

Reinforcement Learning

Computer Engineering Department

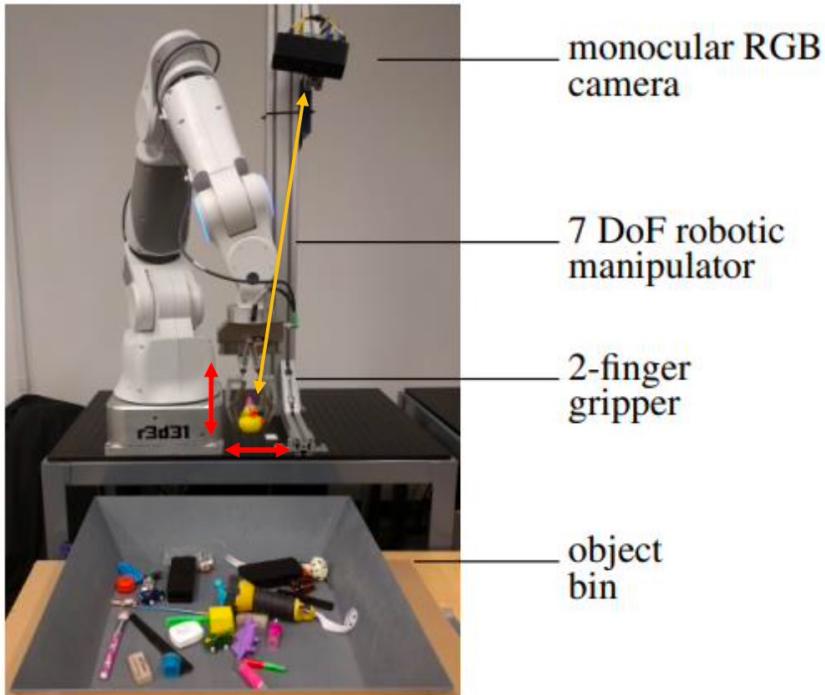
Sharif University of Technology

Mohammad Hossein Rohban, Ph.D.

Spring 2024

Courtesy: Some slides are adopted from CS 285 Berkeley, and CS 234
Stanford, and Pieter Abbeel's compact series on RL.

Motivation



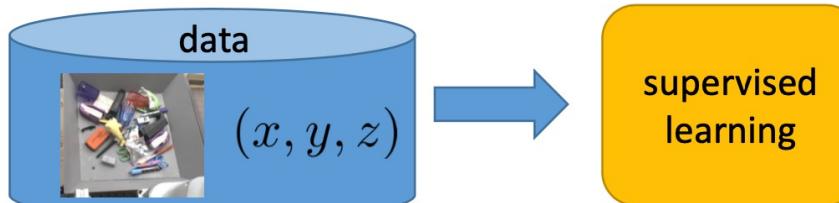
Option 1:

Understand the problem, design a solution



Option 2:

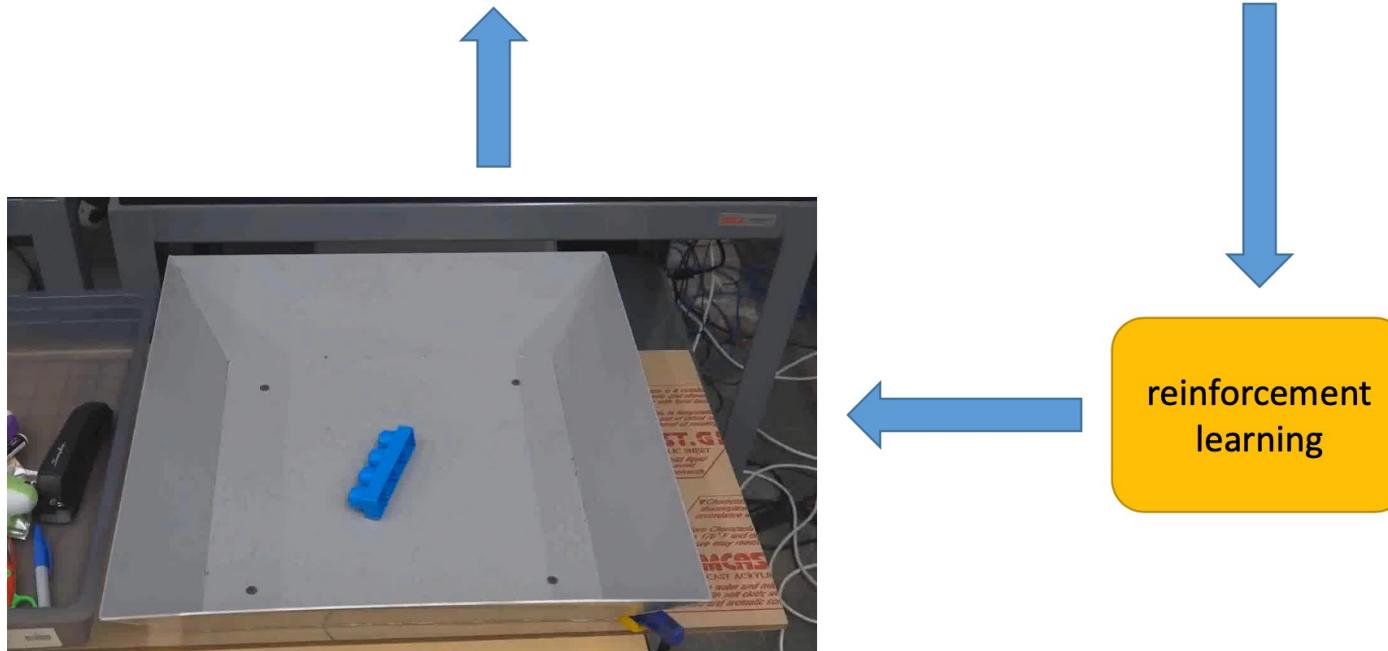
Set it up as a machine learning problem



Motivation (cont.)

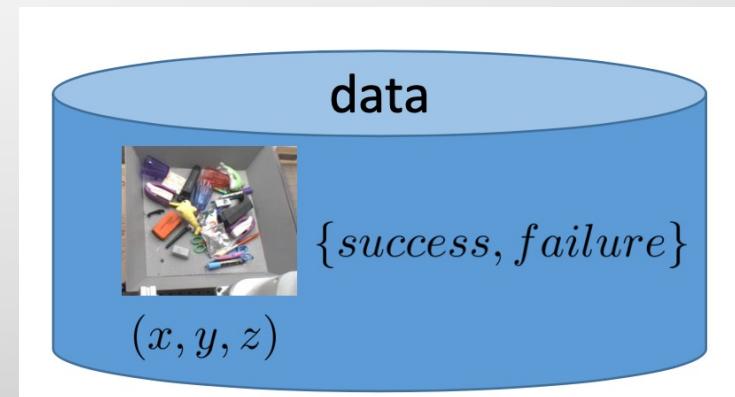


Motivation (cont.)



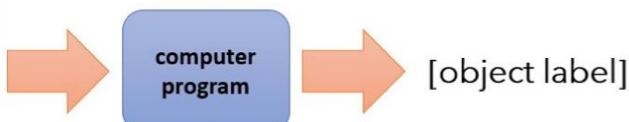
Motivation (cont.)

- Supervised learning:
 - Ground truth is **known** in advance.
 - Training data are usually **static** and **iid**.
- Reinforcement learning:
 - The best action (**policy**) is usually **unknown a priori**.
 - **Sequence of actions** is needed.
 - A series of trial and error (**search**) is performed.
 - Usually **delayed reward** shows goodness of the trial.
 - Data is **dynamic (exploration)** and **non-iid**.



What is Reinforcement Learning?

supervised learning



input: \mathbf{x}

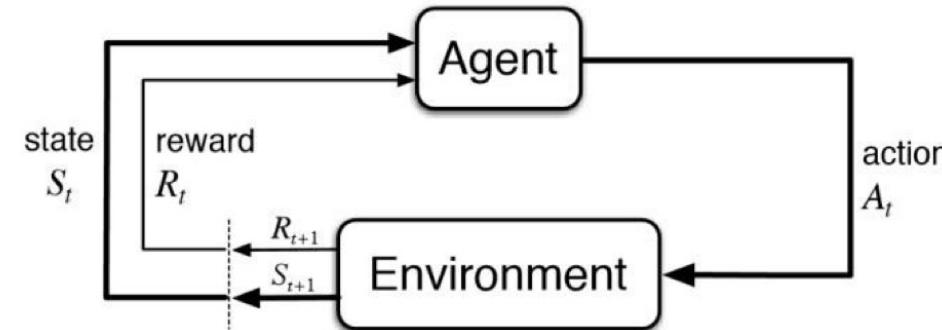
output: \mathbf{y}

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

goal: $f_\theta(\mathbf{x}_i) \approx \mathbf{y}_i$

someone gives
this to you

reinforcement learning



pick your
own actions

input: \mathbf{s}_t at each time step

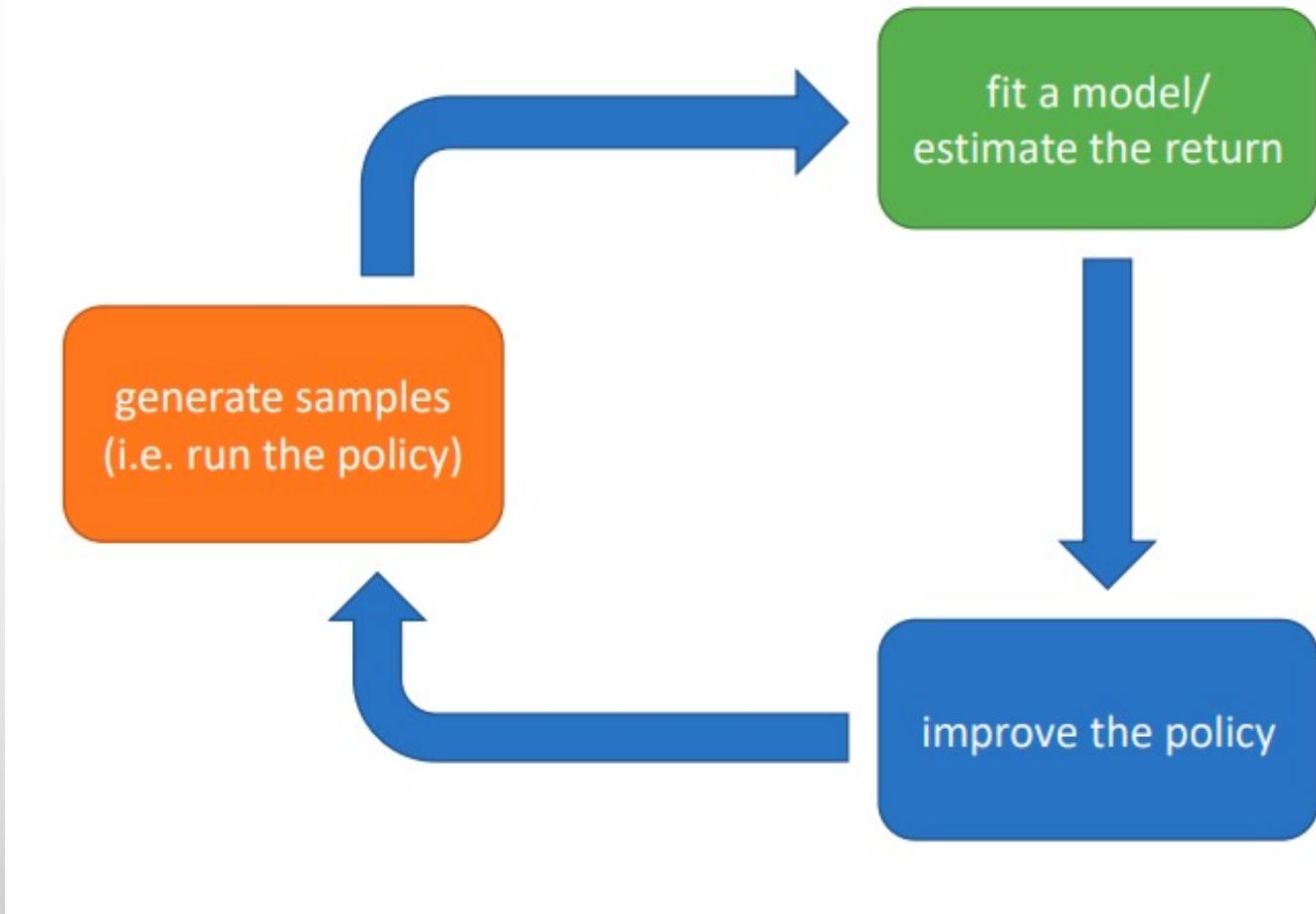
output: \mathbf{a}_t at each time step

data: $(\mathbf{s}_1, \mathbf{a}_1, r_1, \dots, \mathbf{s}_T, \mathbf{a}_T, r_T)$

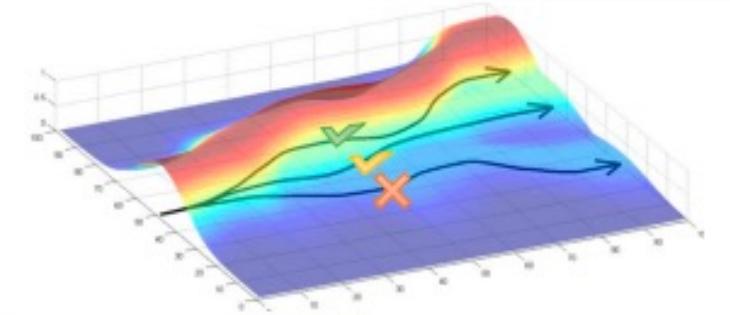
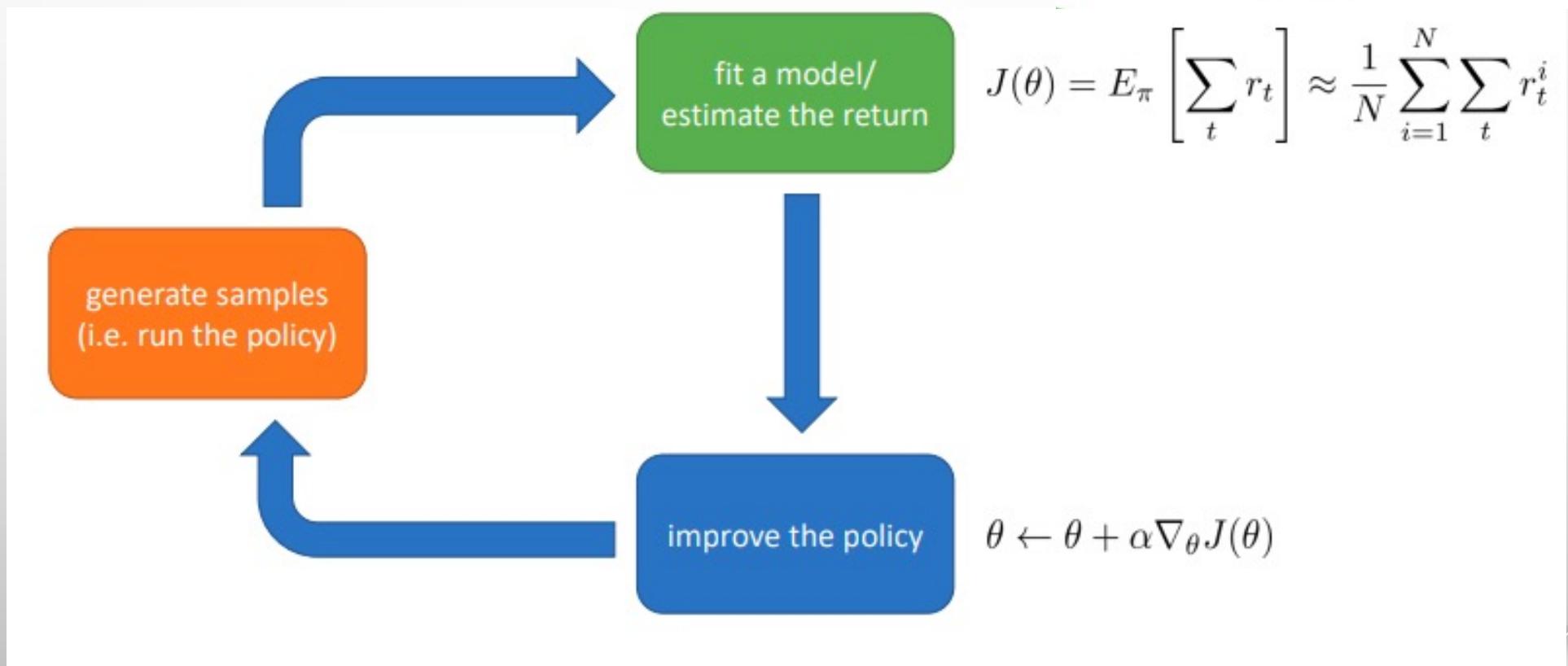
goal: learn $\pi_\theta : \mathbf{s}_t \rightarrow \mathbf{a}_t$

to maximize $\sum_t r_t$

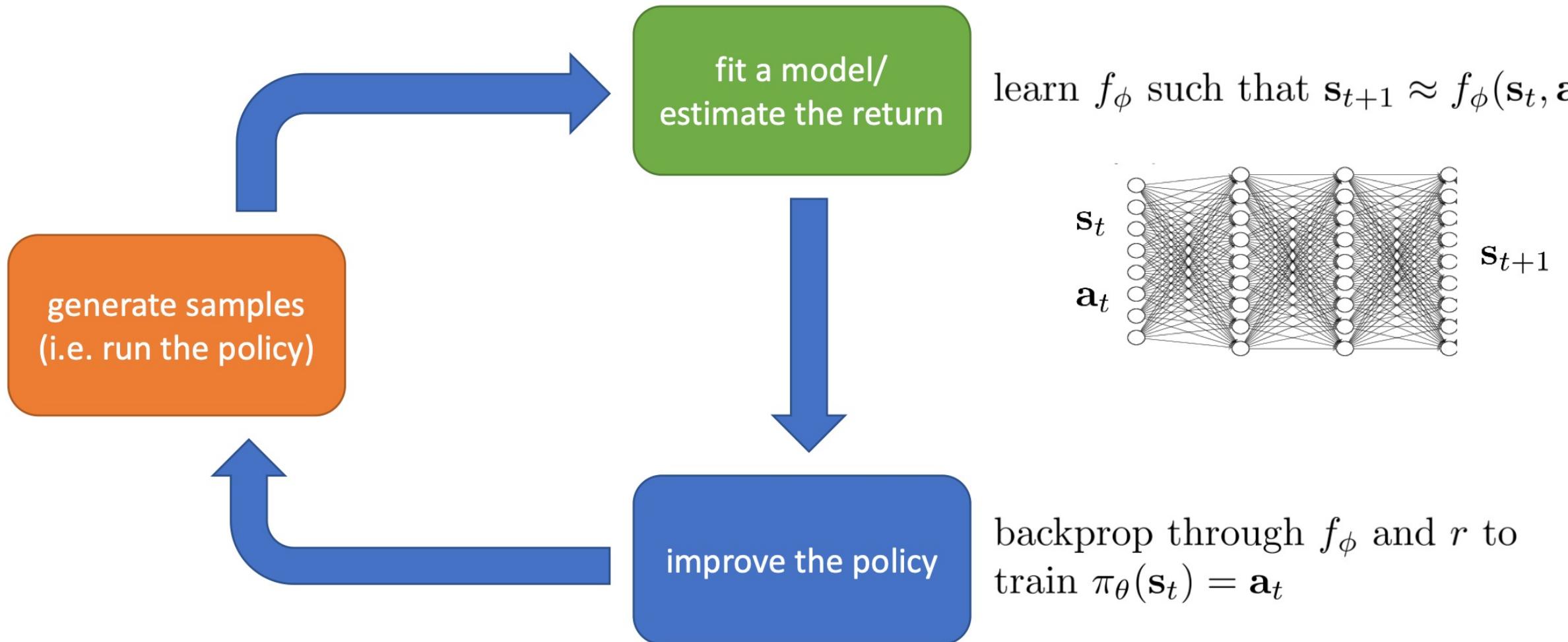
The Anatomy of Reinforcement Learning



A Simple Example



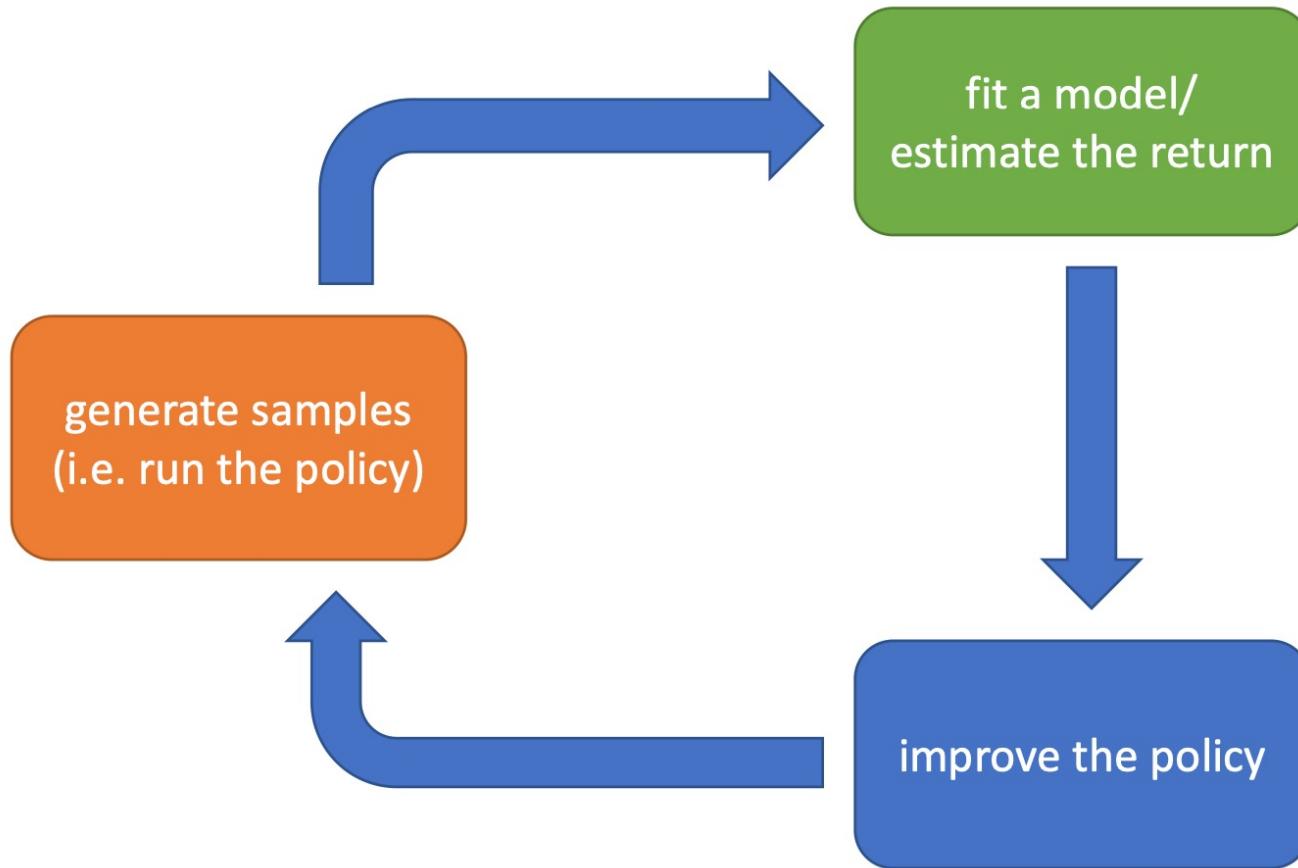
Another Example



Which parts are expensive?

real robot/car/power grid/whatever:
1x real time, until we invent time travel

MuJoCo simulator:
up to 10000x real time



$$J(\theta) = E_{\pi} \left[\sum_t r_t \right] \approx \frac{1}{N} \sum_{i=1}^N \sum_t r_t^i$$

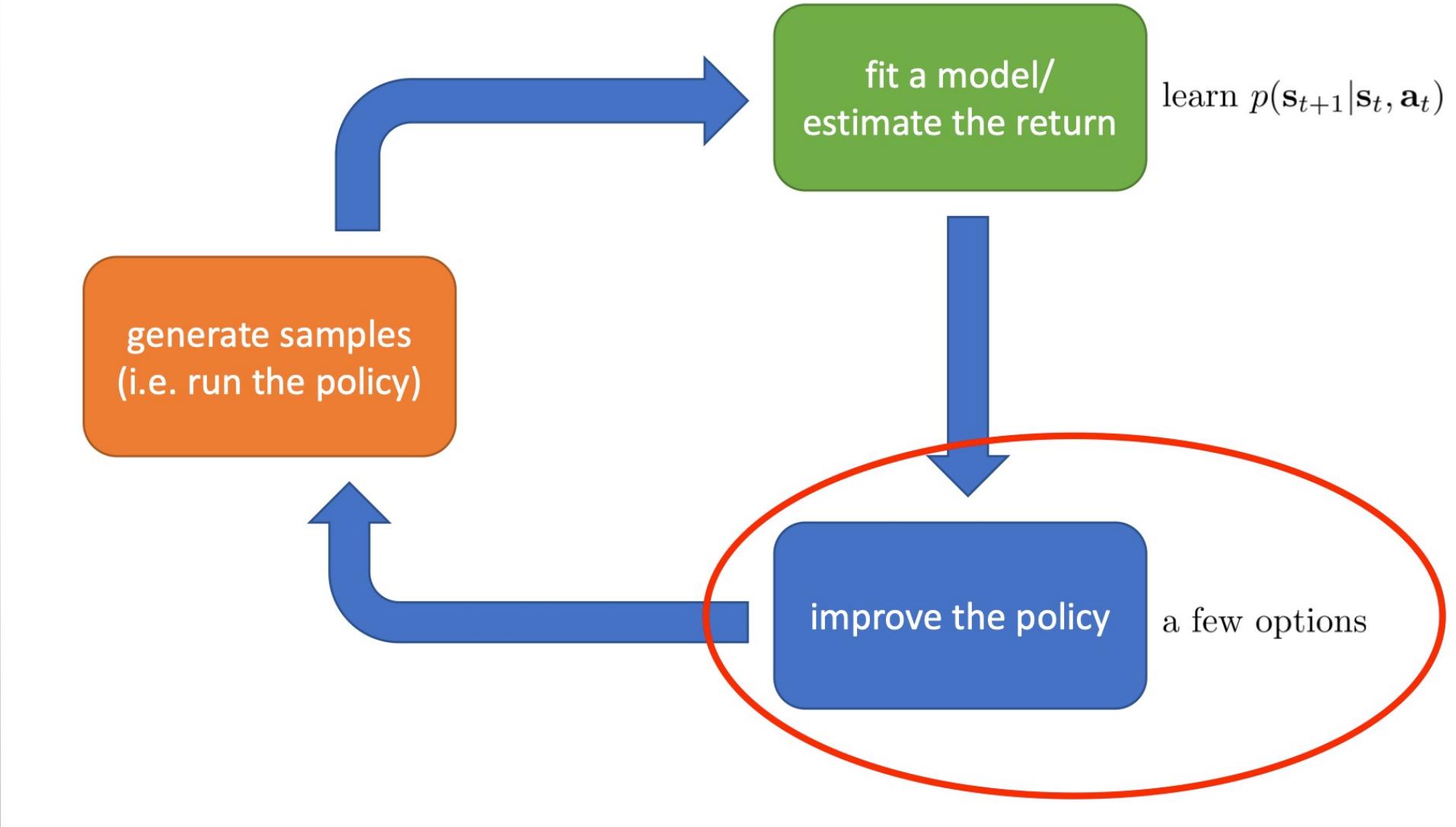
trivial, fast

learn $\mathbf{s}_{t+1} \approx f_{\phi}(\mathbf{s}_t, \mathbf{a}_t)$
expensive

$$\theta \leftarrow \theta + \alpha \nabla_{\theta} J(\theta)$$

backprop through f_{ϕ} and r to
train $\pi_{\theta}(\mathbf{s}_t) = \mathbf{a}_t$

Model-based RL



Value-based RL

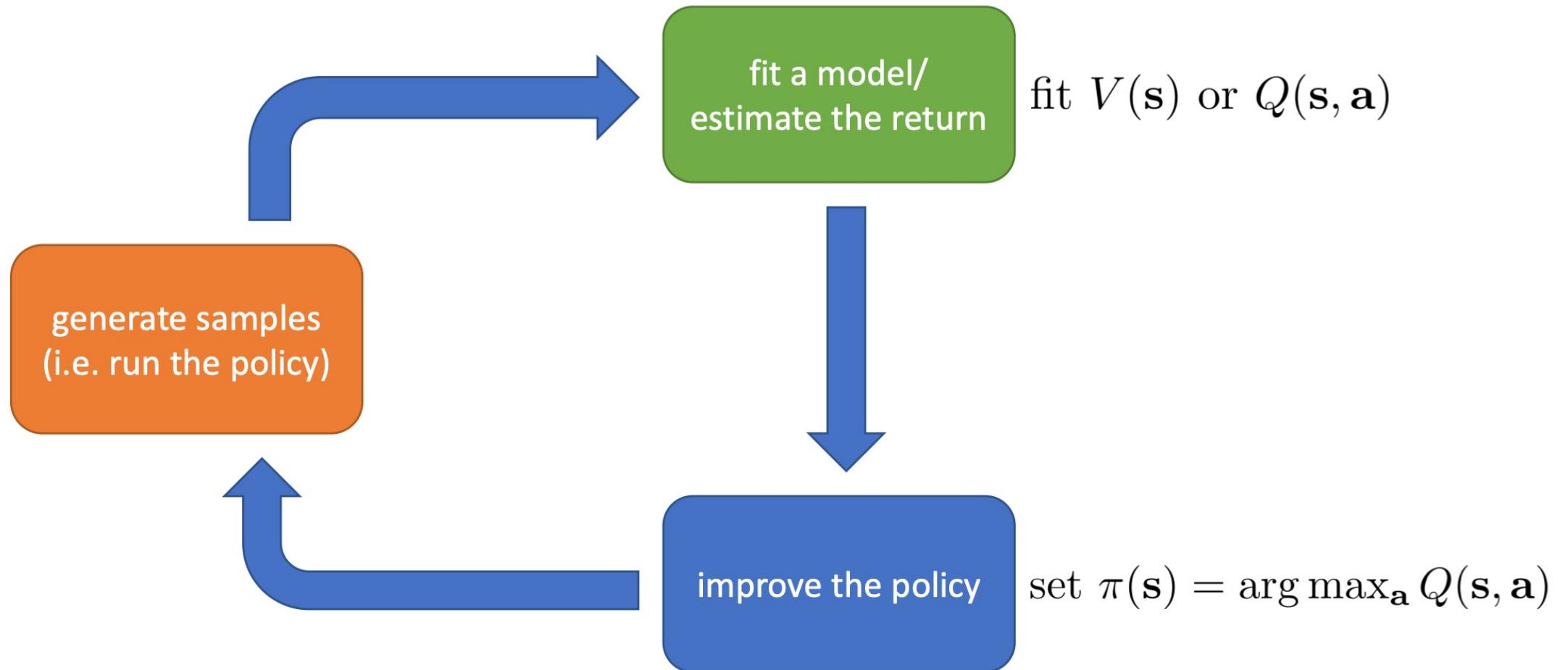
$Q^\pi(\mathbf{s}_t, \mathbf{a}_t) = \sum_{t'=t}^T E_{\pi_\theta} [r(\mathbf{s}_{t'}, \mathbf{a}_{t'}) | \mathbf{s}_t, \mathbf{a}_t]$: total reward from taking \mathbf{a}_t in \mathbf{s}_t

$V^\pi(\mathbf{s}_t) = \sum_{t'=t}^T E_{\pi_\theta} [r(\mathbf{s}_{t'}, \mathbf{a}_{t'}) | \mathbf{s}_t]$: total reward from \mathbf{s}_t

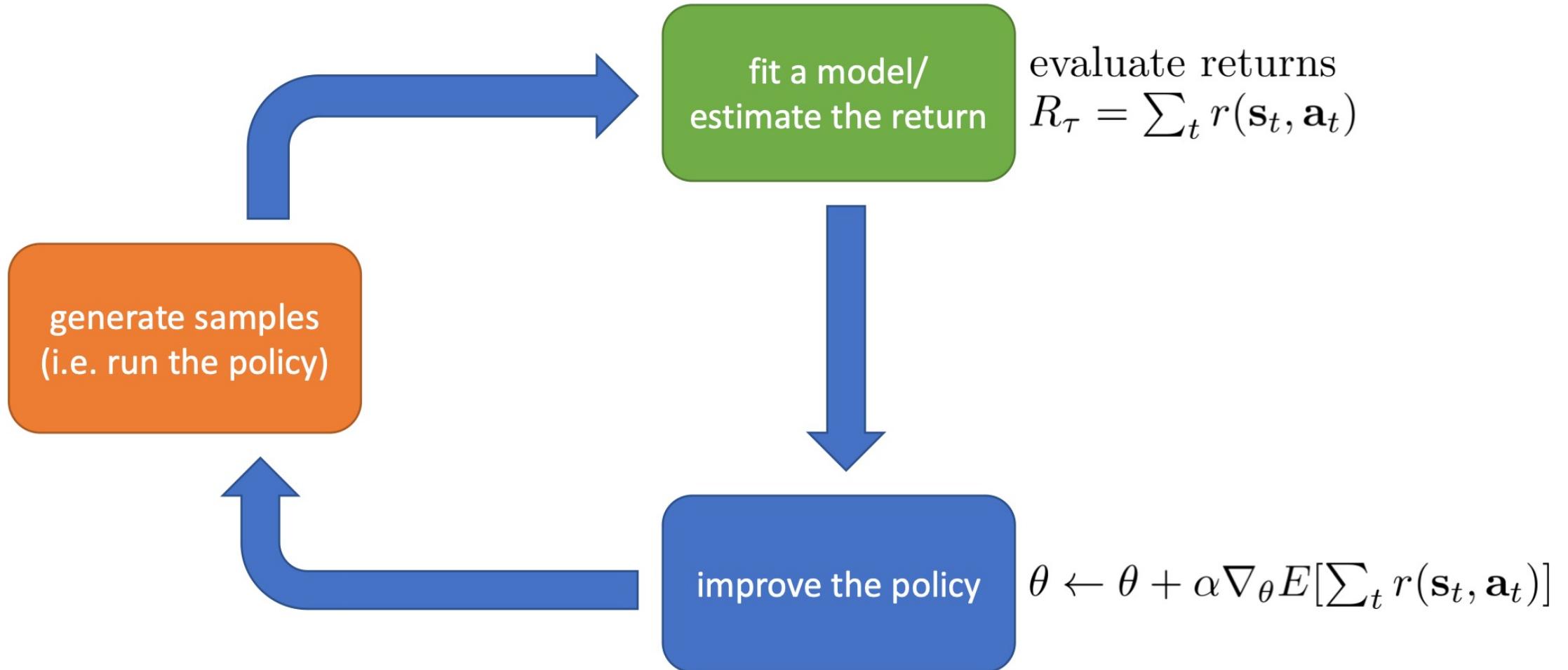
$V^\pi(\mathbf{s}_t) = E_{\mathbf{a}_t \sim \pi(\mathbf{a}_t | \mathbf{s}_t)} [Q^\pi(\mathbf{s}_t, \mathbf{a}_t)]$

$E_{\mathbf{s}_1 \sim p(\mathbf{s}_1)} [V^\pi(\mathbf{s}_1)]$ is the RL objective!

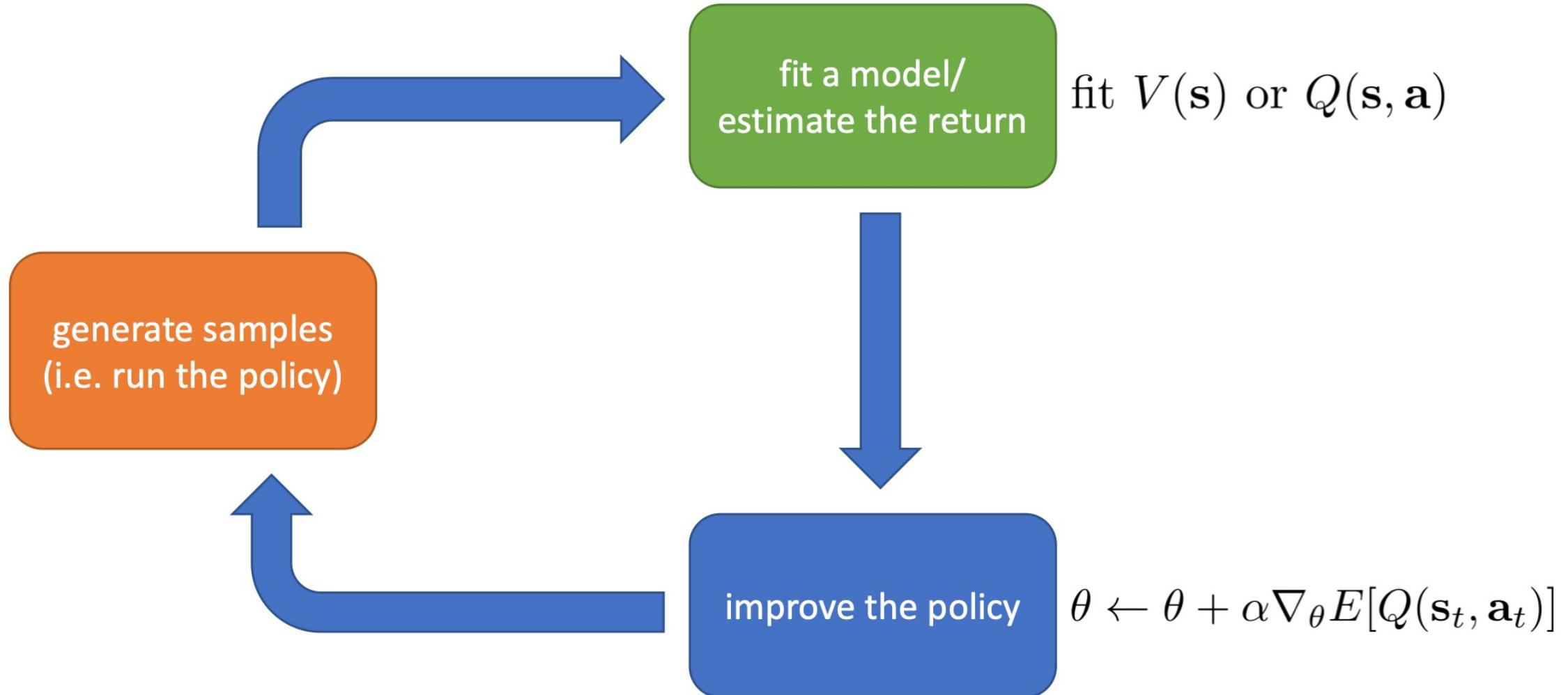
Value-based RL (cont.)



Direct Policy Gradient



Actor-critic: value functions + policy gradients



Where do rewards come from?

- Learning from **demonstrations**
 - Directly **copying** observed behavior
 - **Inferring rewards** from observed behavior (inverse reinforcement learning)
- Learning from observing the world
 - Learning to **predict**
 - **Unsupervised** learning
- Learning from other tasks
 - **Transfer learning**
 - **Meta-learning**: learning to learn



Motivation (cont.)

	AI Planning	SL	UL	RL	IL
Optimization	X			X	X
Learns from experience		X	X	X	X
Generalization	X	X	X	X	X
Delayed Consequences	X			X	X
Exploration				X	

- SL = supervised learning; UL = unsupervised learning; RL = reinforcement learning; IL = imitation learning
- Imitation learning typically assumes input demonstrations of good policies
- IL reduces RL to SL. IL + RL is promising area

Planning vs learning

- Two fundamental problems in sequential decision making
 - Reinforcement learning:
 - The environment is initially **unknown**
 - The agent **interacts** with the environment
 - The agent **improves** its policy
 - Planning:
 - A model of the environment is **known**
 - The agent performs computations with its model (**without any external interaction**)
 - The agent **improves** its policy
 - a.k.a. deliberation, reasoning, introspection, pondering, thought, search

Why should we study deep reinforcement learning?

Impressive because no person had thought of it!



“Move 37” in Lee Sedol AlphaGo match: reinforcement learning “discovers” a move that surprises everyone

Impressive because it looks like something a person might draw!



Data-driven AI vs. RL

Data-Driven AI



All about using data

Explaining a joke

Prompt

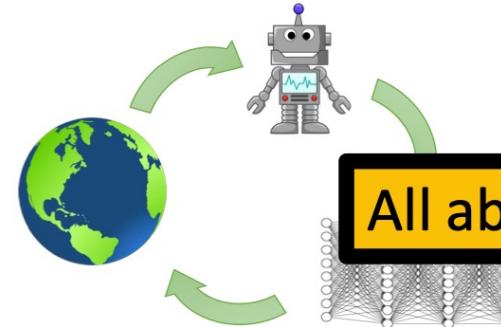
Explain this joke:

Joke: Did you see that Google just hired an eloquent whale for their TPU team? It showed them how to communicate between two



- + learns about the real world from data
- doesn't try to do **better** than the data

Reinforcement Learning



All about optimization



- + optimizes a goal with emergent behavior
- but need to figure out how to use at scale!

Data without optimization
doesn't allow us to solve new
problems in new ways



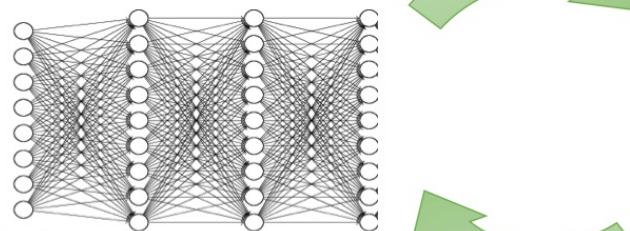
A Bitter Lesson (Richard Sutton)

“We have to learn the bitter lesson that **building in how we think we think** does not work in the long run. The two methods that seem to scale arbitrarily ... are **learning** and **search**

<http://www.incompleteideas.net/Incldeas/BitterLesson.html>

Learning

use **data** to extract **patterns**



allows us to **understand** the world

Search

use **computation** to extract **inferences**

optimization

some optimization process that uses (typically iterative) computation to make rational decisions

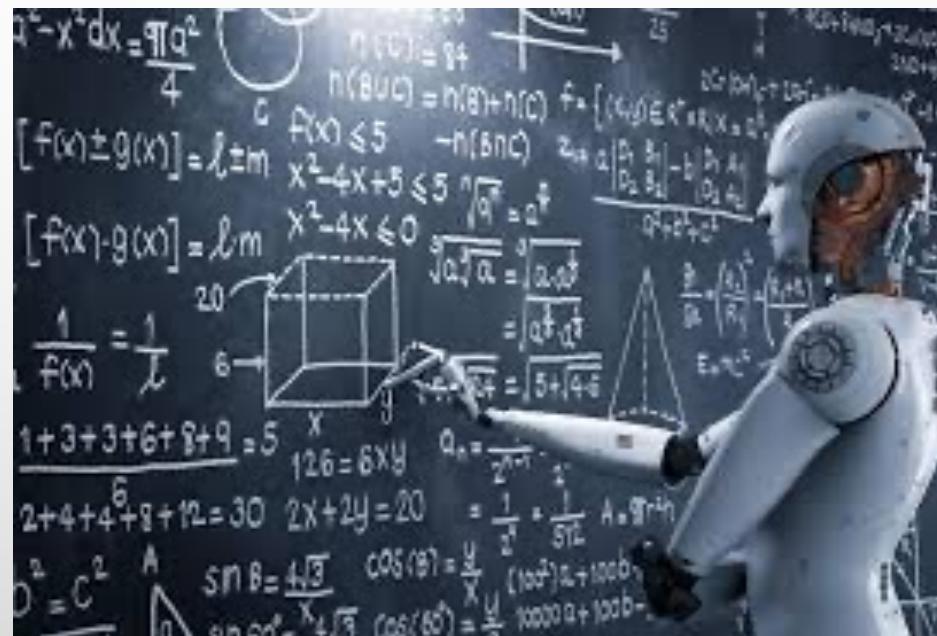
leverages that **understanding** for **emergence**

Data without **optimization** doesn't allow us to solve new problems in new ways

Optimization without **data** is hard to apply to the real world outside of simulators

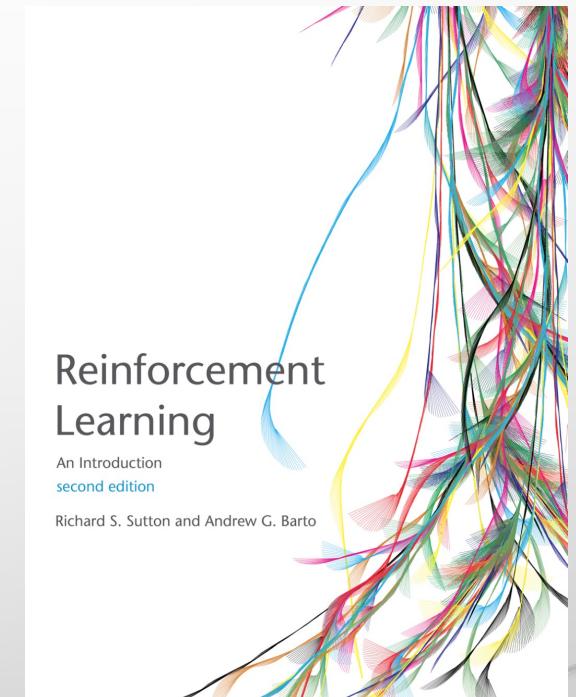
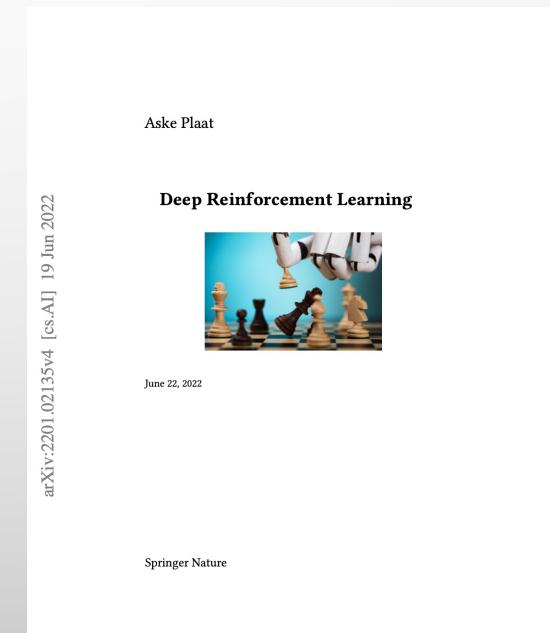
Superintelligence

- The models are trained based on **human annotations** and **preferences**.
- Can they get **smarter** than humans?



References

- Reinforcement Learning: An Introduction by R. Sutton and A. Barto,
2nd Edition, 2020.
- Deep Reinforcement Learning by A. Plaat, 2022.
- Original papers of some methods.



Teaching Assistants

- Hossein Jafarinia (Head TA)
- Mahdi Ghaznavi
- For undergrads to take the course, please fill out:

<https://forms.gle/Hp2c4Fdex1WHKKHLA>

Prereqs.

- Stochastic Processes (Prob. And Stats, Markov Processes, Estimation Theory, Information Theory)
- Optimization (Lagrange Multipliers)
- Deep Learning (Concepts and Pytorch)

Motivation (cont.) ChatGPT; Why RL?!

Step 1

Collect demonstration data
and train a supervised policy.

A prompt is sampled from our prompt dataset.

Explain reinforcement learning to a 6 year old.

A labeler demonstrates the desired output behavior.

We give treats and punishments to teach...

This data is used to fine-tune GPT-3.5 with supervised learning.

SFT



Step 2

Collect comparison data and train a reward model.

A prompt and several model outputs are sampled.

Explain reinforcement learning to a 6 year old.

A
In reinforcement learning, the agent is...
B
Explain rewards...
C
In machine learning...
D
We give treats and punishments to teach...

A labeler ranks the outputs from best to worst.

D > C > A > B

This data is used to train our reward model.

RM



Step 3

Optimize a policy against the reward model using the PPO reinforcement learning algorithm.

A new prompt is sampled from the dataset.

Write a story about otters.

The PPO model is initialized from the supervised policy.

PPO


The policy generates an output.

Once upon a time...

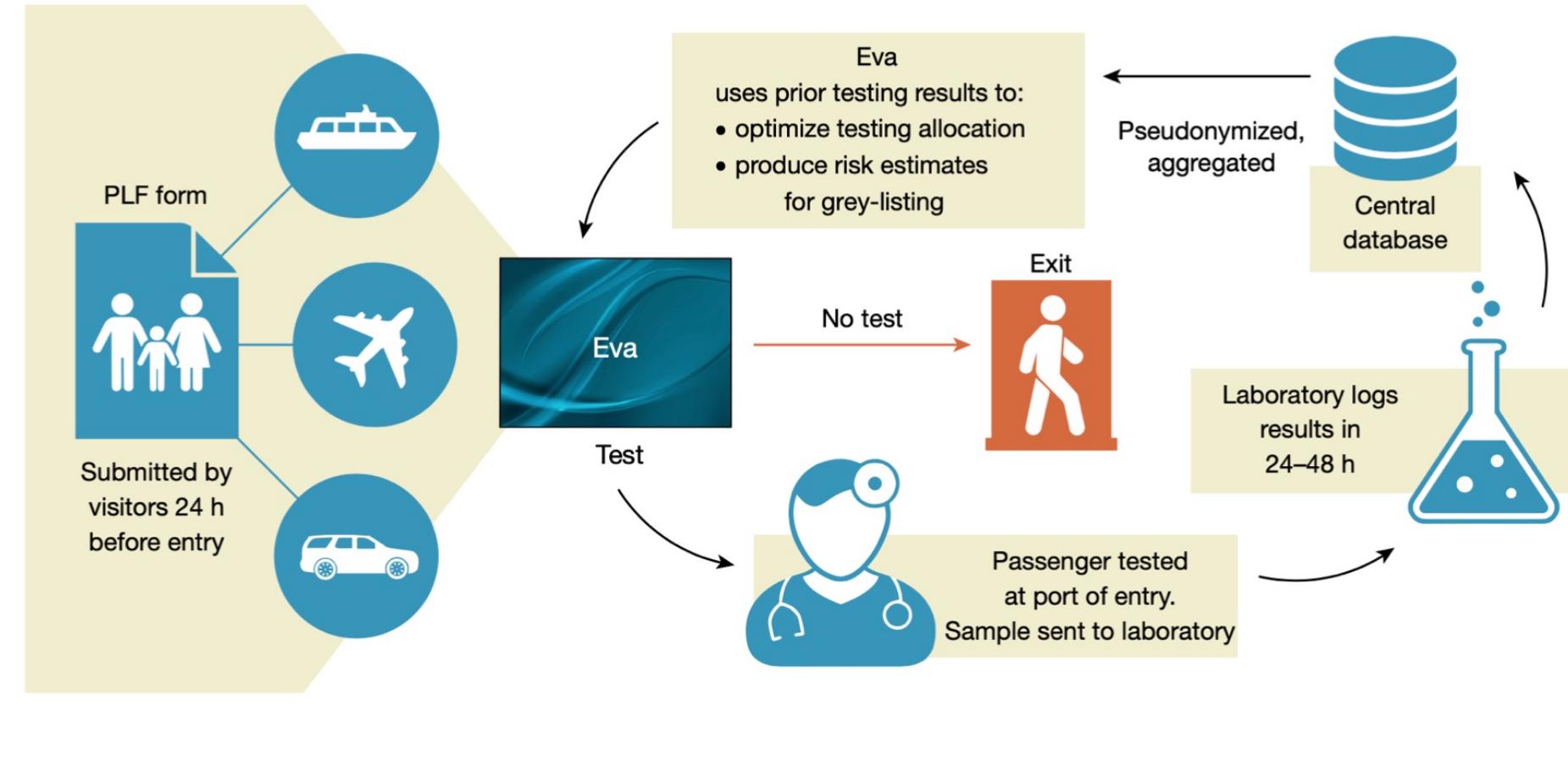
The reward model calculates a reward for the output.

RM


The reward is used to update the policy using PPO.

r_k

Motivation (cont.)



- <https://www.nature.com/articles/s41586-021-04014-z>

History

2013

Atari (DQN)
[Deepmind]



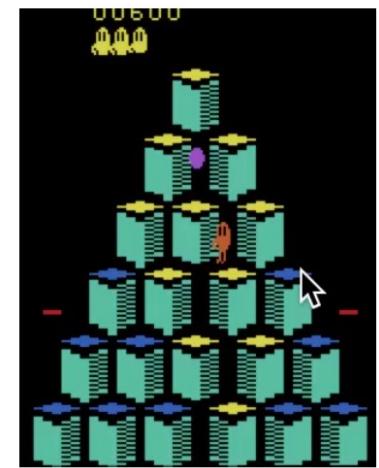
Pong



Enduro



Beamrider



Q*bert

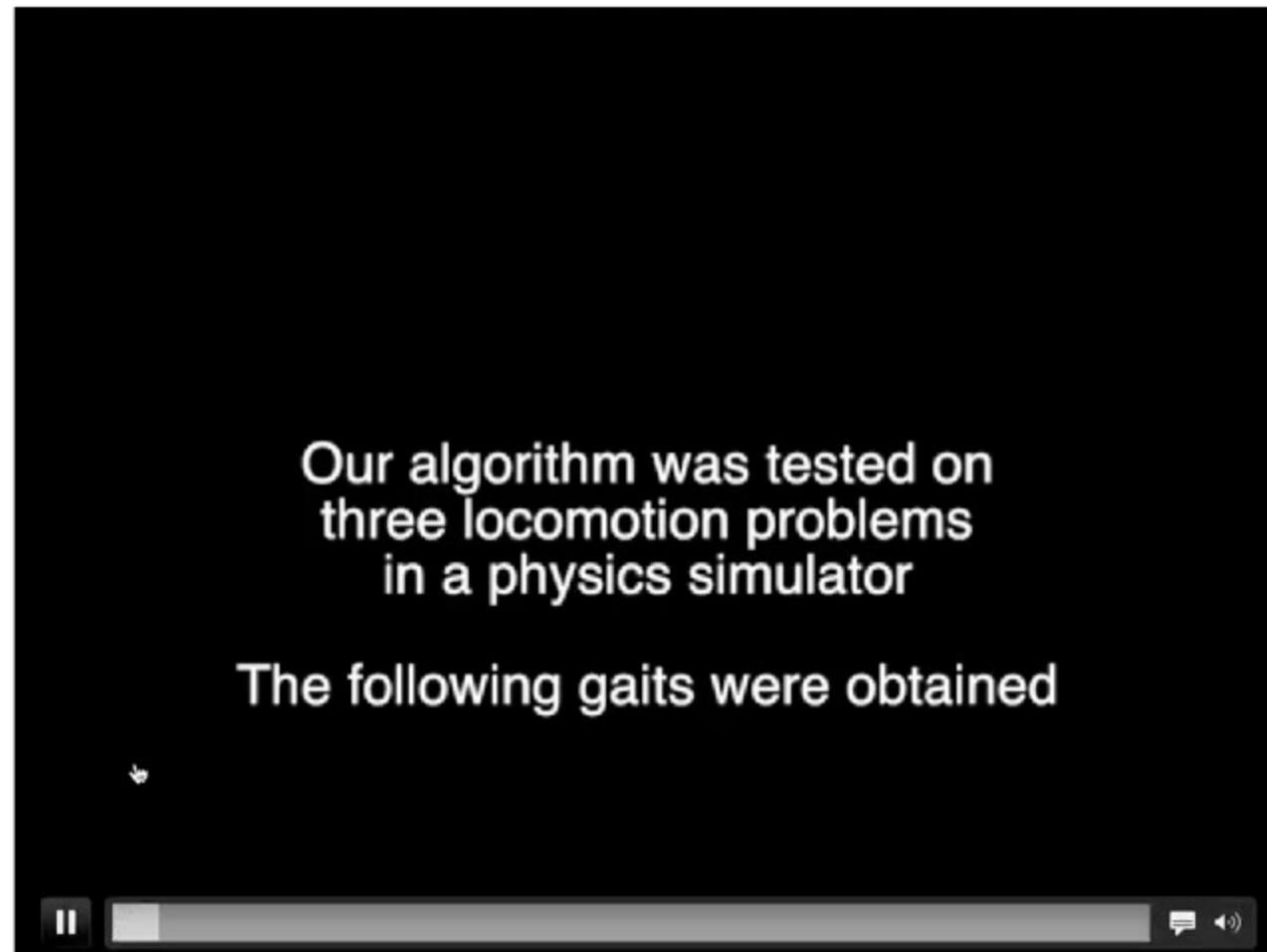
A Few Deep RL Highlights

2013

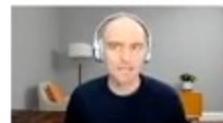
Atari (DQN)
[Deepmind]

2014

2D locomotion (TRPO)
[Berkeley]



Play 0:06 – 0:25



History

2013	Atari (DQN) [Deepmind]
2014	2D locomotion (TRPO) [Berkeley]
2015	AlphaGo [Deepmind]



Tian et al, 2016; Maddison et al, 2014; Clark et al, 2015

A Few Deep RL Highlights

2013

Atari (DQN)
[Deepmind]

2014

2D locomotion (TRPO)
[Berkeley]

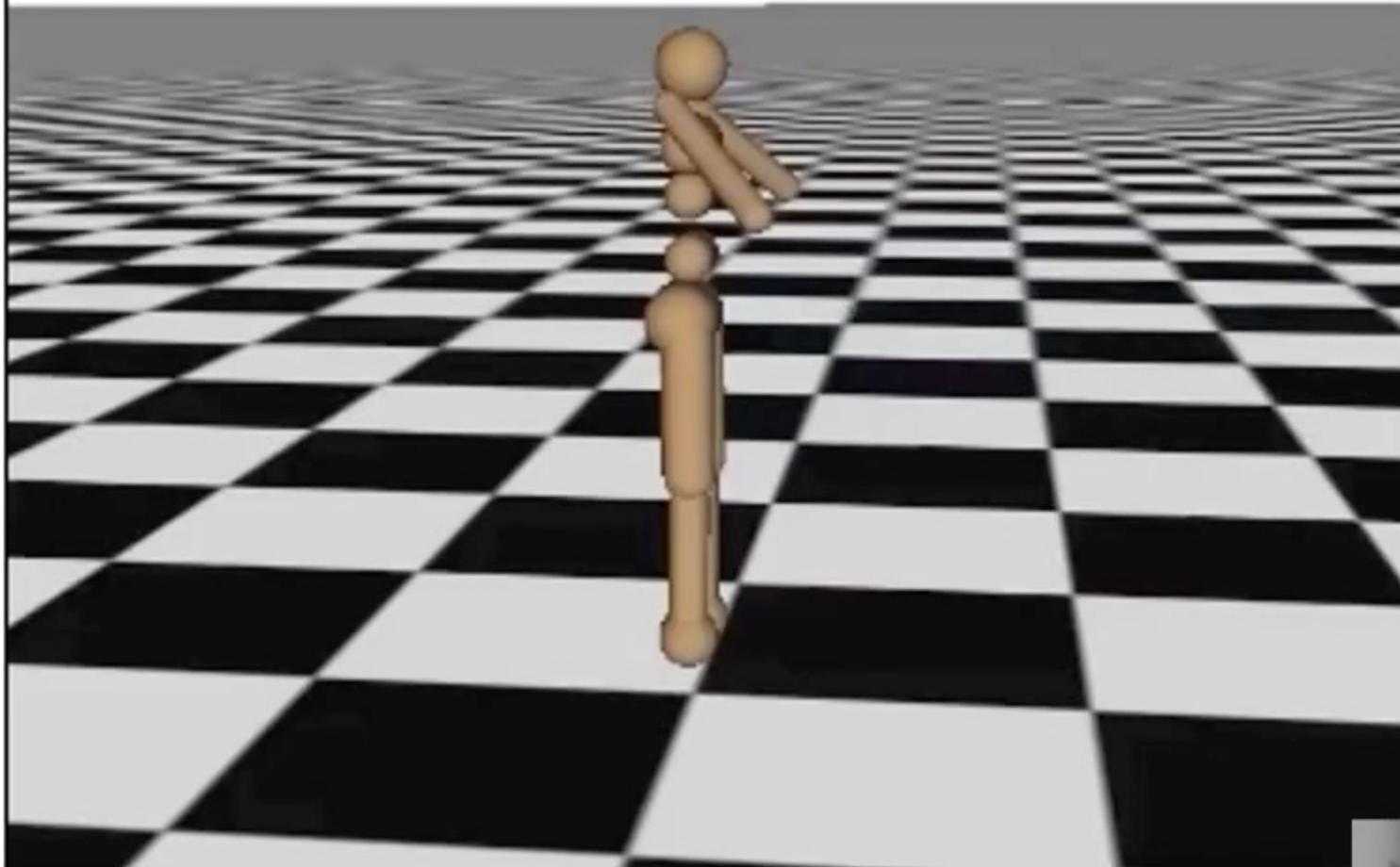
2015

AlphaGo
[Deepmind]

2016

3D locomotion (TRPO+GAE)
[Berkeley]

Iteration 0



[Schulman, Moritz, Levine, Jordan, Abbeel, ICLR 2016]



A Few Deep RL Highlights

2013	Atari (DQN) [Deepmind]
2014	2D locomotion (TRPO) [Berkeley]
2015	AlphaGo [Deepmind]
2016	3D locomotion (TRPO+GAE) [Berkeley]
2016	Real Robot Manipulation (GPS) [Berkeley]

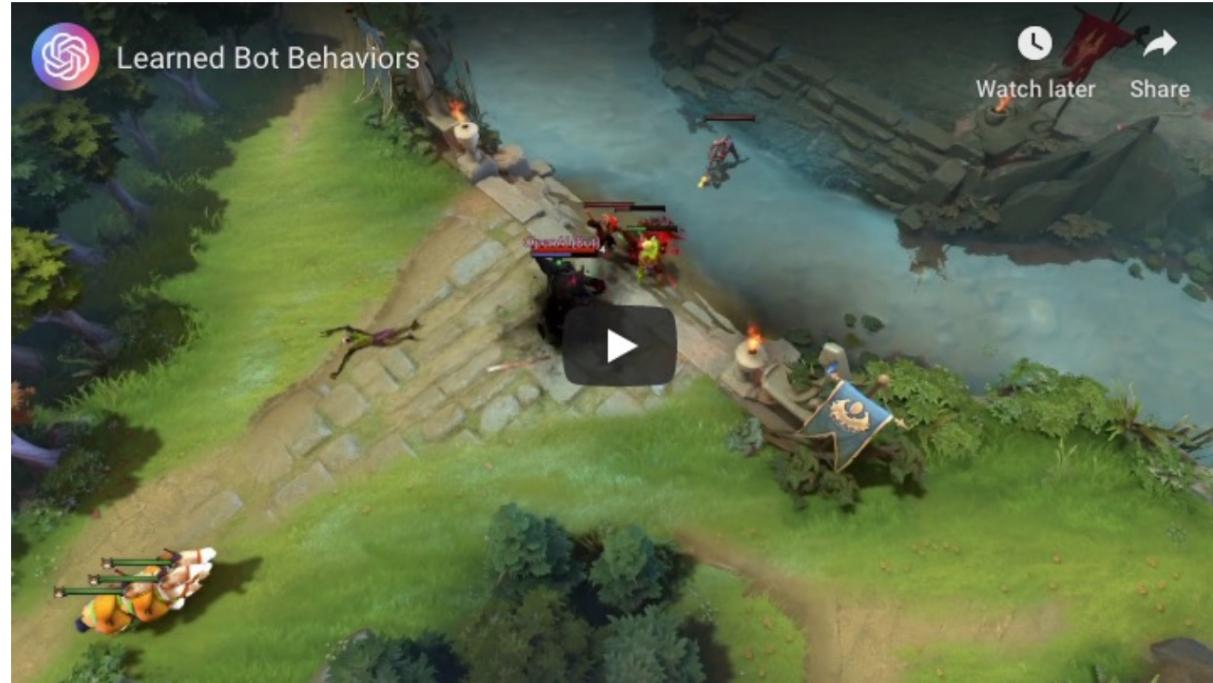


[Levine*, Finn*, Darrell, Abbeel, JMLR 2016]



History

2013	Atari (DQN) [Deepmind]
2014	2D locomotion (TRPO) [Berkeley]
2015	AlphaGo [Deepmind]
2016	3D locomotion (TRPO+GAE) [Berkeley]
2016	Real Robot Manipulation (GPS) [Berkeley, Google]
2017	Dota2 (PPO) [OpenAI]



OpenAI Dota Bot beat best humans 1:1 (Aug 2018)

A Few Deep RL Highlights

2013

Atari (DQN)
[Deepmind]

2014

2D locomotion (TRPO)
[Berkeley]

2015

AlphaGo
[Deepmind]

2016

3D locomotion (TRPO+GAE)
[Berkeley]

2016

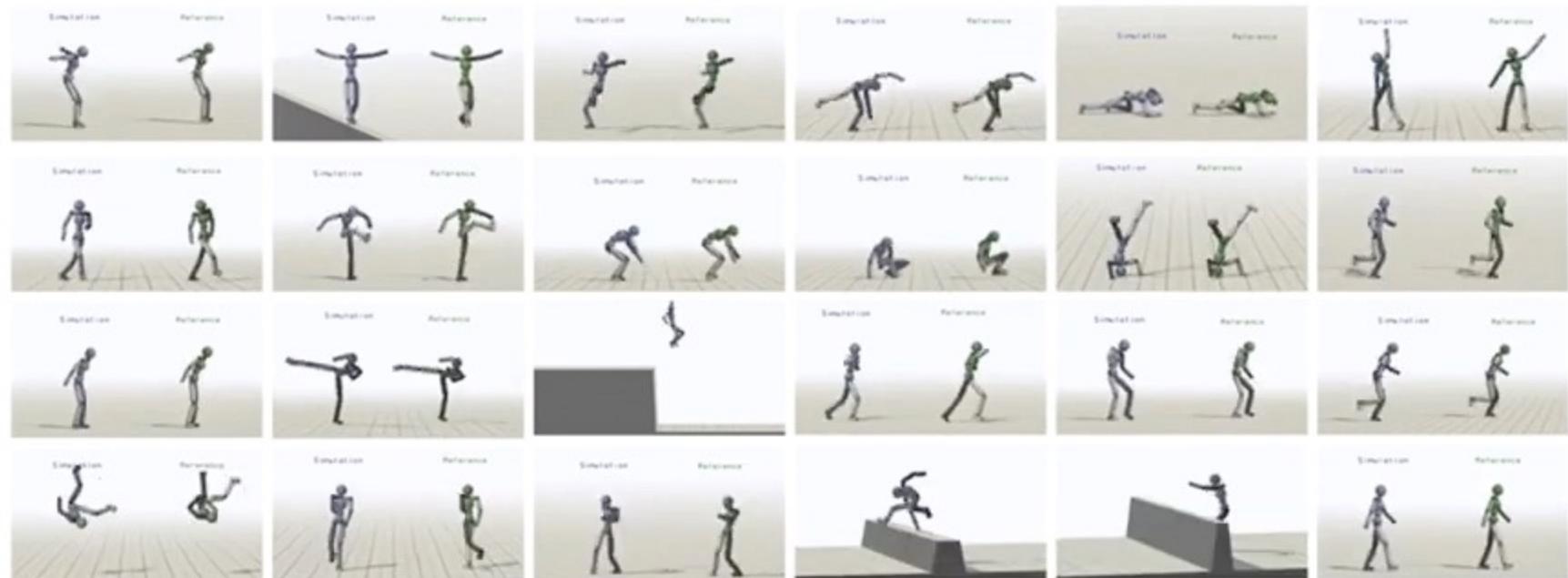
Real Robot Manipulation
(GPS) [Berkeley, Google]

2017

Dota2
(PPO) [OpenAI]

2018

DeepMimic
[Berkeley]



[Peng, Abbeel, Levine, van de Panne, 2018]



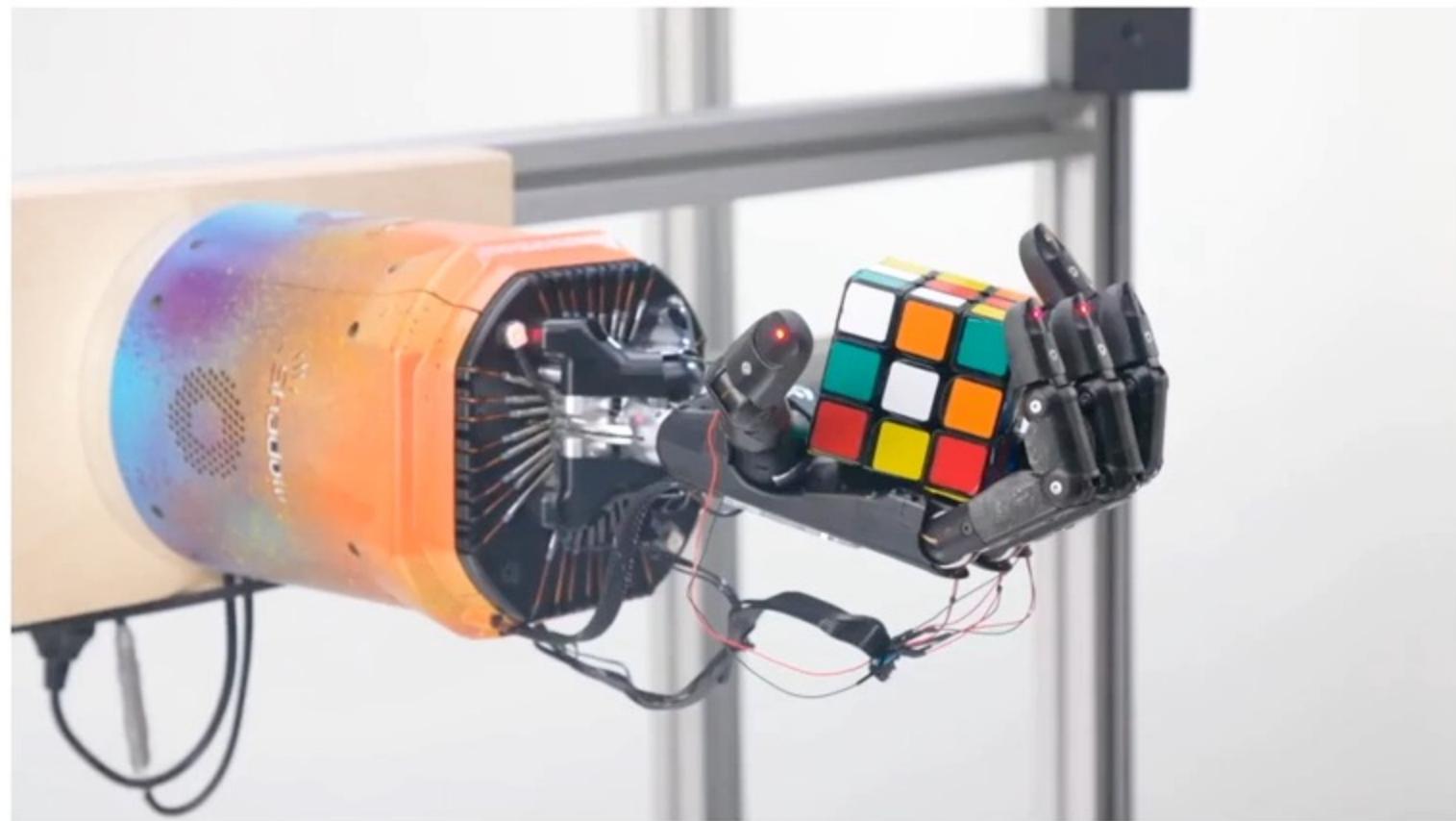
History

2013	Atari (DQN) [Deepmind]
2014	2D locomotion (TRPO) [Berkeley]
2015	AlphaGo [Deepmind]
2016	3D locomotion (TRPO+GAE) [Berkeley]
2016	Real Robot Manipulation (GPS) [Berkeley, Google]
2017	Dota2 (PPO) [OpenAI]
2018	DeepMimic [Berkeley]
2019	AlphaStar [Deepmind]



A Few Deep RL Highlights

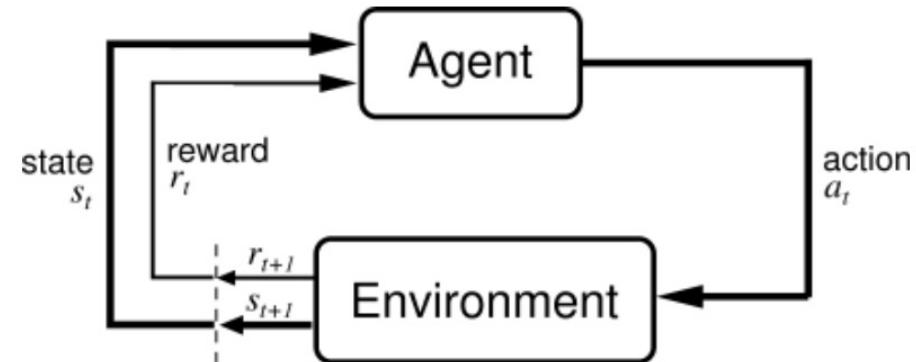
2013	Atari (DQN) [Deepmind]
2014	2D locomotion (TRPO) [Berkeley]
2015	AlphaGo [Deepmind]
2016	3D locomotion (TRPO+GAE) [Berkeley]
2016	Real Robot Manipulation (GPS) [Berkeley, Google]
2017	Dota2 (PPO) [OpenAI]
2018	DeepMimic [Berkeley]
2019	AlphaStar [Deepmind]
2019	Rubik's Cube (PPO+DR) [OpenAI]



Let's Begin: Markov Decision Processes (MDPs)

An MDP is defined by:

- Set of states S
- Set of actions A
- Transition function $P(s' | s, a)$
- Reward function $R(s, a, s')$
- Start state s_0
- Discount factor γ
- Horizon H



The Goal

- The policy is $\pi_\theta: S \rightarrow A$ for infinite horizon or

$\pi_\theta: S \times \{0, \dots, H\} \rightarrow A$ for finite horizon MDP.

MDP (S, A, T, R, γ, H) ,

goal: $\max_{\pi} \mathbb{E} \left[\sum_{t=0}^H \gamma^t R(S_t, A_t, S_{t+1}) | \pi \right]$

Sometimes the policy could be stochastic: $\pi : S \times A \rightarrow [0,1]$, which is

$$\pi(a|s) = \Pr(A_t = a | S_t = s).$$

Example: Grid World

An MDP is defined by:

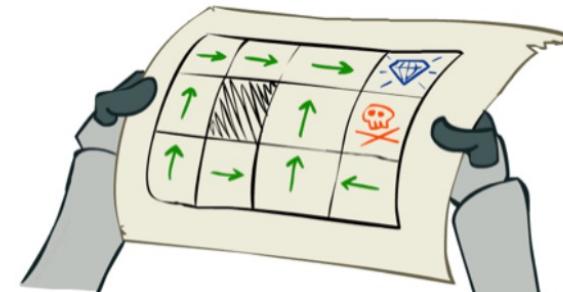
- Set of states S
- Set of actions A
- Transition function $P(s' | s, a)$
- Reward function $R(s, a, s')$
- Start state s_0
- Discount factor γ
- Horizon H



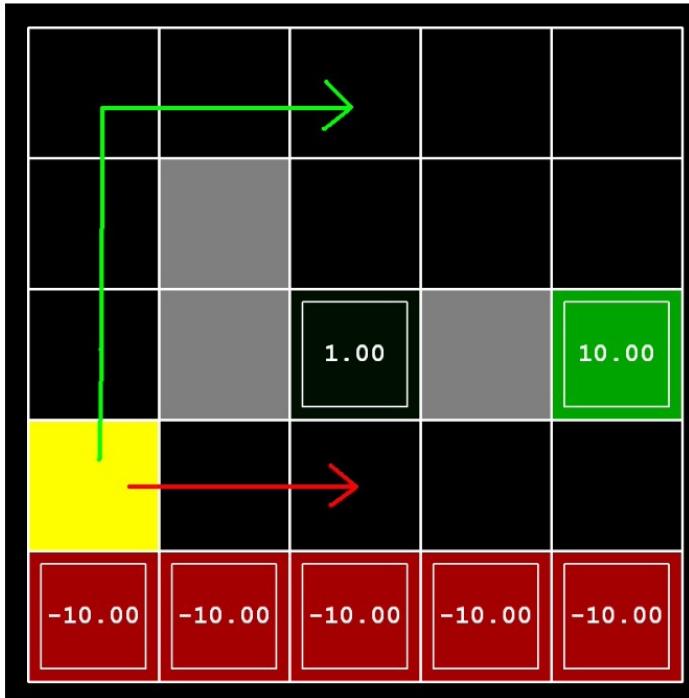
Goal:

$$\max_{\pi} \mathbb{E} \left[\sum_{t=0}^H \gamma^t R(S_t, A_t, S_{t+1}) \middle| \pi \right]$$

π^* :

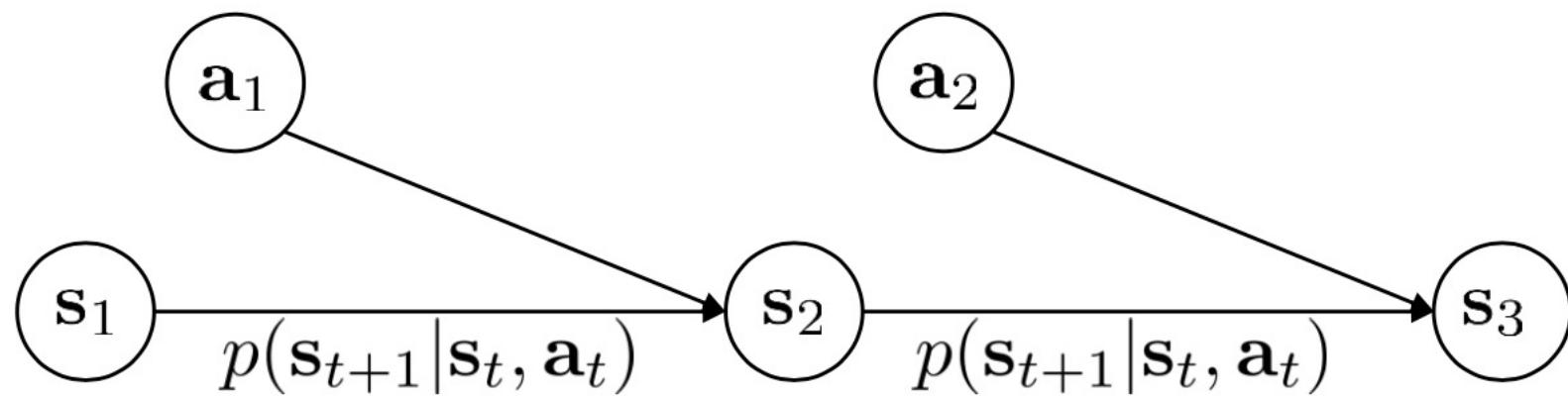


Exercise



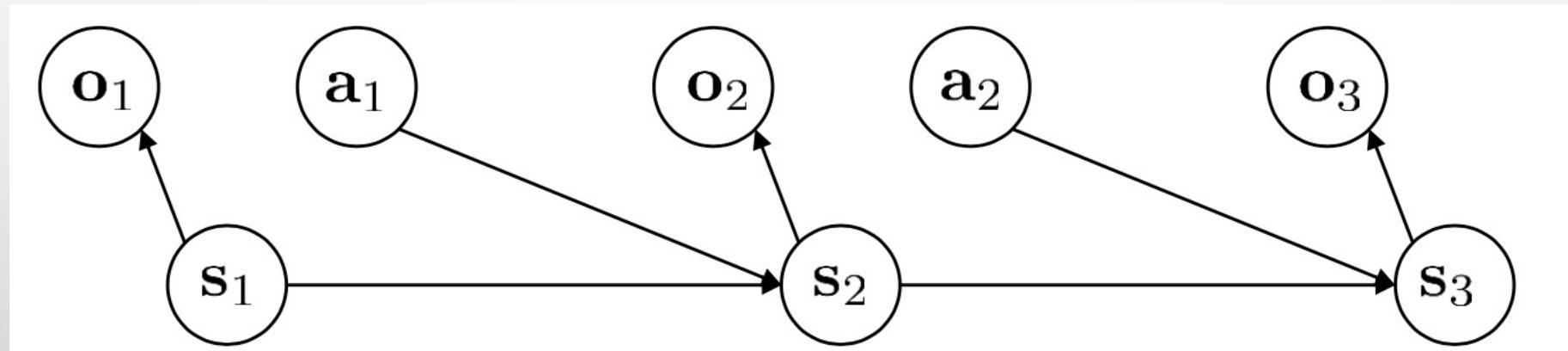
- | | |
|--|-----------------------------------|
| (a) Prefer the close exit (+1), risking the cliff (-10) | (1) $\gamma = 0.1$, noise = 0.5 |
| (b) Prefer the close exit (+1), but avoiding the cliff (-10) | (2) $\gamma = 0.99$, noise = 0 |
| (c) Prefer the distant exit (+10), risking the cliff (-10) | (3) $\gamma = 0.99$, noise = 0.5 |
| (d) Prefer the distant exit (+10), avoiding the cliff (-10) | (4) $\gamma = 0.1$, noise = 0 |

Graphical Model of MDPs



Partially Observable MDPs (POMDPs)

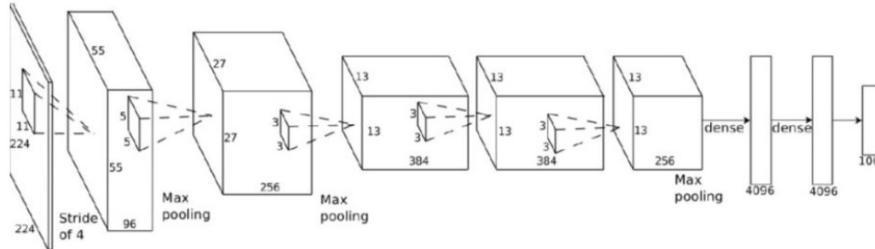
- Often times the state S_t is **hidden** from the agent, and only **noisy** or **incomplete** measurement of it is available O_t .



Policy as a function of S_t or O_t



\mathbf{o}_t



$\pi_\theta(\mathbf{a}_t | \mathbf{o}_t)$



\mathbf{a}_t

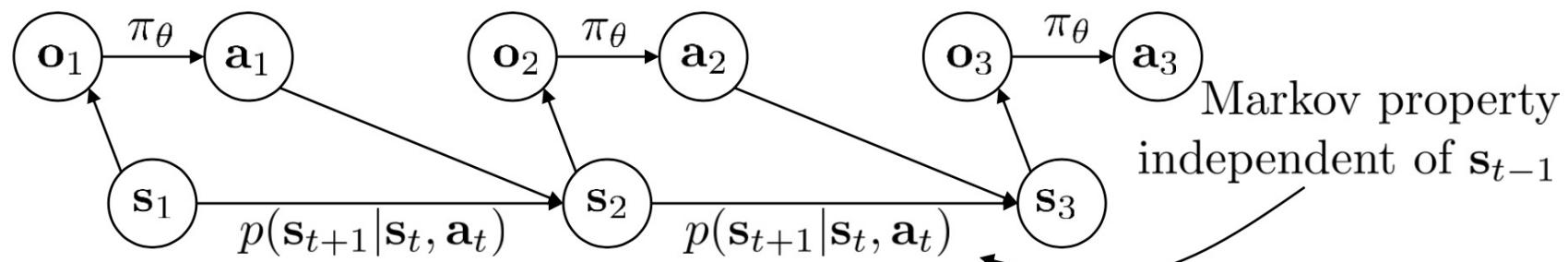
\mathbf{s}_t – state

\mathbf{o}_t – observation

\mathbf{a}_t – action

$\pi_\theta(\mathbf{a}_t | \mathbf{o}_t)$ – policy

$\pi_\theta(\mathbf{a}_t | \mathbf{s}_t)$ – policy (fully observed)



Optimal Value Function

MDP (S, A, T, R, γ, H) ,

goal: $\max_{\pi} \mathbb{E} \left[\sum_{t=0}^H \gamma^t R(S_t, A_t, S_{t+1}) | \pi \right]$

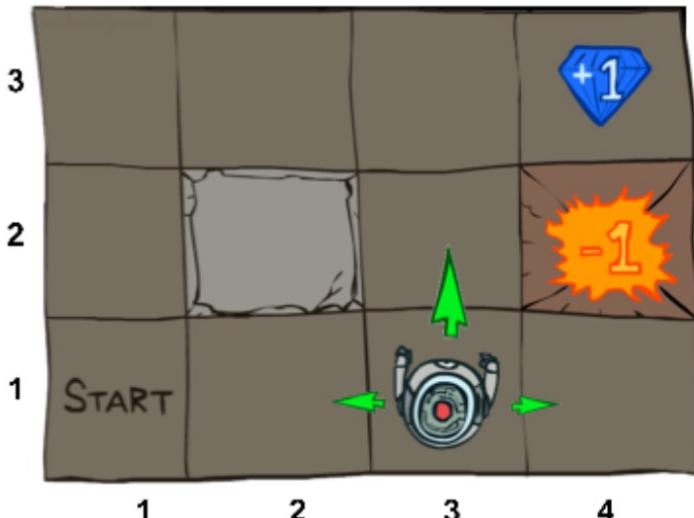
$$V^*(s) = \max_{\pi} \mathbb{E} \left[\sum_{t=0}^H \gamma^t R(s_t, a_t, s_{t+1}) \mid \pi, s_0 = s \right]$$

= sum of discounted rewards when starting from state s and acting optimally

Optimal Value Function

$$V^*(s) = \max_{\pi} \mathbb{E} \left[\sum_{t=0}^H \gamma^t R(s_t, a_t, s_{t+1}) \mid \pi, s_0 = s \right]$$

= sum of discounted rewards when starting from state s and acting optimally



Let's assume:

actions deterministically successful, gamma = 1, H = 100

$$V^*(4,3) =$$

$$V^*(3,3) =$$

$$V^*(2,3) =$$

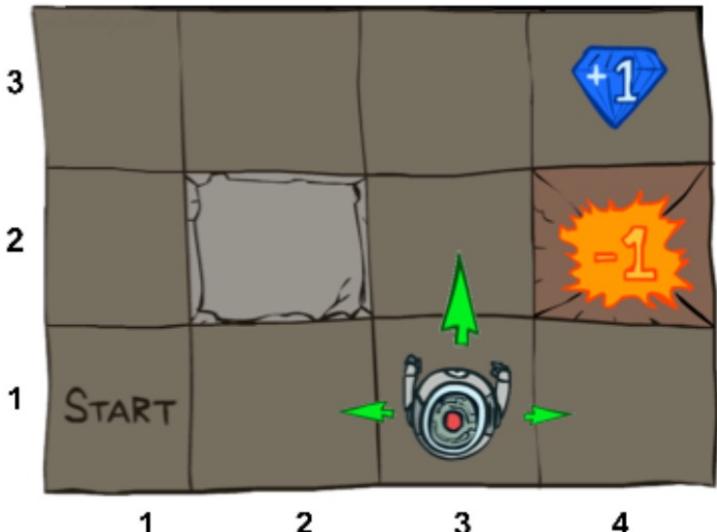
$$V^*(1,1) =$$

$$V^*(4,2) =$$

Optimal Value Function

$$V^*(s) = \max_{\pi} \mathbb{E} \left[\sum_{t=0}^H \gamma^t R(s_t, a_t, s_{t+1}) \mid \pi, s_0 = s \right]$$

= sum of discounted rewards when starting from state s and acting optimally



Let's assume:

actions deterministically successful, gamma = 0.9, H = 100

$$V^*(4,3) =$$

$$V^*(3,3) =$$

$$V^*(2,3) =$$

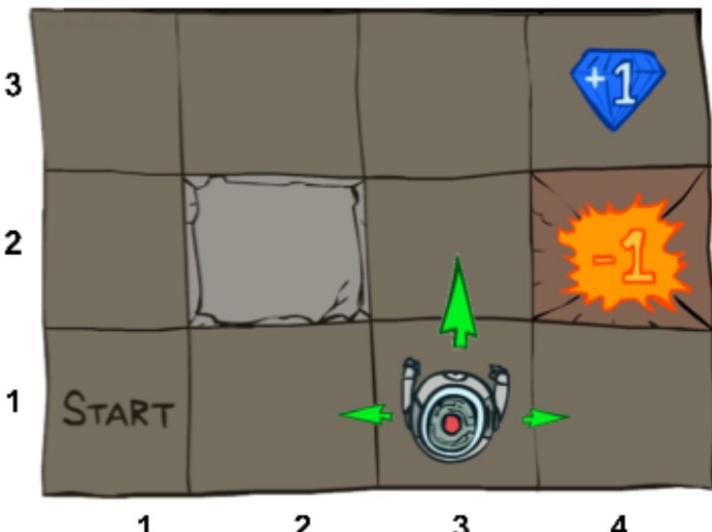
$$V^*(1,1) =$$

$$V^*(4,2) =$$

Optimal Value Function

$$V^*(s) = \max_{\pi} \mathbb{E} \left[\sum_{t=0}^H \gamma^t R(s_t, a_t, s_{t+1}) \mid \pi, s_0 = s \right]$$

= sum of discounted rewards when starting from state s and acting optimally



Let's assume:

actions successful w/probability 0.8, gamma = 0.9, H = 100

$$V^*(4,3) =$$

$$V^*(3,3) =$$

$$V^*(2,3) =$$

$$V^*(1,1) =$$

$$V^*(4,2) =$$