Introduction to RL

Deep Reinforcement Learning

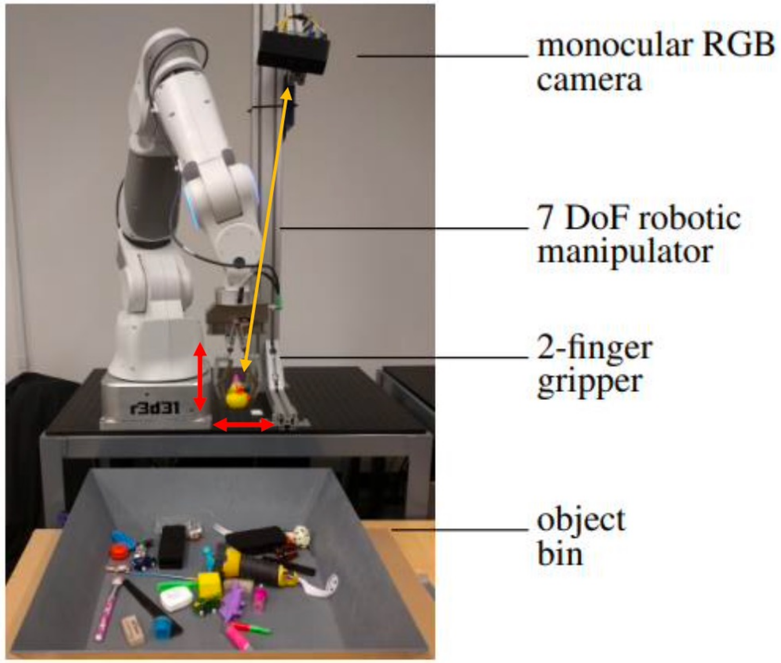
Mohammad Hossein Rohban, Ph.D.

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

RIML Lab, CE Department, Sharif University of Technology

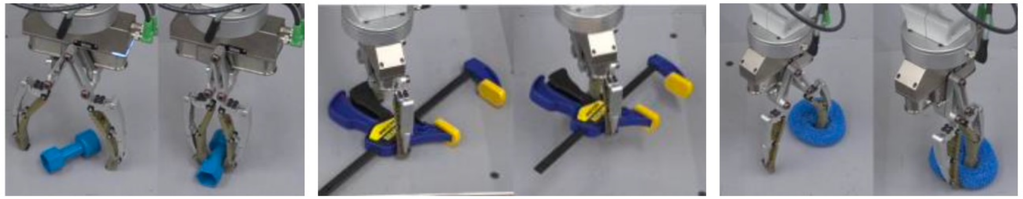
Spring 2026

Motivation



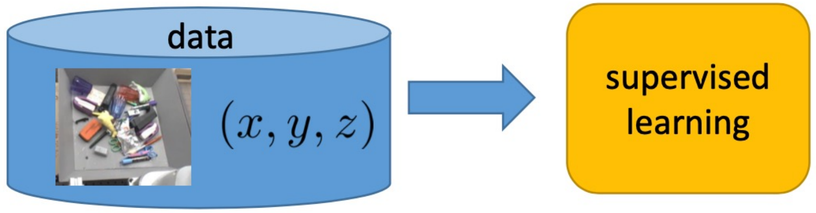
Option 1:

Understand the problem, design a solution



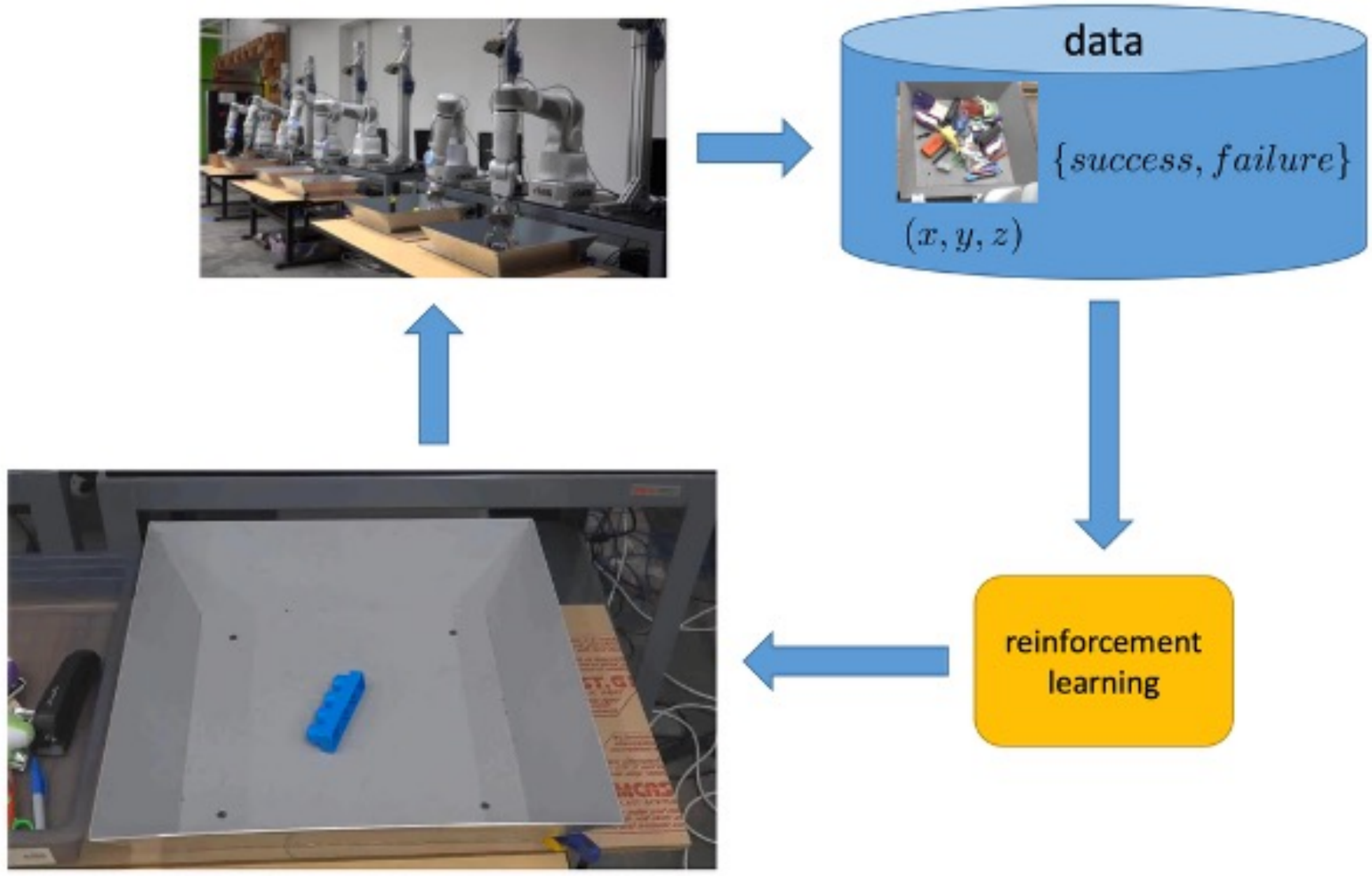
Option 2:

Set it up as a machine learning problem



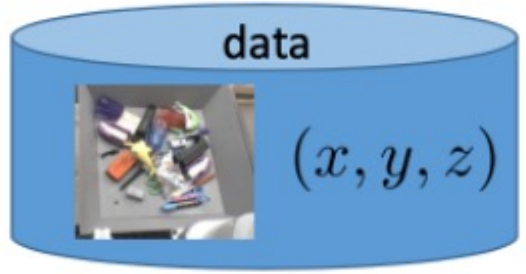
Motivation (cont.)

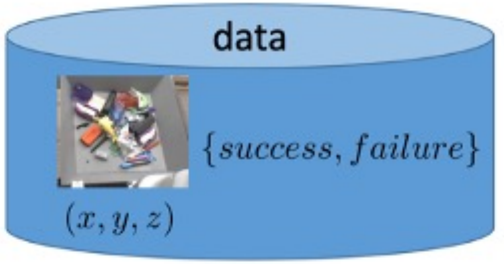
Motivation (cont.)



Motivation (cont.)

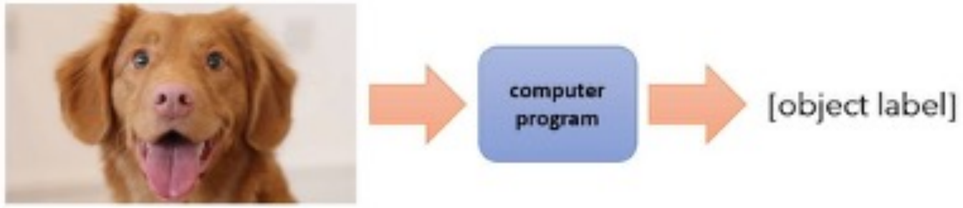
- Supervised learning:
  - Ground truth is known in advance.
  - Training data are usually static and iid.
- Reinforcement learning:
  - The best action (policy) is usually unknown a priori.
  - Sequence of actions is needed.
  - A series of trial and error (search) is performed.
    - Usually delayed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





What is Reinforcement Learning?

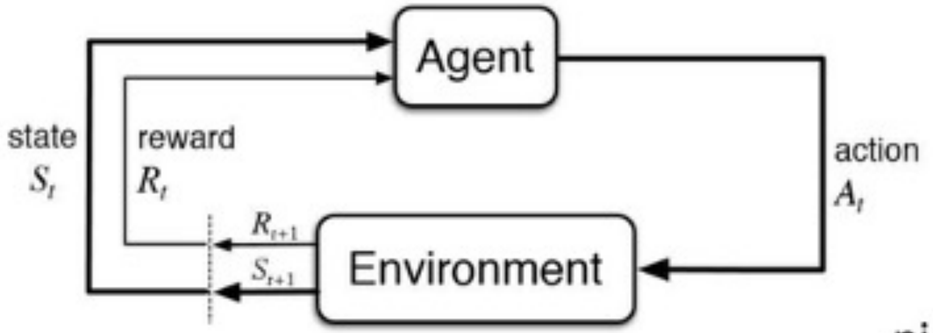
supervised learning



$$\text{input: } \mathbf{x}$$
$$\text{output: } \mathbf{y}$$
$$\text{data: } \mathcal{D} = \{(\mathbf{x}_i, \mathbf{y}_i)\}$$
$$\text{goal: } f_\theta(\mathbf{x}_i) \approx \mathbf{y}_i$$

someone gives this to you

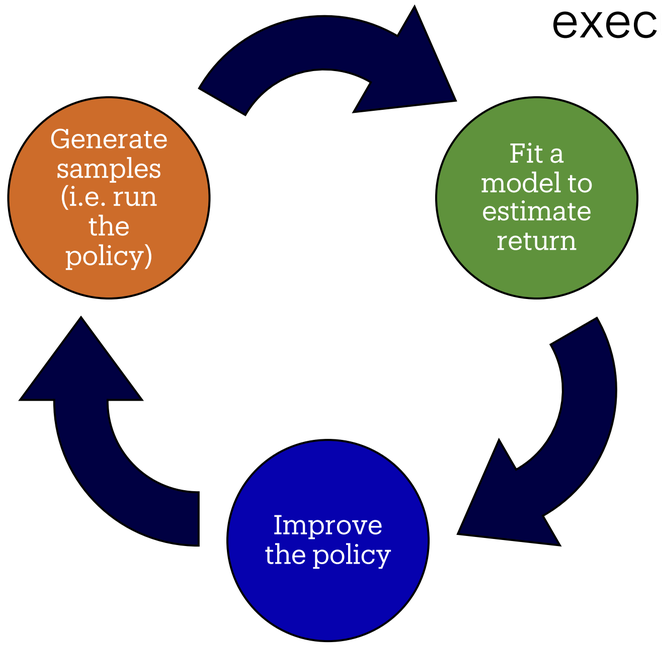
reinforcement learning



pick your own actions

$$\text{input: } \mathbf{s}_t \textit{ at each time step}$$
$$\text{output: } \mathbf{a}_t \textit{ at each time step}$$
$$\text{data: } (\mathbf{s}_1, \mathbf{a}_1, r_1, \ldots, \mathbf{s}_T, \mathbf{a}_T, r_T)$$
$$\text{goal: learn } \pi_\theta : \mathbf{s}_t \rightarrow \mathbf{a}_t$$
$$\text{to maximize } \textstyle\sum_t r_t$$

The Anatomy of Reinforcement Learning



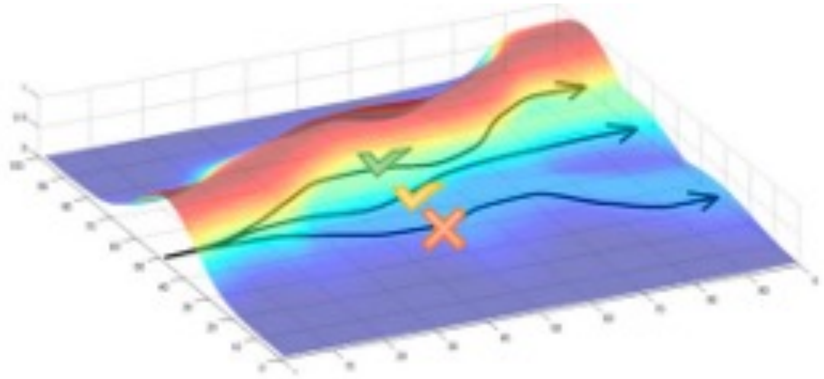
gripping objects on random or else “best guess” locations

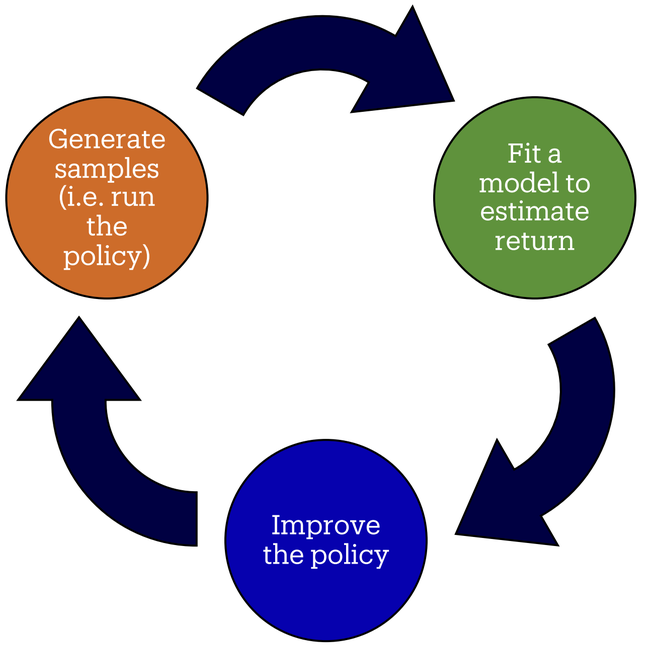
how well was our choice of (x, y, z); execute gripping!

or … when I grip some object on a specific location, what happens physically; i.e. how its (x, y, z)’s change?

improve our “best guess”

A Simple Example

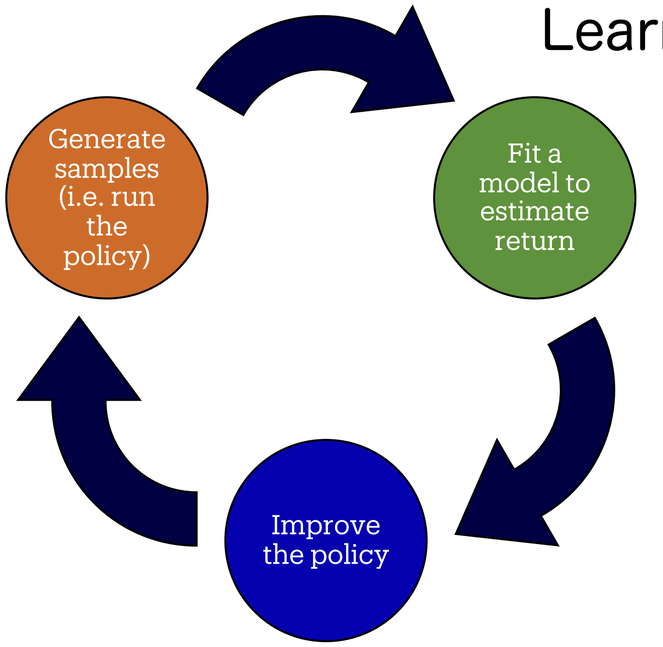




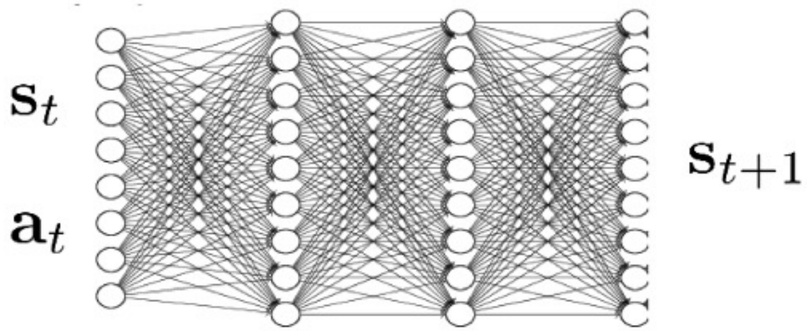
$$J(\theta) = E_\pi \left[\sum_t r_t \right] \approx \frac{1}{N} \sum_{i=1}^{N} \sum_t r_t^i$$

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

Another Example



Learn f_ϕ such that $s_t + 1 \approx f_\phi(s_t, a_t)$

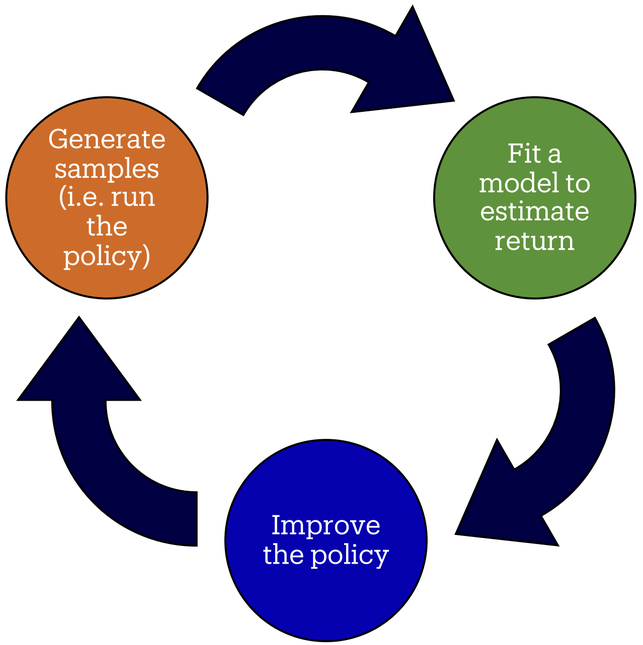


Backprop through f_ϕ and r to train $\pi_\theta(s_t) = a_t$

Which parts are expensive?

Real robot/car/power/gred/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

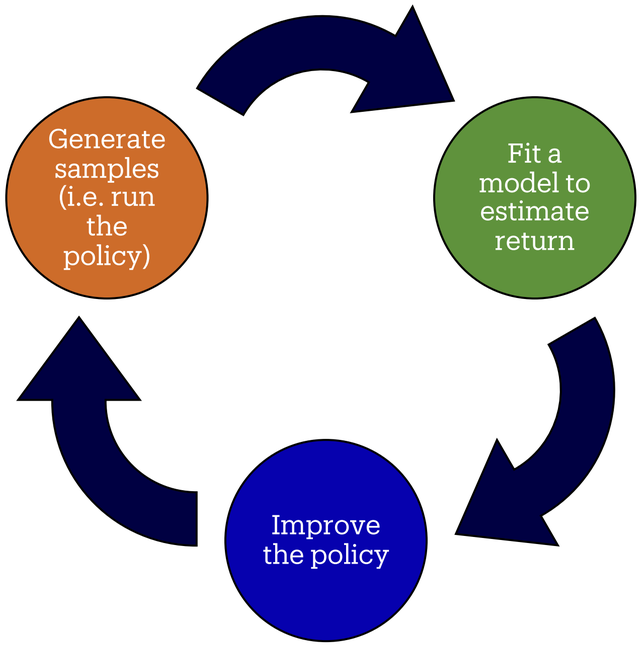
learn $\mathbf{s}_{t+1} \approx f_\phi(\mathbf{s}_t, \mathbf{a}_t)$

Expensive

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

Backprop through f_ϕ and r to train $\pi_\theta(s_t) = a_t$

Model-based RL



Learn $p(s_{t+1}|s_t, a_t)$

A few options

Value-based RL

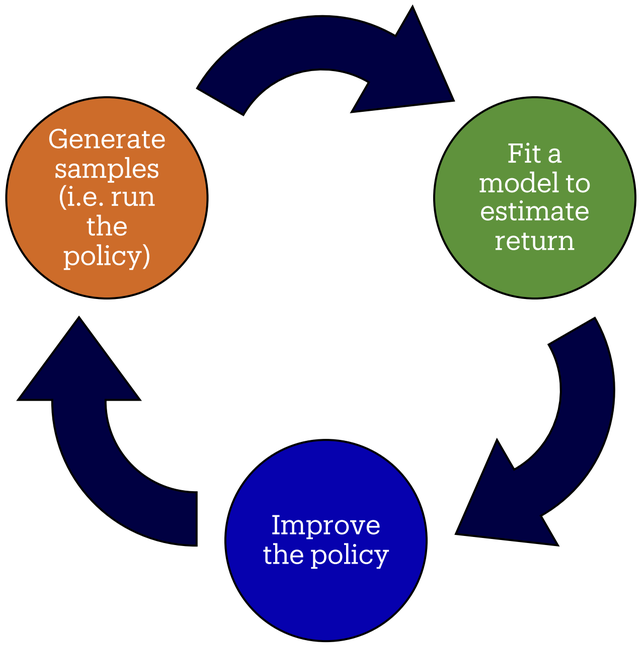
$$Q^{\pi}(\mathbf{s}_t, \mathbf{a}_t) = \sum_{t'=t}^{T} E_{\pi_\theta}[r(\mathbf{s}_{t'}, \mathbf{a}_{t'})|\mathbf{s}_t, \mathbf{a}_t]: \textit{total reward from taking} \mathbf{a}_t \textit{in} \mathbf{s}_t$$

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

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

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

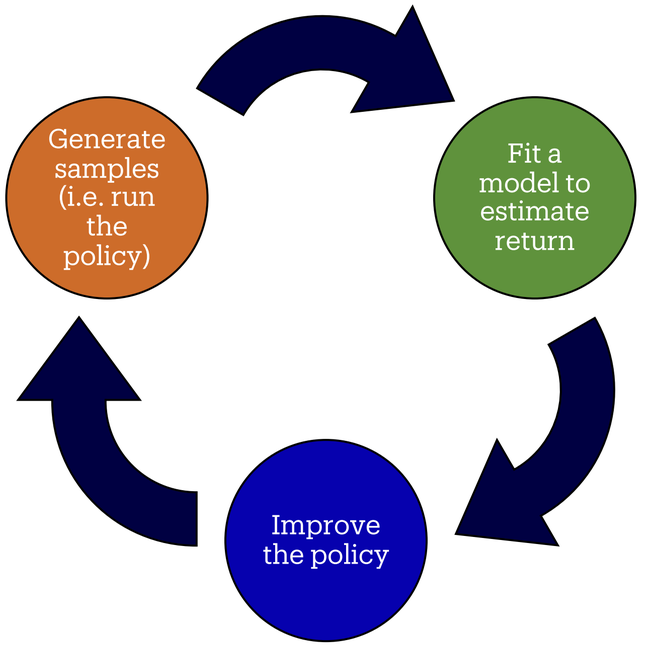
Value-based RL (cont.)



Fit $V(s) or Q(s,a)$

Set $\pi(s) = argmax_a Q(s,a)$

Direct Policy Gradient

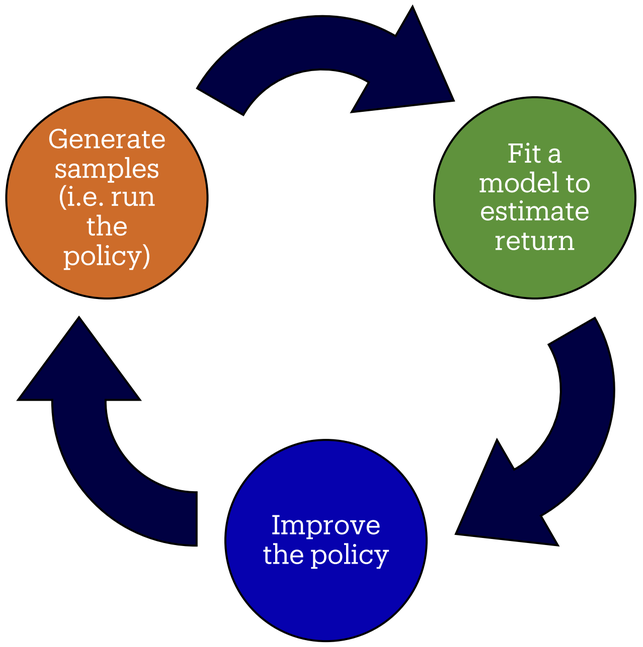


Evaluate returns

$R_\tau = \sum_t r(\mathbf{s}_t, \mathbf{a}_t)$

$$\theta \leftarrow \theta + \alpha \nabla_\theta E\left[\sum_t r(\mathbf{s}_t, \mathbf{a}_t)\right]$$

Actor-critic: value functions + policy gradients



Fit $V(s) or Q(s,a)$

$$\theta \leftarrow \theta + \alpha \nabla_\theta E[Q(s_t, a_t)]$$

Where do rewards come form?

- An expert gives us the reward
- Learning from demonstrations
  - Directly copying observed behavior
  - Inferring rewards from observed behavior (inverse reinforcement learning)

Motivation (cont.)

- 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

	AI Planing	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	

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

+ learns about the real world from data

- doesn't try to do **better** than the data

Reinforcement Learning

+ optimizes a goal with emergent behavior

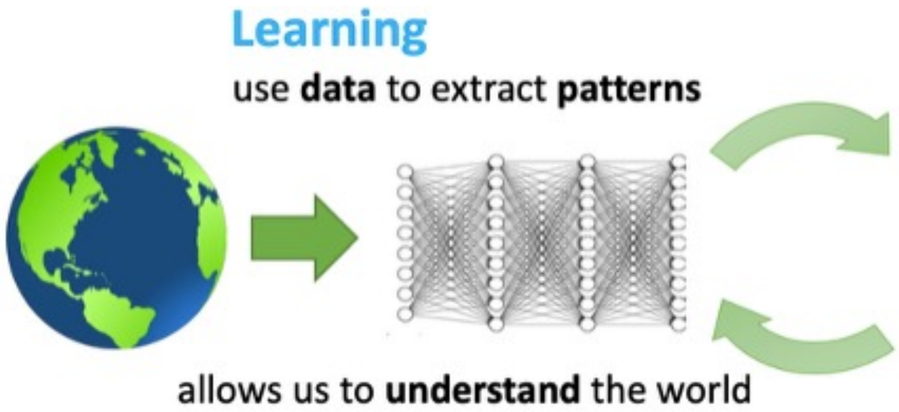
- but need to figure out how to use at scale!

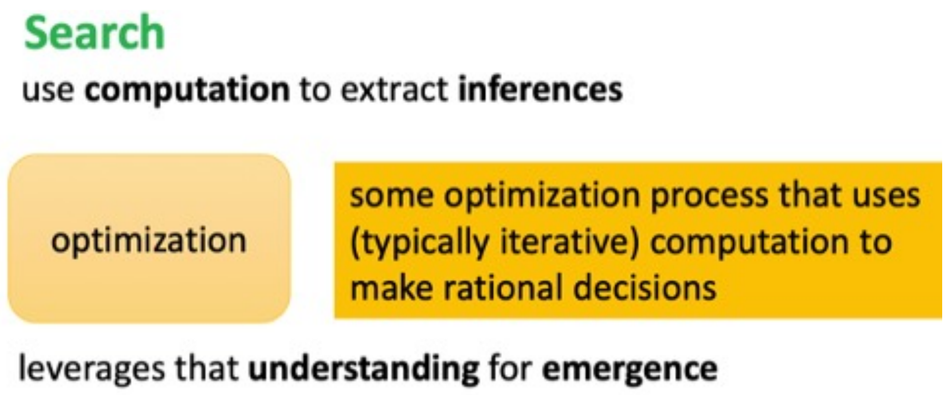
Data without **optimization** doesn't allow us to solve new problems in new ways

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/IncIdeas/BitterLesson.html





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

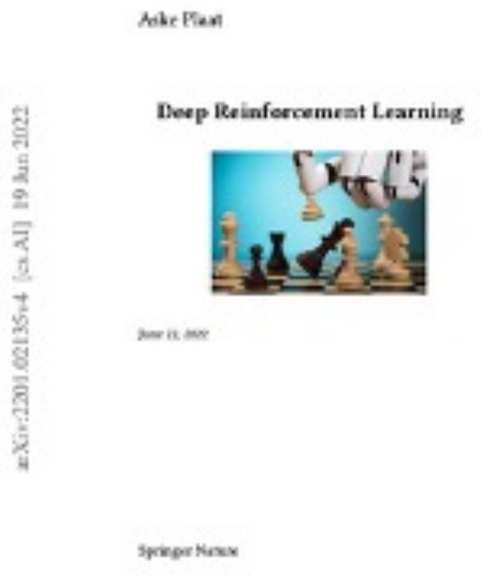
Optimization without **data** is hard to apply to the real world outside of simulators

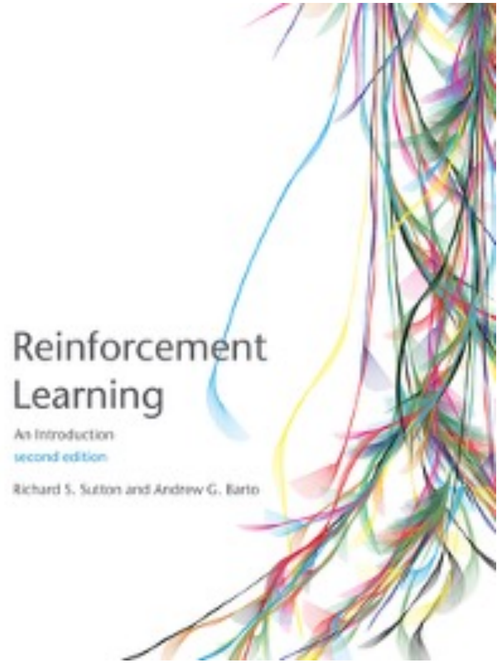
Superintelligence

- 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.





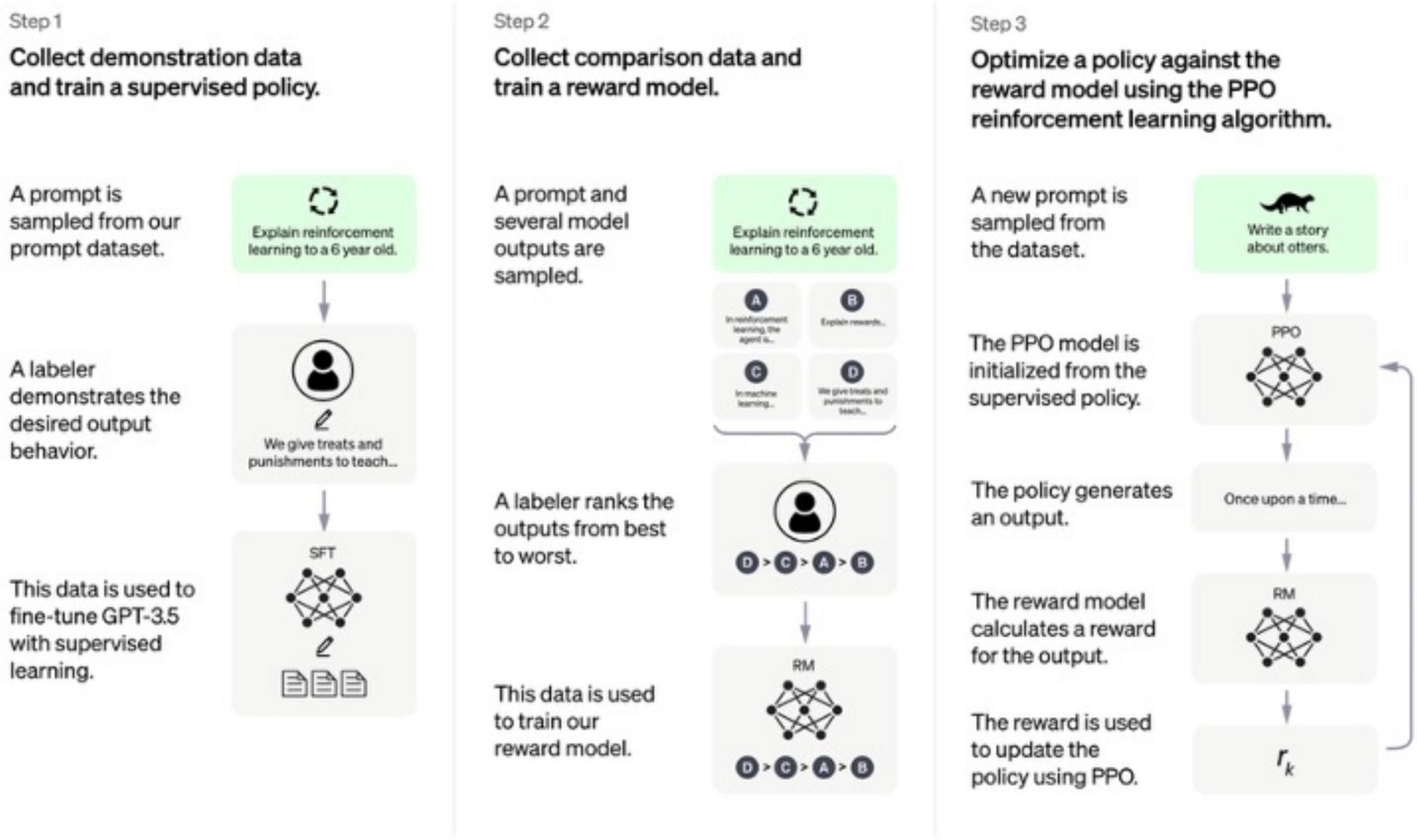
Teaching Assistants

- Ali Najar (Head TA)
- Pouya Toroghi (Head TA)

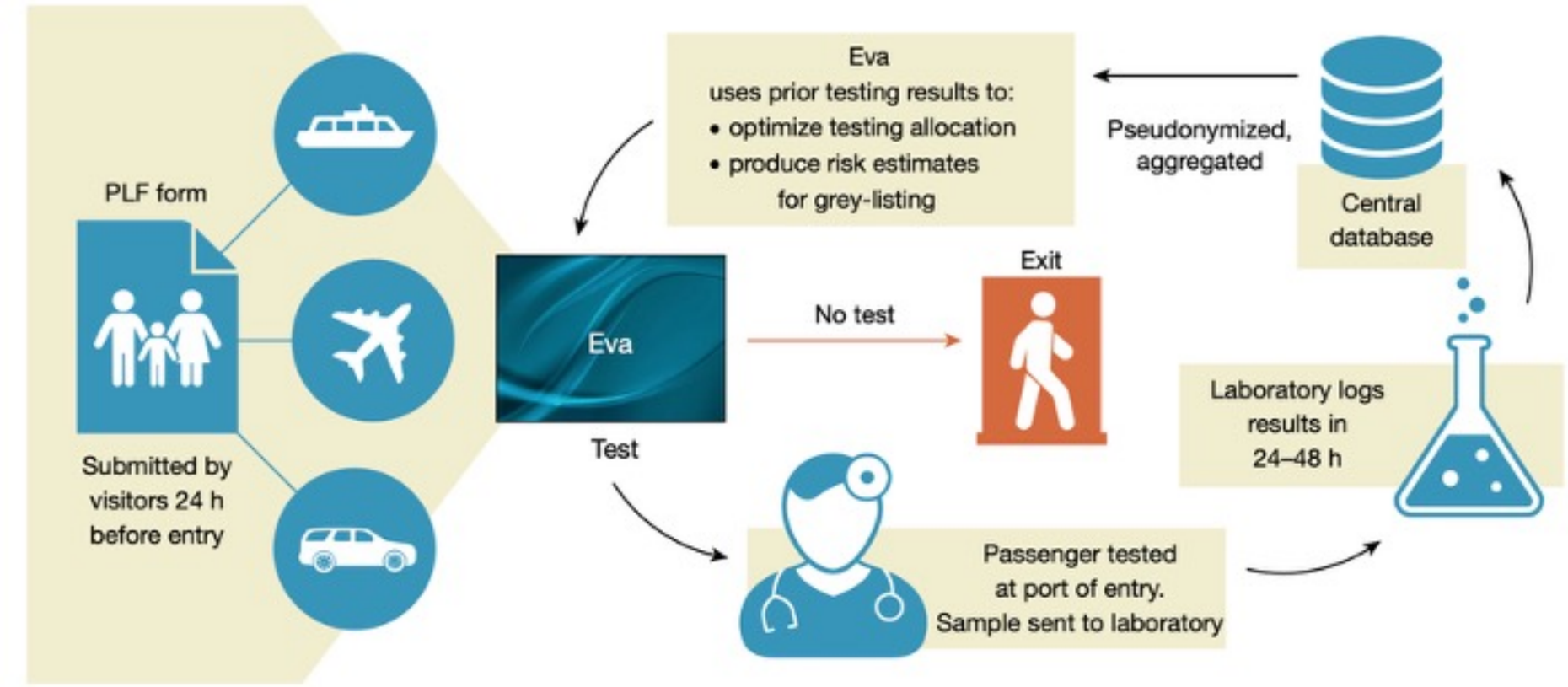
Prereqs

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

Motivation (cont.) ChatGPT; Why RL?!



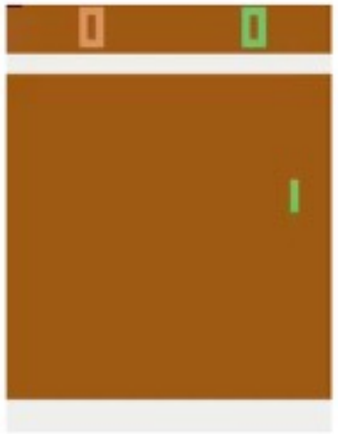
Motivation (cont.)



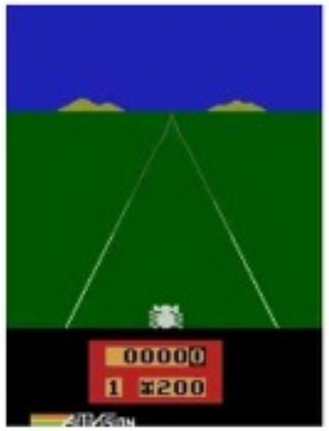
https://www.nature.com/articles/s41586-021-04014-z

History

2013 — **Atari (DQN) [Deepmind]**

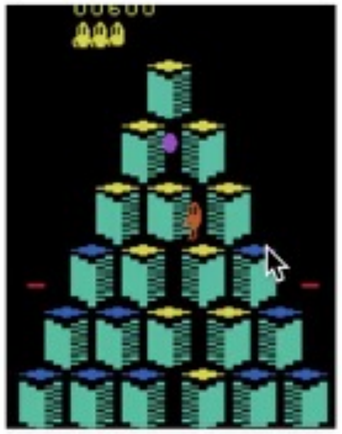


Pong



Enduro

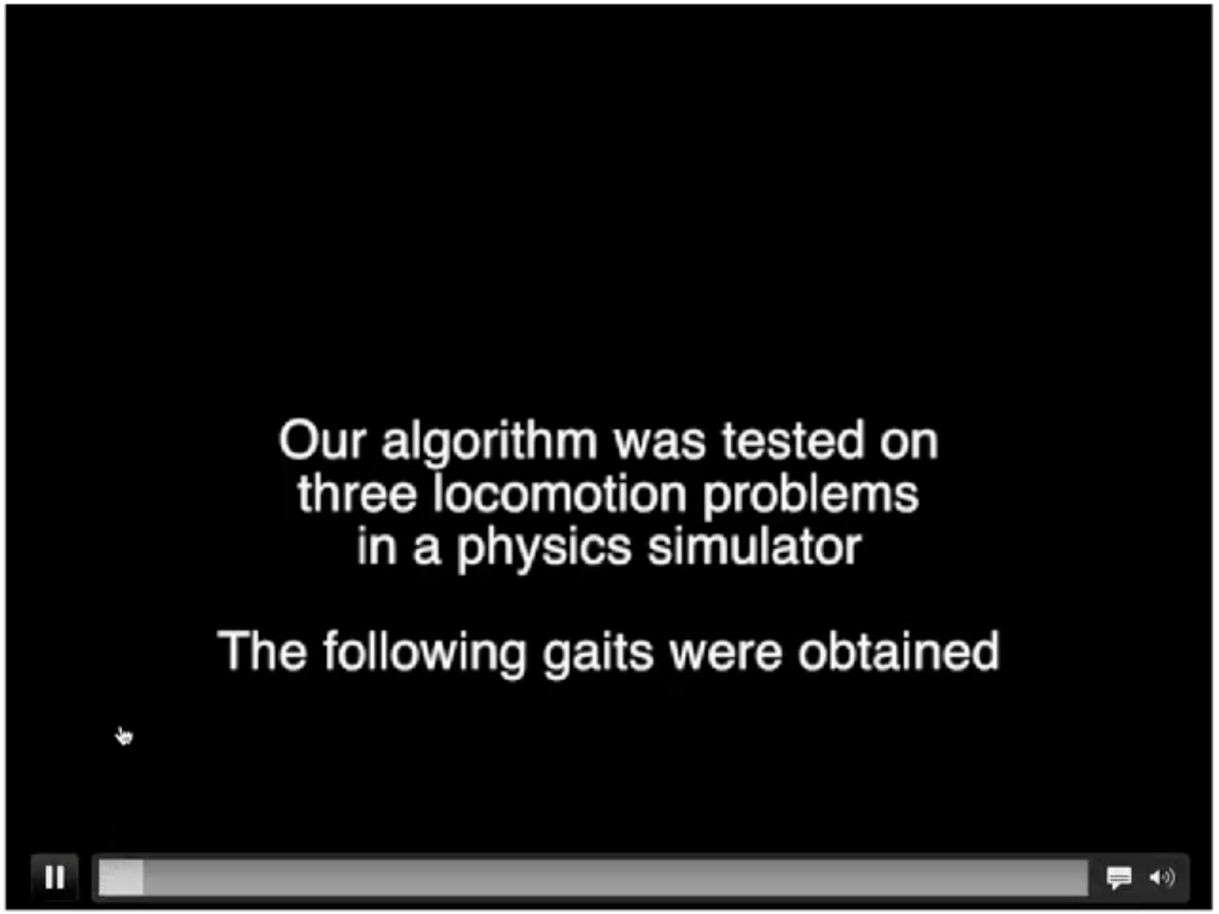
Beamrider



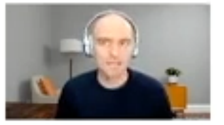
Q*bert

A Few Deep RL Highlights

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



Play 0:06 – 0:25



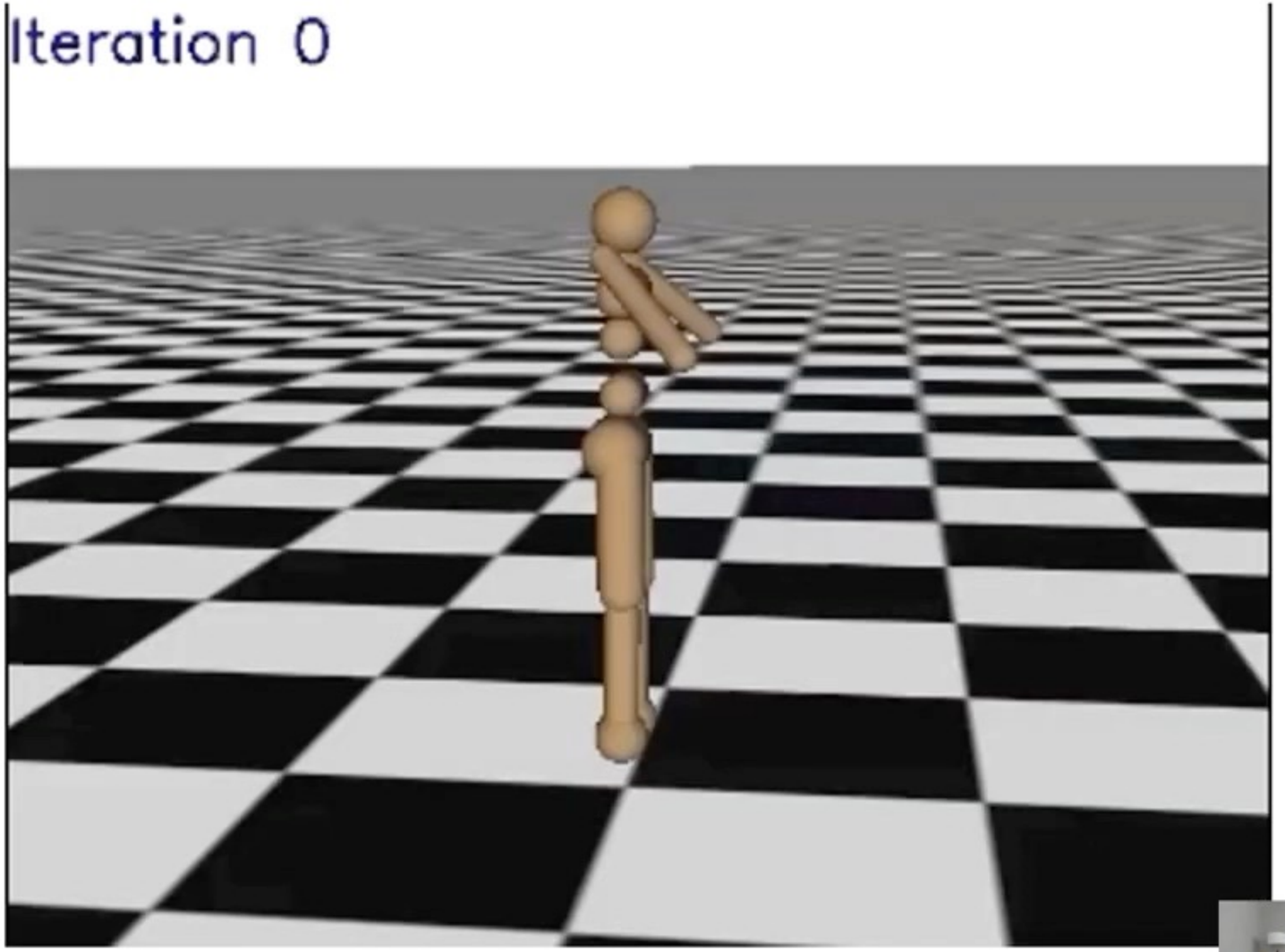
History

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

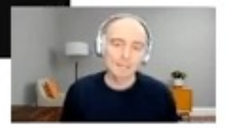
Tian et al, 2016; Maddison et al, 2014; Clark et al, 2015

A Few Deep RL Highlights

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



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



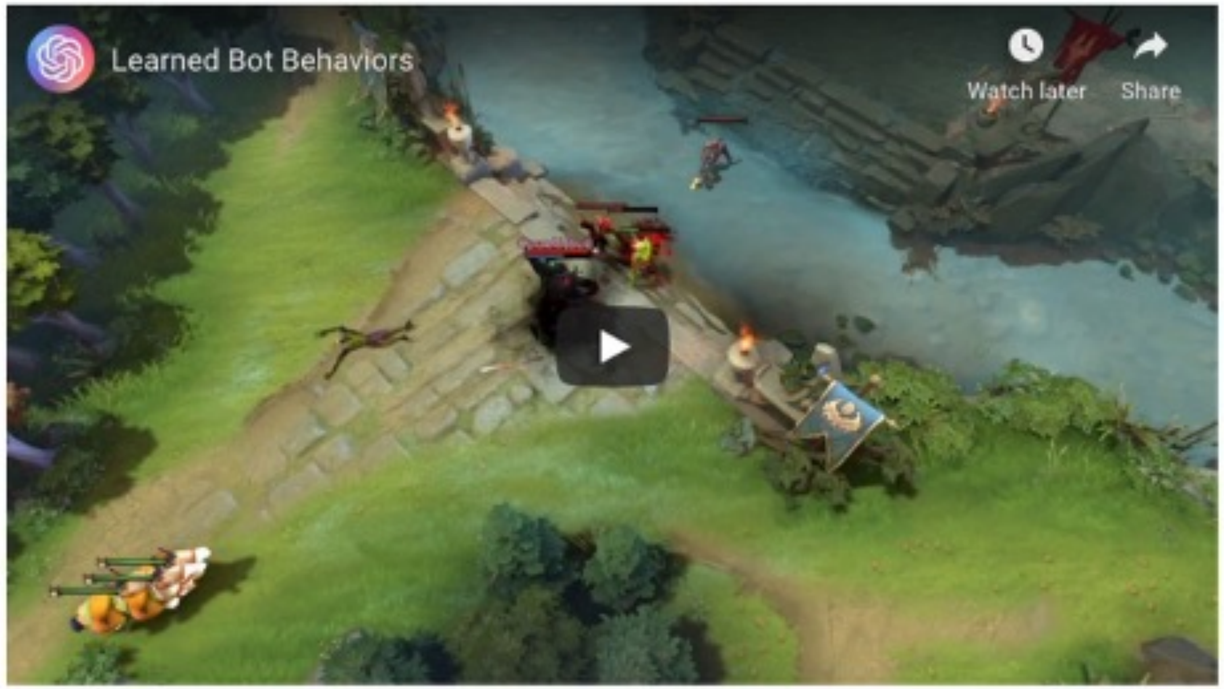
A Few Deep RL Highlights

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

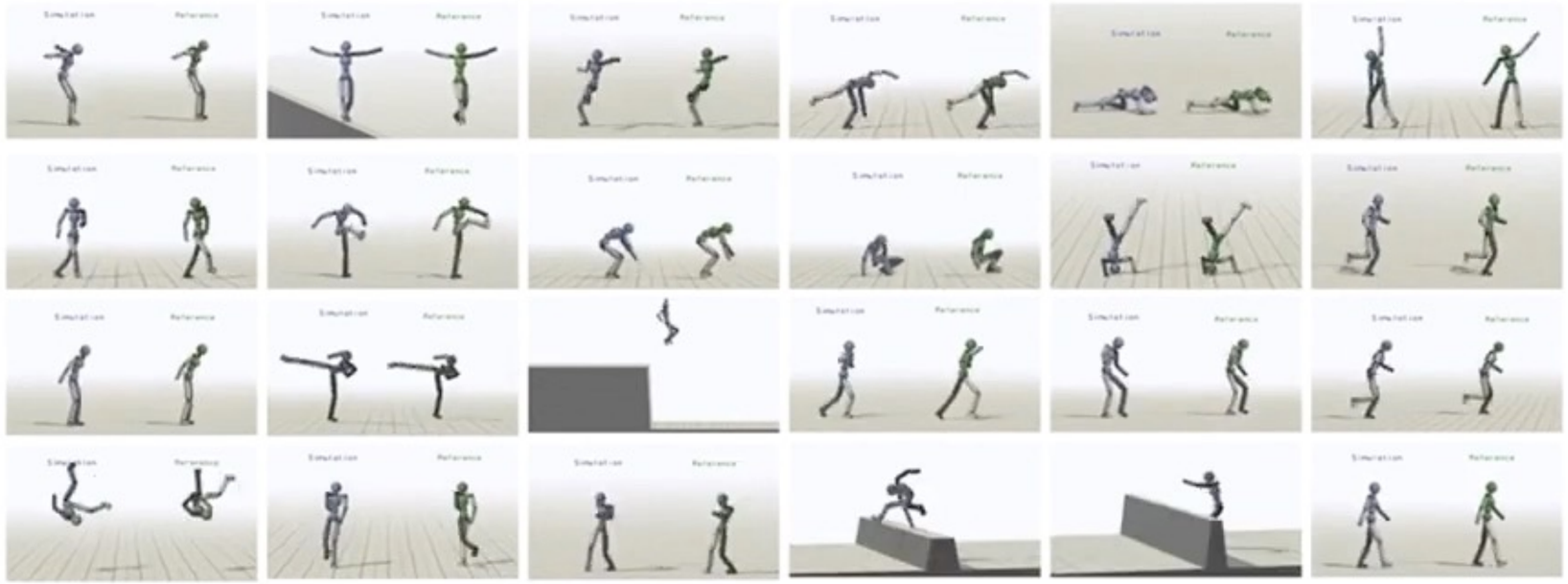
2013	Atari (DQN) [Deepmind]
2014	2D locomotion (TRPO) [Berkeley]
2015	AlphaGo [Deepmind]
2016	3D locomotion (TRPO+GAE) [Berkeley]
2016	Real Robot Manipulation (GPS) [Berkeley, Google]
2017	**Dota2 (PPO) [OpenAI]**



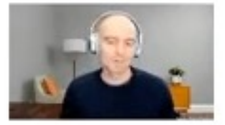
OpenAI Dota Bot beat best humans 1:1 (Aug 2018)

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]

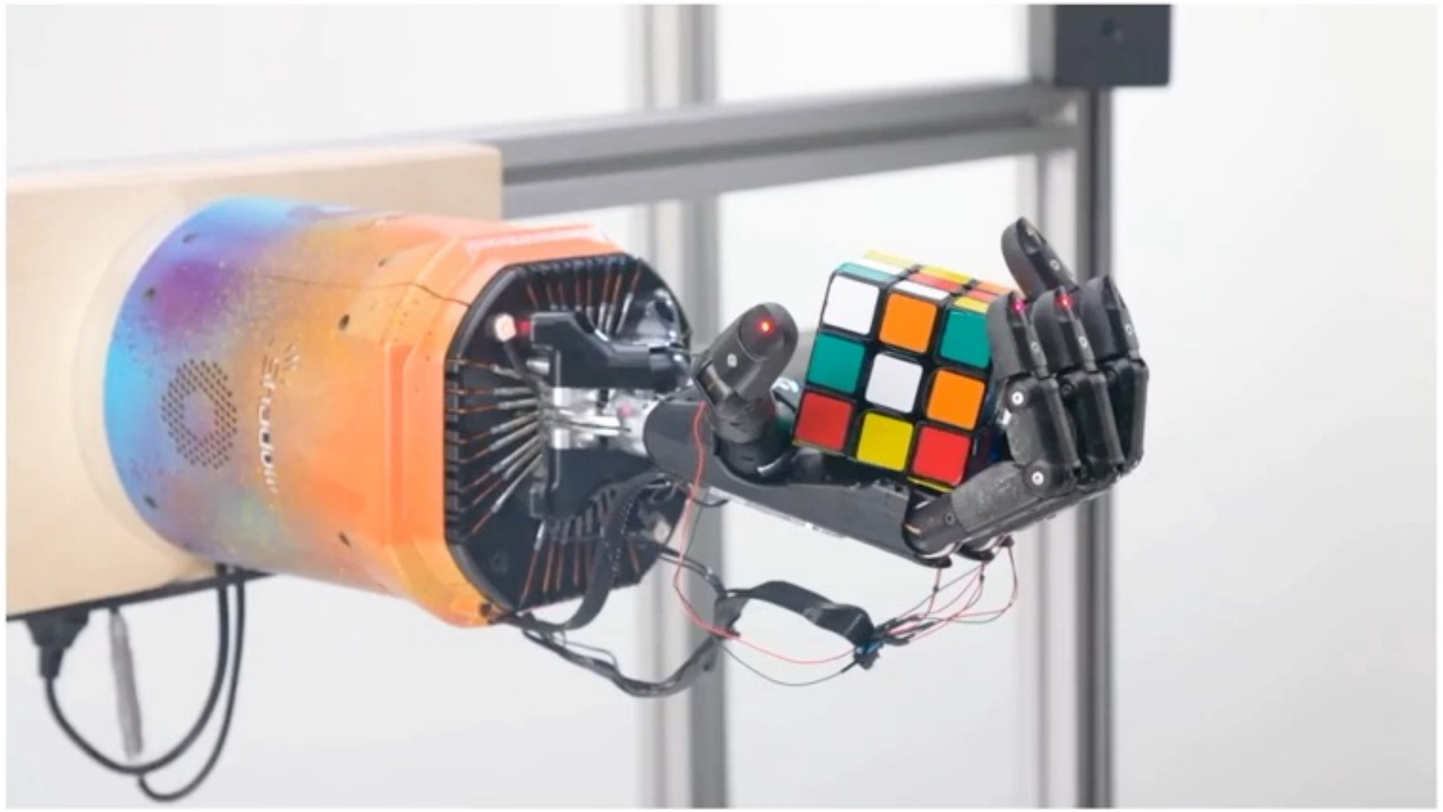


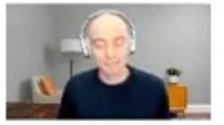
History

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]**

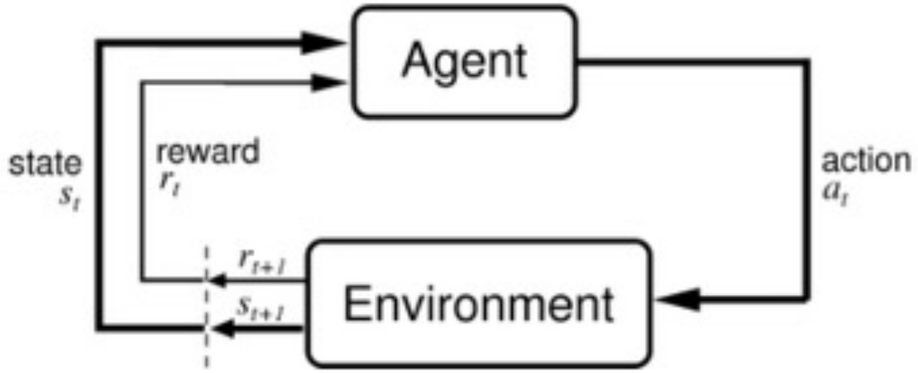




Let's Begin: Markov Decision Processes (MDPs)

An MDP is defined by:

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



The Goal

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

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

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

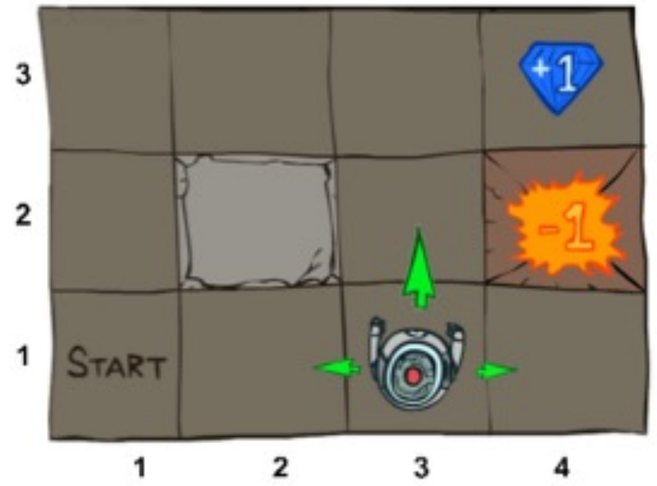
Sometimes the policy could be stochastic: $\pi: S \times A \to [0,1]$, which is

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

Example: Grid World

An MDP is defined by:

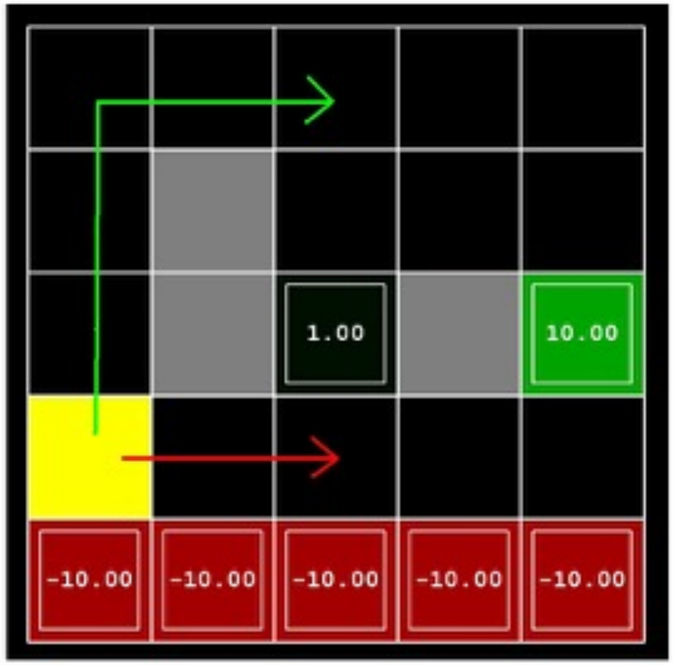
- Set of states S
- Set of actions A
- Transition function $P(s' \mid s, a)$
- Reward function $R(s, a, s')$
- Start state s_0
- Discount factor γ
- Horizon H



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

π:

Exercise



(a) Prefer the close exit (+1), risking the cliff (-10)

(b) Prefer the close exit (+1), but avoiding the cliff (-10)

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

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

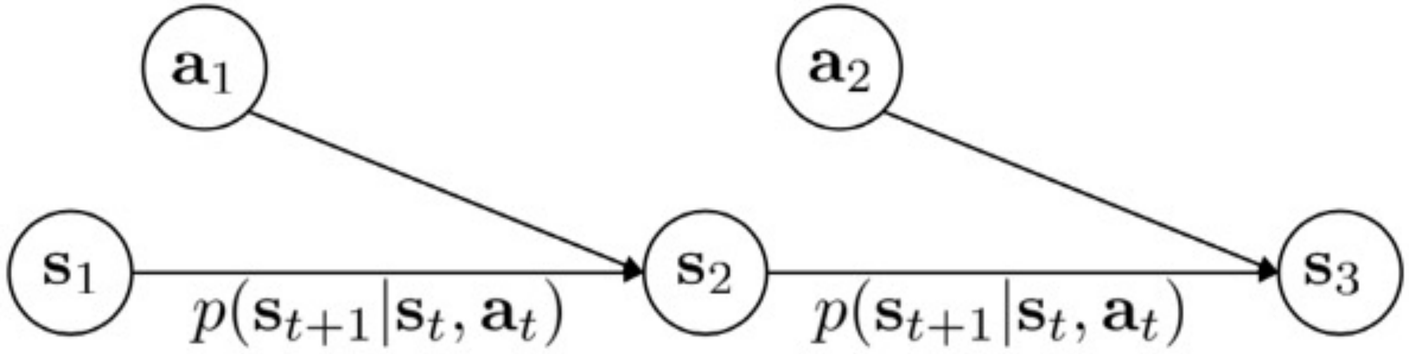
(1) $\gamma = 0.1$, noise = 0.5

(2) $\gamma = 0.99$, noise = 0

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

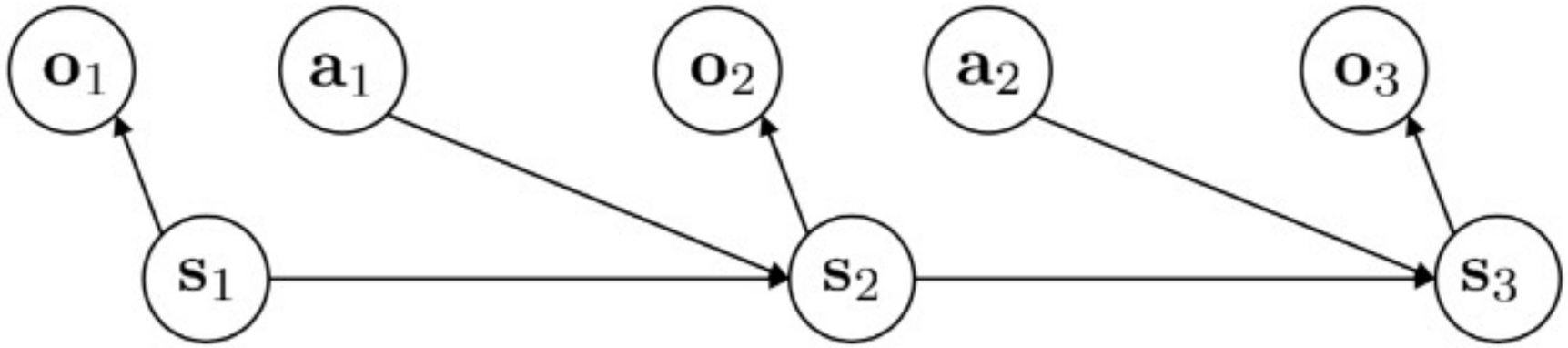
(4) $\gamma = 0.1$, noise = 0

Graphical Model of MDPs

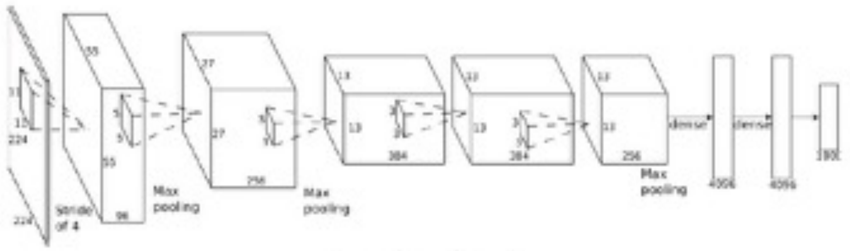


Partially Observable MDPs (POMDPs)

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



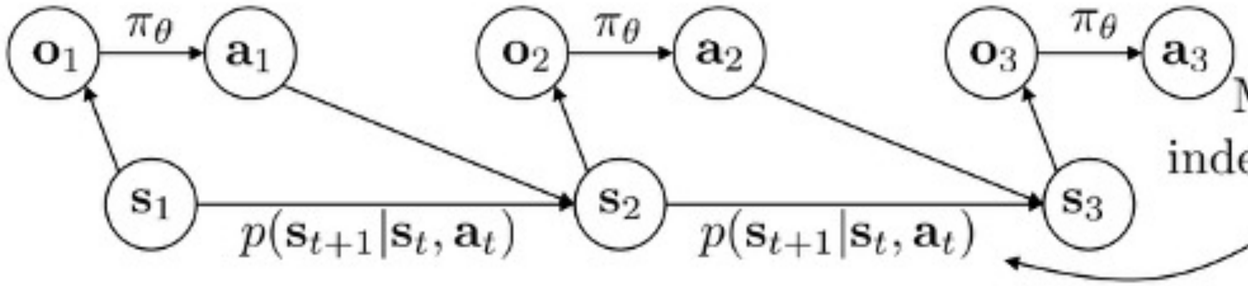
Policy as a function of S_t or O_t



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

$\mathbf{s}_t$ – 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)



Markov property
independent of $\mathbf{s}_{t-1}$

Optimal Value Function

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

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

= 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

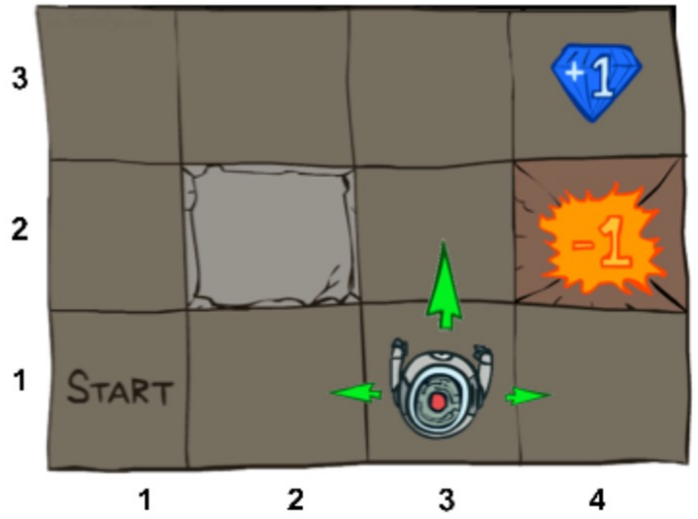
$V^*(4,3) =$

$V^*(3,3) =$

$V^*(2,3) =$

$V^*(1,1) =$

$V^*(4,2) =$



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

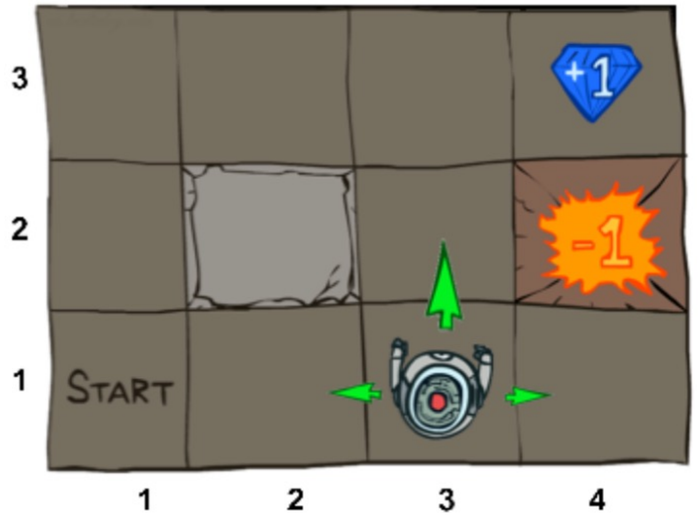
$V^*(4,3) =$

$V^*(3,3) =$

$V^*(2,3) =$

$V^*(1,1) =$

$V^*(4,2) =$



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 with probability 0.8, gamma = 0.9, H=100

$V^*(4,3) =$

$V^*(3,3) =$

$V^*(2,3) =$

$V^*(1,1) =$

$V^*(4,2) =$