



# Reinforcement Learning

## Computer Engineering Department Sharif University of Technology

**Mohammad Hossein Rohban, Ph.D.**

Spring 2025

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

# Motivation (cont.) ChatGPT; Why RL?!

Step 1

**Collect demonstration data and train a supervised policy.**

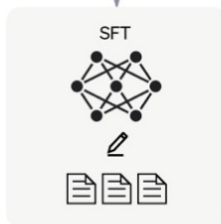
A prompt is sampled from our prompt dataset.



A labeler demonstrates the desired output behavior.



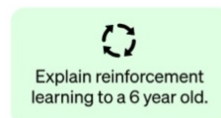
This data is used to fine-tune GPT-3.5 with supervised learning.



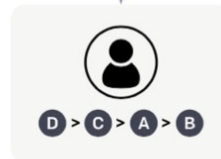
Step 2

**Collect comparison data and train a reward model.**

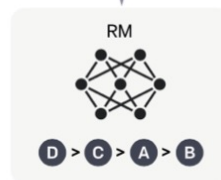
A prompt and several model outputs are sampled.



A labeler ranks the outputs from best to worst.



This data is used to train our reward model.



Step 3

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

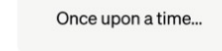
A new prompt is sampled from the dataset.



The PPO model is initialized from the supervised policy.



The policy generates an output.



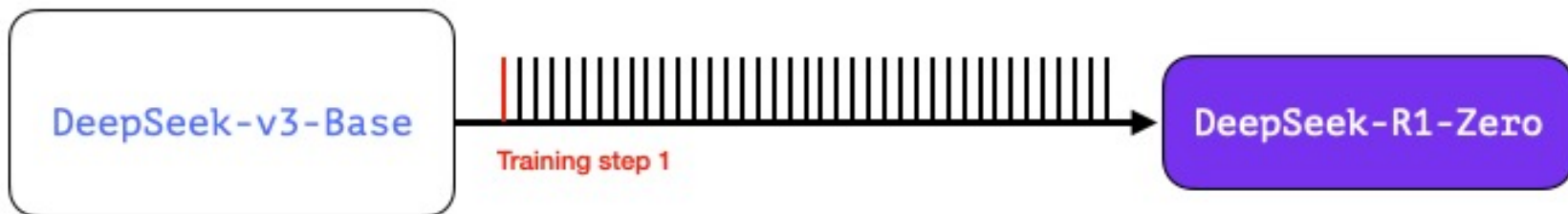
The reward model calculates a reward for the output.



The reward is used to update the policy using PPO.



# Large-scale Reasoning-Oriented Reinforcement Learning



Solution score (reward)

Training prompt

Write python code that takes a list of numbers, returns them in a sorted order, but also adds 42 at the start.

Model checkpoint under training

Generate 4 possible solutions

here's a joke about frogs

Low

echo 42

Low

def sort(a)  
...  
...

Low

def sort\_and\_prepend(a)  
...  
...

High

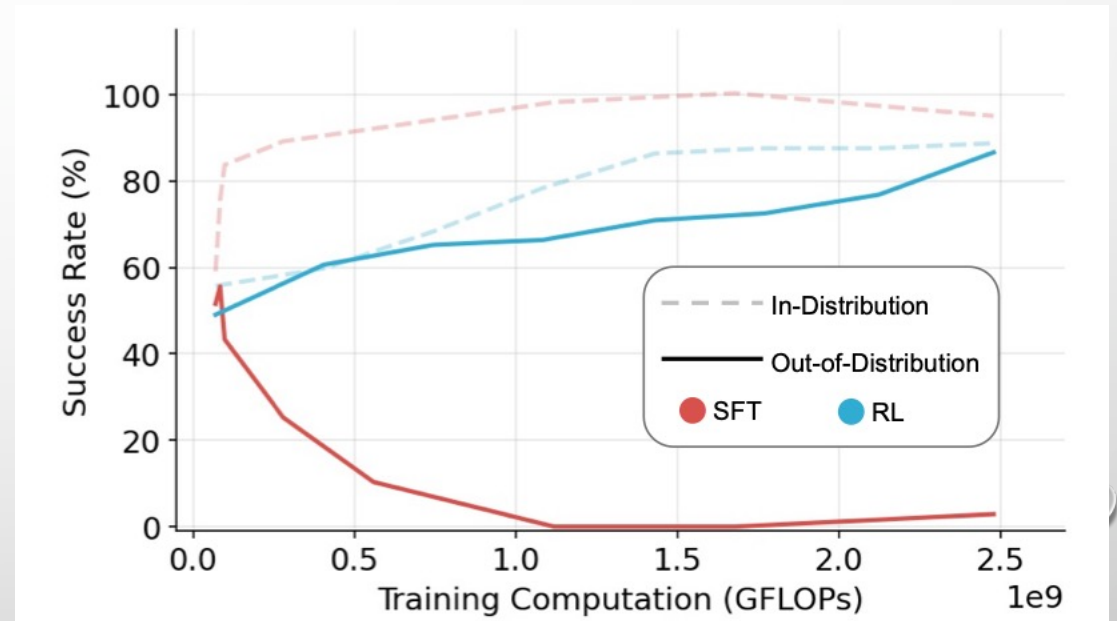
Update the model so its less likely to output low score solutions like these and more likely to output high-score solutions in response to such a prompt

# Motivation (cont.)

## SFT Memorizes, RL Generalizes: A Comparative Study of Foundation Model Post-training



★ First, **turn slightly right** towards the northeast and walk a short distance until you reach the next intersection, where you'll see **The Dutch** on your right. Next, make a **sharp left turn** to head northwest. Continue for a while until you reach the next intersection, where **Lola Taverna** will be on your right. Finally, **turn slightly right** to face northeast and walk a short distance until you reach your destination, **Shuka**, which will be on your right.



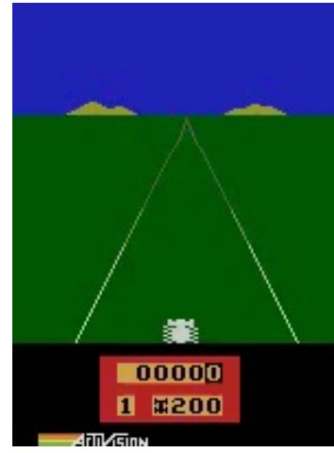
# History

2013

Atari (DQN)  
[Deepmind]



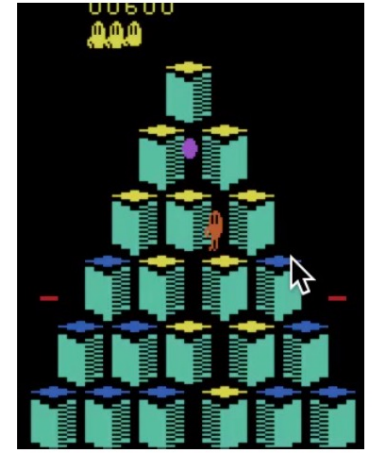
Pong



Enduro



Beamrider



Q\*bert



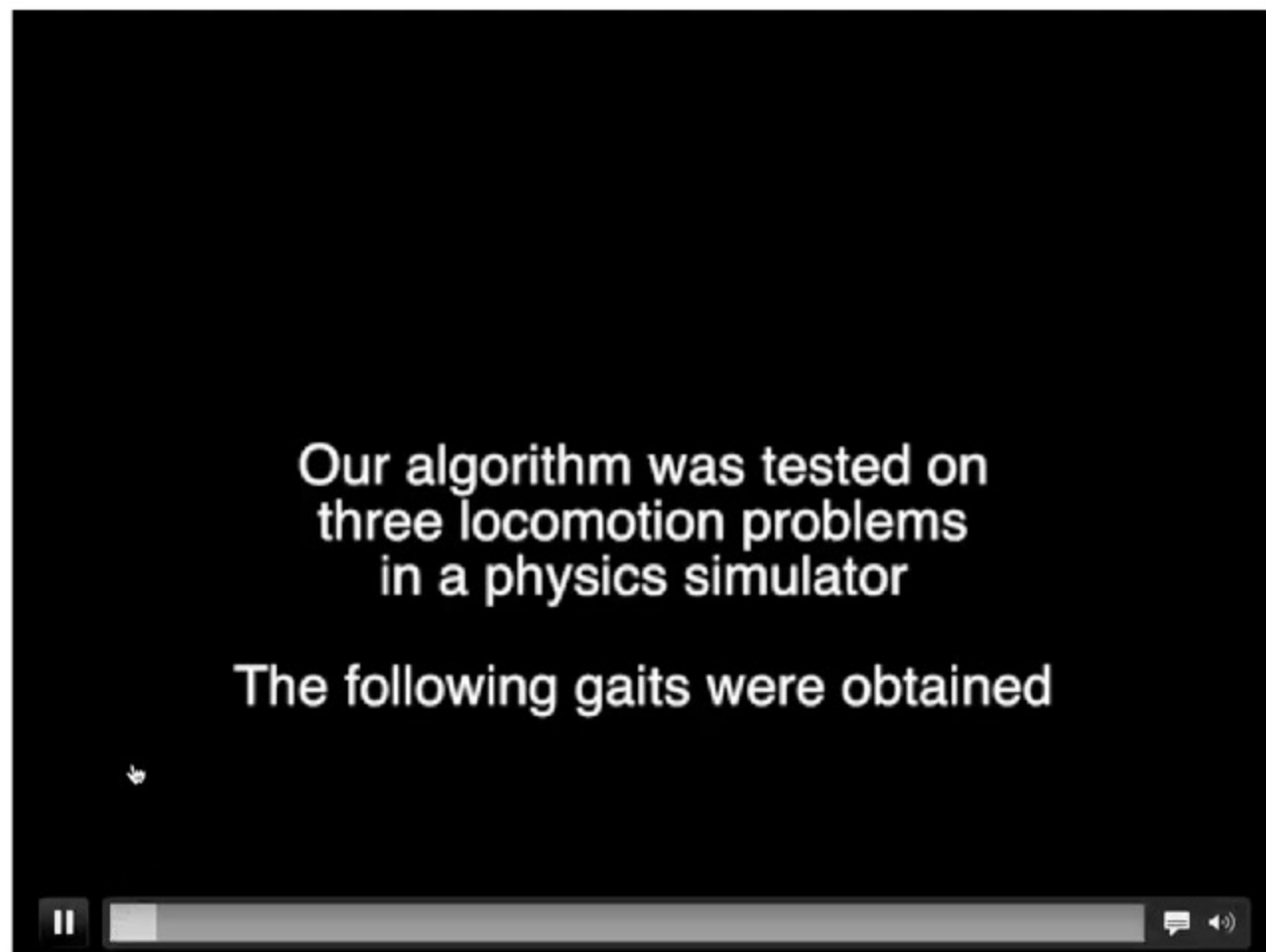
# A Few Deep RL Highlights

2013

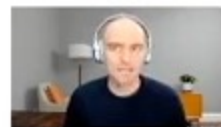
Atari (DQN)  
[Deepmind]

2014

2D locomotion (TRPO)  
[Berkeley]



Play 0:06 – 0:25



# History

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

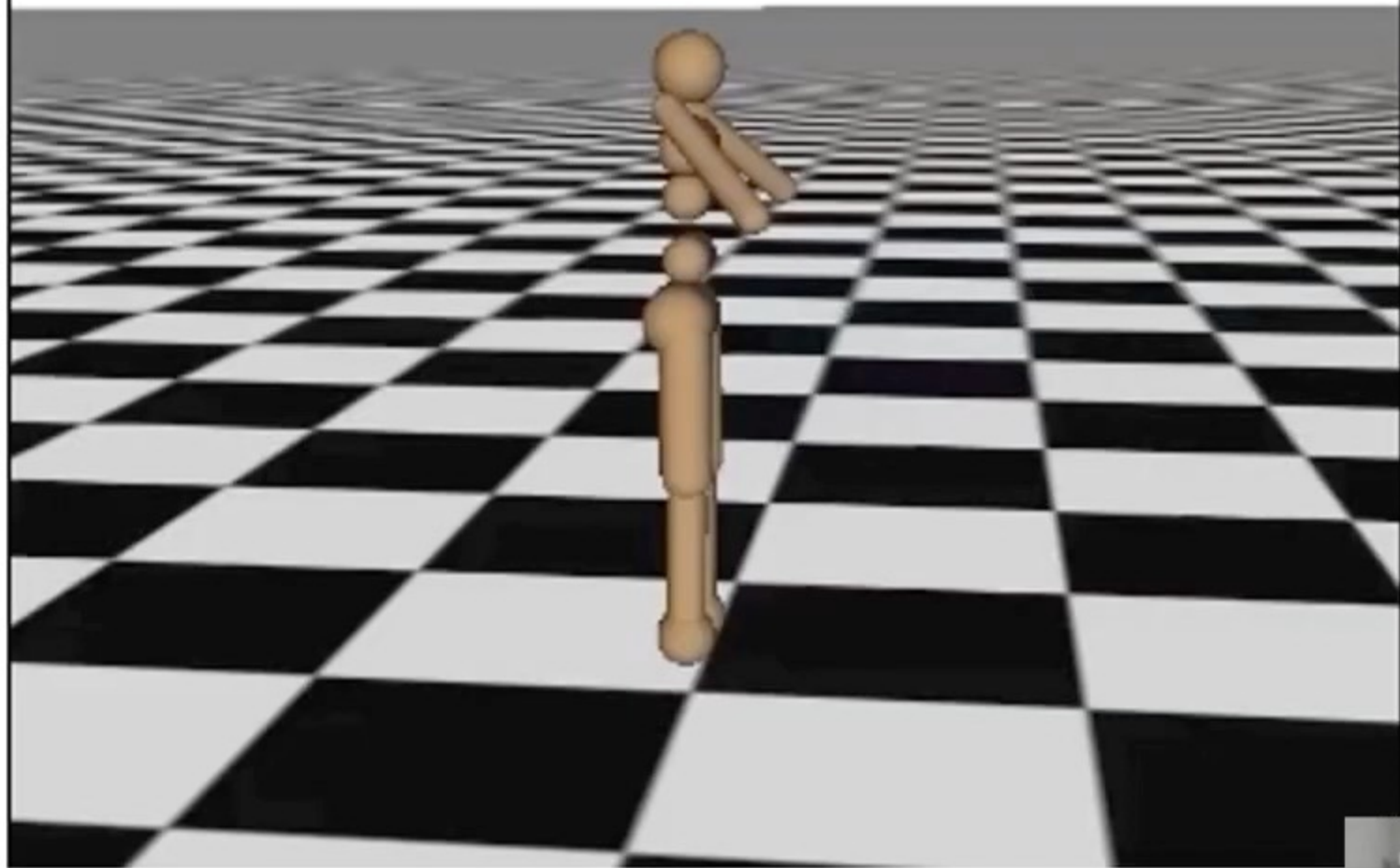


Tian et al, 2016; Maddison et al, 2014; Clark et al, 2015

# 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]



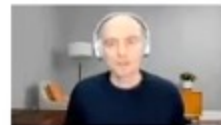


# A Few Deep RL Highlights

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

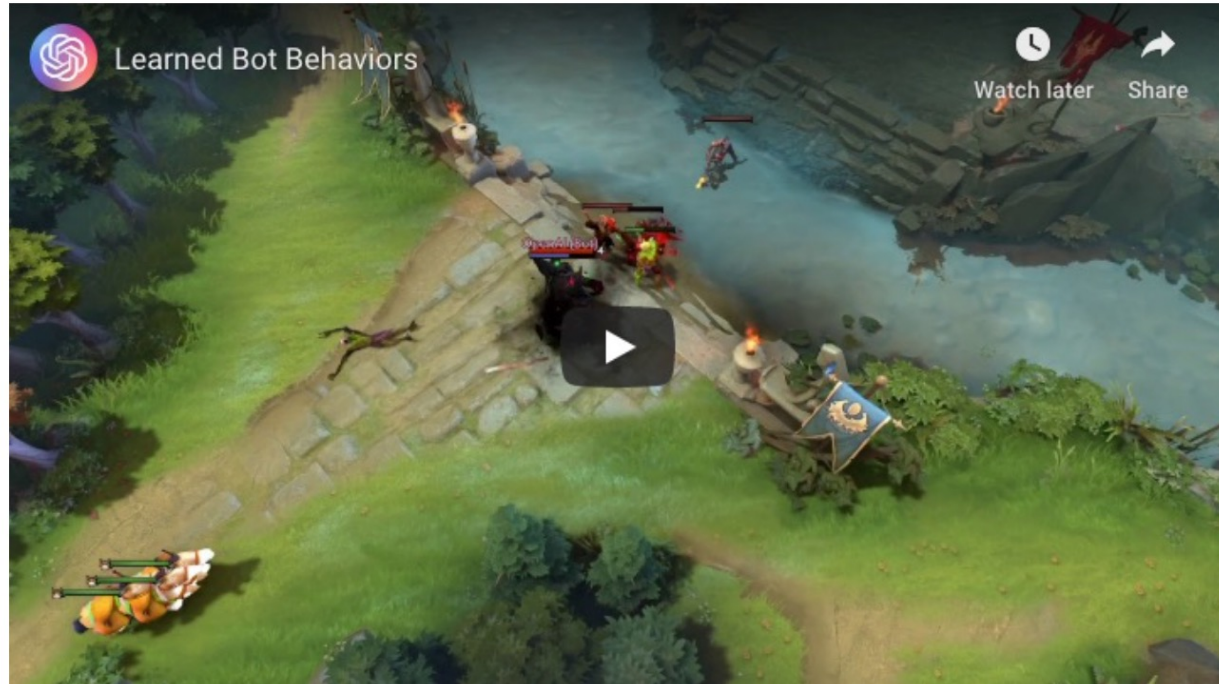


[Levine\*, Finn\*, Darrell, Abbeel, JMLR 2016]



# History

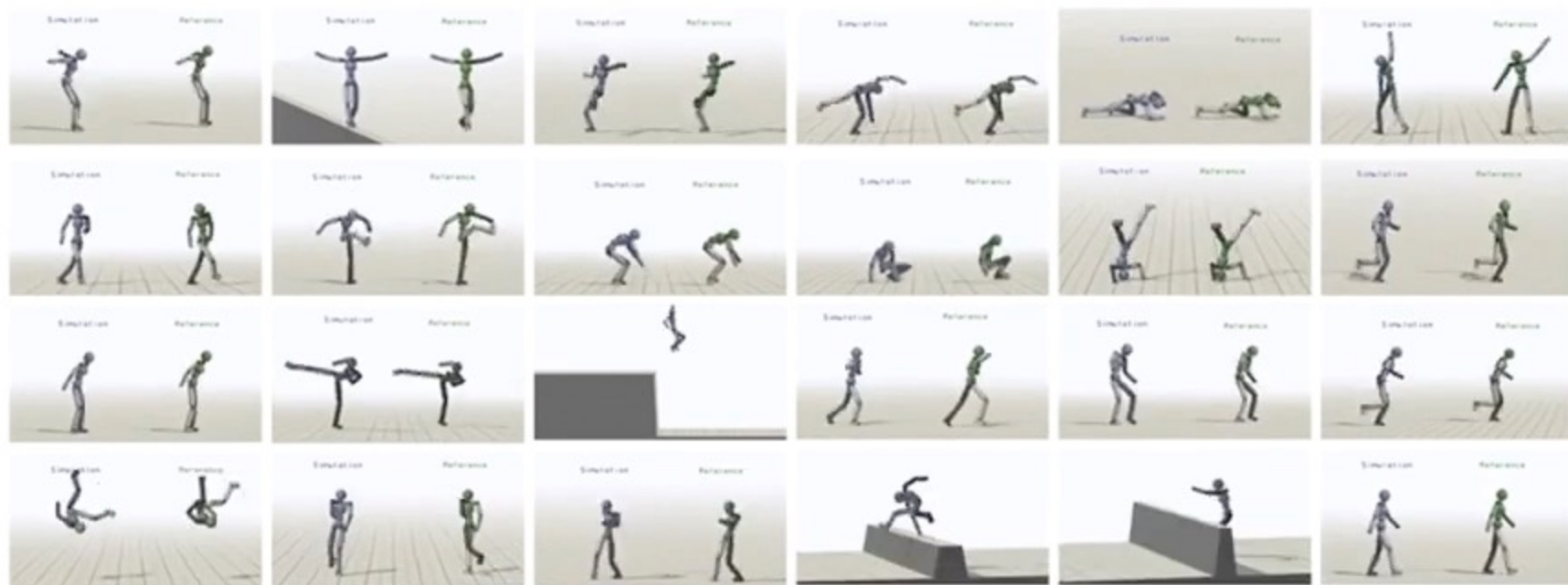
- 2013 Atari (DQN)  
[Deepmind]
- 2014 2D locomotion (TRPO)  
[Berkeley]
- 2015 AlphaGo  
[Deepmind]
- 2016 3D locomotion (TRPO+GAE)  
[Berkeley]
- 2016 Real Robot Manipulation  
(GPS) [Berkeley, Google]
- 2017 **Dota2**  
**(PPO) [OpenAI]**



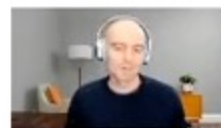
OpenAI Dota Bot beat best humans 1:1 (Aug 2018)

# 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

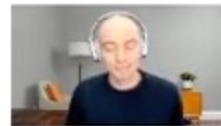
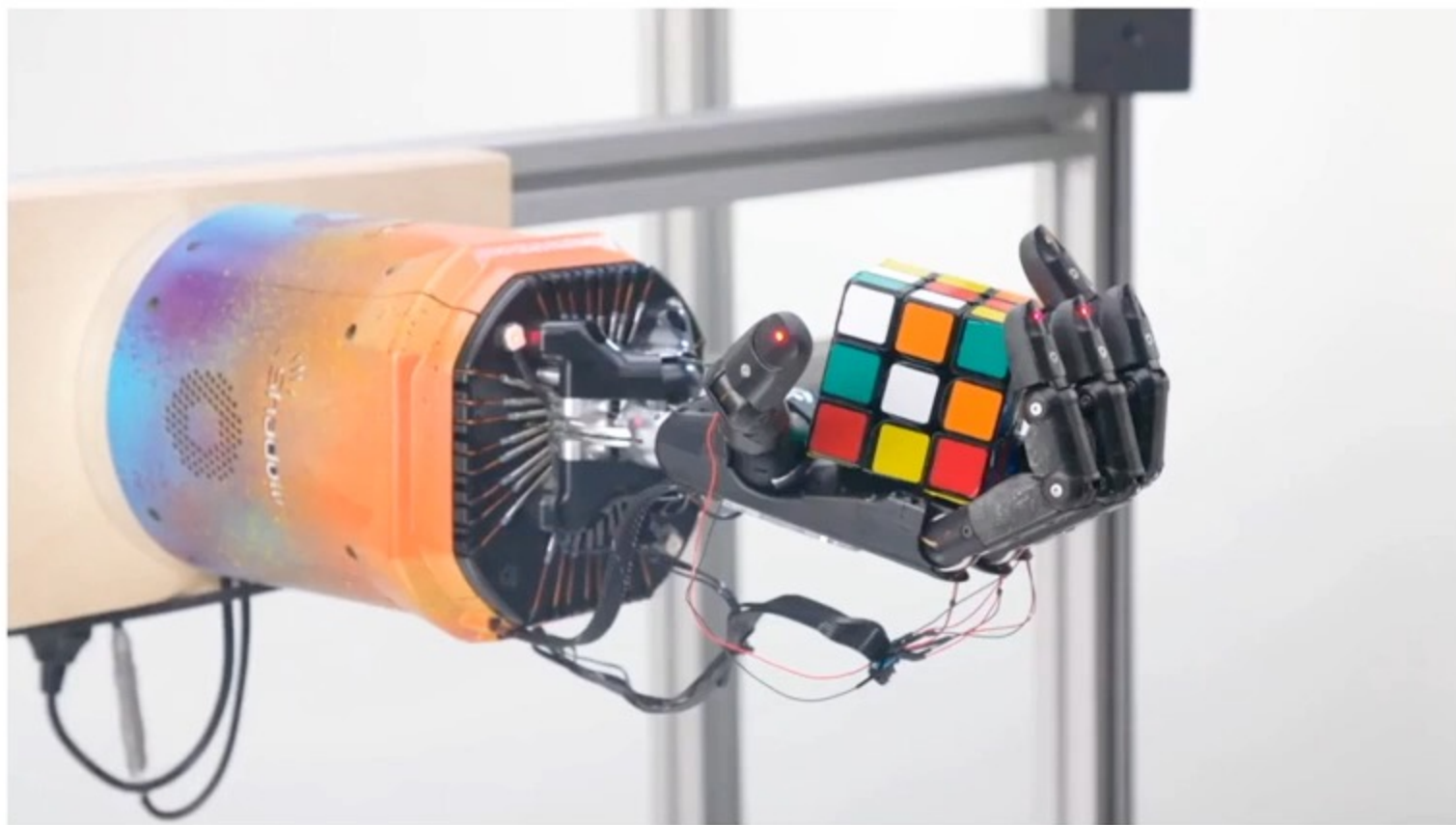
- 2013 Atari (DQN)  
[Deepmind]
- 2014 2D locomotion (TRPO)  
[Berkeley]
- 2015 AlphaGo  
[Deepmind]
- 2016 3D locomotion (TRPO+GAE)  
[Berkeley]
- 2016 Real Robot Manipulation  
(GPS) [Berkeley, Google]
- 2017 Dota2  
(PPO) [OpenAI]
- 2018 DeepMimic  
[Berkeley]
- 2019 AlphaStar  
[Deepmind]**





# A Few Deep RL Highlights

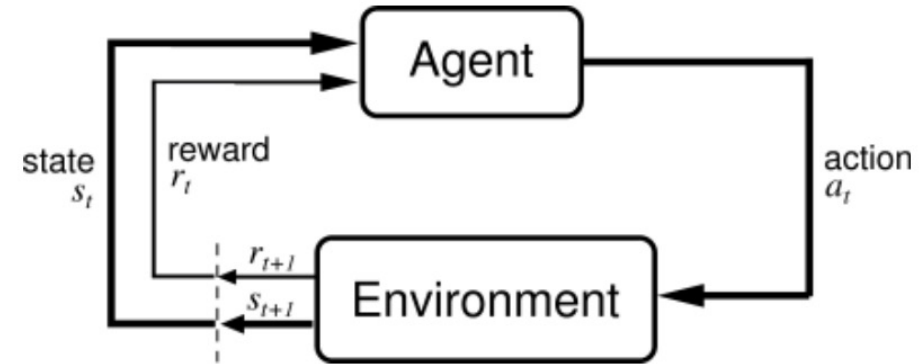
- 2013 Atari (DQN)  
[Deepmind]
- 2014 2D locomotion (TRPO)  
[Berkeley]
- 2015 AlphaGo  
[Deepmind]
- 2016 3D locomotion (TRPO+GAE)  
[Berkeley]
- 2016 Real Robot Manipulation  
(GPS) [Berkeley, Google]
- 2017 Dota2  
(PPO) [OpenAI]
- 2018 DeepMimic  
[Berkeley]
- 2019 AlphaStar  
[Deepmind]
- 2019 **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  $\gamma$
- 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  $(S, A, T, R, \gamma, H)$ ,

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

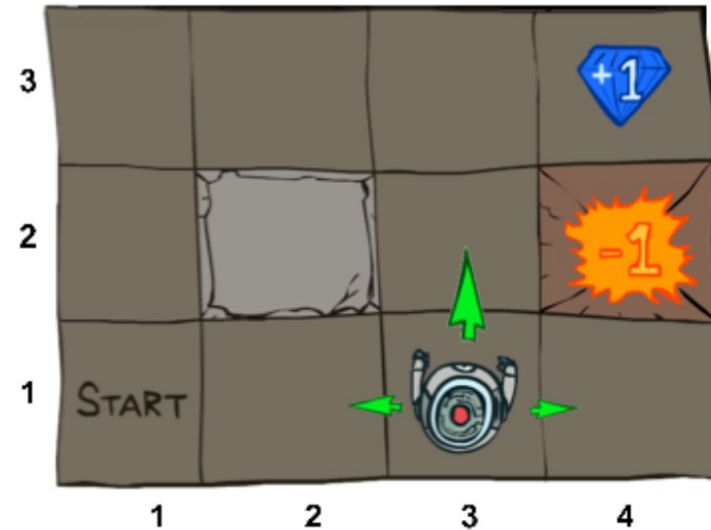
Sometimes the policy could be stochastic:  $\pi: S \times A \rightarrow [0, 1]$ , which is

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

# Example: Grid World

An MDP is defined by:

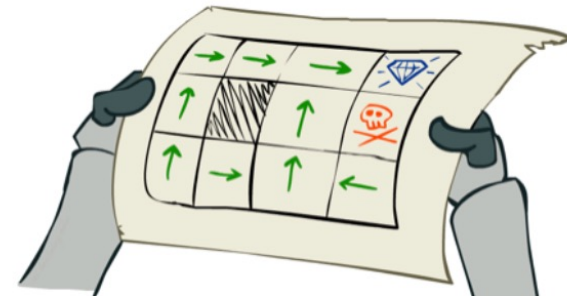
- Set of states  $S$
- Set of actions  $A$
- Transition function  $P(s' | s, a)$
- Reward function  $R(s, a, s')$
- Start state  $s_0$
- Discount factor  $\gamma$
- Horizon  $H$



Goal:

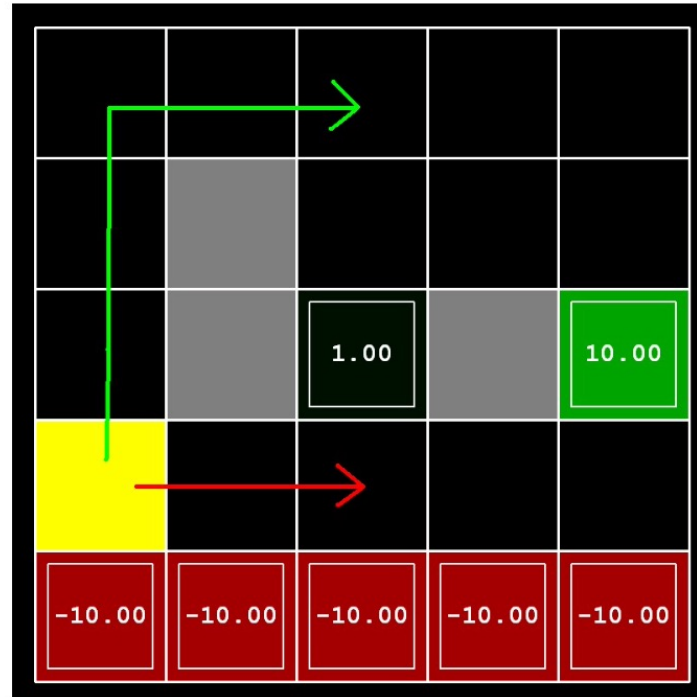
$$\max_{\pi} E\left[\sum_{t=0}^H \gamma^t R(S_t, A_t, S_{t+1}) | \pi\right]$$

$\pi$ :





# 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

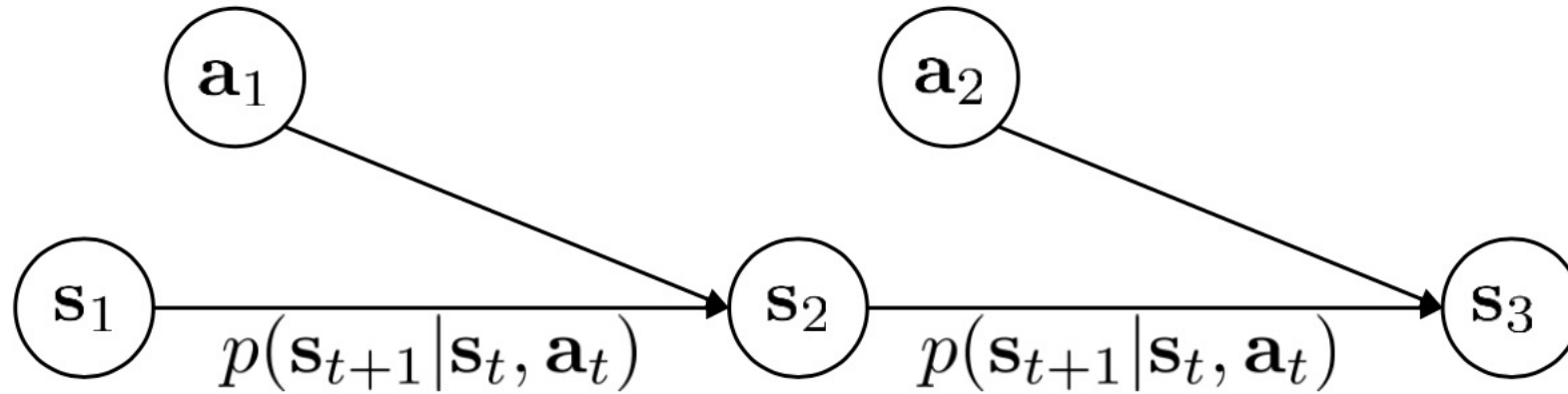
(c) Prefer the distant exit (+10), risking the cliff (-10)

(3)  $\gamma = 0.99$ , noise = 0.5

(d) Prefer the distant exit (+10), avoiding the cliff (-10)

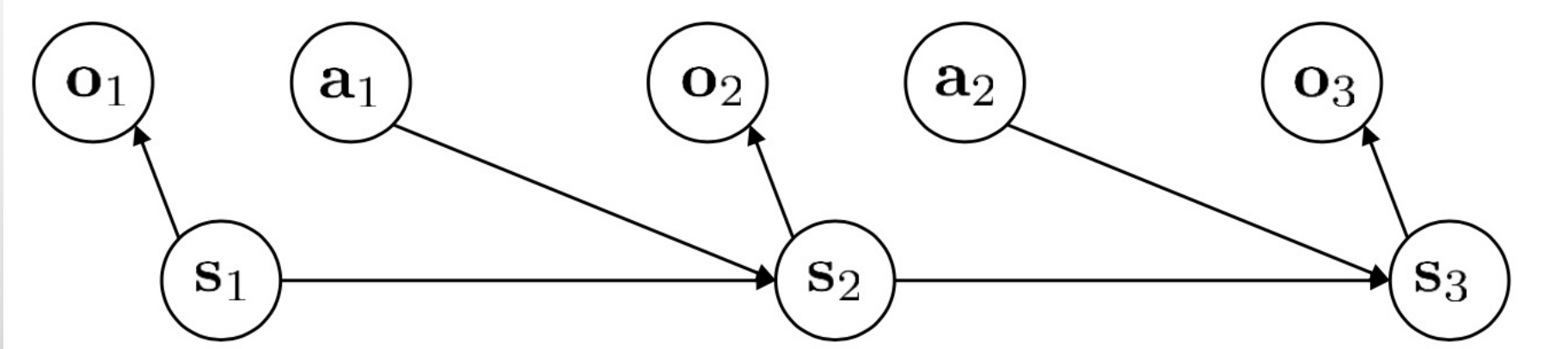
(4)  $\gamma = 0.1$ , noise = 0

# Graphical Model of MDPs



# Partially Observable MDPs (POMDPs)

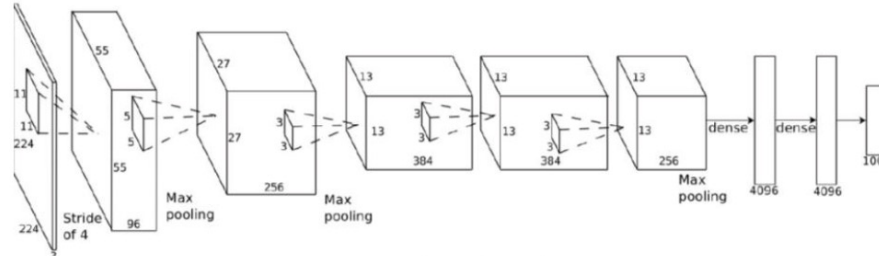
- Often times the state  $S_t$  is **hidden** from the agent,  
and only **noisy** or **incomplete** measurement of it is available  $O_t$ .



# Policy as a function of $S_t$ or $O_t$



$\mathbf{o}_t$



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



$\mathbf{a}_t$

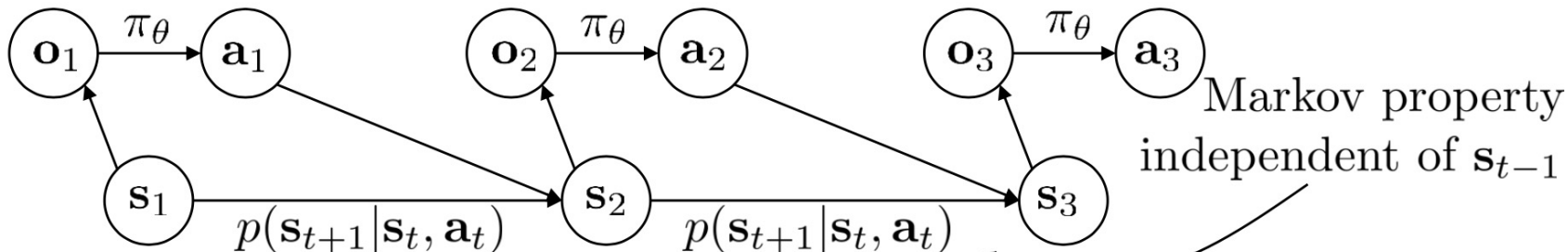
$\mathbf{s}_t$  – state

$\mathbf{o}_t$  – 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  $(S, A, T, R, \gamma, H)$ ,

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

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

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

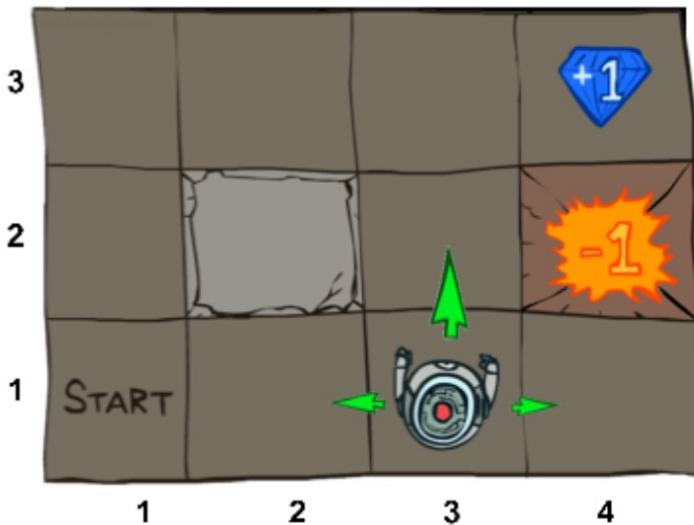
# Optimal Value Function

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

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

Let's assume:

actions deterministically successful,  $\gamma = 1$ ,  $H = 100$



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

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

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

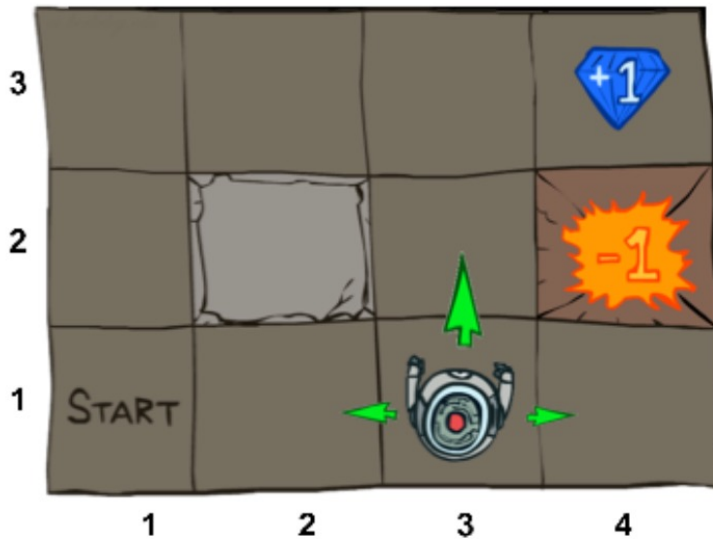
$$V^*(1,1) =$$

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

# 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,  $\gamma = 0.9$ ,  $H = 100$

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

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

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

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

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

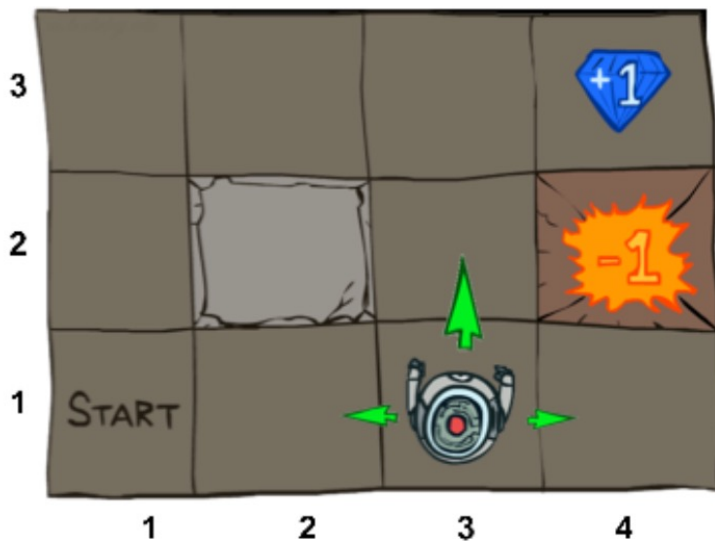
# Optimal Value Function

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

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

Let's assume:

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



$V^*(4,3) =$

$V^*(3,3) =$

$V^*(2,3) =$

$V^*(1,1) =$

$V^*(4,2) =$