CSCI 699: Robotic Perception

Yue Wang

Sep 24th, 2025

Administratives

Project

- Project proposal
  - 2-4 pages
    - Introduction
    - Related work
    - Proposed method / pipeline
    - (Optionally) risk mitigation
    - (Tentative) work assignment
  - Due today
  - Group project update
    - Starting today
    - Each group presents their progress in a 25-minute presentation

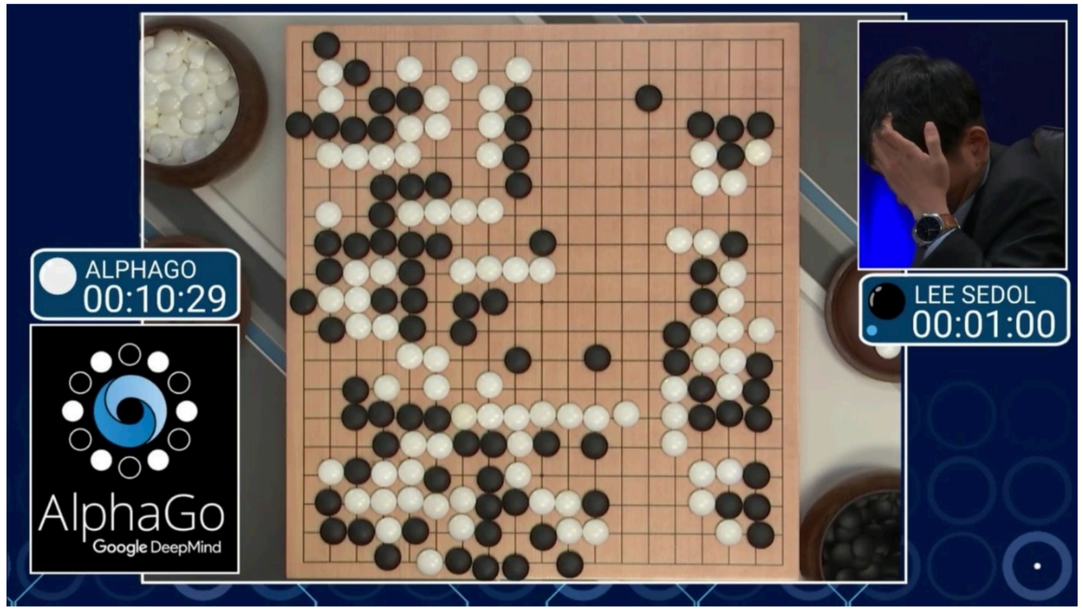
Why RL?

Limitations of Imitation Learning

- Need experts (e.g., human)
  - Could be expensive
- Can not go beyond the expert level
  - Not enough for “super intelligence”
- Ideally, AI should be able to learn autonomously!

RL Success

- AlphaGo, 2016



Summary [edit]

Game	Date	Black	White	Result	Moves
1	9 March 2016	Lee Sedol	**AlphaGo**	Lee Sedol resigned	186 Game 1
2	10 March 2016	**AlphaGo**	Lee Sedol	Lee Sedol resigned	211 Game 2
3	12 March 2016	Lee Sedol	**AlphaGo**	Lee Sedol resigned	176 Game 3
4	13 March 2016	AlphaGo	**Lee Sedol**	AlphaGo resigned	180 Game 4
5	15 March 2016	Lee Sedol[note 1]	**AlphaGo**	Lee Sedol resigned	280 Game 5
Result: **AlphaGo 4 – 1 Lee Sedol**					

RL Success

- AlphaZero: chess, go, shoji

Configuration and strength[62]

Versions	Hardware	Elo rating	Date	Results
AlphaGo Fan	176 GPUs,[53] distributed	3,144[52]	Oct 2015	5:0 against Fan Hui
AlphaGo Lee	48 TPUs,[53] distributed	3,739[52]	Mar 2016	4:1 against Lee Sedol
AlphaGo Master	4 TPUs,[53] single machine	4,858[52]	May 2017	60:0 against professional players; Future of Go Summit
AlphaGo Zero (40 block)	4 TPUs,[53] single machine	5,185[52]	Oct 2017	100:0 against AlphaGo Lee 89:11 against AlphaGo Master
AlphaZero (20 block)	4 TPUs, single machine	5,018[63]	Dec 2017	60:40 against AlphaGo Zero (20 block)

RL Success

- AlphaStar: Grandmaster level in StartCraft II

RL Success

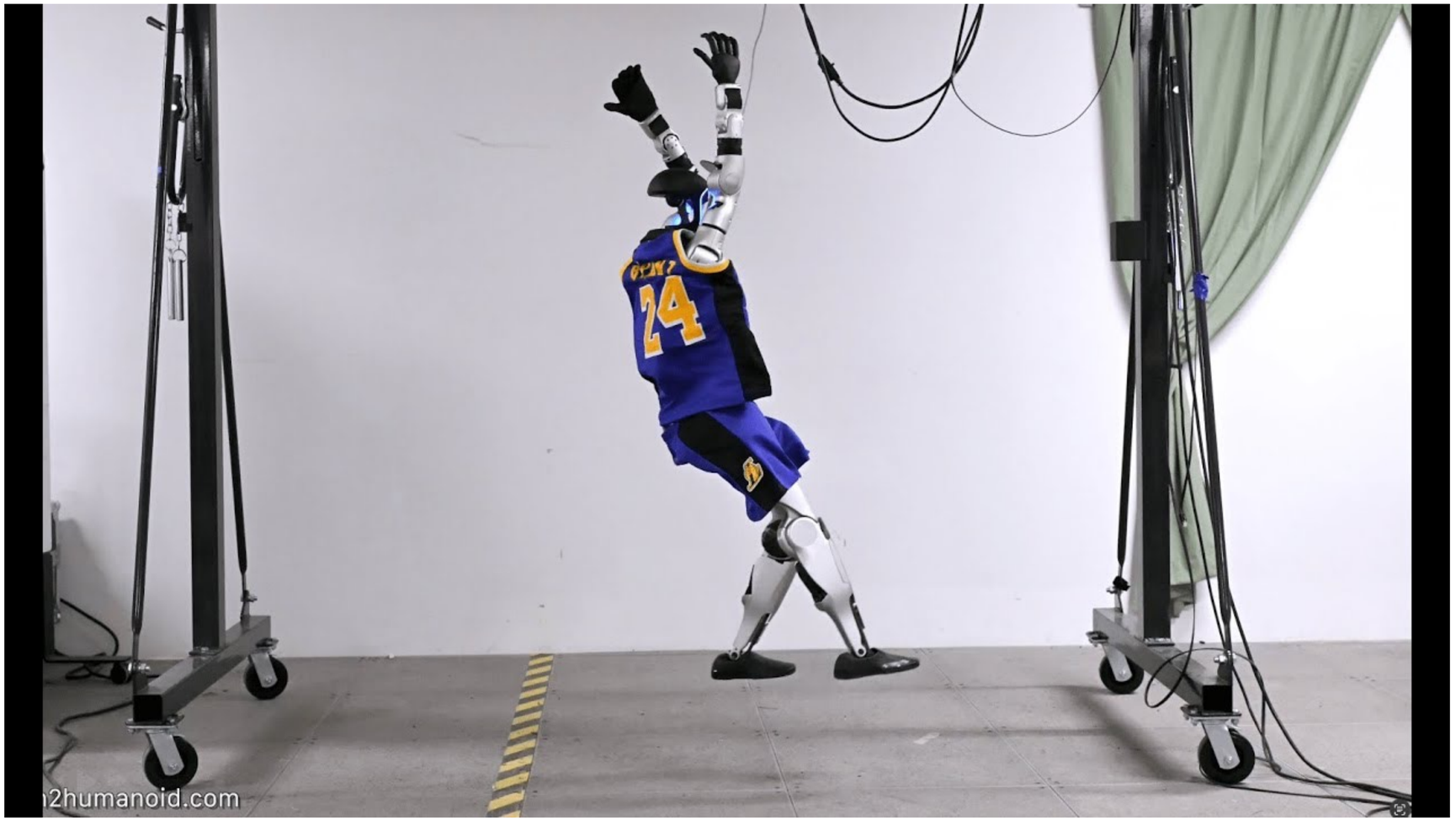
- Jumping, 400m dash of Cassie (biped)

RL Success

- Legged locomotion in challenging terrains using egocentric vision

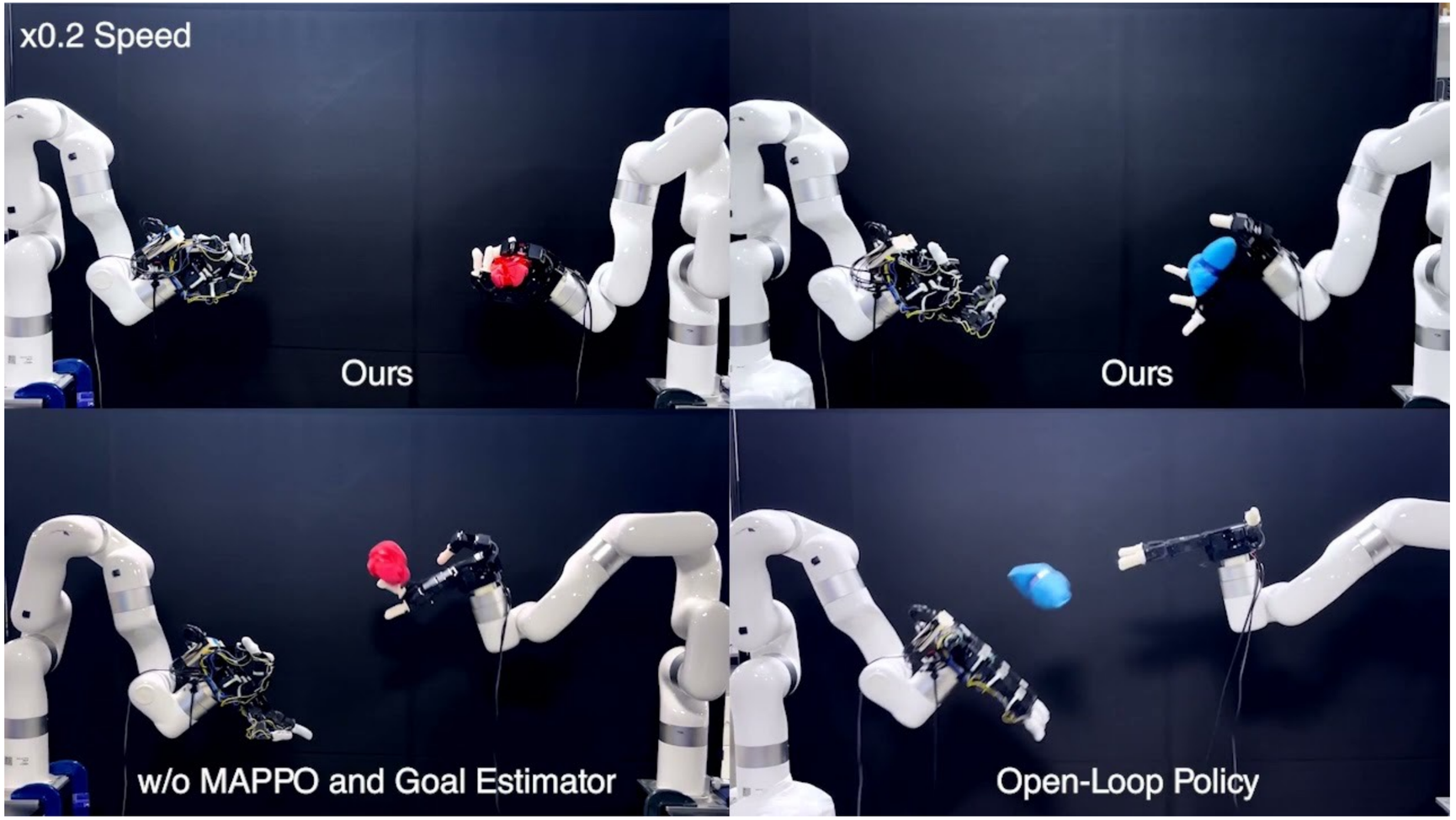
RL Success

- ASAP: Aligning Simulation and Real-World Physics for Learning Agile Humanoid Whole-Body Skills



RL Success

- Dynamic Handover: Throw and Catch with Bimanual Hands

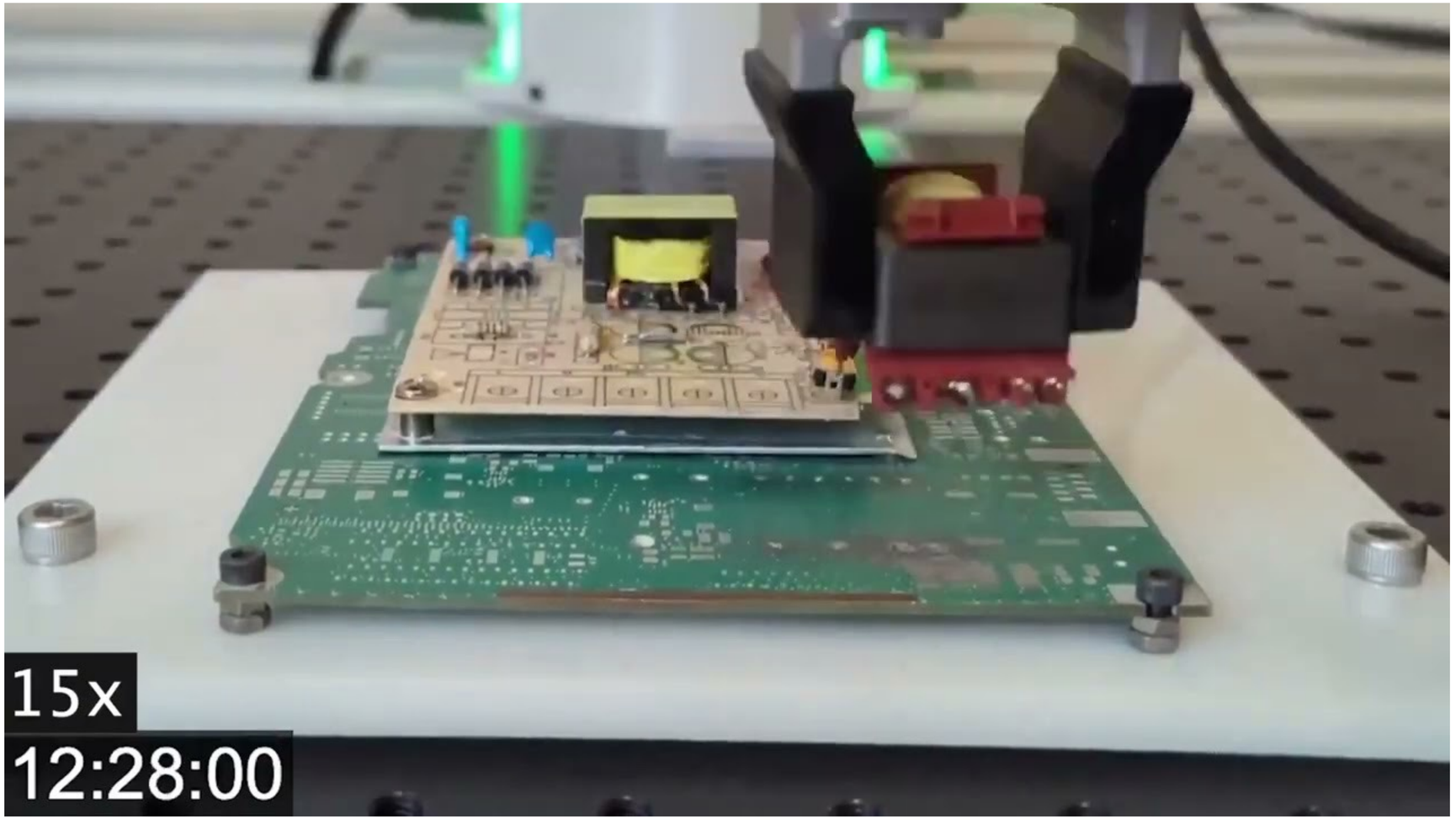


RL Success

- Champion-level drone racing using deep reinforcement learning

RL Success

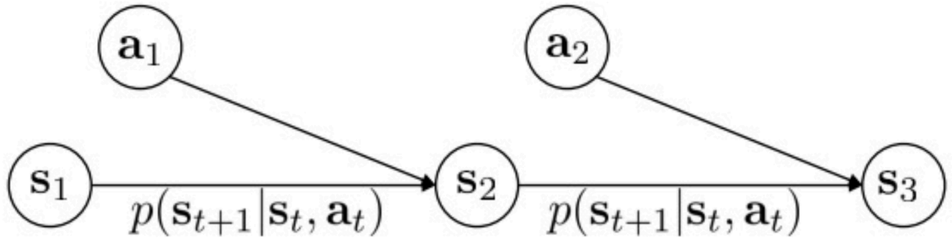
- SERL: Robot Reinforcement Learning Software Suite



Notations, MDP, POMDP, return, value functions

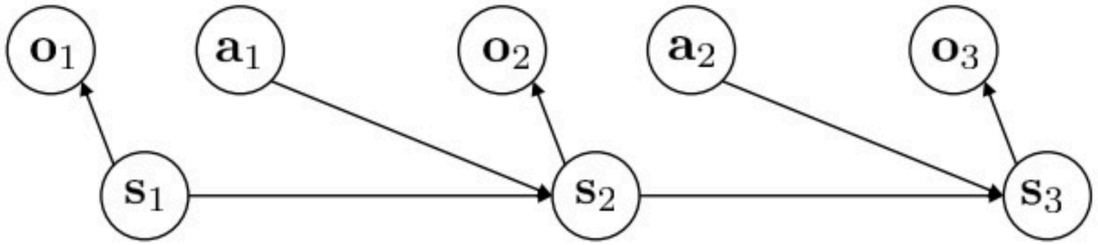
MDP

- Definitions
  - S is the state space. $s_t \in S$ is the state at time step t.
  - A is the action space. $a_t \in A$ is the action at time step t.
  - O is the observation space. $o_t \in O$ is the observation at time step t.
  - p is the one-step transition probability (a.k.a, dynamics): $s_{t+1} \sim p(\cdot \mid s_t, a_t)$.
  - h is the observation model: $o_t \sim h(\cdot \mid s_t)$
  - $r : S \times A \to \mathbb{R}$ is the reward function.
- A Markov decision process (MDP) is a tuple (S, A, p, r)
  - Goal: learn a policy $\pi_\theta(a_t \mid s_t)$



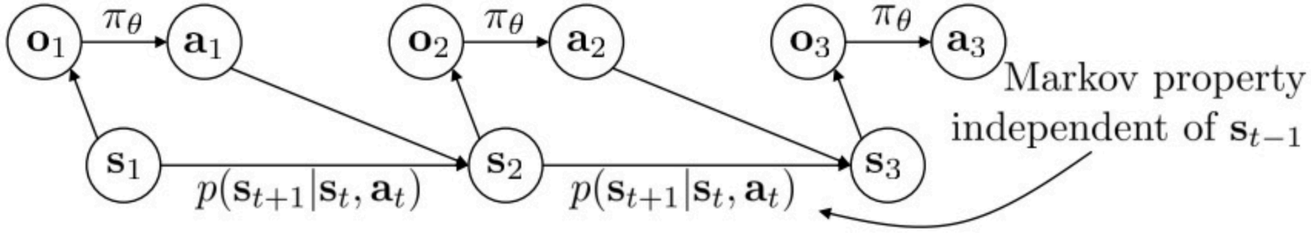
POMDP

- Definitions
  - S is the state space. $s_t \in S$ is the state at time step t.
  - A is the action space. $a_t \in A$ is the action at time step t.
  - O is the observation space. $o_t \in O$ is the observation at time step t.
  - p is the one-step transition probability (a.k.a, dynamics): $s_{t+1} \sim p(\cdot \mid s_t, a_t)$.
  - h is the observation model: $o_t \sim h(\cdot \mid s_t)$
  - $r : S \times A \to \mathbb{R}$ is the reward function.
- A partially observed Markov decision process (POMDP) is a tuple (S, A, O, p, h, r)
  - Goal: learn a policy $\pi_\theta(a_t \mid o_t)$



POMDP

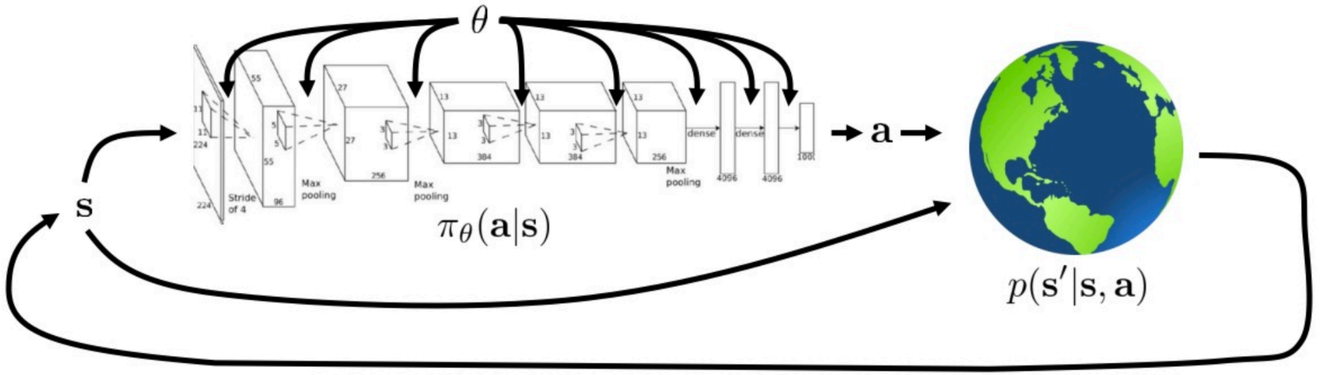
- Definitions
  - S is the state space. $s_t \in S$ is the state at time step t.
  - A is the action space. $a_t \in A$ is the action at time step t.
  - O is the observation space. $o_t \in O$ is the observation at time step t.
  - p is the one-step transition probability (a.k.a, dynamics):
    $s_{t+1} \sim p(\cdot \mid s_t, a_t)$.
  - h is the observation model: $o_t \sim h(\cdot \mid s_t)$
  - $r : S \times A \to \mathbb{R}$ is the reward function.
- Why Markov?
  - Allow us to throw away history once state is known!



Andrey Markov

The Goal of Reinforcement Learning and Control

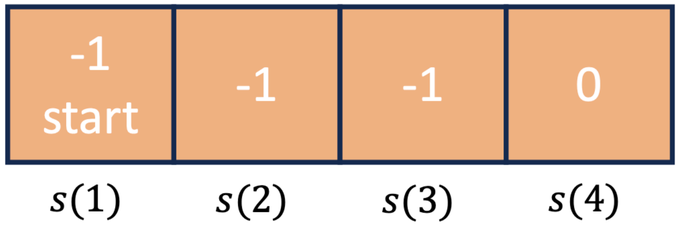
- Finite horizon case: T is finite
- Infinite horizon case: $T = \infty$
- The cumulative reward can be discounted: $\sum_t \gamma^t r(s_t, a_t)$ where $0 < \gamma \leq 1$
- Goal: find a policy to maximize the cumulative reward



$$\underbrace{p_\theta(\mathbf{s}_1, \mathbf{a}_1, \ldots, \mathbf{s}_T, \mathbf{a}_T)}_{\pi_\theta(\tau)} = p(\mathbf{s}_1) \prod_{t=1}^{T} \pi_\theta(\mathbf{a}_t|\mathbf{s}_t) p(\mathbf{s}_{t+1}|\mathbf{s}_t, \mathbf{a}_t)$$

$$\theta^\star = \arg\max_\theta E_{\tau \sim p_\theta(\tau)} \left[\sum_t r(\mathbf{s}_t, \mathbf{a}_t) \right]$$

Notion Explanation: 1D Grid World



- Four states: $s(1)$ (start), $s(2)$, $s(3)$, $s(4)$
- Three actions: left, right, stay
- $0 \leq \pi(a \mid s) \leq 1$ is π's probability of choosing a at state s
- Example (a smart deterministic policy): $\pi(a = \text{right} \mid s(2)) = 1$
- Example (a random policy): $\pi(a \mid s(2)) = 1/3$ for all actions
- $\pi(s)$: (slightly abuse the notation) A mapping from state to action!
- $0 \leq p(s' \mid s, a) \leq 1$ is probability of transiting to s' from s, with action a
- Example (deterministic system): $p(s(3) \mid s(2), a = \text{right}) = 1$
- Example (stochastic): $p(s(3) \mid s(2), a = \text{right}) = 0.9, p(s(2) \mid s(2), a = \text{right}) = 0.1$

Returns G_t

- Sum of future rewards: $G_t = r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+3} \ldots = \sum_{k=0}^{\infty} \gamma^k r_{t+k+1}$
- Episodic setting (finite horizon): $G_t = r_{t+1} + r_{t+2} + \ldots + r_T$
- The action sequence is $s_0, a_0, r_1, s_1, a_1, r_2, \ldots$
  - Some books/papers might use different index
- We will focus on the discounted infinite horizon case today

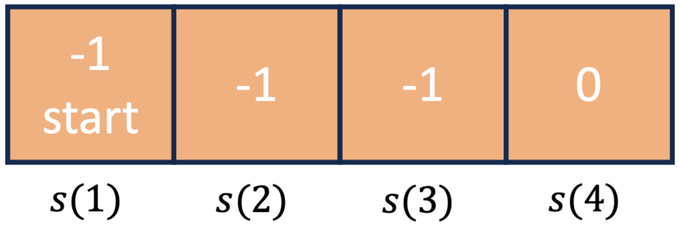
Returns G_t

- State-value function (V): the expected return starting from state s, and then following policy π: $V^{\pi}(s) = \mathbb{E}_{\pi}[G_0 \,|\, s_0 = s]$
- Action-value function (Q): the expected return starting from state s, taking action a, and then following policy π: $Q^{\pi}(s, a) = \mathbb{E}_{\pi}[G_0 \,|\, s_0 = s, a_0 = a]$
- Question: Why do we need both V and Q? What does $Q^{\pi}(s, a) > V^{\pi}(s)$ mean?

Optimal Value Functions

- Optimal V: $V^*(s) = \max_\pi V^\pi(s) = \max_\pi \mathrm{E}_\pi[G_0 \,|\, s_0 = s]$
- Optimal Q: $Q^*(s,a) = \max_\pi Q^\pi(s,a) = \max_\pi \mathrm{E}_\pi[G_0 \,|\, s_0 = s, a_0 = a]$
- Optimal policy: $\pi^*(s) = \mathrm{argmax}_\pi V^\pi(s)$
- Properties: $V^*(s) = \max_a Q^*(s,a)$ and $\pi^*(s) = \mathrm{argmax}_a Q^*(s,a)$
- If we know $Q^*(s,a)$, we immediately have the optimal policy!
  - If we only know $V^*(s)$, we need the dynamics to do one step lookahead to choose the optimal action

Example: 1D Grid World



- Four states: $s(1)$ (start), $s(2)$, $s(3)$, $s(4)$
- Three actions: left, right, stay
- Assume no discount and deterministic transitions
- $V^*(s(1)) = -3, V^*(s(2)) = -2, V^*(s(3)) = -1, V^*(s(4)) = 0$
- $Q^*(s(1), a = \text{stay}) = -4, Q^*(s(1), a = \text{right}) = -3$, so $\pi^*(s(1)) = \text{right}$

Acknowledgement

- Many contents are taken from UT Austin CS391, CMU 16-831