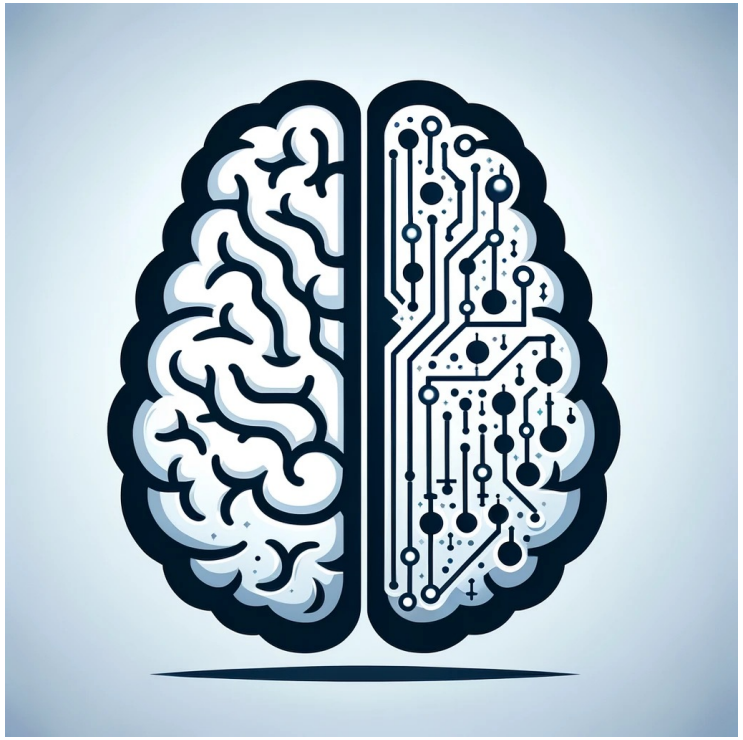
EN 601.473/601.673: Cognitive Artificial Intelligence (CogAI)



Lecture 20: Theory of Mind

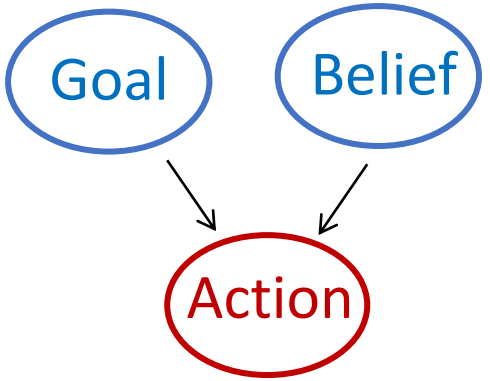
Tianmin Shu

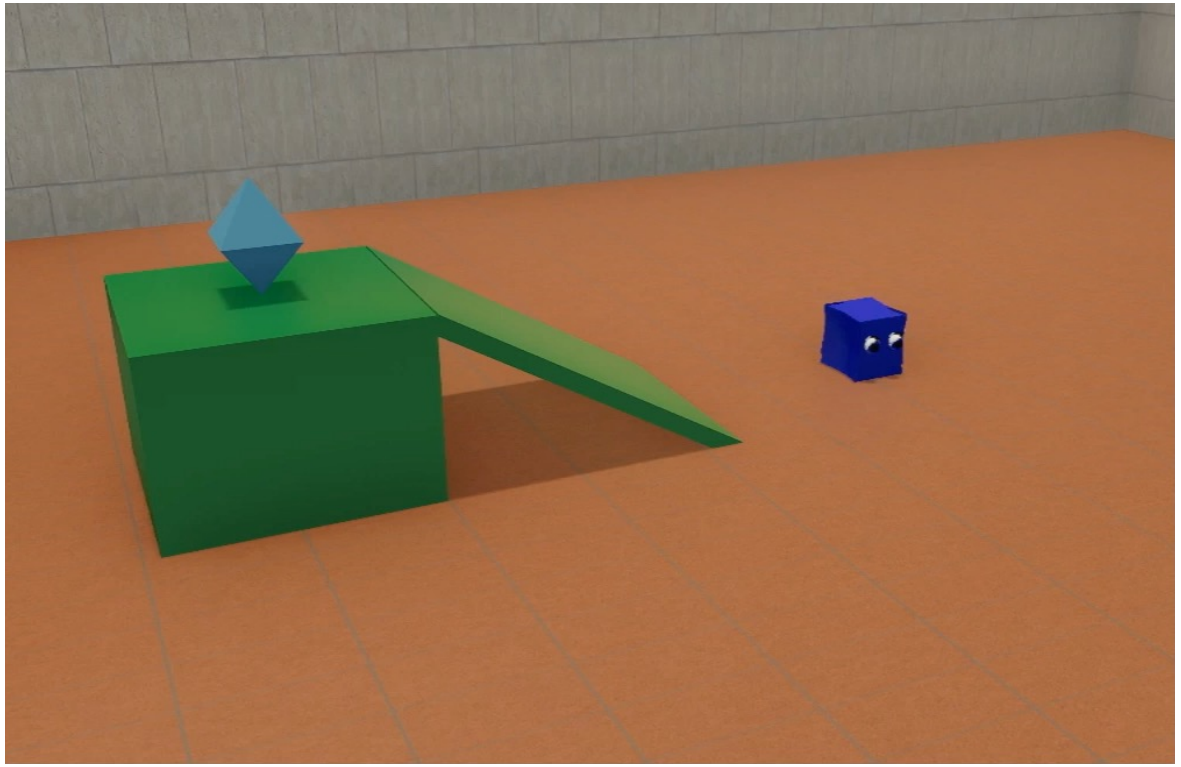
Theory of Mind

- Basic concepts and evaluation
- Single-agent decision making
- Inverse single-agent decision making

Theory of Mind in humans

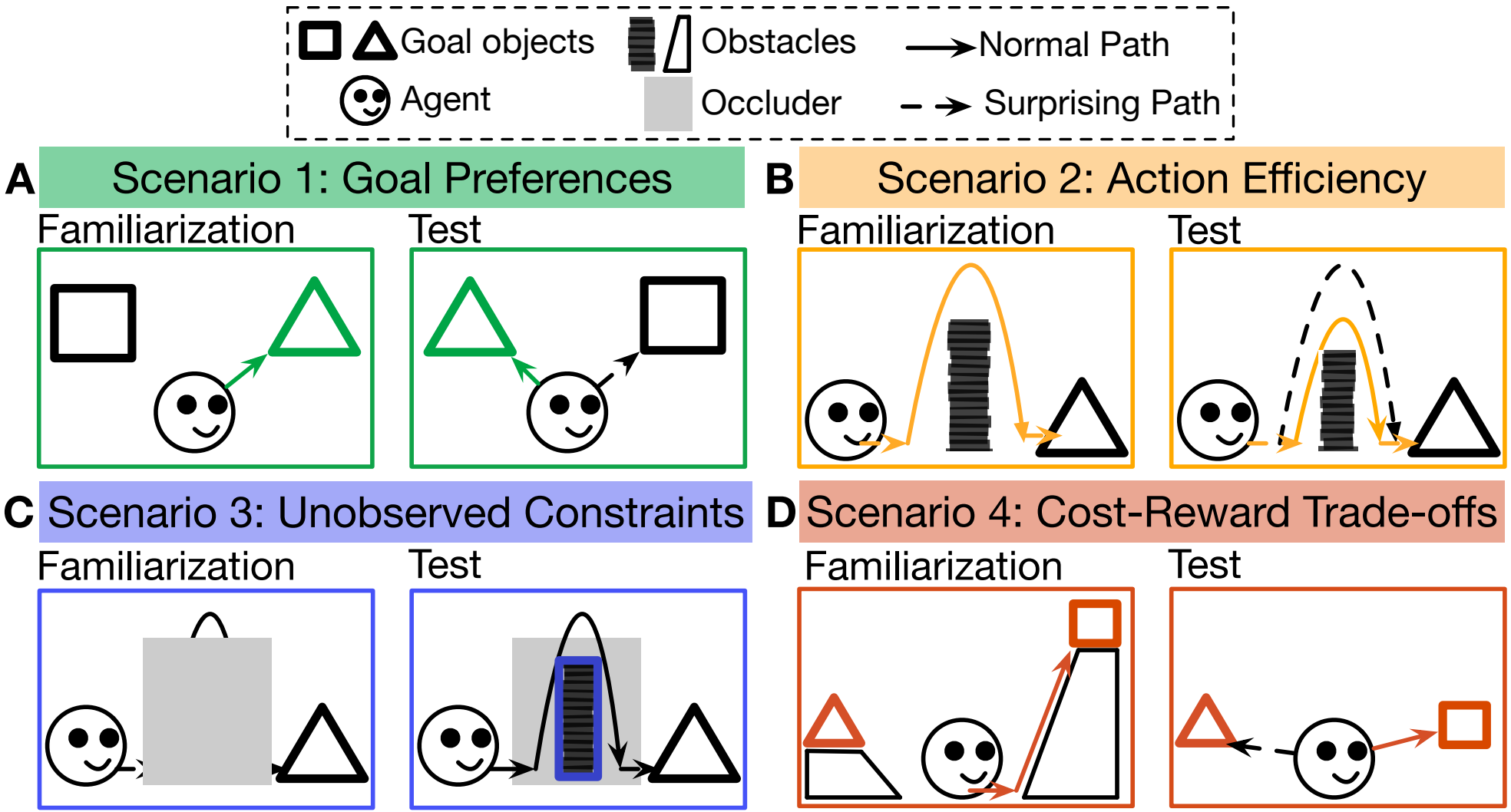
Theory of Mind: Reasoning about **hidden mental** variables that drive **observable actions**





The agent takes the most ***efficient action*** to reach its ***goal*** (blue diamond), maximizing its ***reward*** and minimizing the ***cost,*** subject to ***constraints***

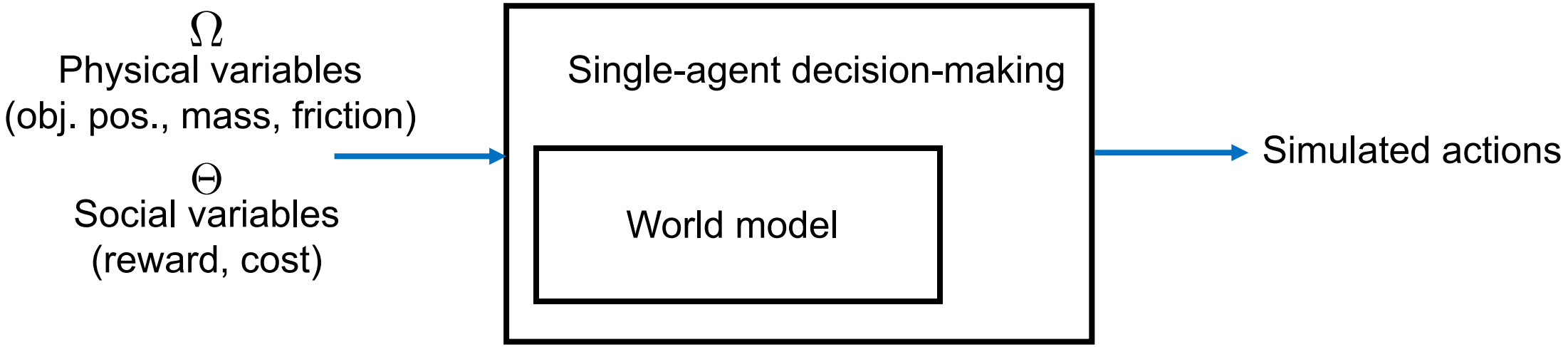
AGENT: Action, Goal, Efficiency, coNstraint, uTility



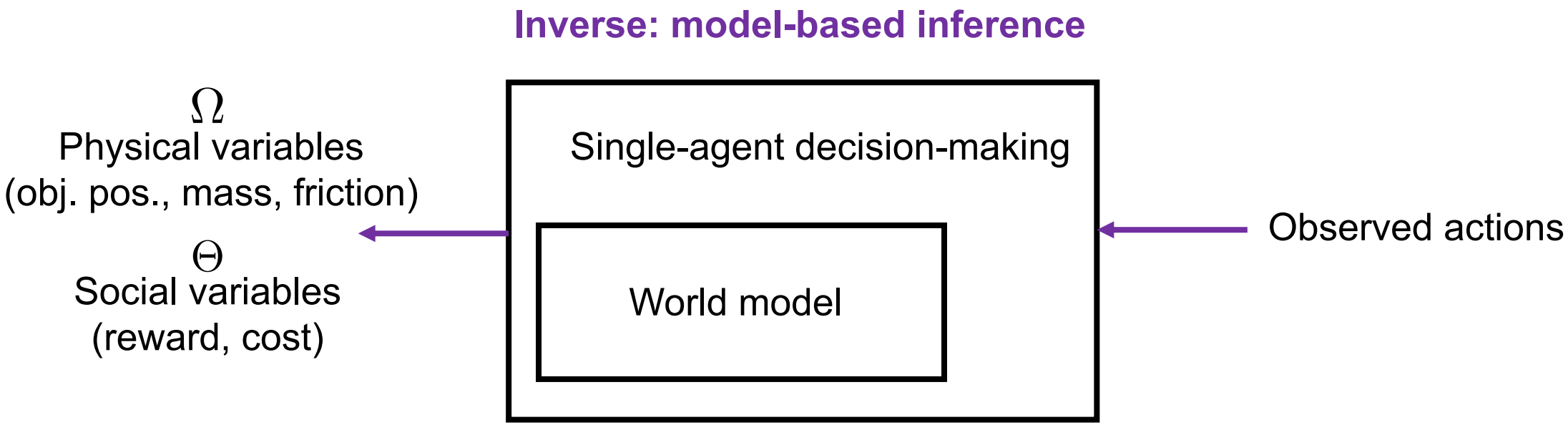
Shu et al. AGENT: A Benchmark for Core Psychological Reasoning. *ICML 2021*

A generative model of single-agent behavior

Forward: simulating human-like behavior



A generative model of single-agent behavior



Theory of Mind

- Basic concepts and evaluation
- Single-agent decision making
- Inverse single-agent decision making

Single-agent decision making

- Markov decision process (MDP)
- Value, policy
- Model-based approaches for solving MDPs
- Model-free RL (Q-learning)

Model-based approaches

- Assume that we know the state transition probabilities (i.e., the world model)
- Policy improvement
- Policy iteration
- Value iteration

Policy improvement theorem

- V(s) tells us how good it is to following current policy from s. Can we do better with a new policy? I.e., take a different action at s.

$$q^{\pi}(s,a) = E[r_{t+1} + \gamma v^{\pi}(s_{t+1}) | s_t = s, a_t = a]$$
$$= \sum_{s',r} p(s', r|s, a)[r + \gamma v^{\pi}(s')]$$

- Policy improvement theorem:
- Given any pair of deterministic policies such that, for all states $s \in \mathcal{S}$

$$q^{\pi}(s, \pi'(s)) \geq v^{\pi}(s)$$

- Then the policy π' must be as good as, or better than, π. That is, it must obtain greater or equal expected return from all states $s \in \mathcal{S}$:

$$v^{\pi'}(s) \geq v^{\pi}(s)$$

Policy improvement

- Constructing a greedy policy, which meets the condition of the policy improvement theorem → as good as or better than the original policy

$$\begin{aligned}\pi'(s) &= \arg\max_a q^\pi(s,a) \\ &= \arg\max_a \sum_{s',r} p(s',r|s,a)[r+\gamma v^\pi(s')]\end{aligned}$$

Policy iteration

$$\pi_0 \xrightarrow{\text{E}} v_{\pi_0} \xrightarrow{\text{I}} \pi_1 \xrightarrow{\text{E}} v_{\pi_1} \xrightarrow{\text{I}} \pi_2 \xrightarrow{\text{E}} \cdots \xrightarrow{\text{I}} \pi_* \xrightarrow{\text{E}} v_*$$

Policy Iteration (using iterative policy evaluation) for estimating $\pi \approx \pi_*$

1. Initialization
   $V(s) \in \mathbb{R}$ and $\pi(s) \in \mathcal{A}(s)$ arbitrarily for all $s \in \mathcal{S}$; $V(\mathit{terminal}) \doteq 0$

2. Policy Evaluation
   Loop:
   $\Delta \leftarrow 0$
   Loop for each $s \in \mathcal{S}$:
   $v \leftarrow V(s)$
   $V(s) \leftarrow \sum_{s',r} p(s', r \mid s, \pi(s)) \left[r + \gamma V(s') \right]$
   $\Delta \leftarrow \max(\Delta, |v - V(s)|)$
   until $\Delta < \theta$ (a small positive number determining the accuracy of estimation)

Drawback:
Each iteration involves policy evaluation.
Do we must wait for exact convergence?

3. Policy Improvement
   policy-stable $\leftarrow$ *true*
   For each $s \in \mathcal{S}$:
   old-action $\leftarrow \pi(s)$
   $\pi(s) \leftarrow \arg\max_a \sum_{s',r} p(s', r \mid s, a) \left[r + \gamma V(s') \right]$
   If *old-action* $\neq \pi(s)$, then *policy-stable* $\leftarrow$ *false*
   If *policy-stable*, then stop and return $V \approx v_*$ and $\pi \approx \pi_*$; else go to 2

Value iteration

- Key idea: Policy evaluation is stopped after just one sweep (one update of each state) instead of wait until convergence

Value Iteration, for estimating $\pi \approx \pi_*$

Algorithm parameter: a small threshold $\theta > 0$ determining accuracy of estimation
Initialize $V(s)$, for all $s \in \mathcal{S}^+$, arbitrarily except that $V(terminal) = 0$

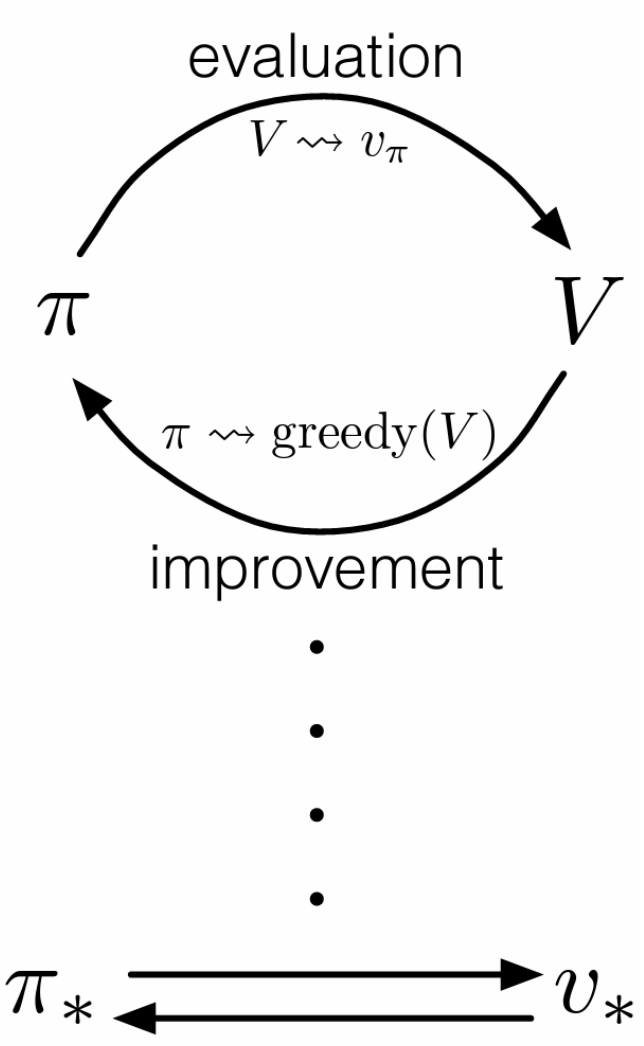
Loop:
  $\Delta \leftarrow 0$
  Loop for each $s \in \mathcal{S}$:
    $v \leftarrow V(s)$
    $V(s) \leftarrow \max_a \sum_{s',r} p(s', r|s, a)\left[r + \gamma V(s')\right]$
    $\Delta \leftarrow \max(\Delta, |v - V(s)|)$
until $\Delta < \theta$

Output a deterministic policy, $\pi \approx \pi_*$, such that
  $\pi(s) = \arg\max_a \sum_{s',r} p(s', r|s, a)\left[r + \gamma V(s')\right]$

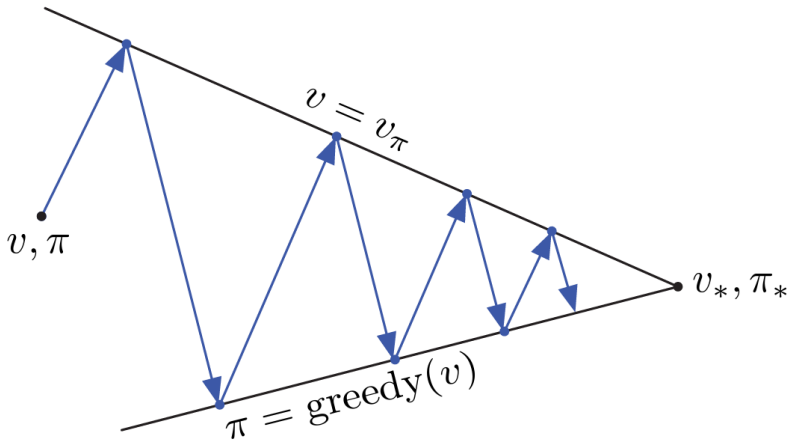
Combines the policy improvement and truncated policy evaluation steps

Generalization policy iteration (GPI)

- The general idea of letting policy-evaluation and policy-improvement process interact
- Almost all reinforcement learning methods are well described as GPI
- All have identifiable policies and value functions, with the policy always being improved with respect to value function and the value function always being driven toward the value function for the policy



Simplified 2D space



How to solve the Bellman equation for $Q^*(s, a)$

$$Q^*(s, a) = R(s, a) + \gamma \max_{a' \in \mathcal{A}} Q^*(s', a')$$

$Q^*(s_t, a_t)$

$\rightarrow Q^*(s_{t+1}, a_{t+1})$

$\rightarrow Q^*(s_{t+2}, a_{t+2})$

$\rightarrow \ldots$

Many iterative methods:

- Policy iteration
- Value iteration
- Sarsa
- Q-learning

Model-free RL: without access of the world model

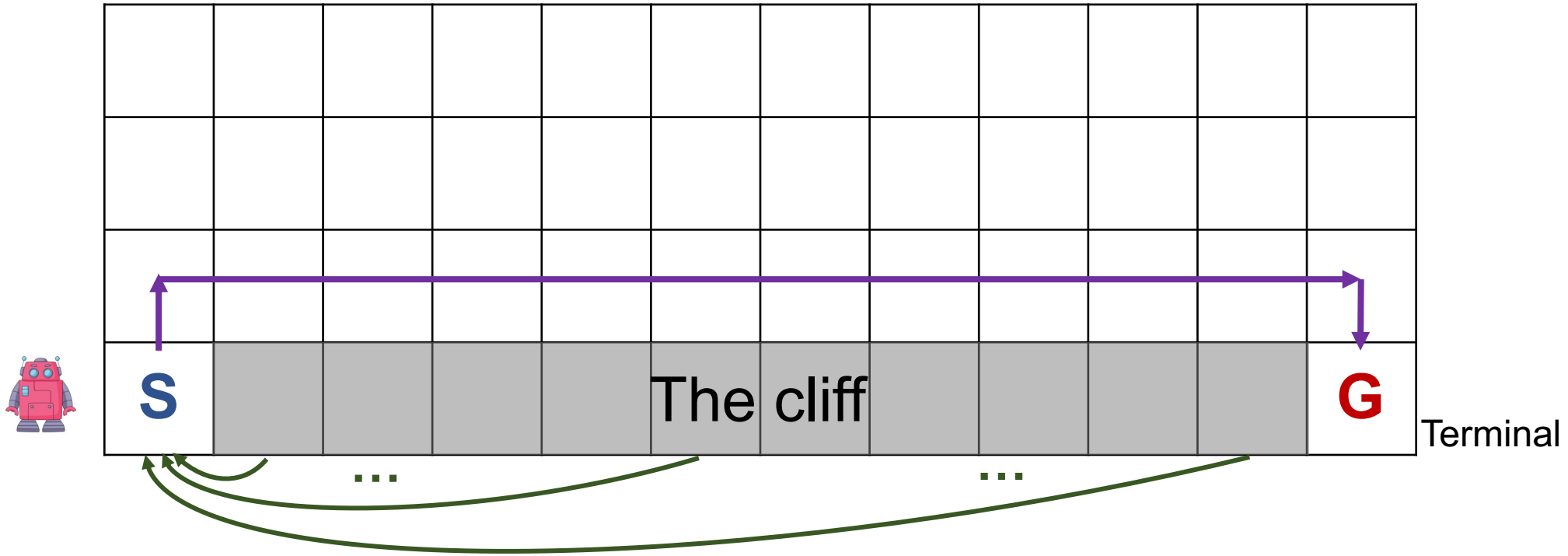
An example: cliff walk in a 2D grid word

State s: (x, y)

Action a:

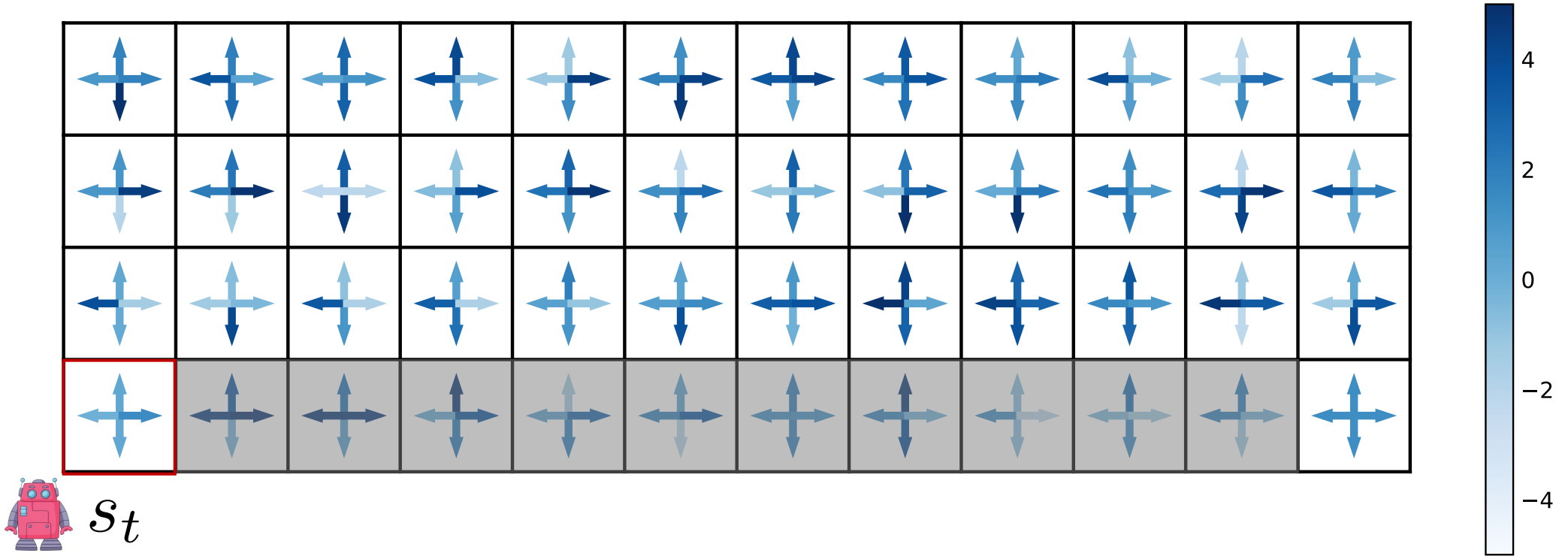
Reward $R(s,a) = \begin{cases} -10 & \text{if step into the cliff} \\ -1 & \text{otherwise} \end{cases}$

$\gamma = 0.95$

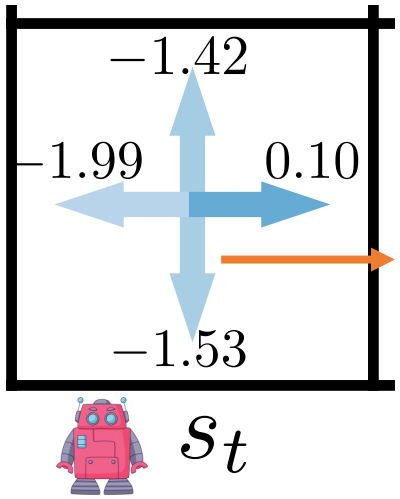


Q-learning

Start from a random estimation $Q(s,a), \forall s \in \mathcal{S}, \forall a \in \mathcal{A}$



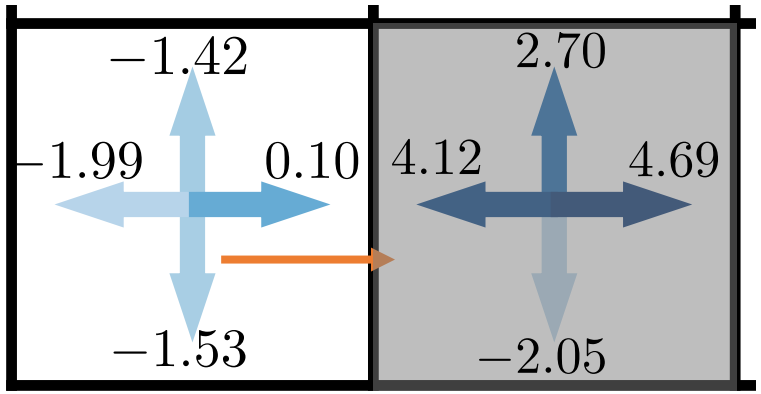
Q-learning



$$a_t = \underset{a_t}{\operatorname{argmax}}\, Q(s_t, a_t)$$

$$Q(s_t, a_t) = 0.10$$

Q-learning



② $r_t = -10$

① $a_t = \underset{a_t}{\operatorname{argmax}}\, Q(s_t, a_t)$

$Q(s_t, a_t) = \boxed{0.10}$

$$Q^{\text{new}}(s_t, a_t) = r_t + \gamma \max_{a_{t+1} \in \mathcal{A}} \boxed{Q(s_{t+1}, a_{t+1})}$$

$$= -10 + 0.95 * 4.69$$

$$= \boxed{-5.54}$$

③ $Q(s_t, a_t) = (1 - \alpha) Q(s_t, a_t) + \alpha Q^{\text{new}}(s_t, a_t)$

A better estimation than $Q(s_t, a_t)$!

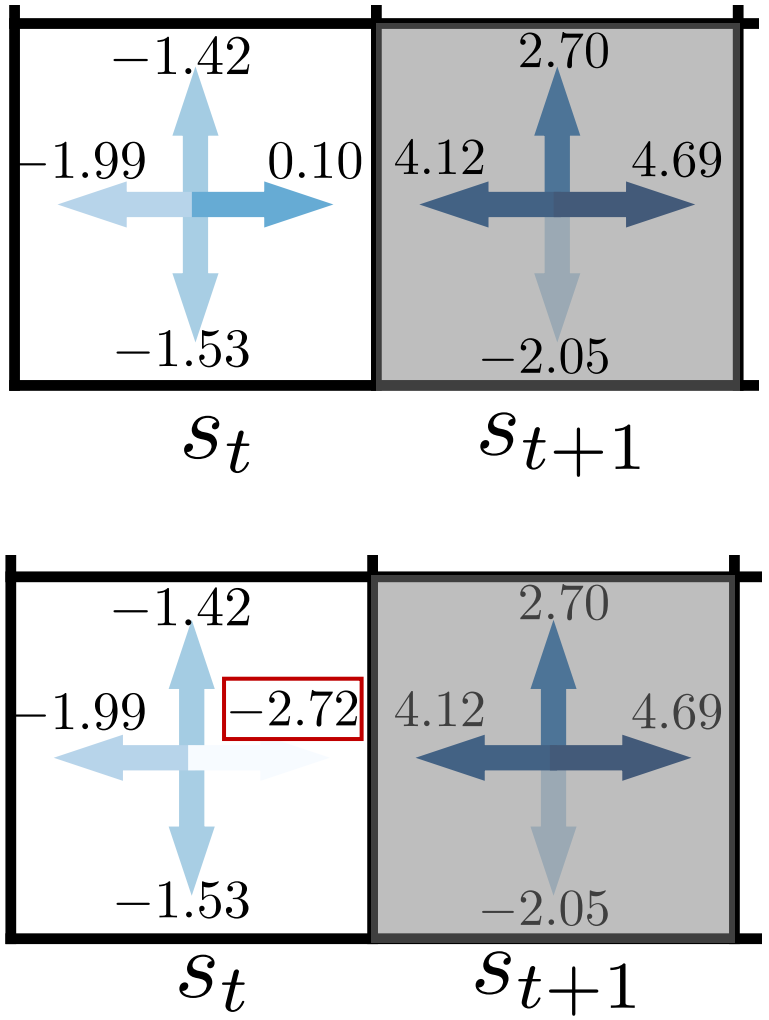
$\alpha \in (0, 1]$ Why weighted average? Q^{new} is not yet the optimal value

After one-step update

$Q(s_t, a_t) = 0.10$

$\alpha = 0.5$

$Q(s_t, a_t) = -2.72$



Pseudocode for Q-learning

Set the algorithm parameter: $\alpha \in (0, 1]$

Initialize $Q(s, a)$, for all $s \in \mathcal{S}$, $a \in \mathcal{A}$, randomly except that $Q(\text{terminal}, \cdot) = 0$

Loop for each episode:

Agent starts from the initial state S_0

Loop for each step t of episode:

Choose a_t from s_t using policy derived from $Q(s_t, a_t)$

Take action a_t, receive r_t and s_{t+1}

$$Q^{\text{new}}(s_t, a_t) = r_t + \gamma \max_{a_{t+1} \in \mathcal{A}} Q(s_{t+1}, a_{t+1})$$

$$Q(s_t, a_t) = (1 - \alpha) Q(s_t, a_t) + \alpha Q^{\text{new}}(s_t, a_t)$$

Until reach the terminal state

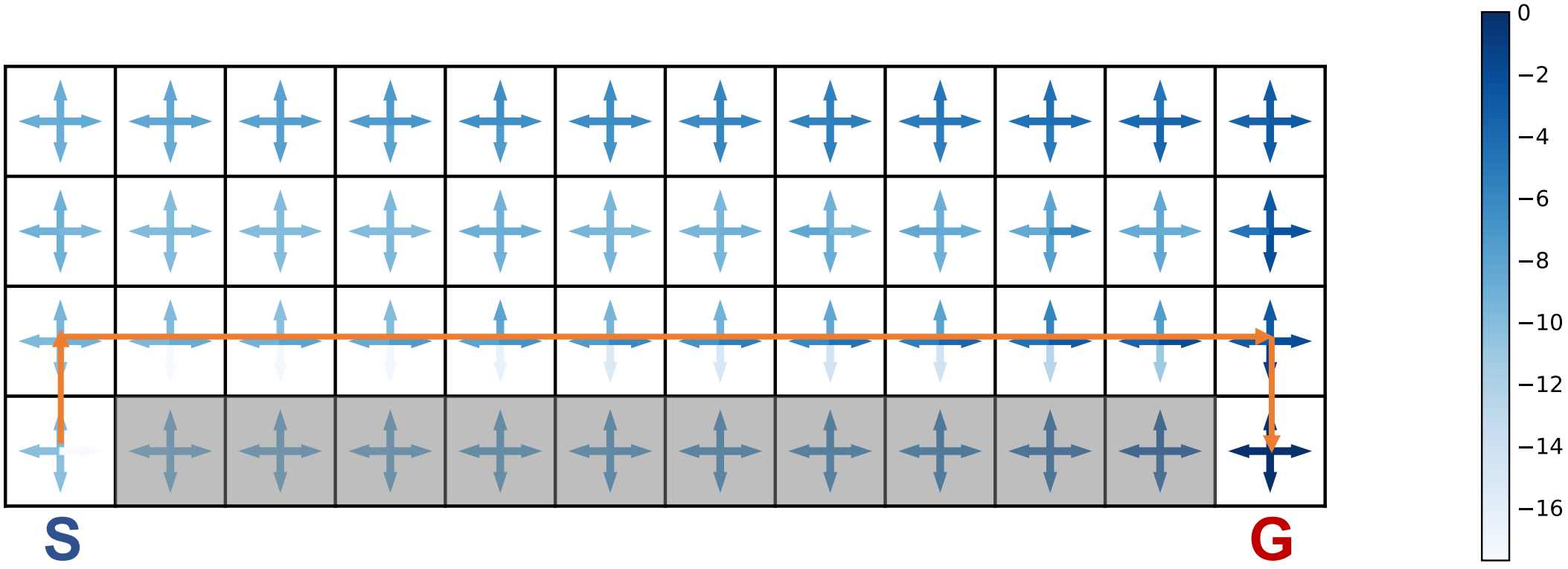
Temporal difference (TD) error

$$Q(s_t, a_t) = Q(s_t, a_t) + \alpha(Q^{\text{new}}(s_t, a_t) - Q(s_t, a_t))$$

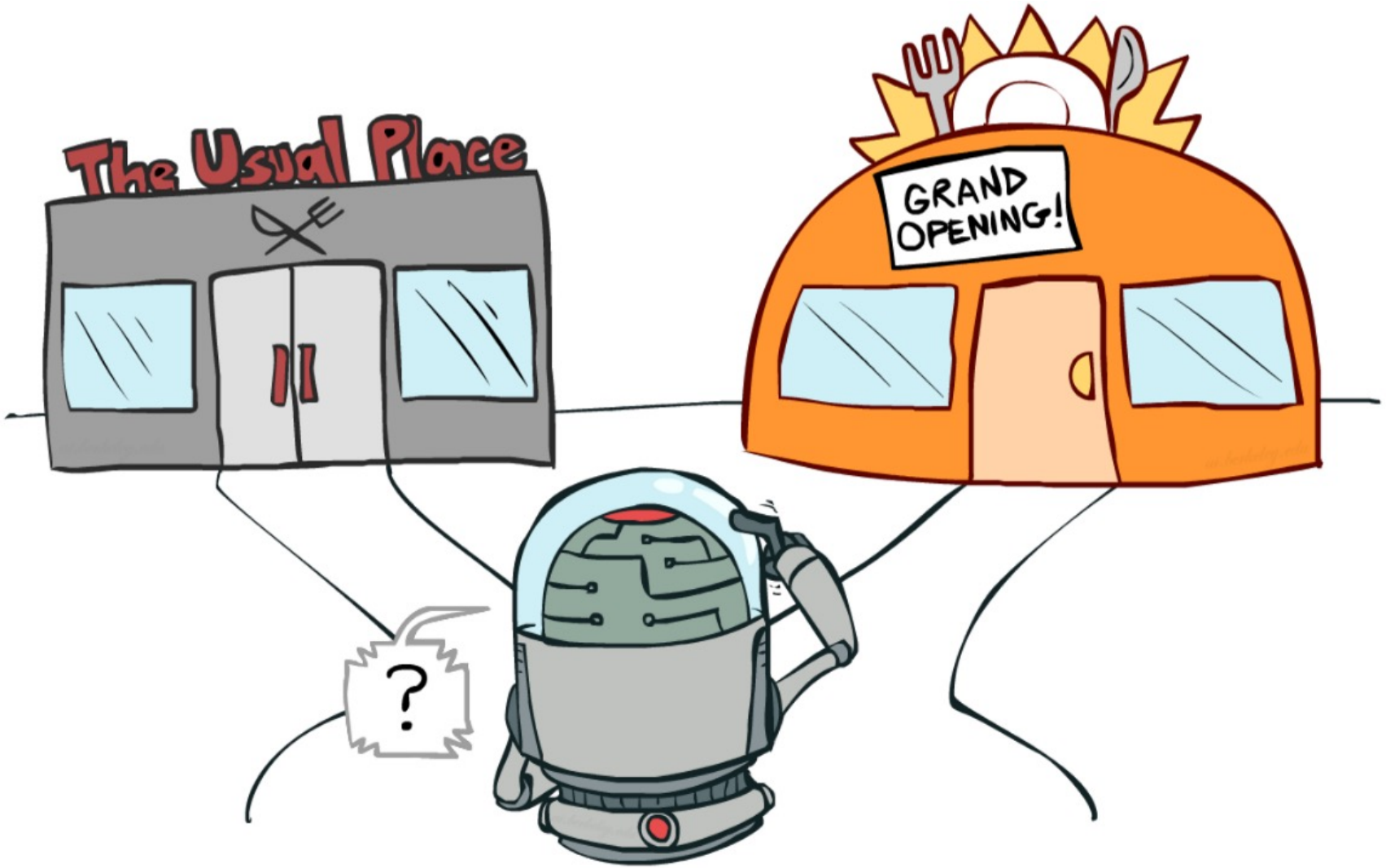
Training results

$$\alpha = 0.5$$

Learned value function after 200 episodes



Exploration vs Exploitation Dilemma



ε-greedy

- Given $\epsilon \in (0,1)$
- At each step, with a probability of ϵ, sample a random action instead

- There are more advanced approaches
- E.g., soft Q-learning (Haarnoja et al., 2017)

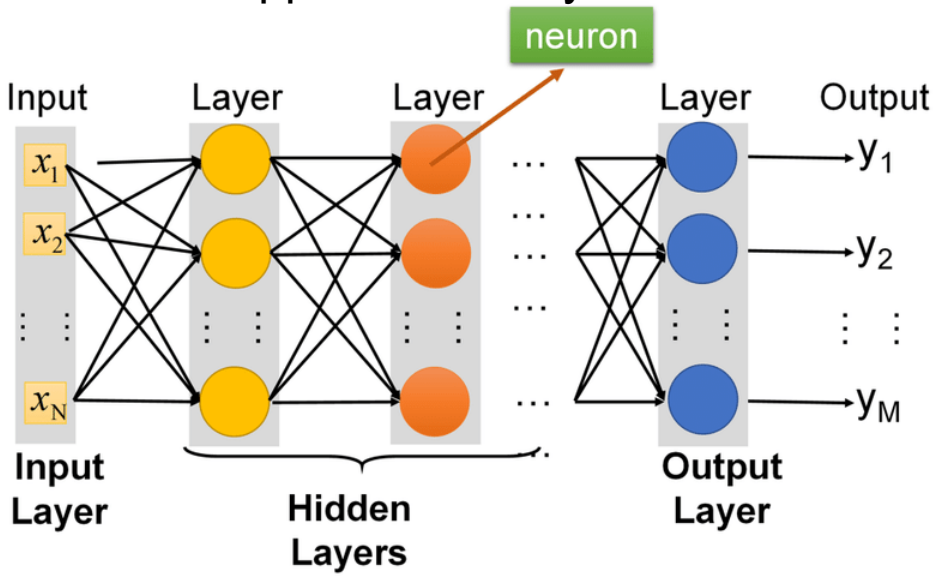
Beyond 2D grid words?

S					The cliff						G

$$Q($$  $$, a)$$

Deep neural networks

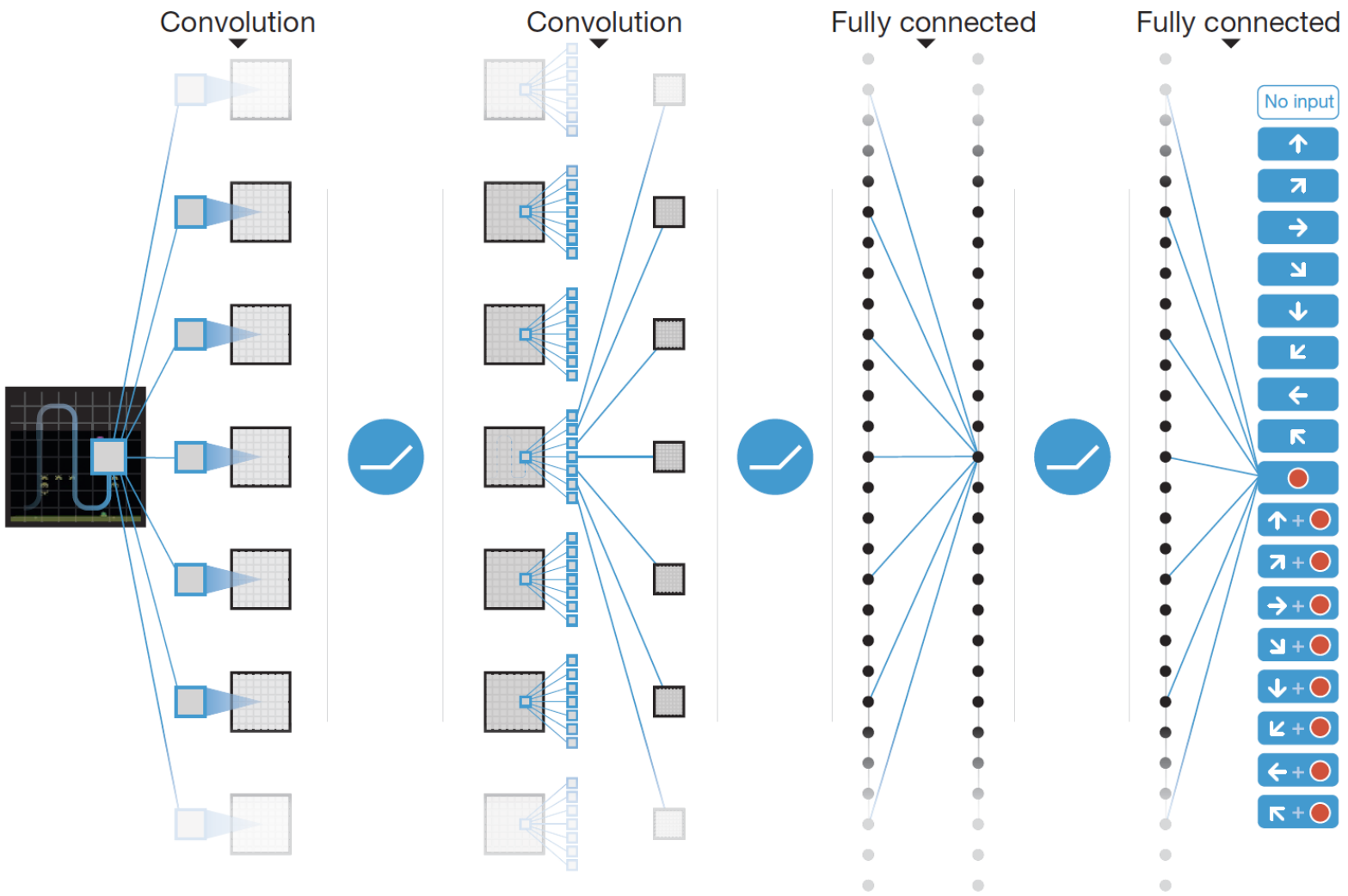
Can approximate any function



Deep neural networks to approximate the value function

Deep Q-network (DQN)

S



$Q(s, a)$

Further readings

For learning about how to combine Q-learning with deep neural networks

Human-level control through deep reinforcement learning

Volodymyr Mnih[1]*, Koray Kavukcuoglu[1]*, David Silver[1]*, Andrei A. Rusu[1], Joel Veness[1], Marc G. Bellemare[1], Alex Graves[1], Martin Riedmiller[1], Andreas K. Fidjeland[1], Georg Ostrovski[1], Stig Petersen[1], Charles Beattie[1], Amir Sadik[1], Ioannis Antonoglou[1], Helen King[1], Dharshan Kumaran[1], Daan Wierstra[1], Shane Legg[1] & Demis Hassabis[1]

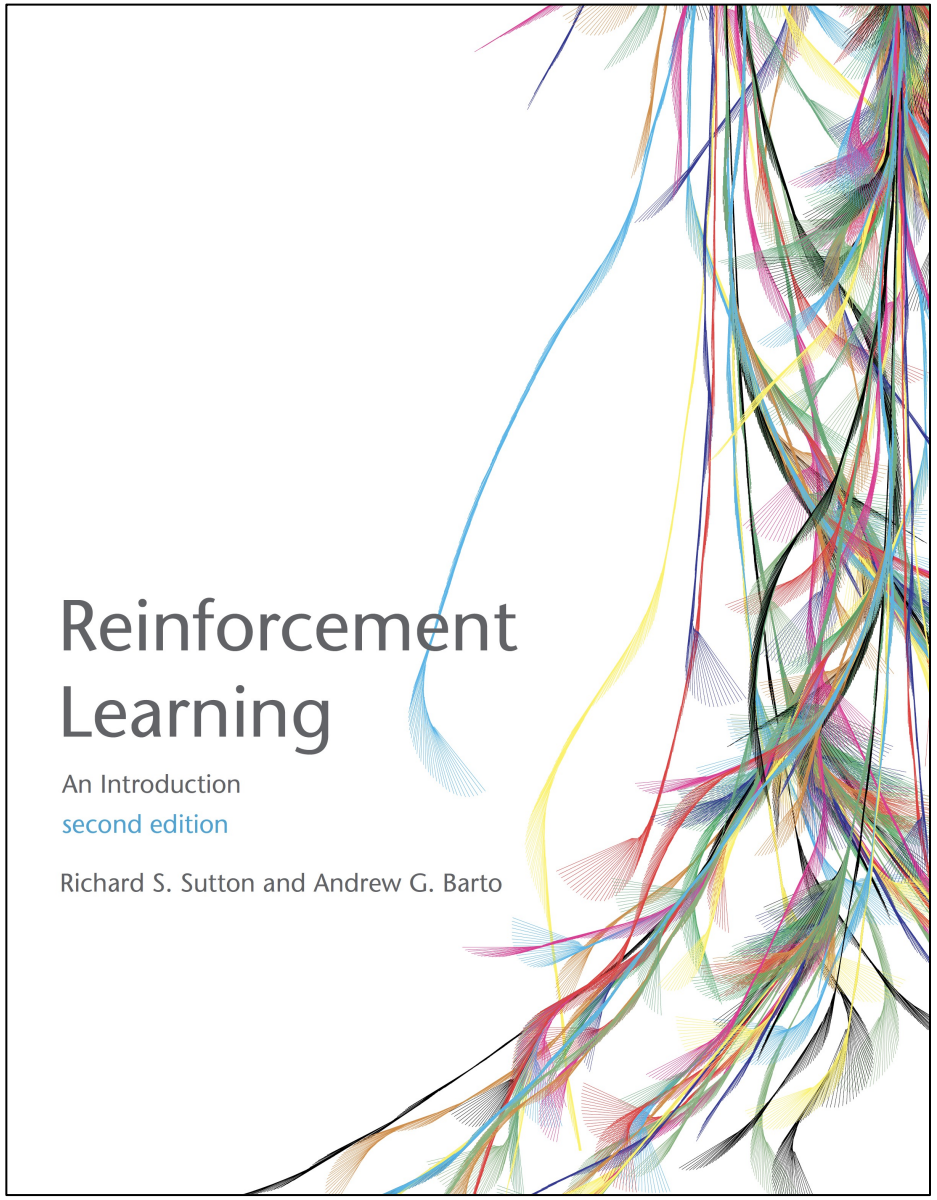
Further readings

For learning about more advanced topics in RL

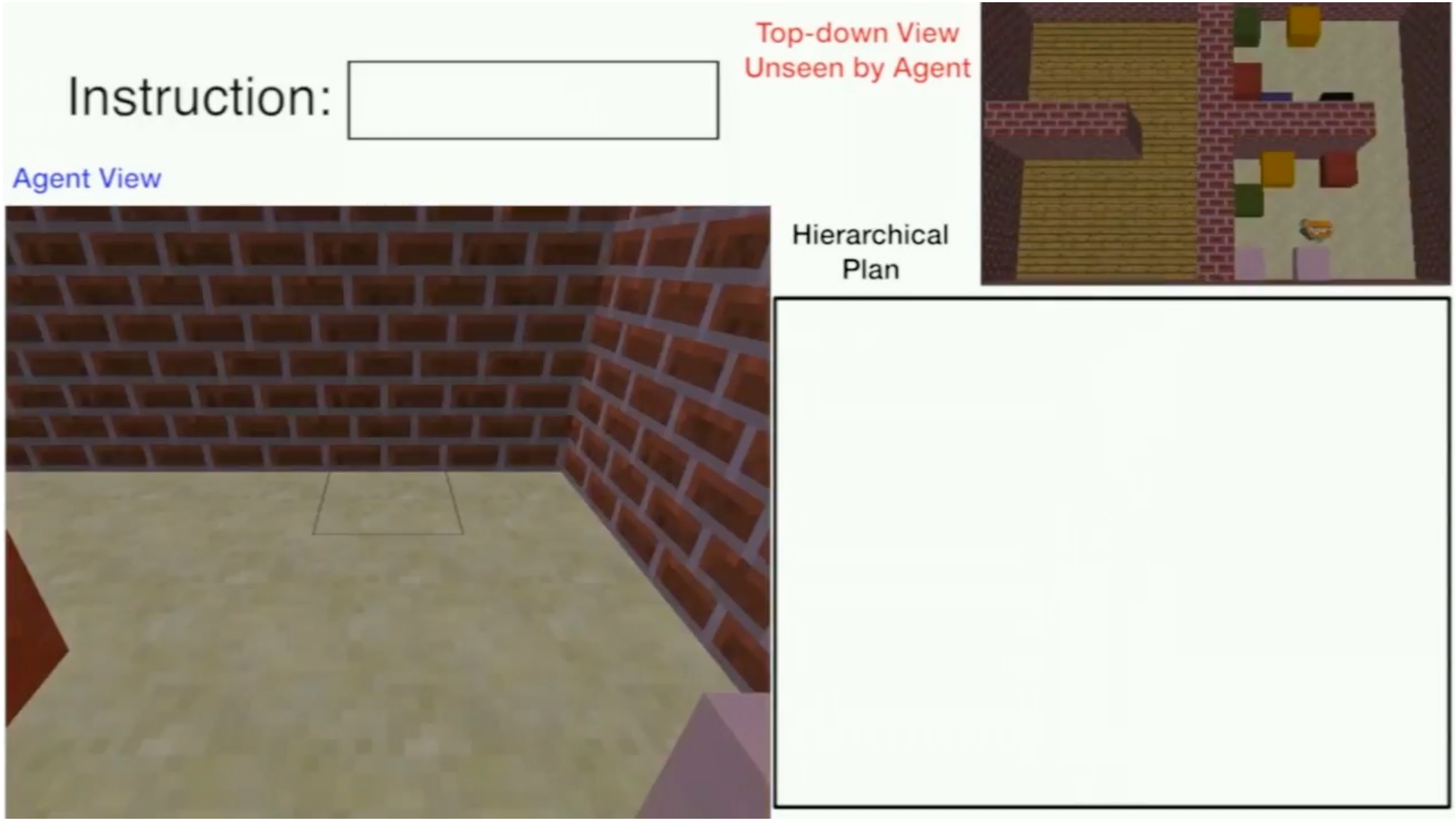
Pioneers in RL:

Richard S. Sutton

Andrew G. Barto



Multimodal inputs and outputs



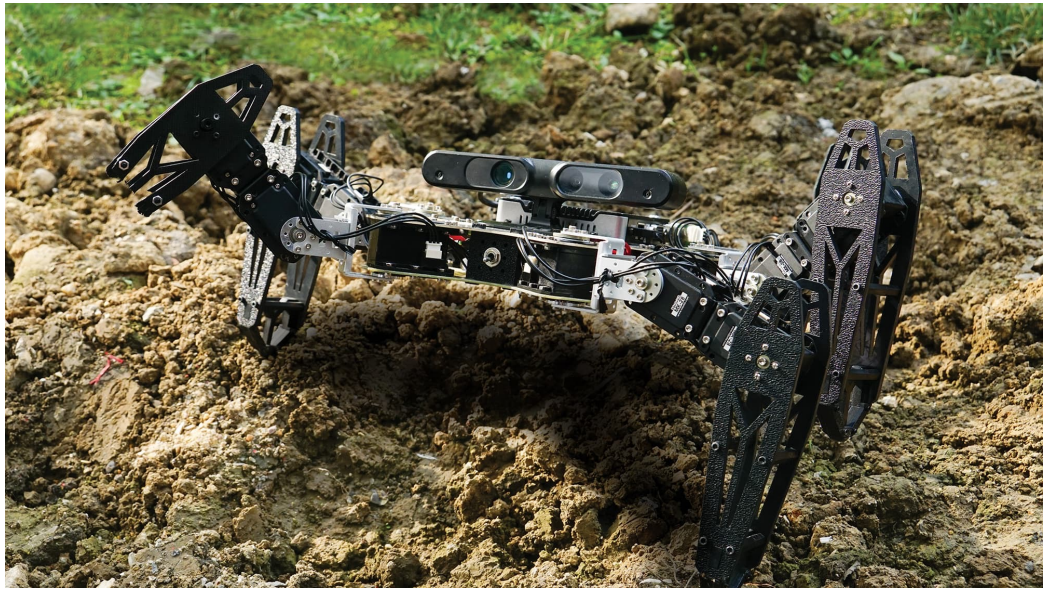
Shu et al. (2018)

Summary

- MDPs, policies and value functions
- Bellman equations
- Policy iteration, value iteration
- Q-learning

Challenges for RL in the real world: safety

S					The cliff						G

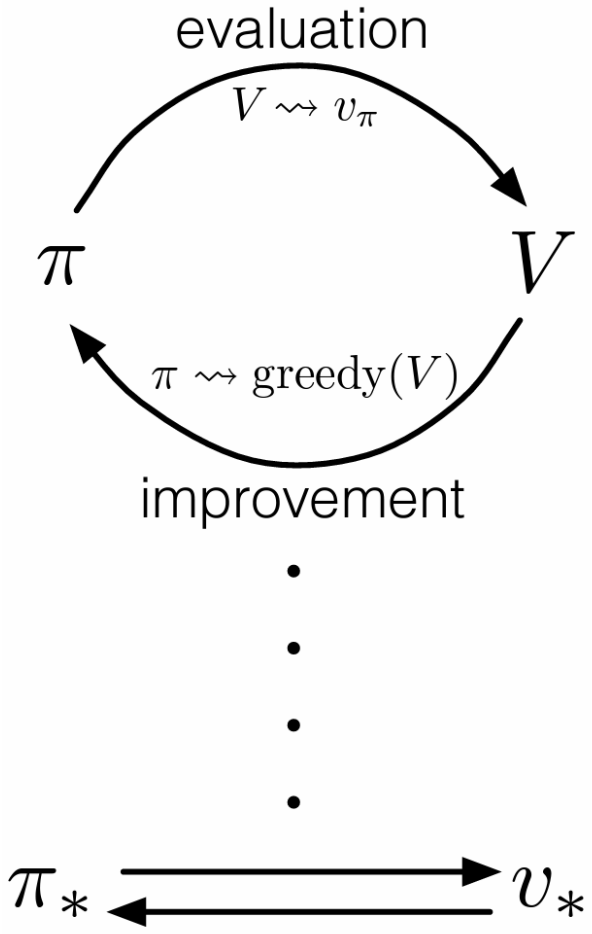


Challenges for RL in the real world: generalization

Challenges for RL in the real world: generalization

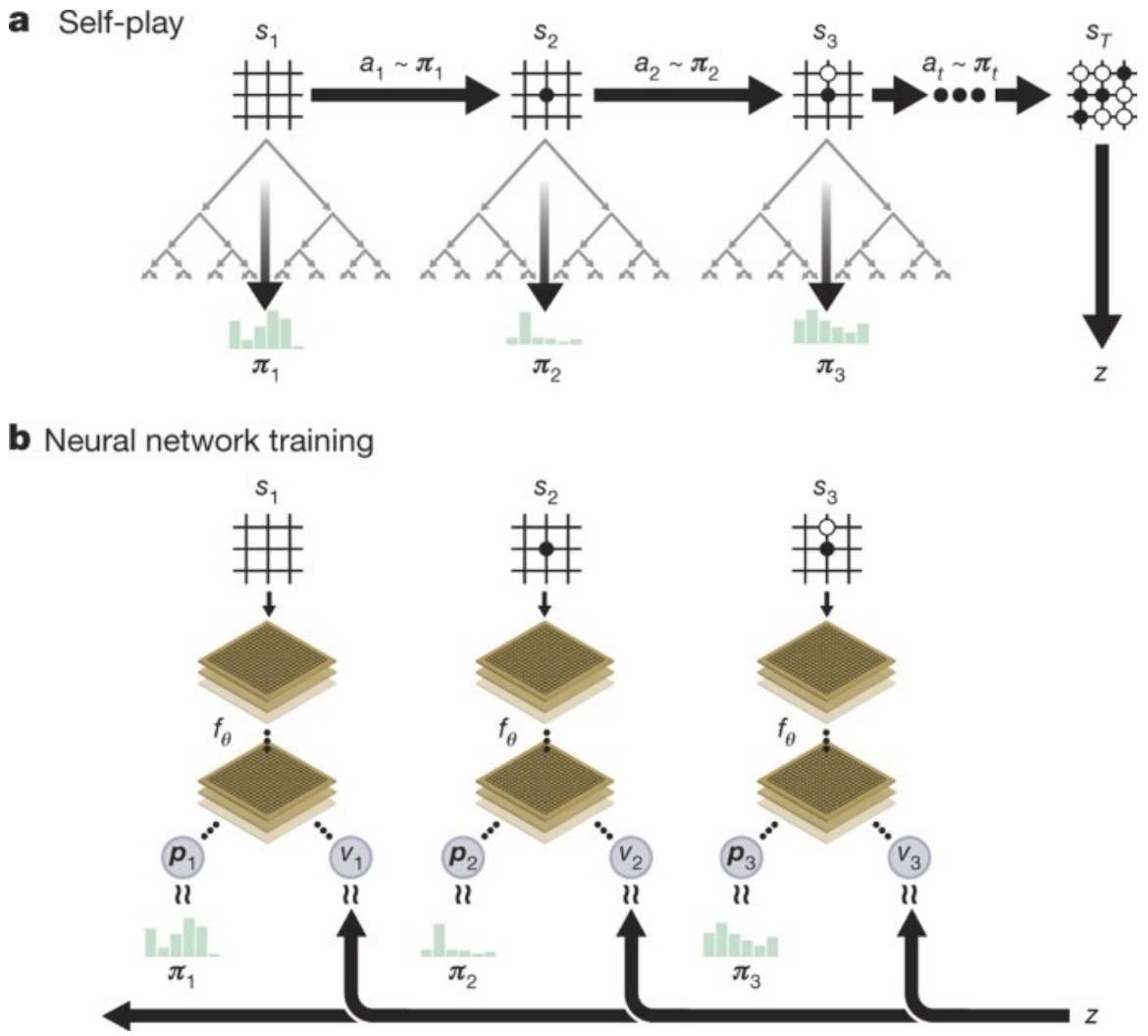
Model-based learning

- Beyond traditional planning algorithms
- World model → model-based planning
- Policy and value function



Model-based learning

- Beyond traditional planning algorithms
- AlphaGo



Model-based learning

- Beyond traditional planning algorithms

- Vision-language models to serve as both policies and value functions
- Text-to-video models as dynamics models.

Video Language Planning

Video Generation | Real Execution

Put the fruits into the top drawer

Execution