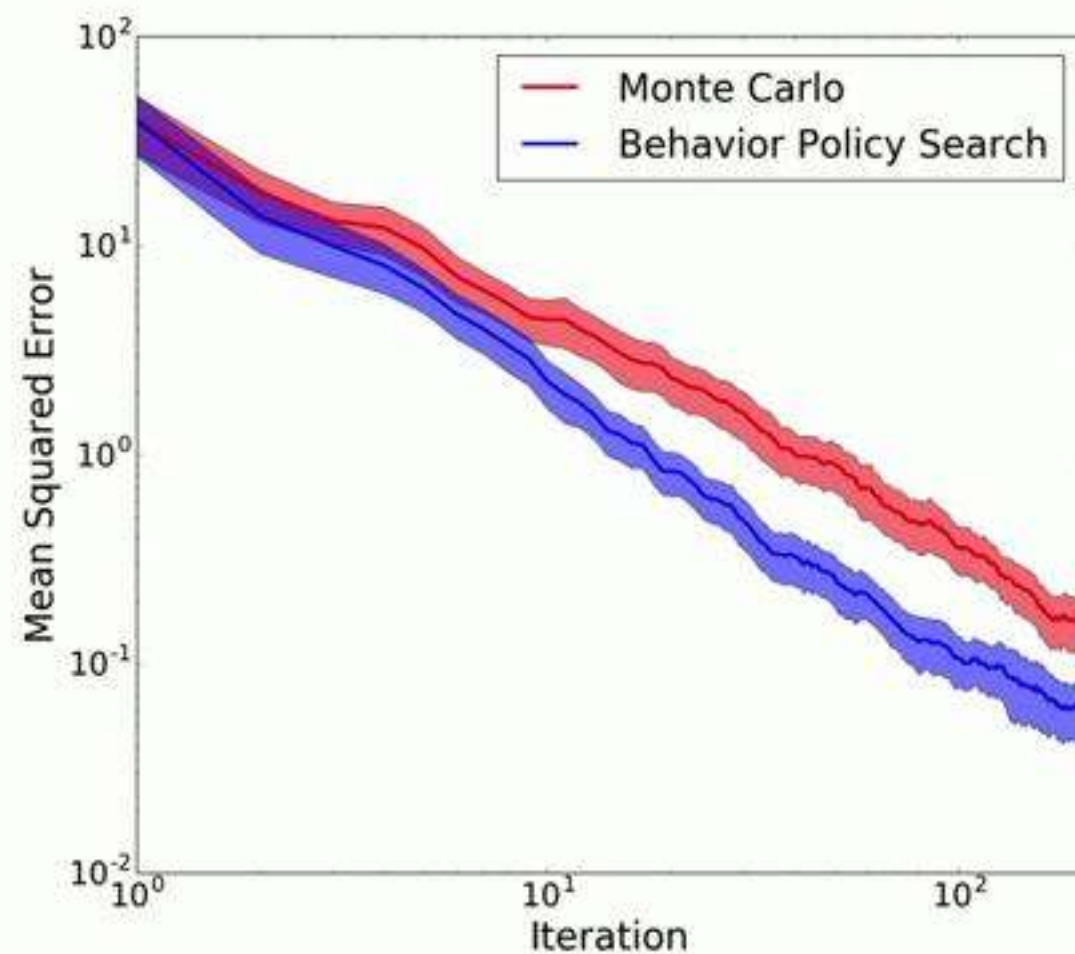


Cart-pole Swing-up

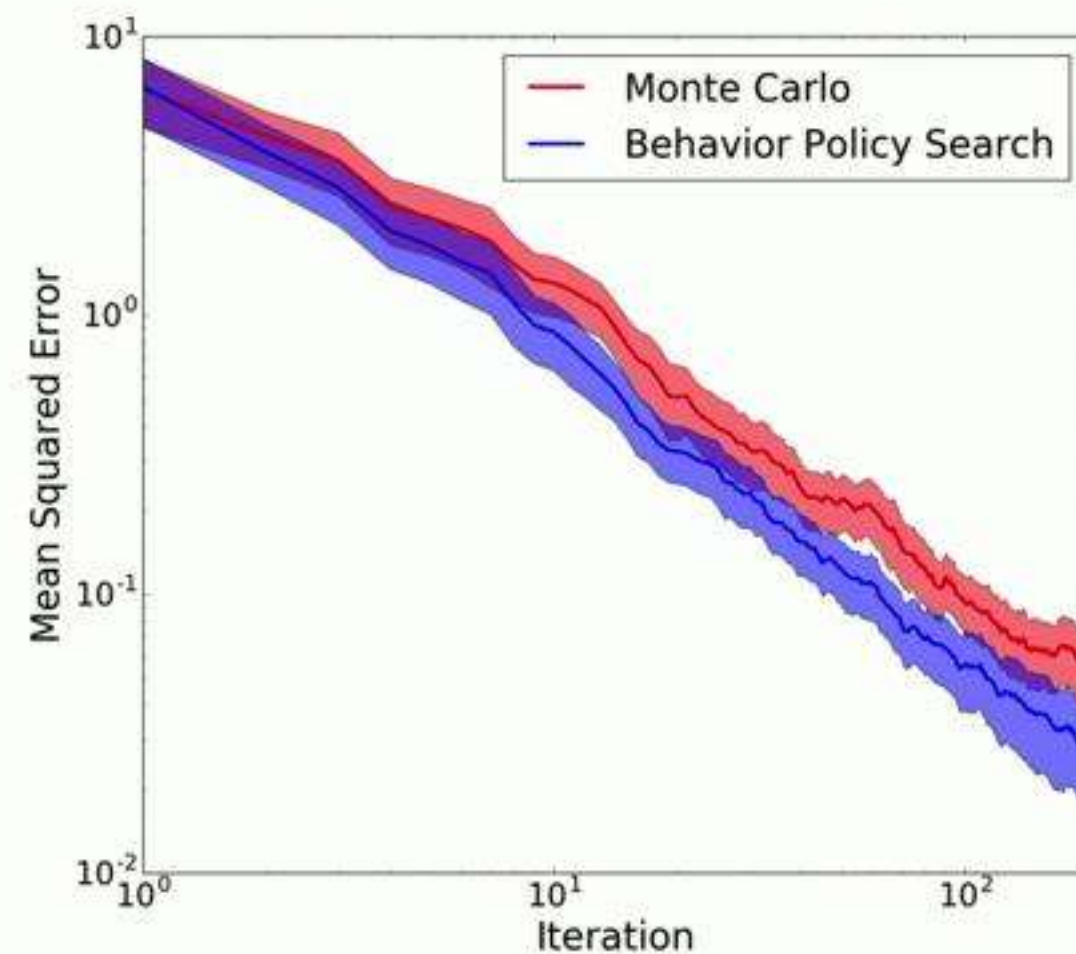


Acro-“bot”

Empirical Results



Cart-pole Swing-up



Acro-“bot”

1. Policy evaluation is critical for **safe and reliable** deployment of learning systems.
2. **Active** data collection is necessary for efficient reinforcement learning.

1. Policy evaluation is critical for **safe and reliable** deployment of learning systems.
2. **Active** data collection is necessary for efficient reinforcement learning.
3. Behavior policy search connects to **counterfactual** reasoning.

$$\frac{\pi_e(A_t|O_t)}{\pi_b(A_t|O_t)} \rightarrow \frac{\pi_e(A_t|O_t)}{\hat{\pi}_b(A_t|O_t)}$$

Josiah Hanna, Peter Stone (AAMAS 2019)

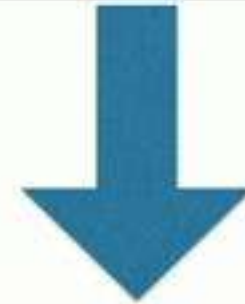
Josiah Hanna, Peter Stone, Scott Niekum (AAMAS 2017)

Josiah Hanna, Scott Niekum, Peter Stone (ICML 2019)

How can an agent efficiently learn to predict the effects of its actions?



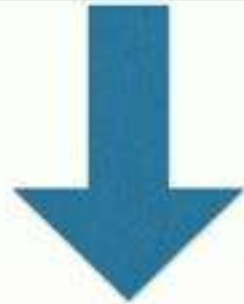
Estimate task performance for a fixed policy.



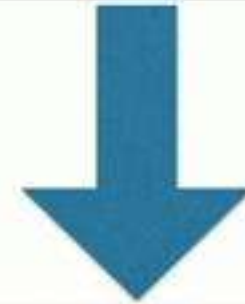
Predicting how actions change the state of the world.

Can reinforcement learning be data efficient enough for real world applications?

How can an agent efficiently learn to predict the effects of its actions?



Estimate task performance for a fixed policy.



Predicting how actions change the state of the world.

Can reinforcement learning be data efficient enough for real world applications?

Predicting Action Effects



Predicting Action Effects

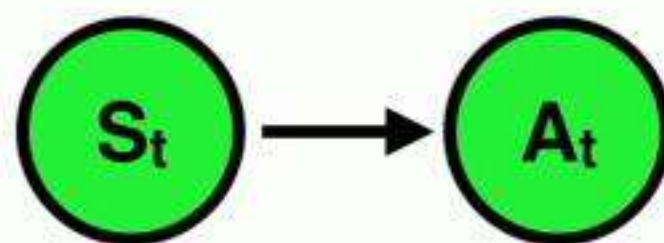


Accelerate

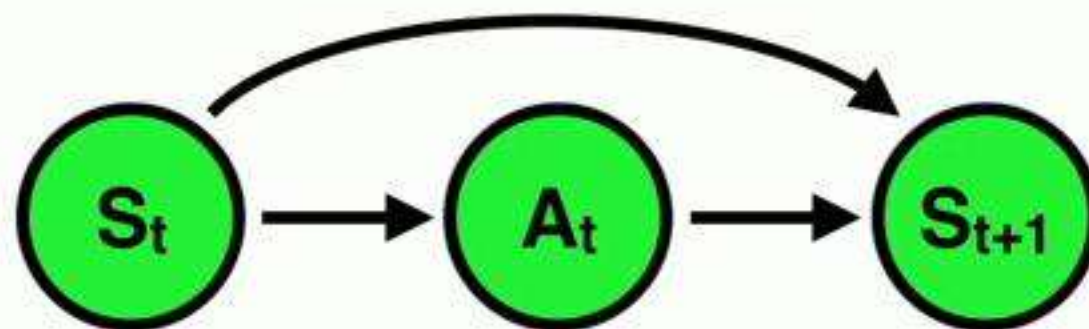
Predicting Action Effects



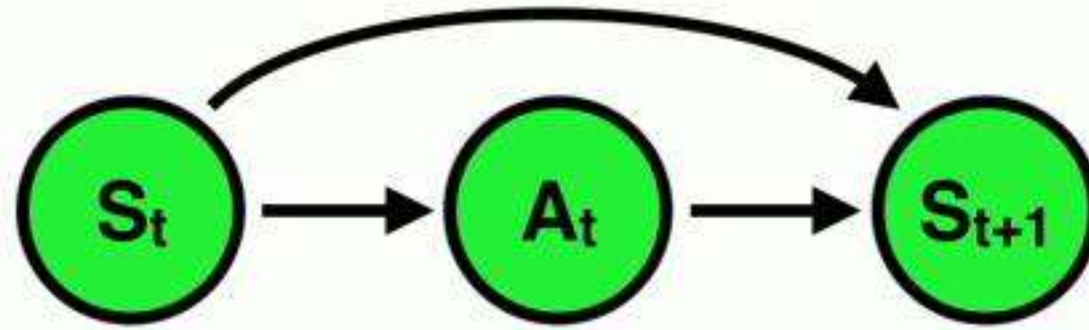
Predicting Action Effects



Predicting Action Effects



Predicting Action Effects



1. Data efficient policy evaluation
 - Simulate outcomes.
2. Data efficient policy learning
 - Learn from simulated experience.

Simulated Robotics



Credit: Patrick MacAlpine

Real World Robotics



Credit: StreamTeam HTWK



Credit: Patrick MacAlpine



Credit: Patrick MacAlpine



Learning Robot Control



State: Position / velocity of robot's parts

Action: Joint commands

Reward: Forward velocity

Policy: Joint positions to joint commands

Learning in Simulation



π

Learning in Simulation



π

$$S_0, A_0, R_0, \dots, S_L, A_L, R_L$$

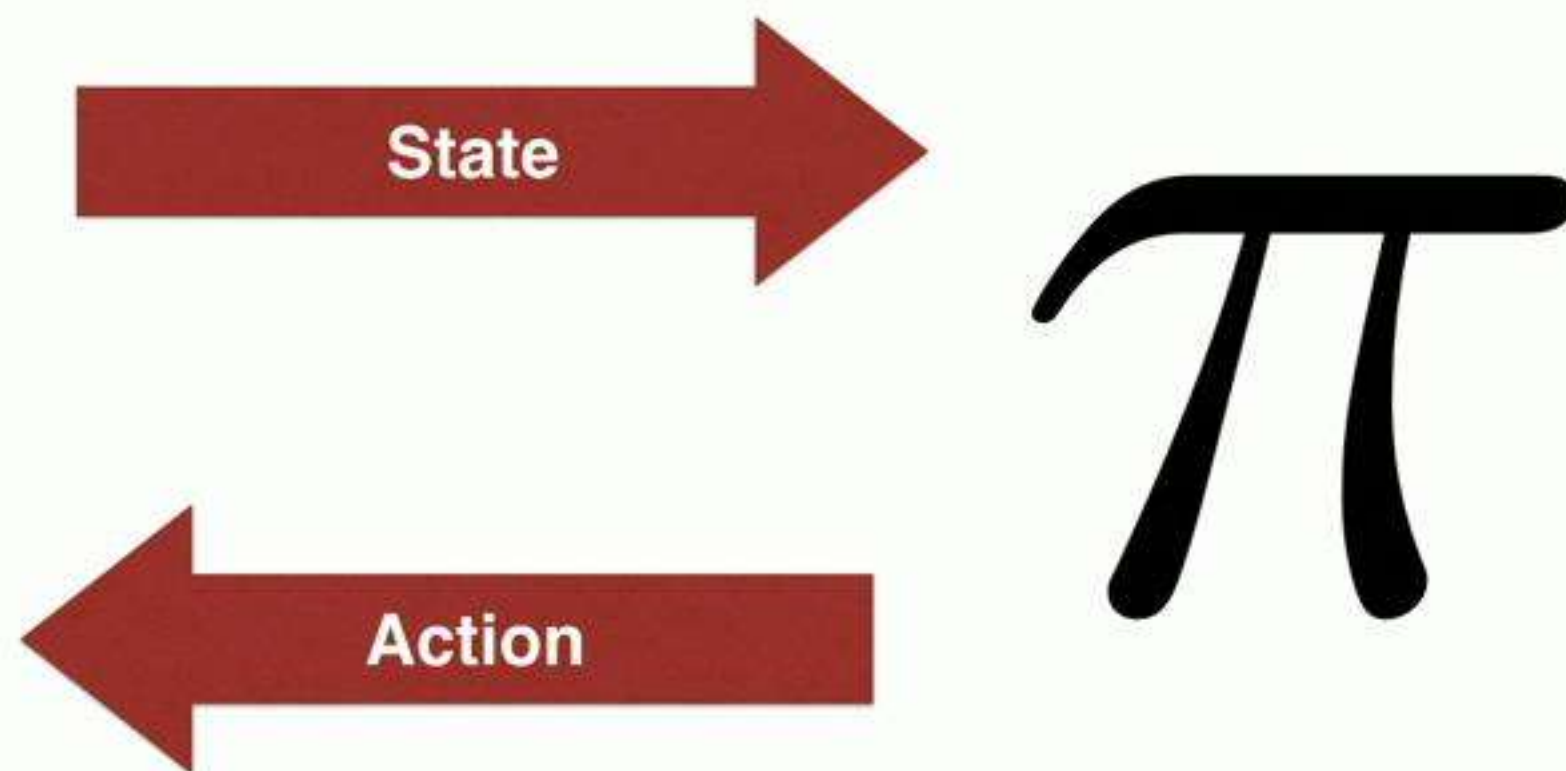
Learning in Simulation



π

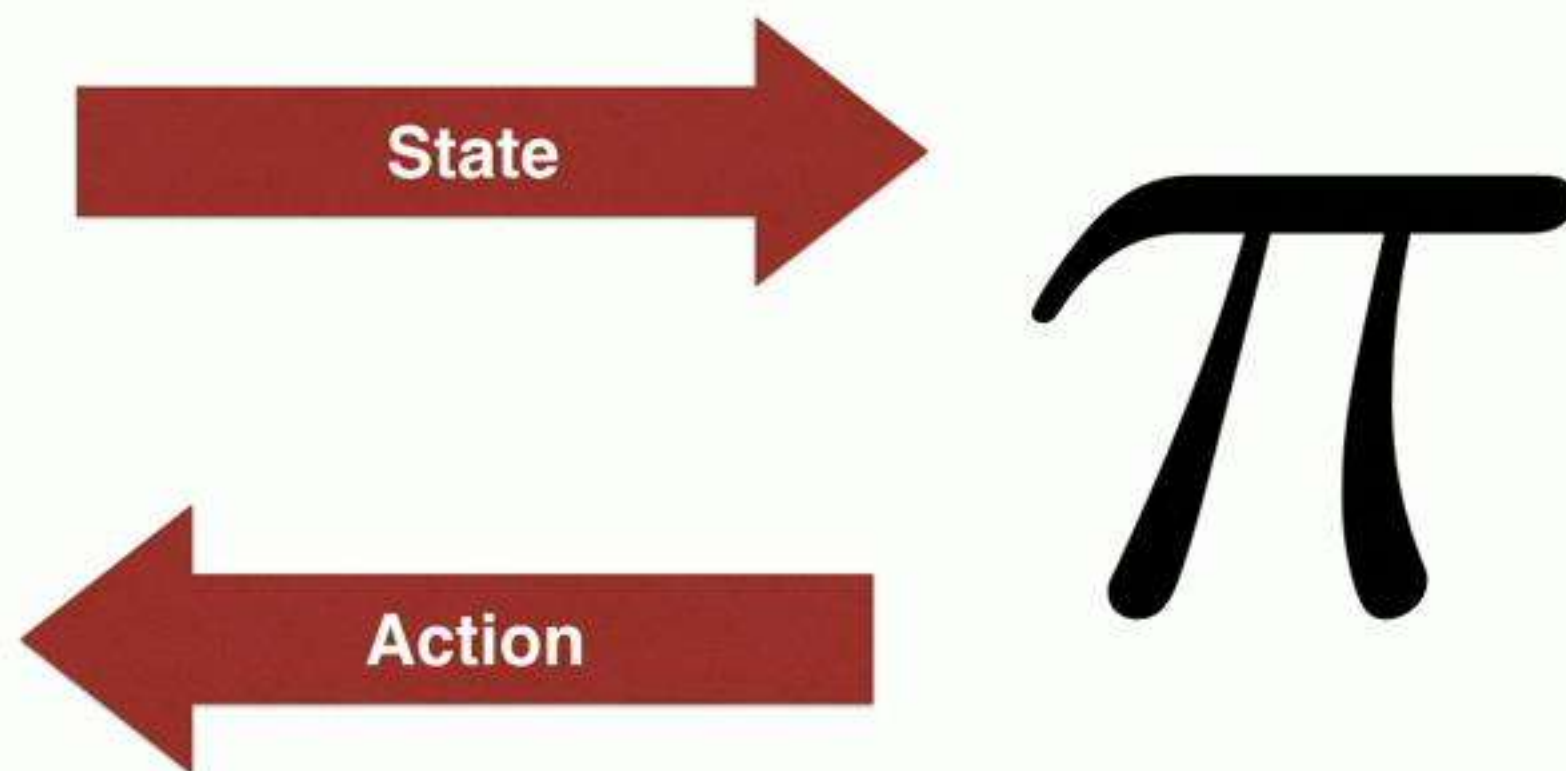
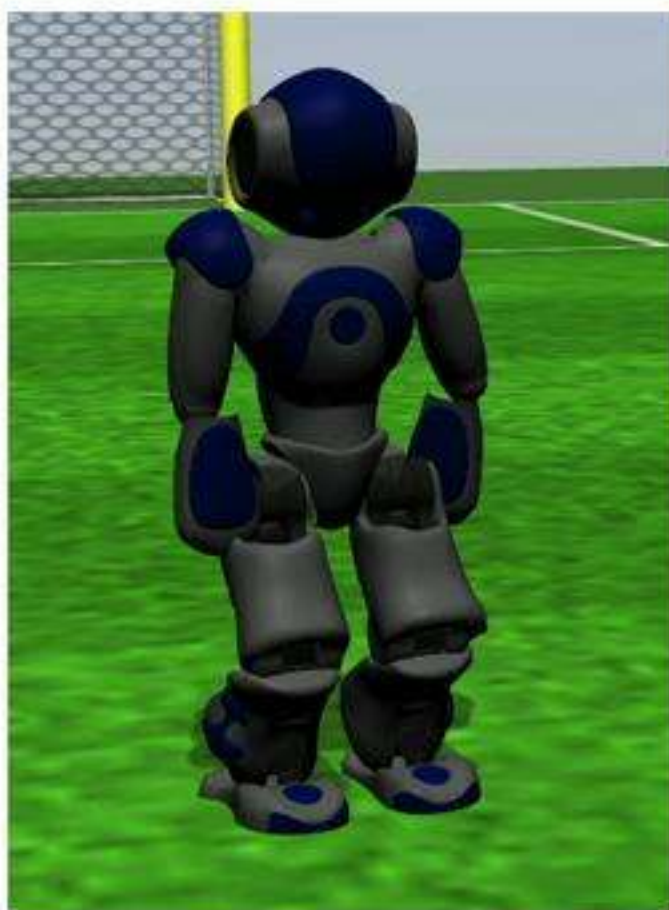
$$S_0, A_0, R_0, \dots, S_L, A_L, R_L$$

Learning in Simulation



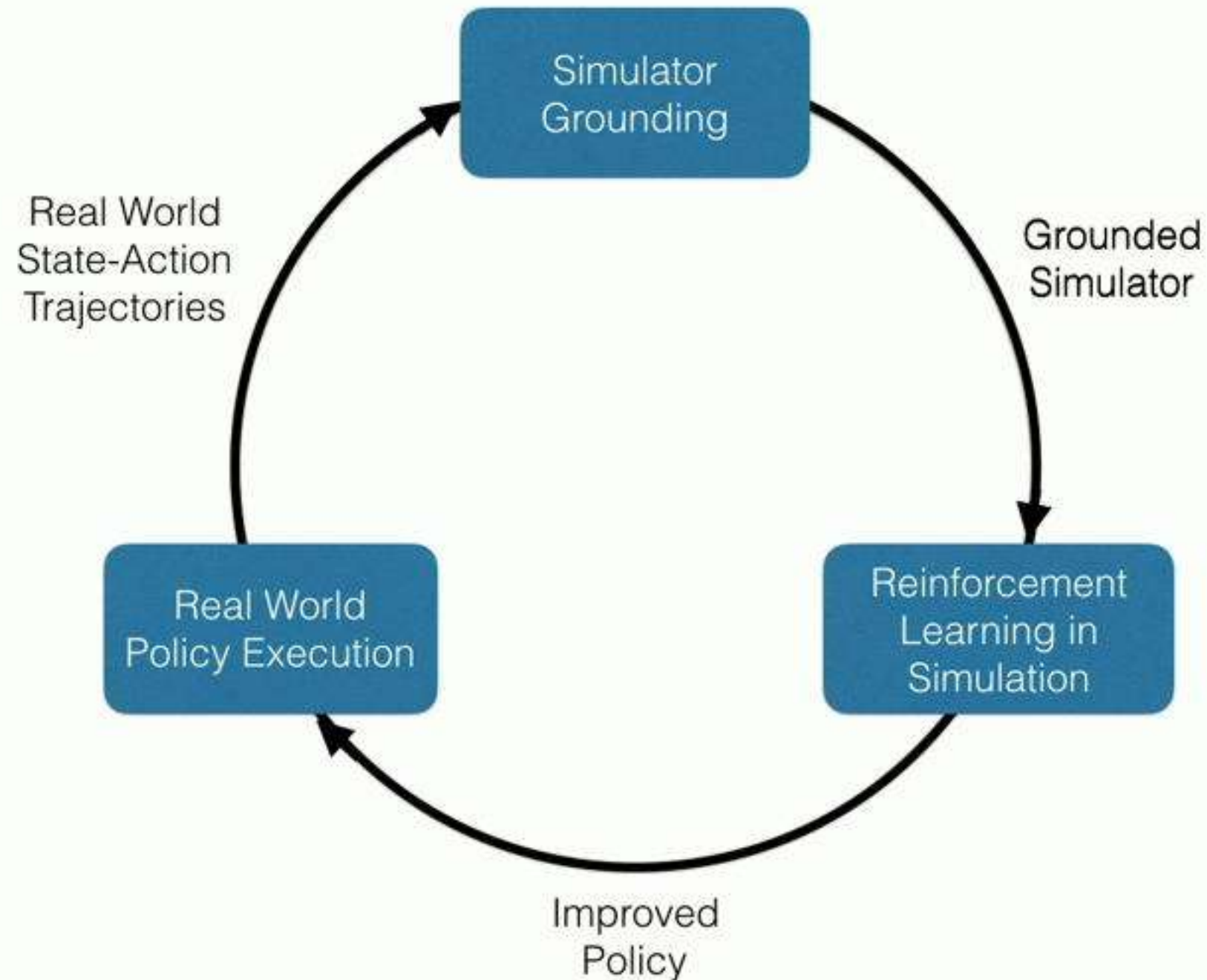
$$S_0, A_0, R_0, \dots, S_L, A_L, R_L$$

Learning in Simulation

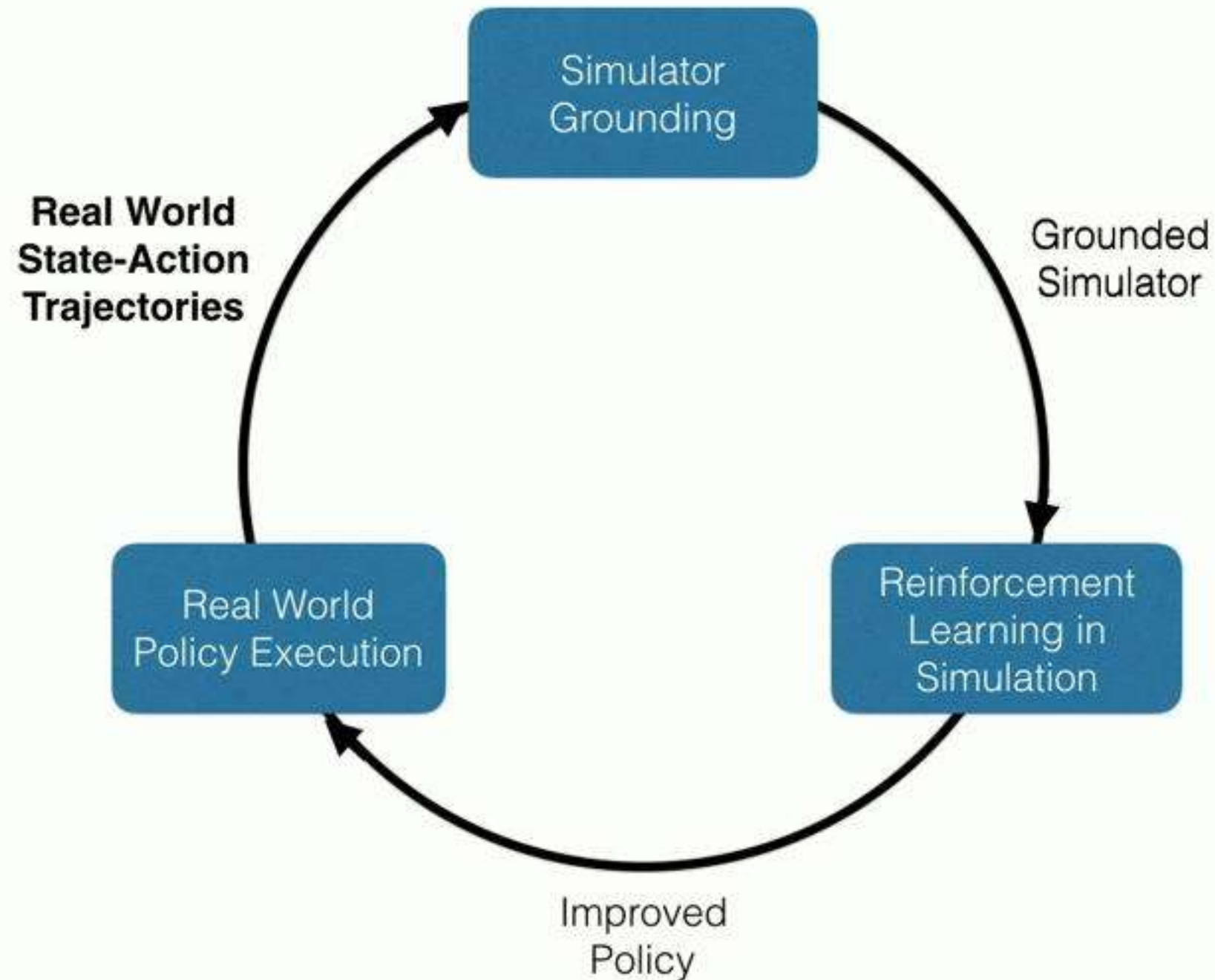


$$S_0, A_0, R_0, \dots, S_L, A_L, R_L$$

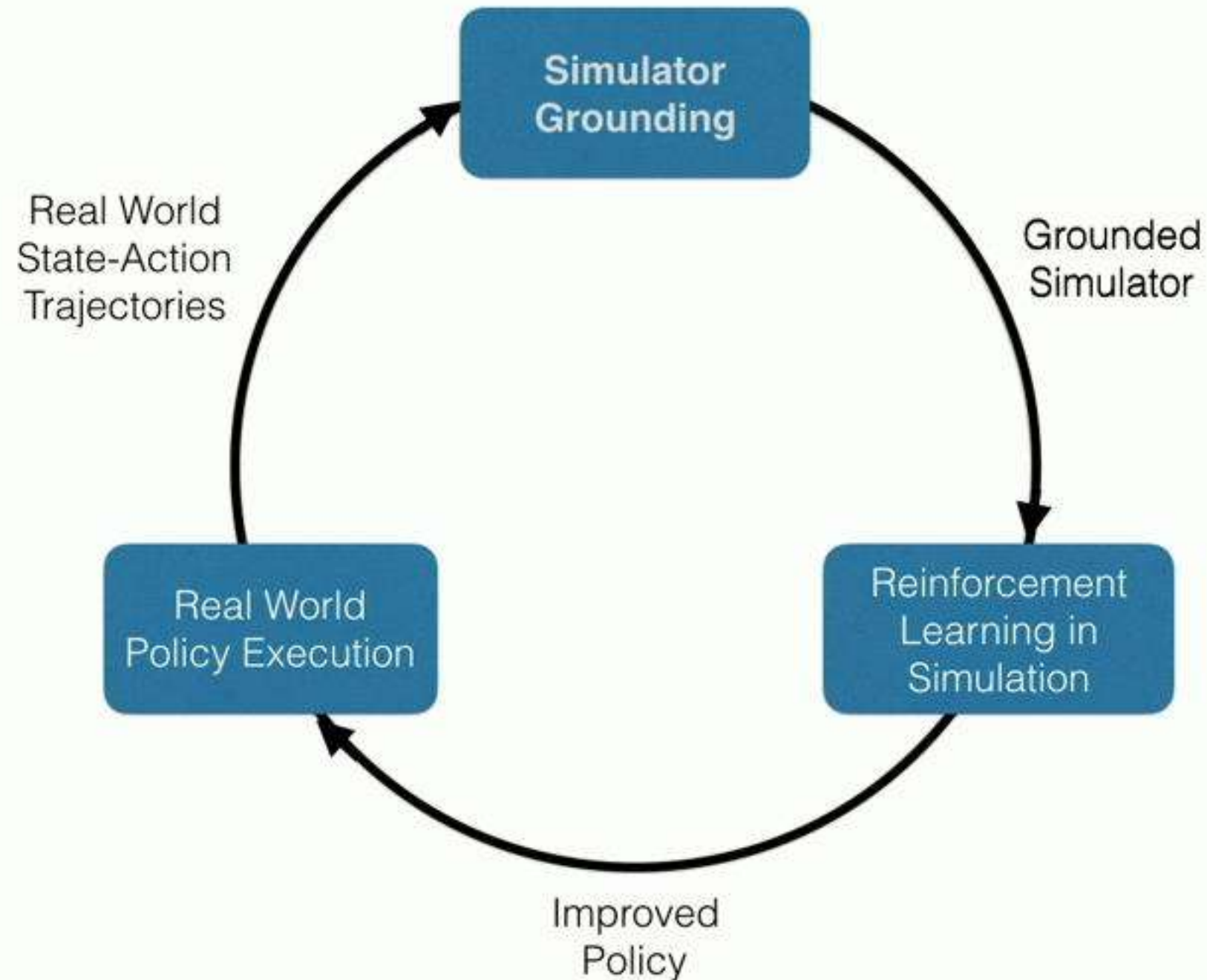
Grounded Simulation Learning



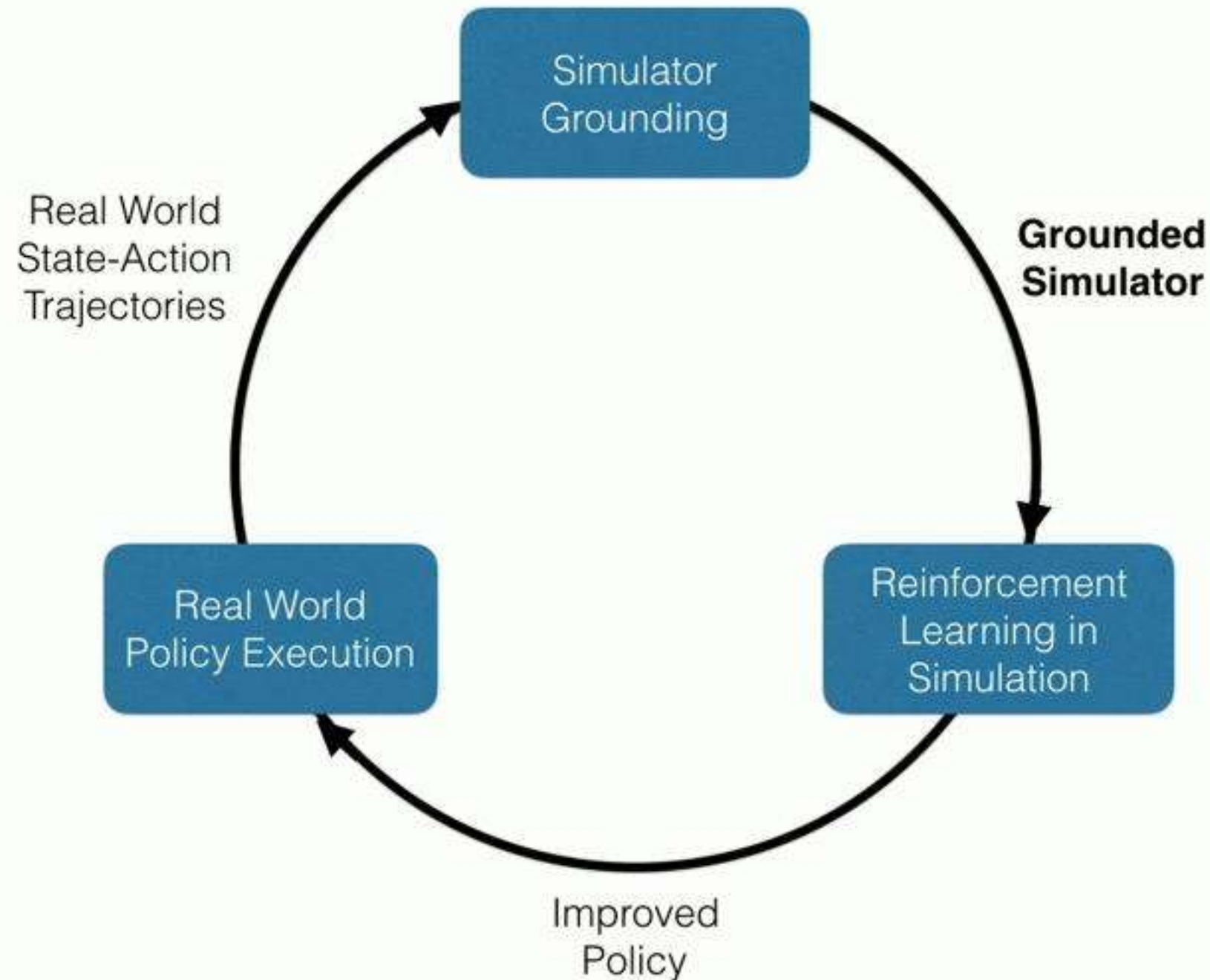
Grounded Simulation Learning



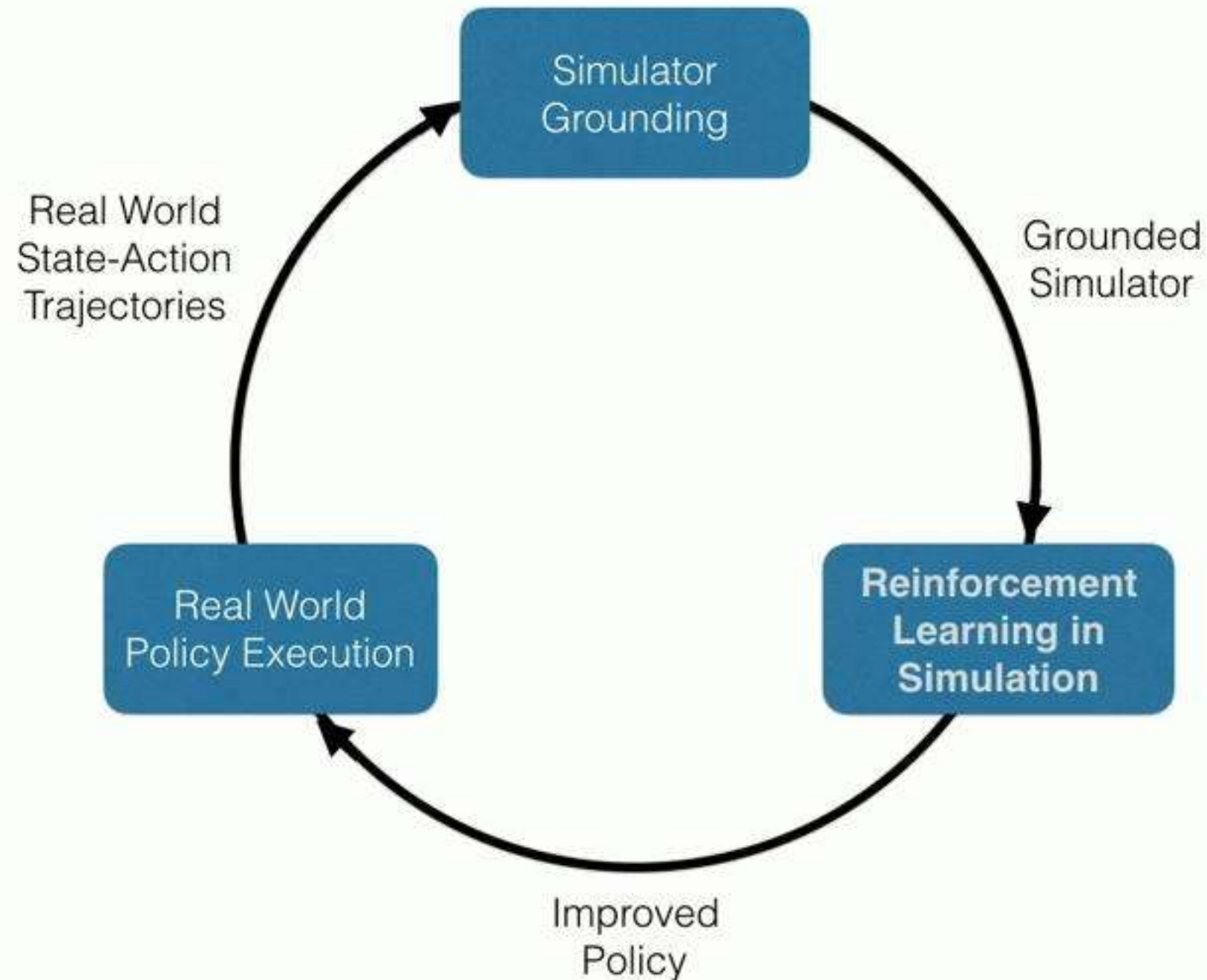
Grounded Simulation Learning



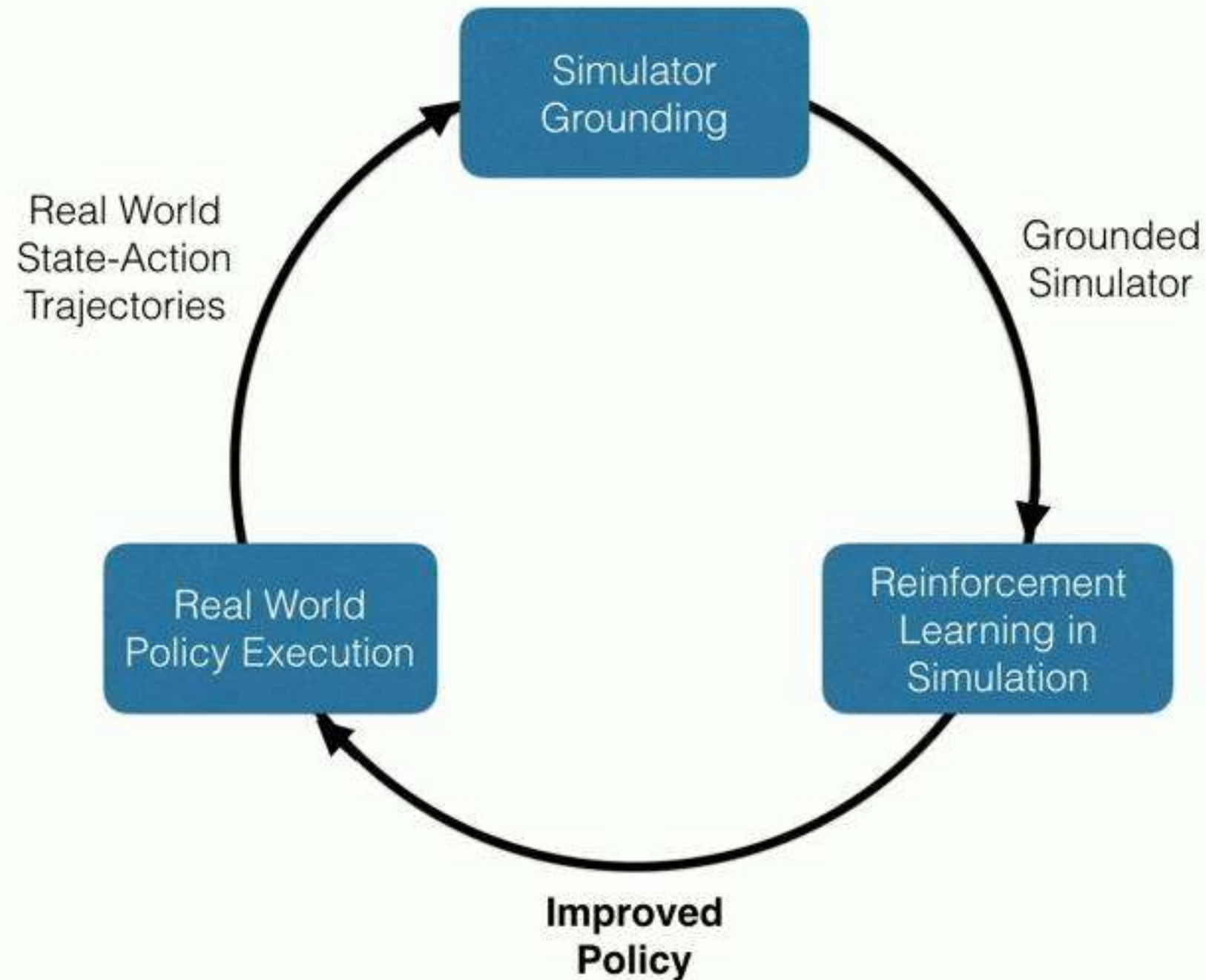
Grounded Simulation Learning



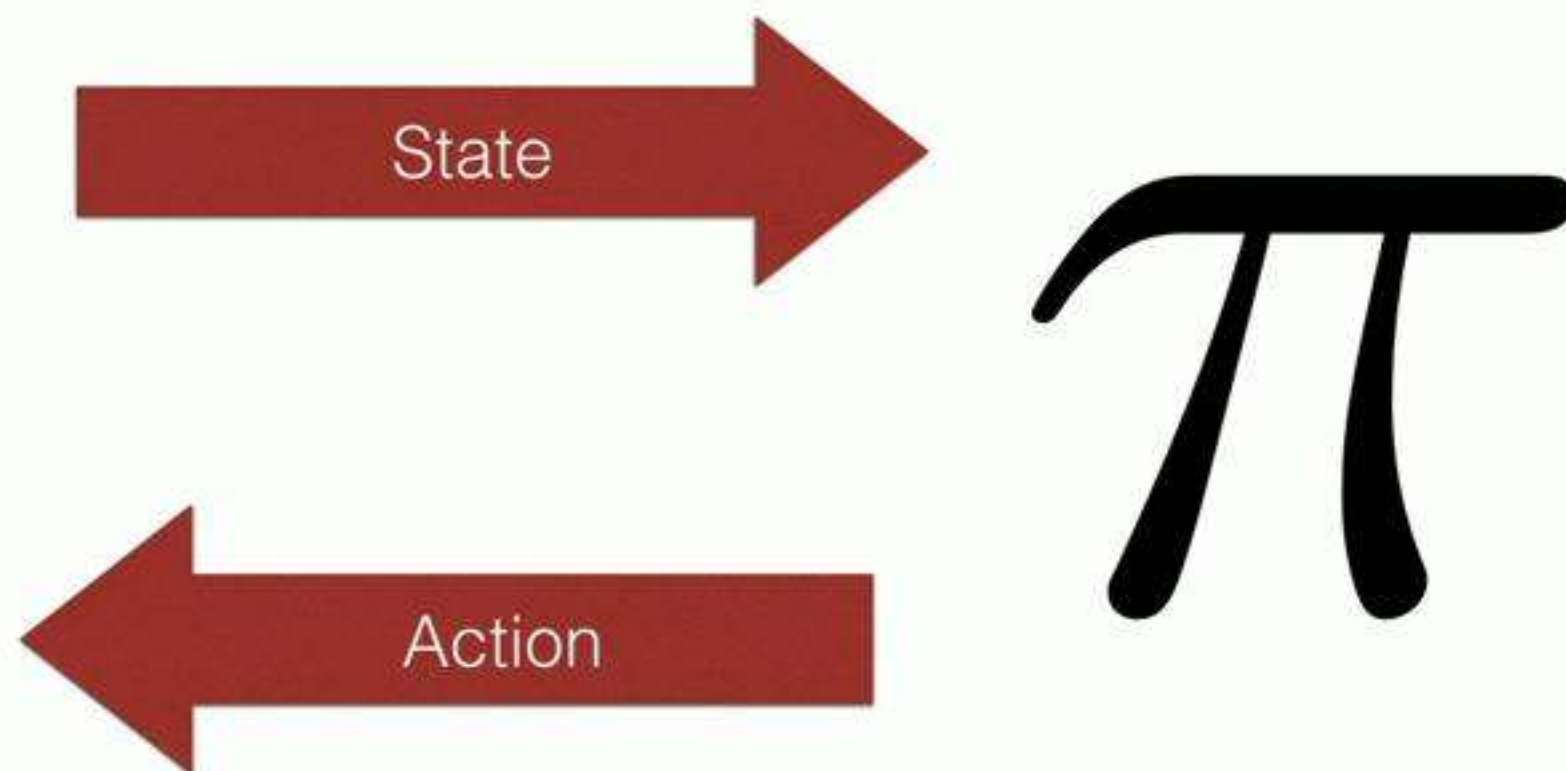
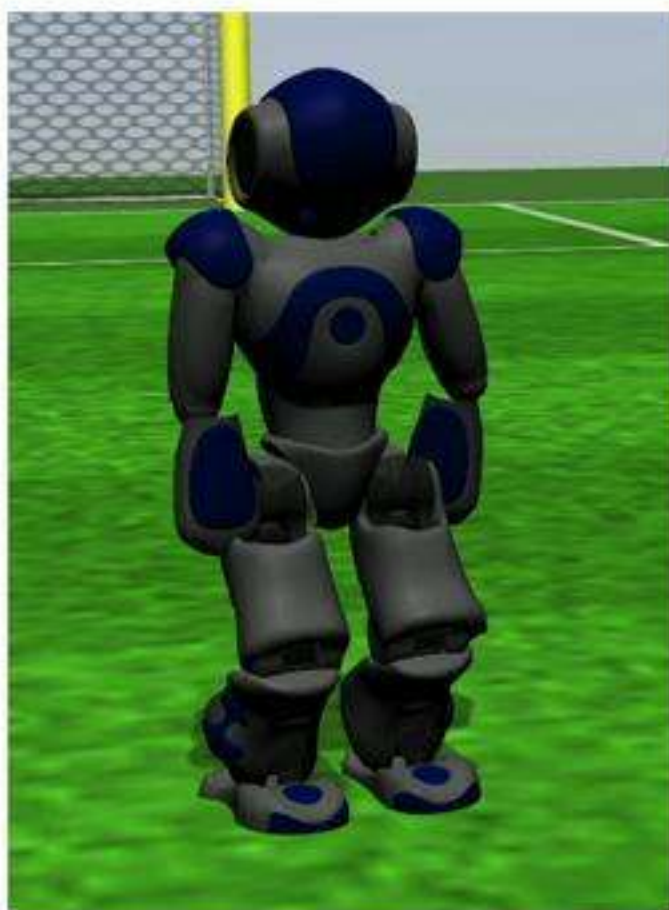
Grounded Simulation Learning



Grounded Simulation Learning



How do we make simulation more realistic?



How do we make simulation more realistic?

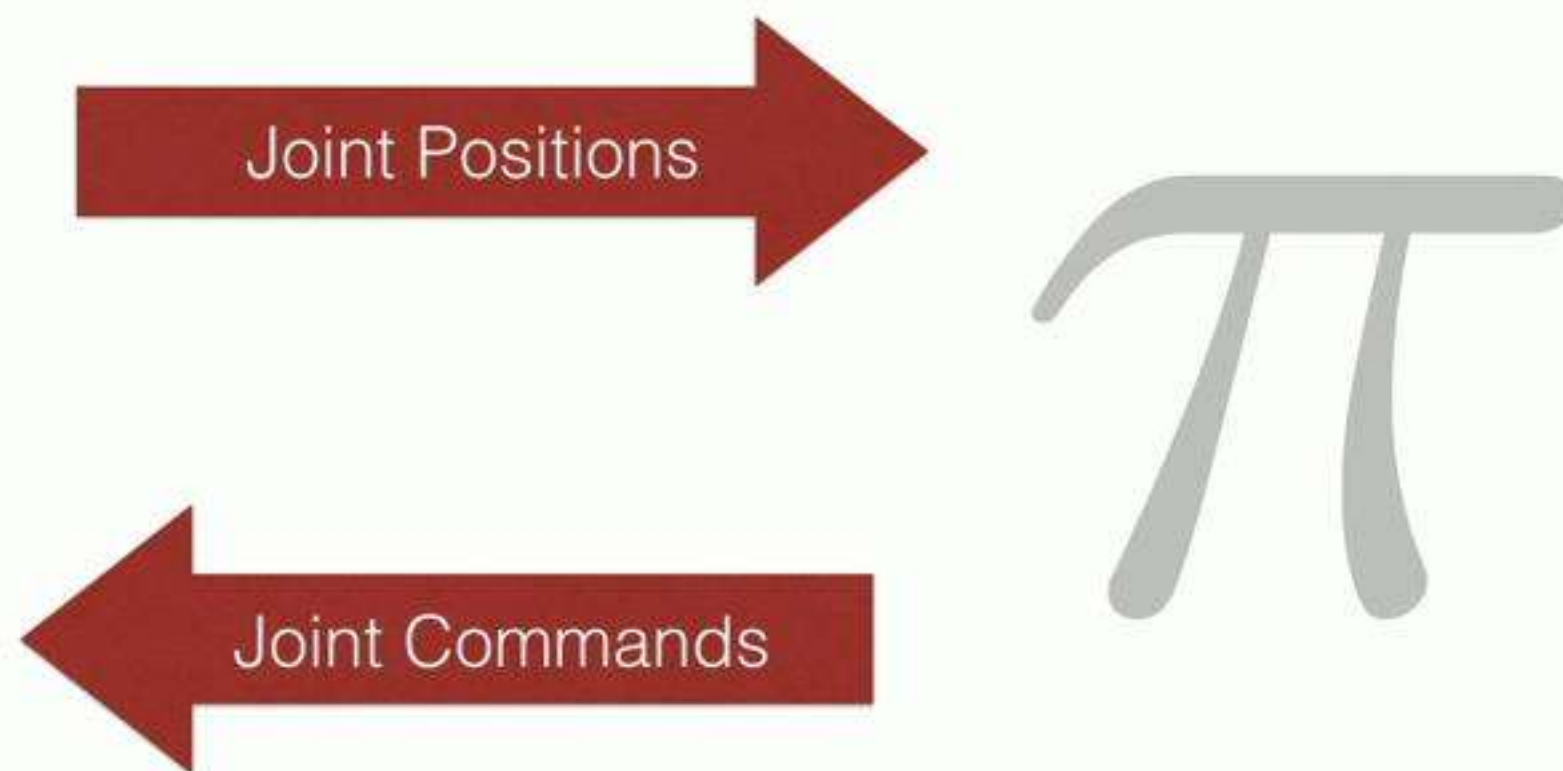


Joint Positions

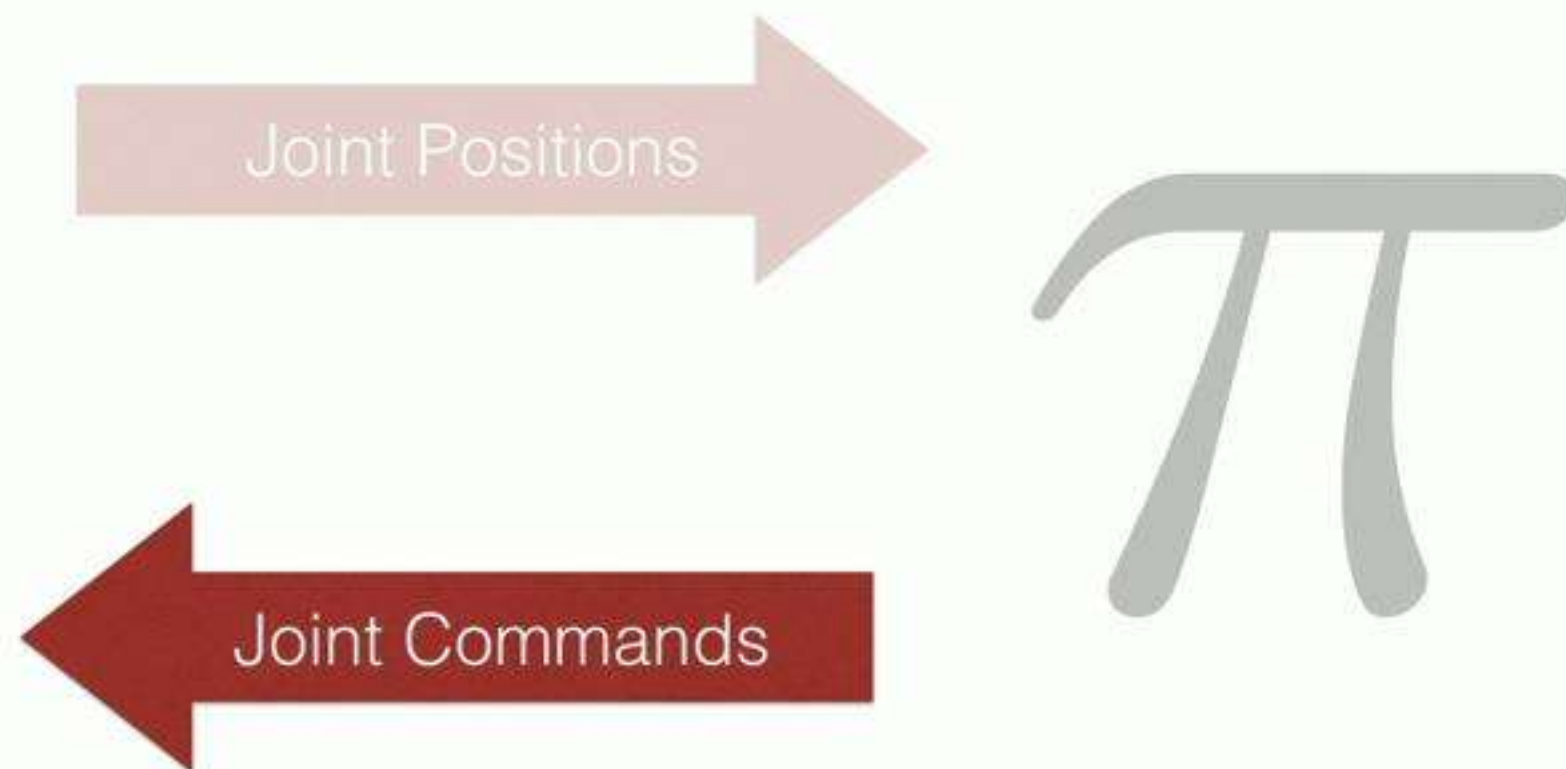
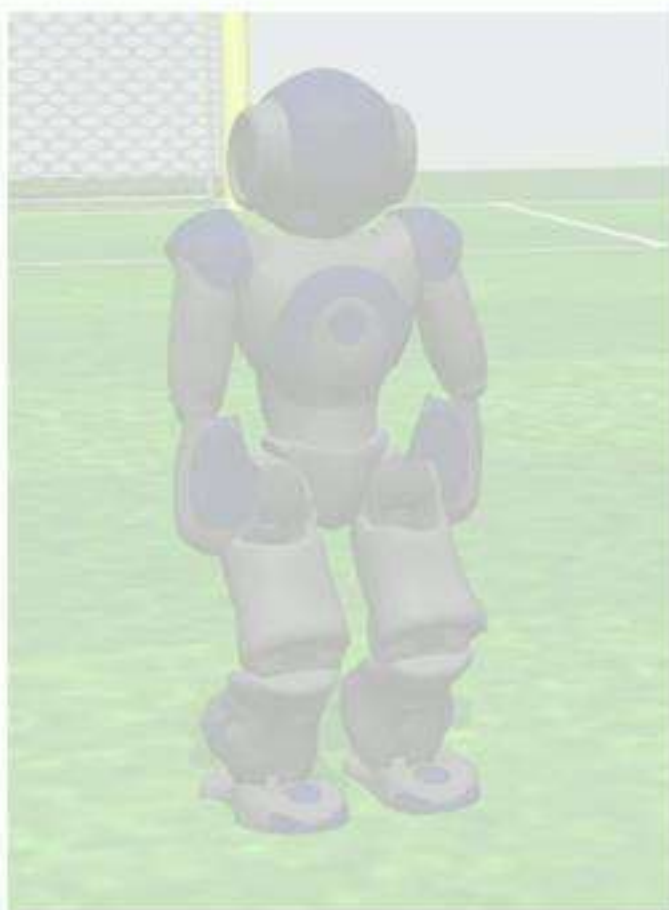
π

Joint Commands

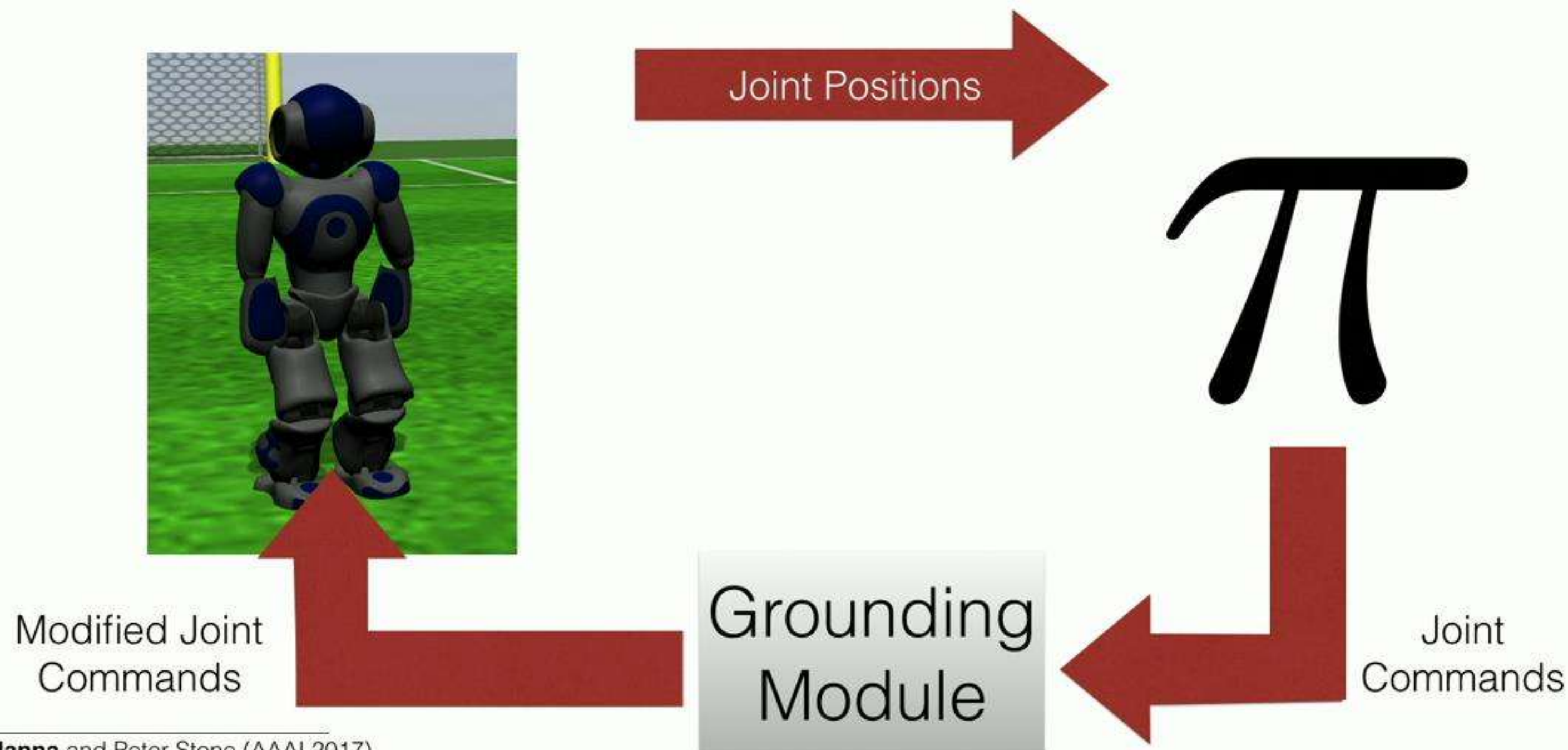
How do we make simulation more realistic?



How do we make simulation more realistic?



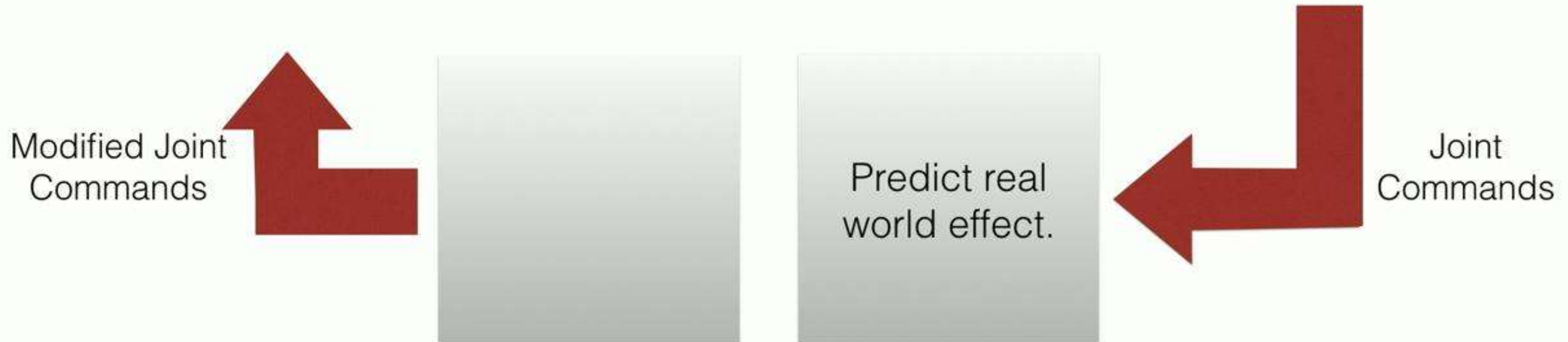
How do we make simulation more realistic?



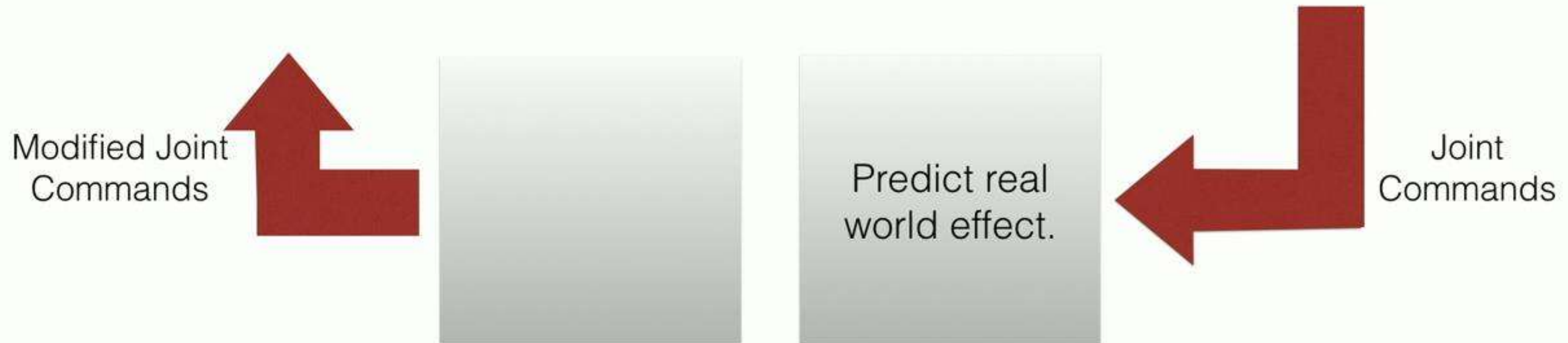
How do we modify the policy's actions?



How do we modify the policy's actions?



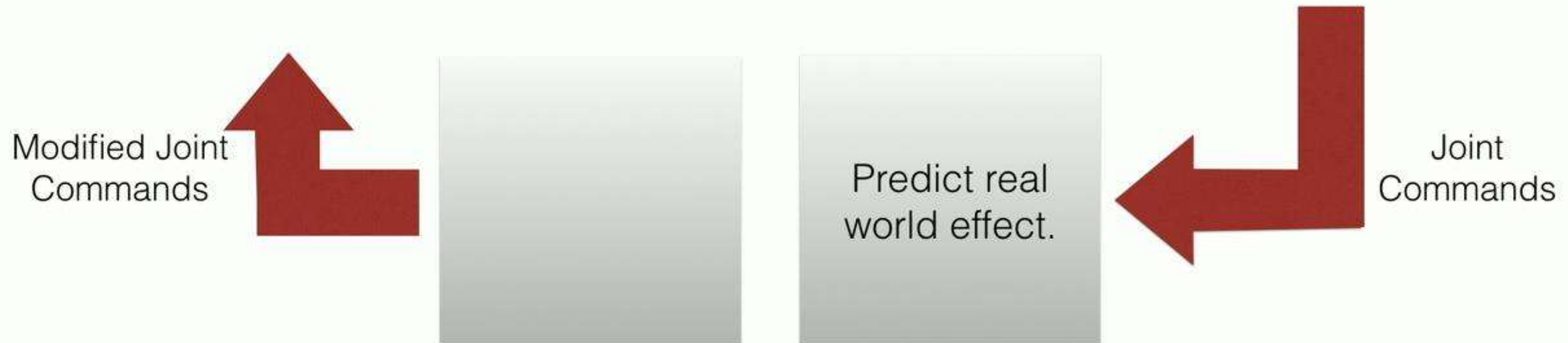
How do we modify the policy's actions?



Real world data:

$$S_0, A_0, R_0, \dots, S_L, A_L, R_L$$

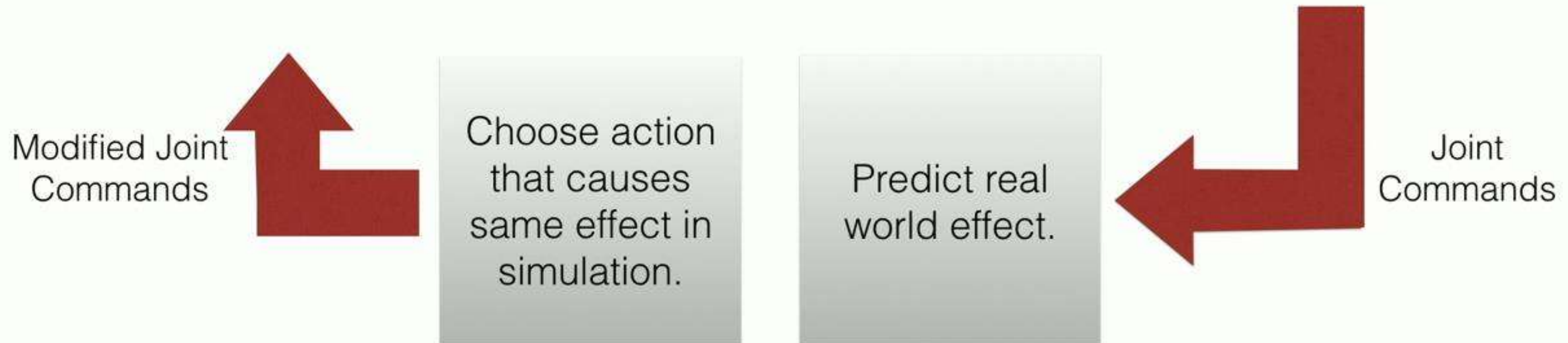
How do we modify the policy's actions?



Real world data:

$$S_0, A_0, R_0, \dots, S_L, A_L, R_L \longrightarrow \{S_t, A_t\} \rightarrow S_{t+1}$$

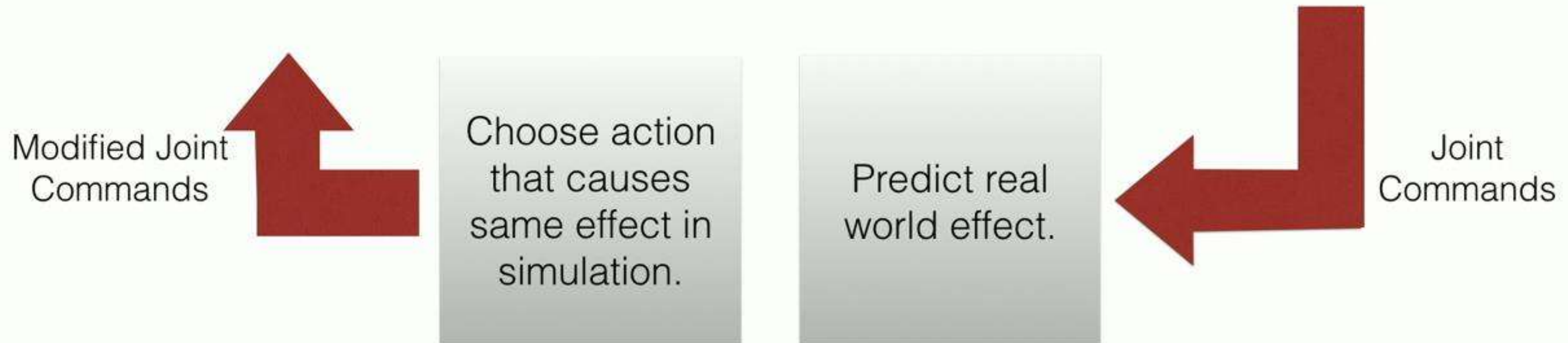
How do we modify the policy's actions?



Real world data:

$$S_0, A_0, R_0, \dots, S_L, A_L, R_L \longrightarrow \{S_t, A_t\} \rightarrow S_{t+1}$$

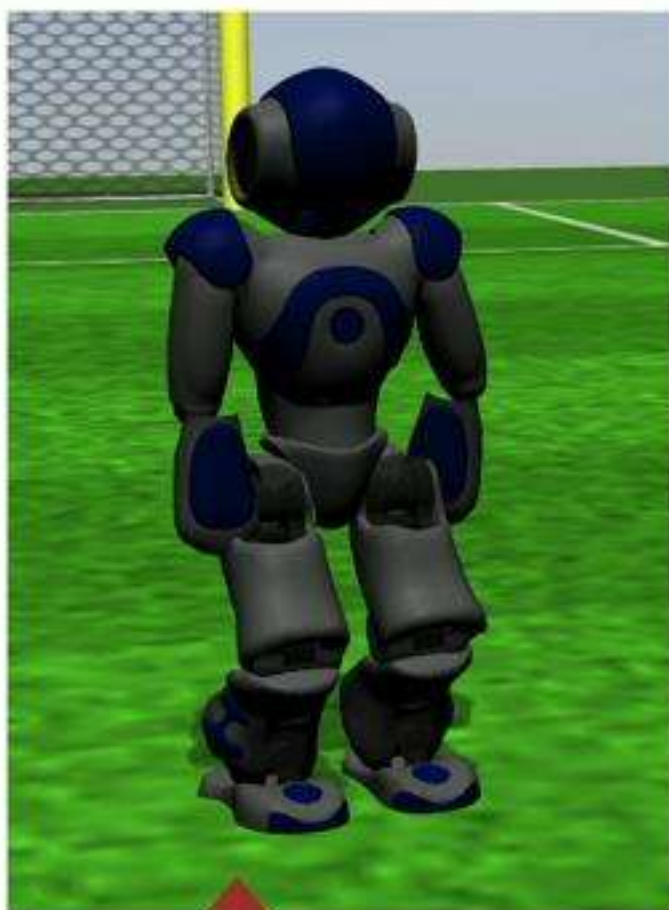
How do we modify the policy's actions?



Simulated data:

$$S_0, A_0, R_0, \dots, S_L, A_L, R_L \longrightarrow \{S_t, S_{t+1}\} \rightarrow A_t$$

How do we make simulation more realistic?



Joint Positions

π

Modified Joint
Commands

Choose action
that causes
same effect in
simulation.

Predict real
world effect.

Joint
Commands

Application to Robot Walking



Application to Robot Walking



5 minutes of walking

Application to Robot Walking



5 minutes of walking



5,000 minutes of walking

Application to Robot Walking

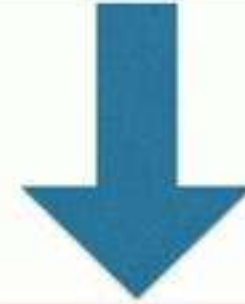


1. Better **action modeling** will lead to faster learning.
2. **Domain simulations** offer a starting point for action modeling.

How can an agent efficiently learn to predict the effects of its actions?



Estimate task performance for a fixed policy.



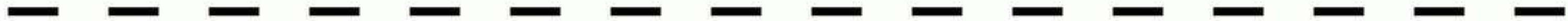
Predicting how actions change the state of the world.

Can reinforcement learning be data efficient enough for real world applications?





Haipeng Chen, Guni Sharon, **Josiah Hanna**, Peter Stone, and Bo An (AAAI 2018)
Josiah Hanna, Guni Sharon, Steven Boyles, and Peter Stone (AAAI 2019)
Guni Sharon, **Josiah Hanna**, Steven Boyles, and Peter Stone (TRB Part C 2017)



Haipeng Chen, Guni Sharon, **Josiah Hanna**, Peter Stone, and Bo An (AAAI 2018)
Josiah Hanna, Guni Sharon, Steven Boyles, and Peter Stone (AAAI 2019)
 Guni Sharon, **Josiah Hanna**, Steven Boyles, and Peter Stone (TRB Part C 2017)

How can an agent efficiently learn to predict the effects of its actions?

Estimate task performance for a fixed policy.

Predicting how actions change the state of the world.

Can reinforcement learning be data efficient enough for real world applications?

How can an agent efficiently learn to predict the effects of its actions?

Estimate task performance for a fixed policy.

Predicting how actions change the state of the world.

State and Action Abstractions

Properties of Prediction Methods

From Prediction to Policy Learning

Can reinforcement learning be data efficient enough for real world applications?

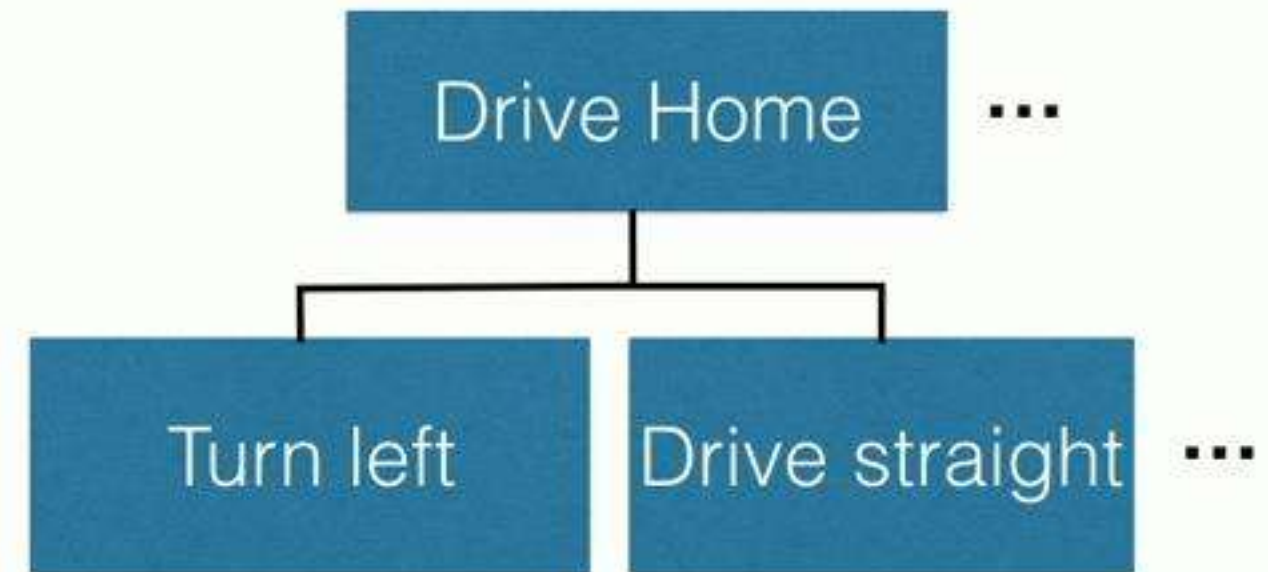
State and Action Abstraction



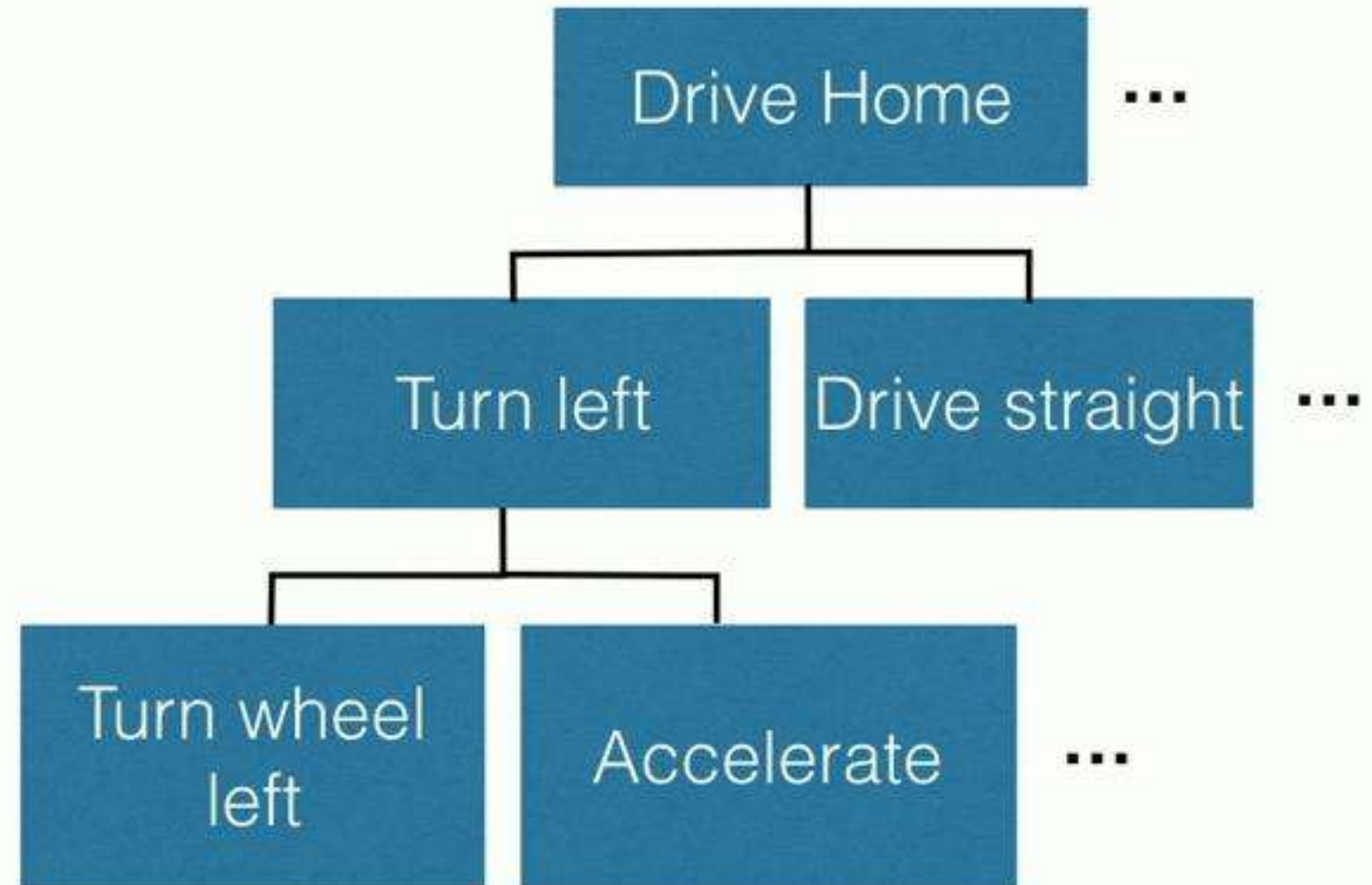
Drive Home

...

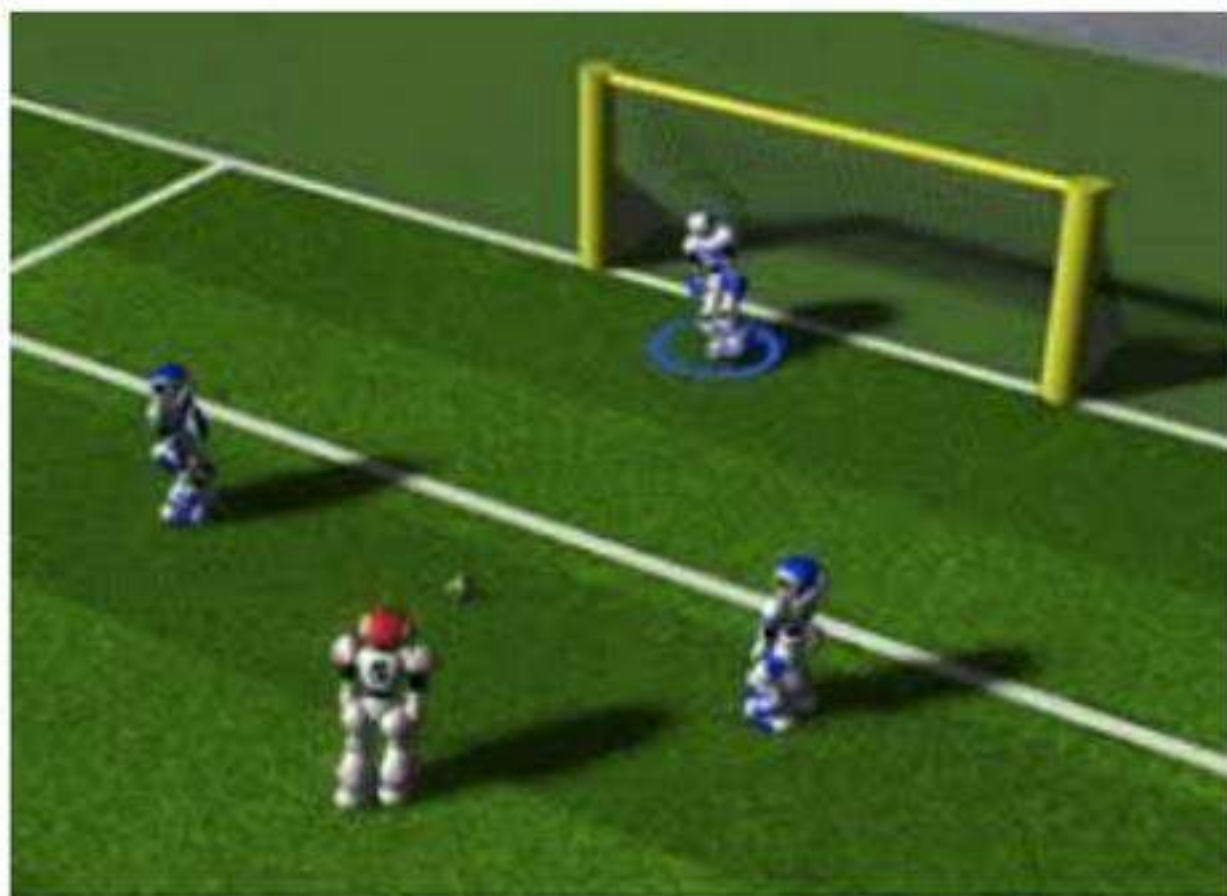
State and Action Abstraction



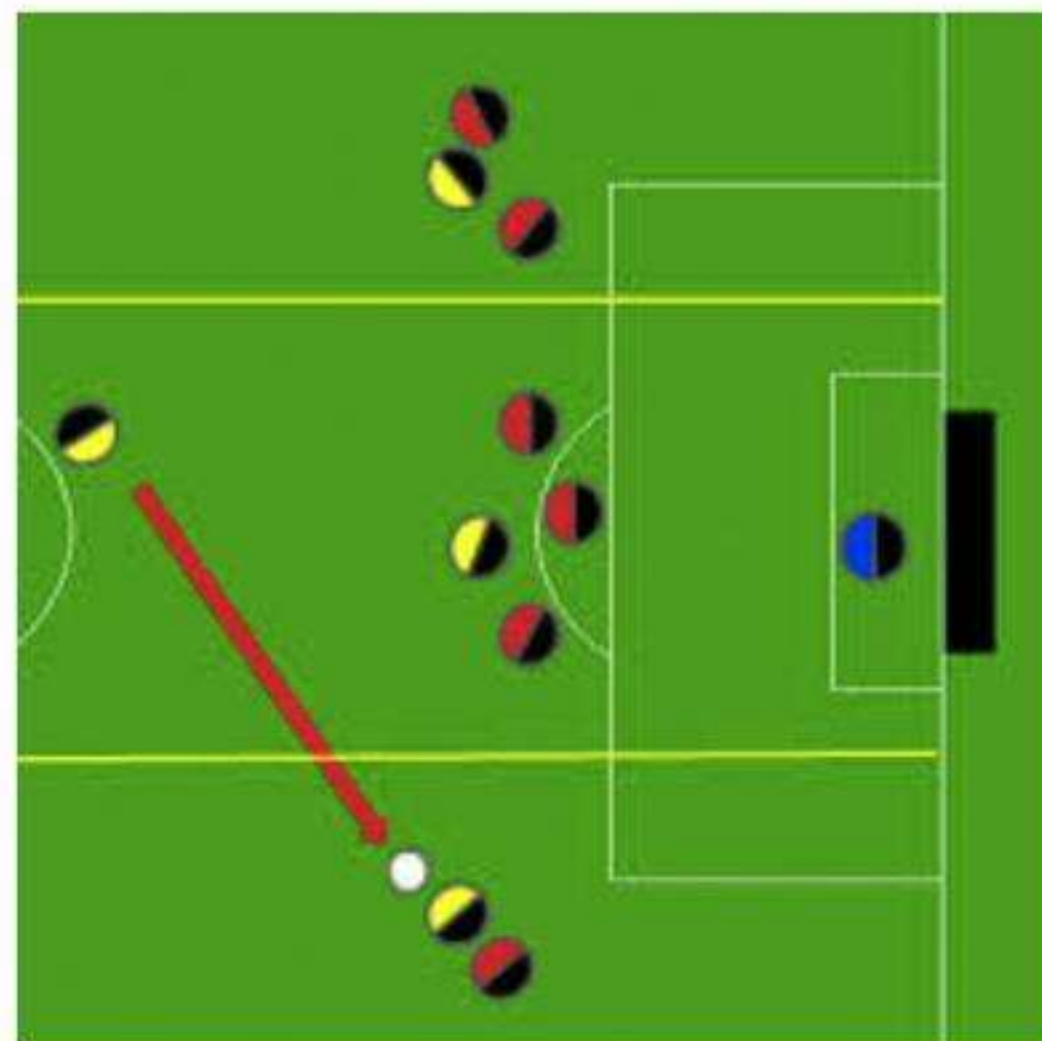
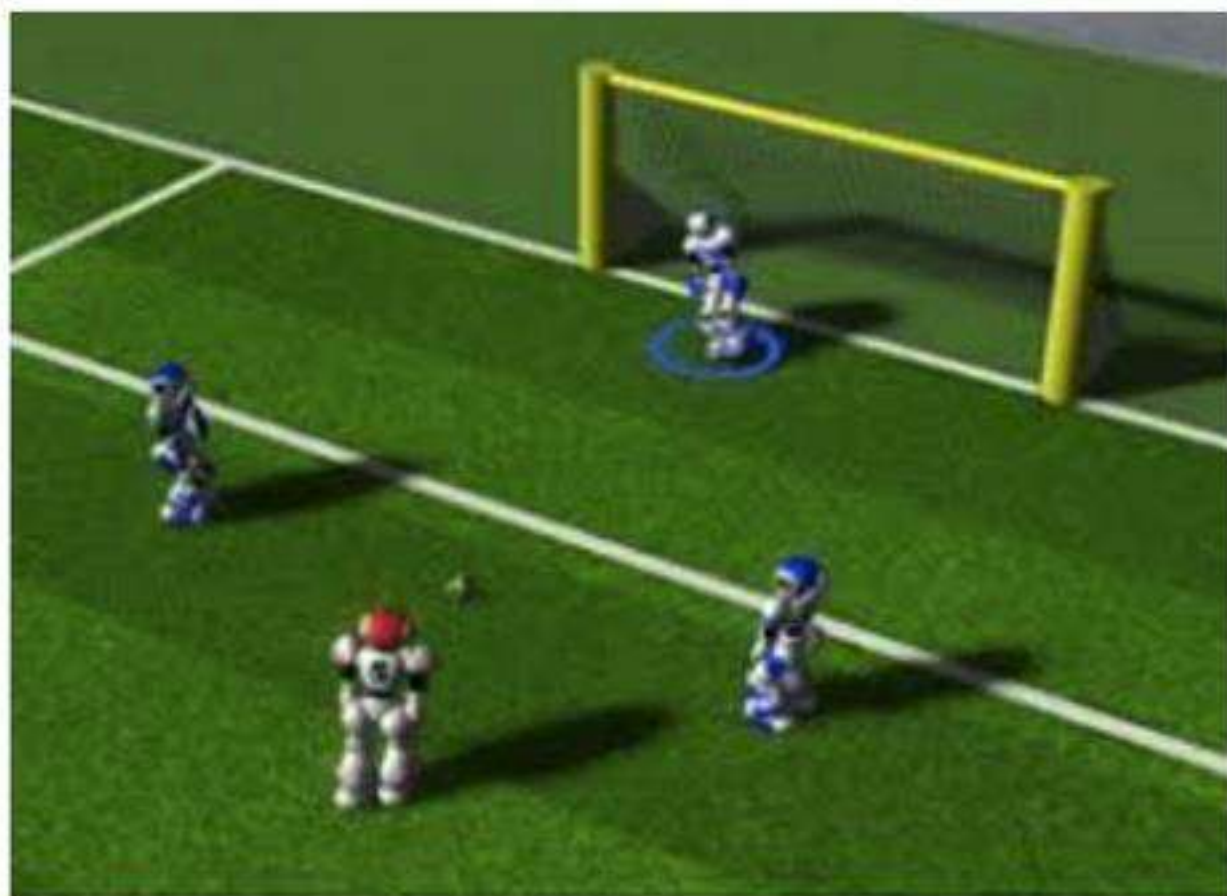
State and Action Abstraction



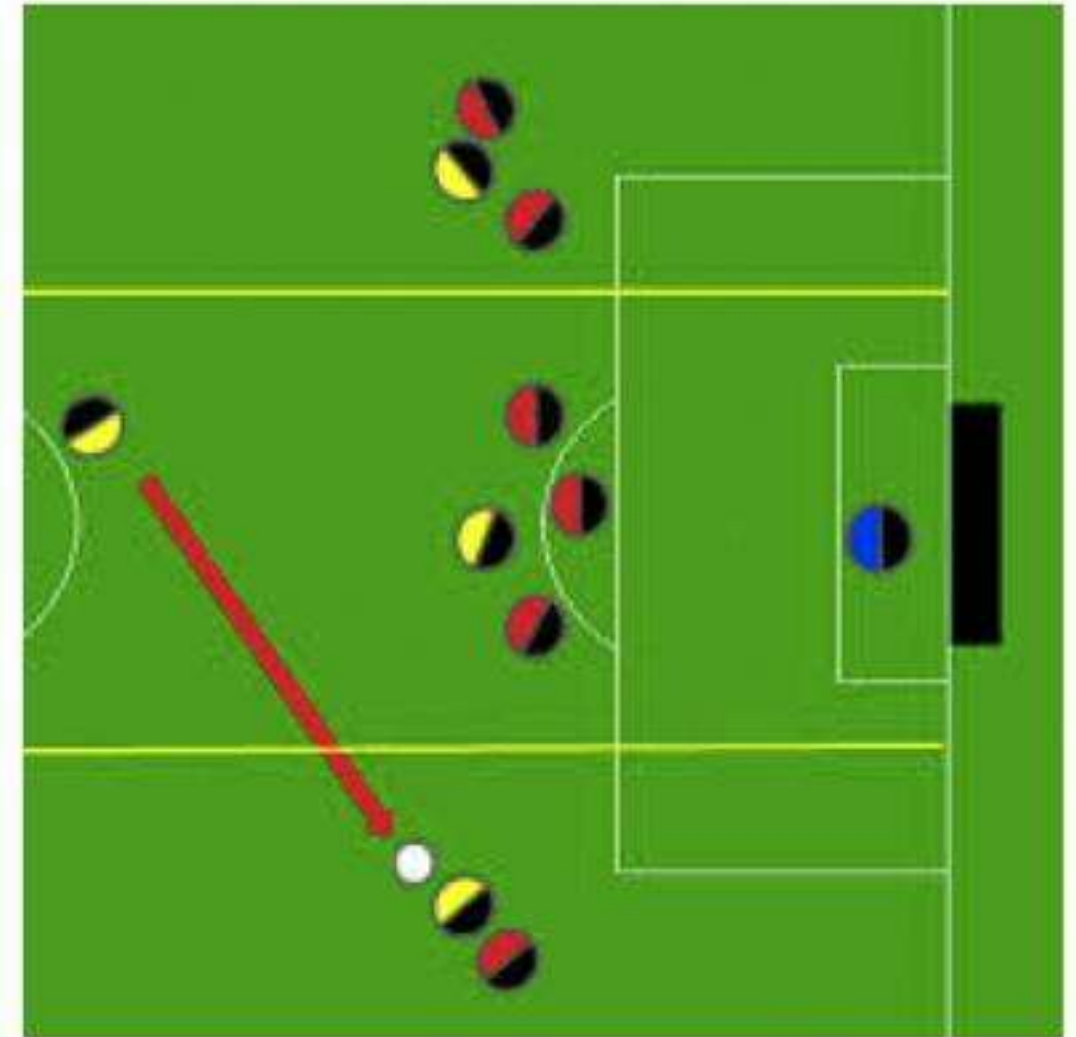
Learning in an abstract simulation



Learning in an abstract simulation



Learning in an abstract simulation



Prediction Guarantees

Prediction Guarantees

Statistical Properties:

1. Confidence?

Prediction Guarantees

Statistical Properties:

1. Confidence?
2. Consistent?



**Testing
Time**

Deployment Time

Prediction Guarantees

Statistical Properties:

1. Confidence?
2. Consistent?
3. Unbiased?



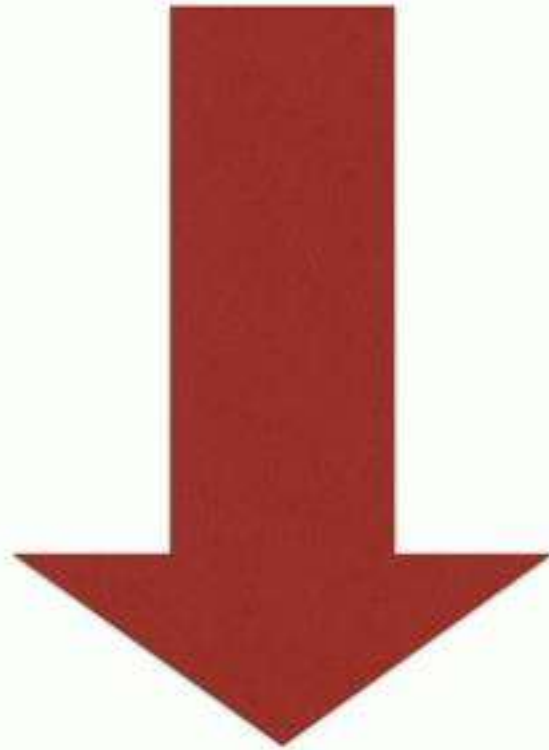
**Testing
Time**

Deployment Time

Practical Properties:

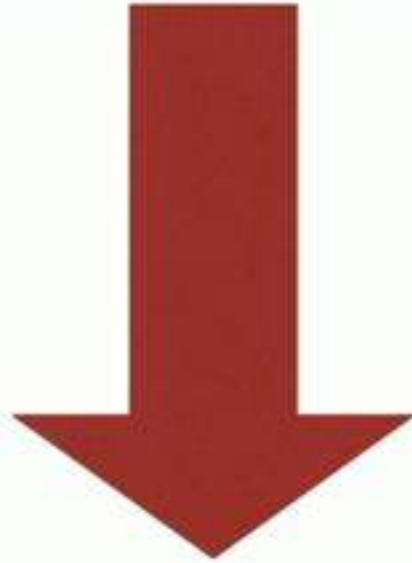
1. Scalable?
2. Explainable?

More efficient algorithms for
predicting effects of actions.



More efficient algorithms for
reinforcement learning.

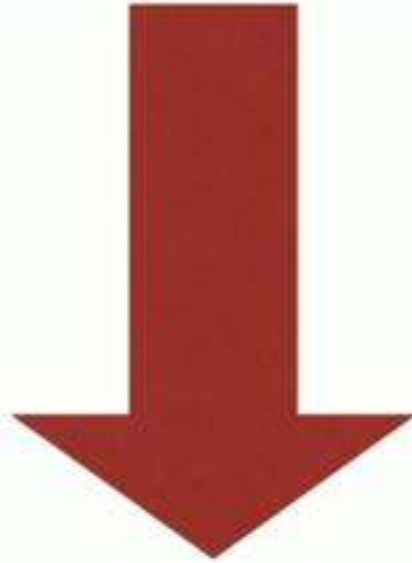
More efficient algorithms for
predicting effects of actions.



More efficient algorithms for
reinforcement learning.

1. Searching the space of possible policies.
2. Transfer evaluation and learning.

More efficient algorithms for predicting effects of actions.



Reinforcement learning can be data efficient enough for real world applications

1. Searching the space of possible policies.
2. Transfer evaluation and learning.

Acknowledgments



Peter Stone



Scott Niekum



Phil Thomas



Guni Sharon



Steve Boyles



Tarun Rambha



Michael Albert



Michael Levin

Josiah Hanna