

# CS 277: Control and Reinforcement Learning

## Winter 2022

# Lecture 1: Introduction

**Roy Fox**

Department of Computer Science

Bren School of Information and Computer Sciences

University of California, Irvine



# Today's lecture

---

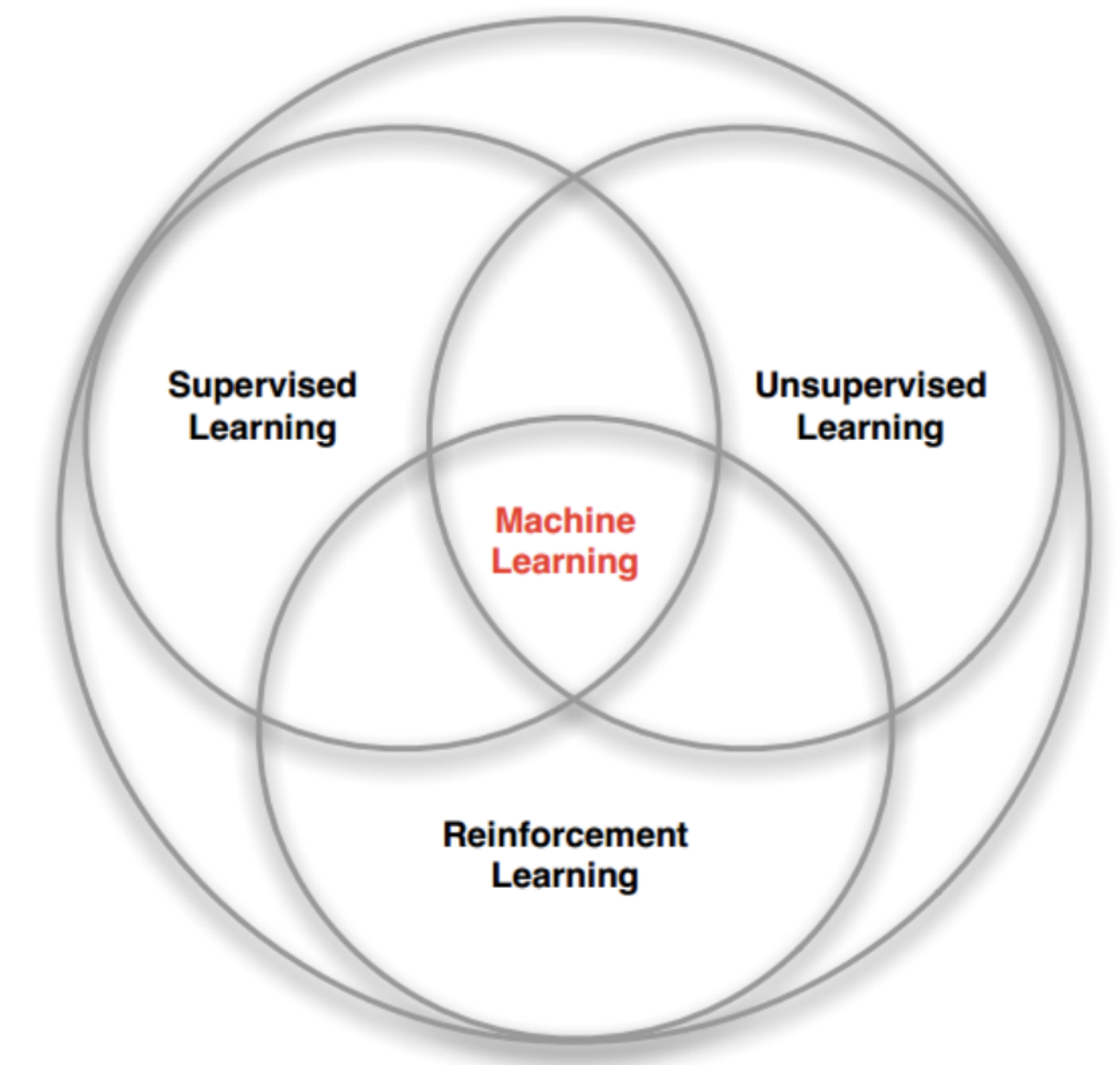
What is reinforcement learning?

Course logistics

Basic RL concepts

# RL $\subseteq$ control learning $\subseteq$ ML

- Reinforcement Learning = learning from reinforcement (**rewards**)
  - But it came to encompass many settings of **learning to control**
  - Distinguished by **sequential decision making** and **learning**
- Many consider RL a separate ML paradigm, but it can involve:
  - **Supervised** learning
  - **Unsupervised** learning
  - **Active** learning
  - **Online** learning



# What is machine learning

- Can we build “intelligent” machines? **Intelligence** = good decision making
- **Learning** = taking in information to “know” more than you did before
- **Machine learning** = use data to make better decisions than before [Mitchell 1997]
- ML can help when other AI methods fail:

- ▶ **Experts** are scarce
- ▶ **Rules / logic** are hard to specify
- ▶ **Search** space is too large
- ▶ **Models** are unknown / hard to specify

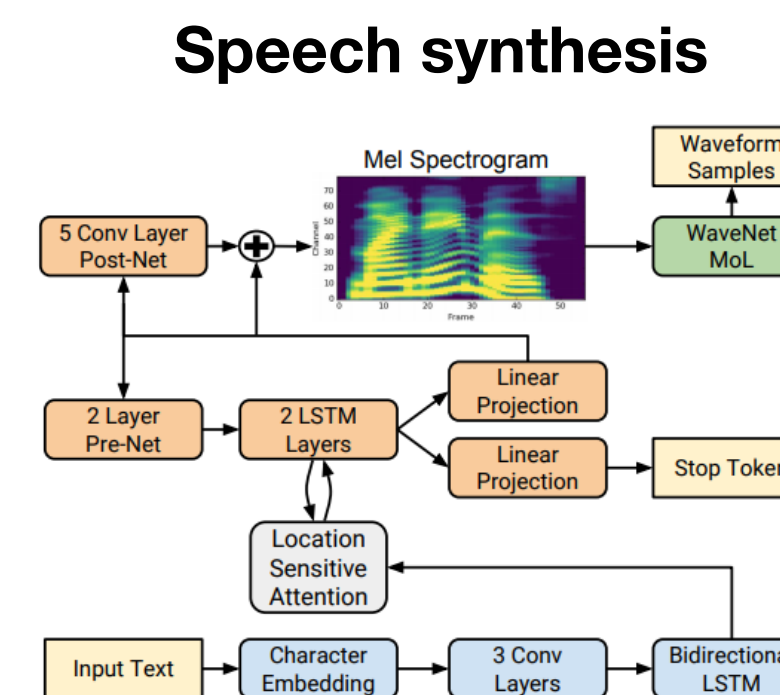
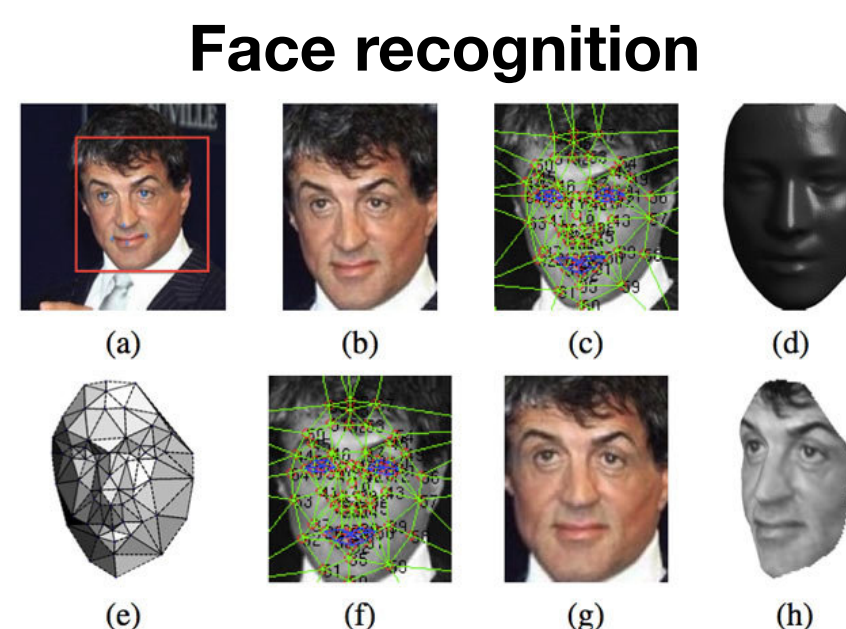
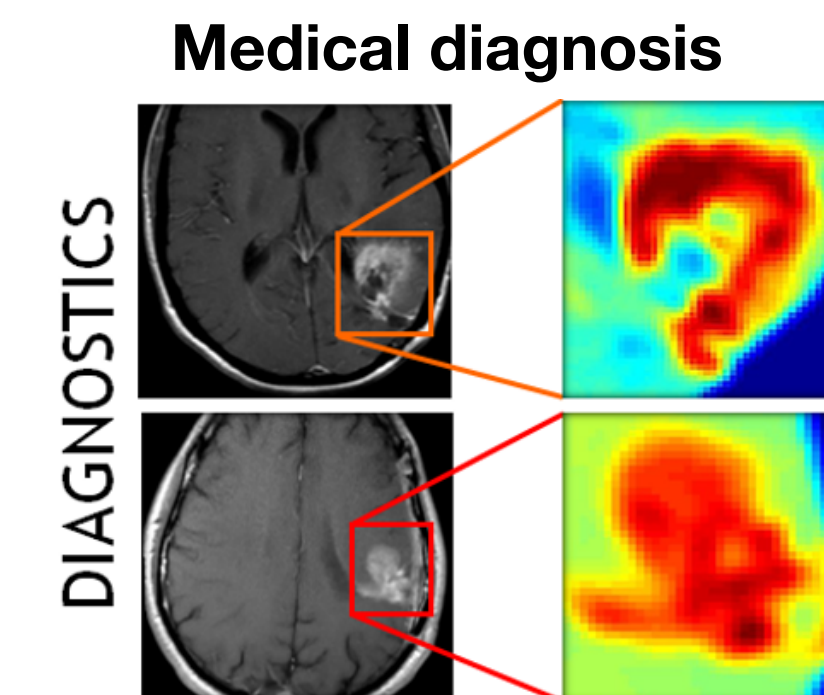


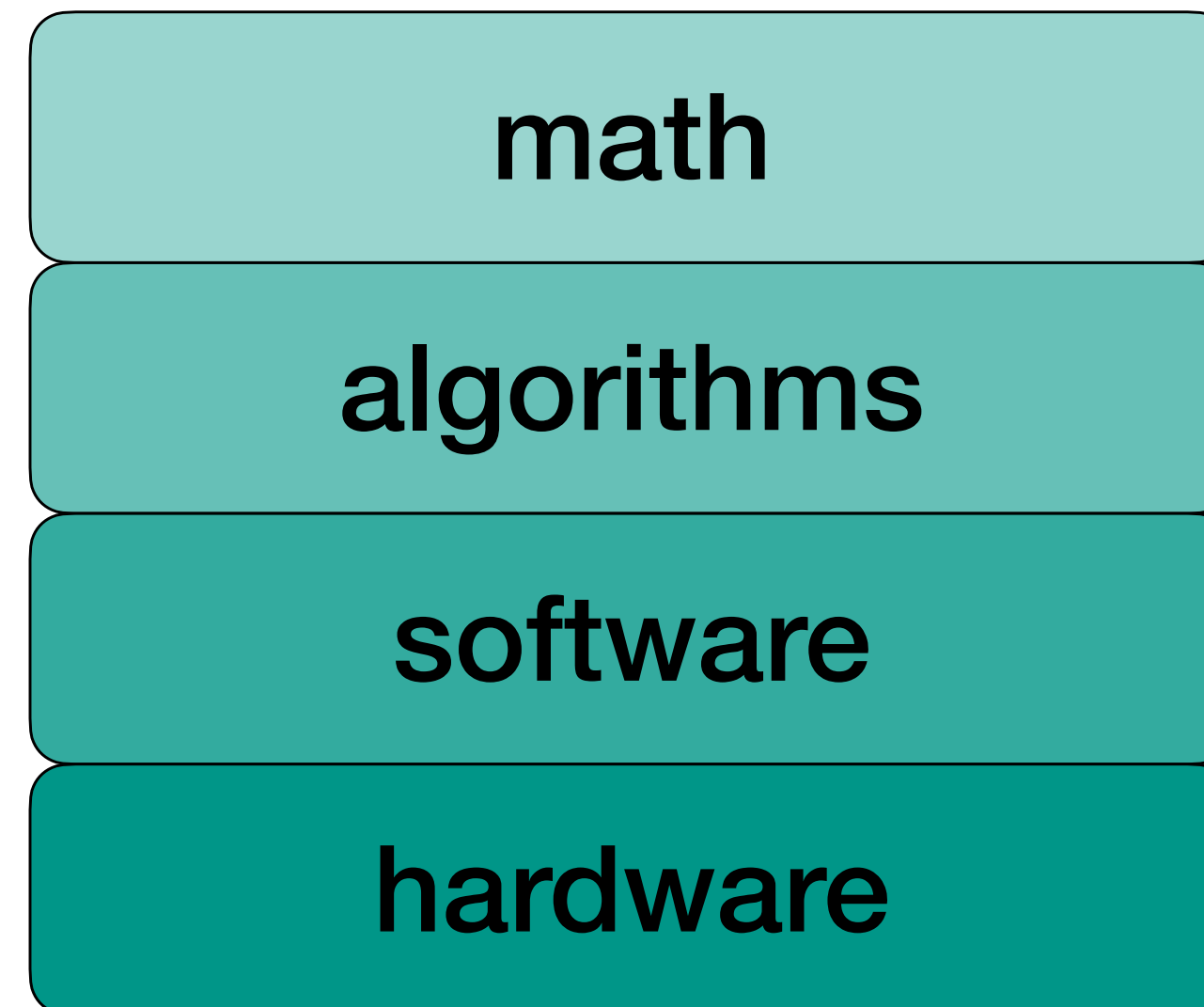
Fig. 1. Block diagram of the Tacotron 2 system architecture.



[Taigman et al., 2014; Shen et al., 2018]

# The ML stack

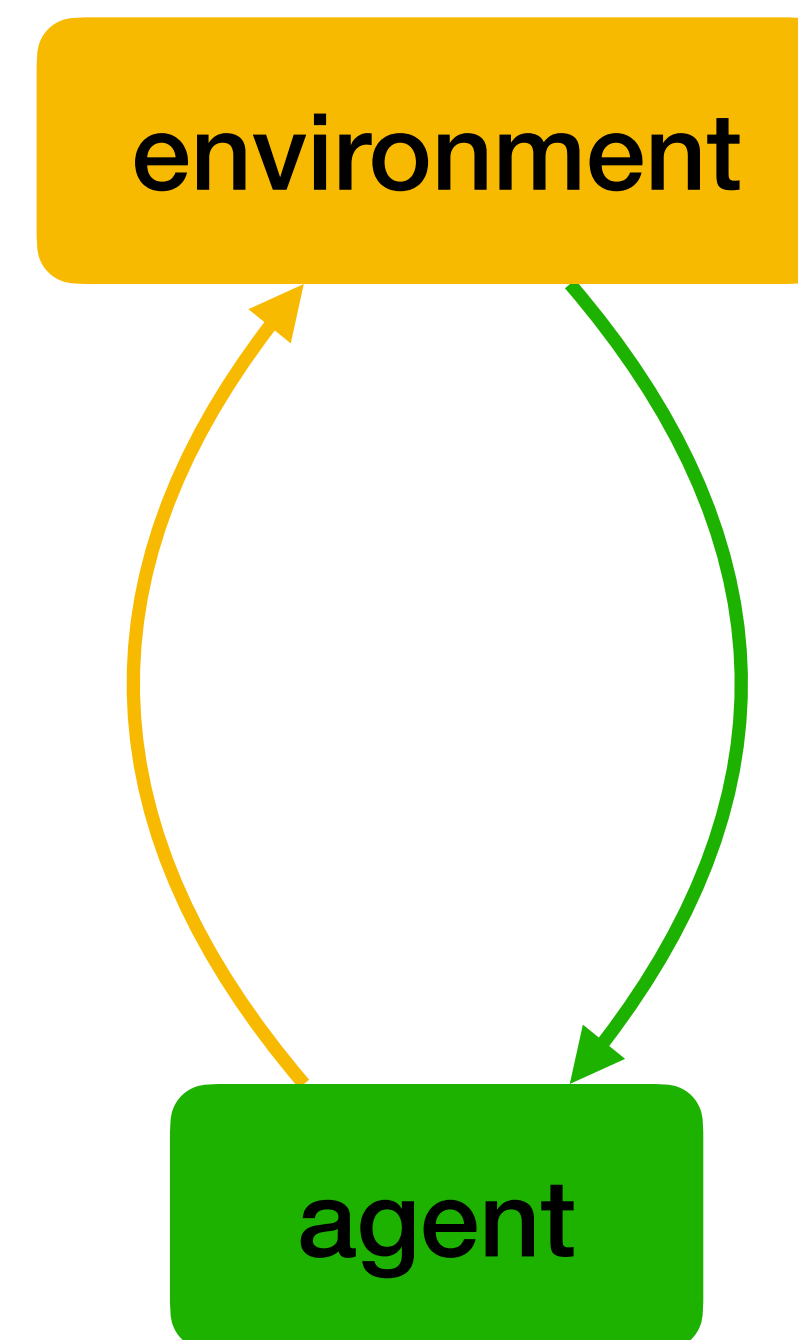
---



- **Math:** probability theory, (linear) algebra, computational learning theory
- **Algorithms:** ML algorithms, optimization, data structures
- **Software:** ML frameworks, databases, testing, deployment
- **Hardware:** cloud computing, distributed systems, cyber-physical systems

# What is control learning?

- Intelligence appears in interaction with a complex system, not in isolation
  - An **agent** interacting with an **environment**
- **Control** = sequential decision making
  - Sense environment state  $s$
  - Take action  $a$
  - Repeat
- Success can be measured by matching good actions — **imitation learning (IL)**
  - Or by accumulating high rewards  $r(s, a)$  — **reinforcement learning (RL)**

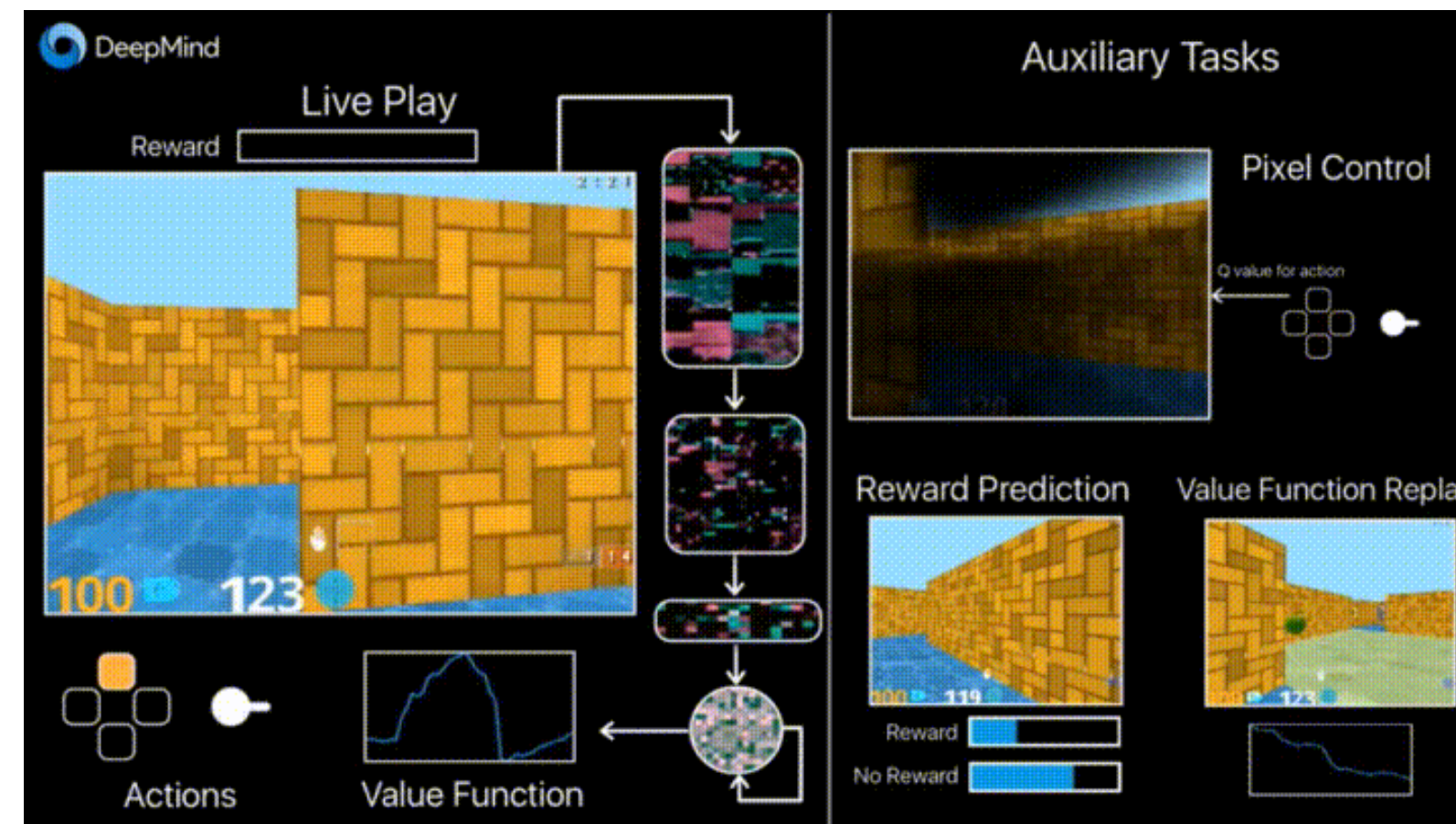


# Examples of learned controllers

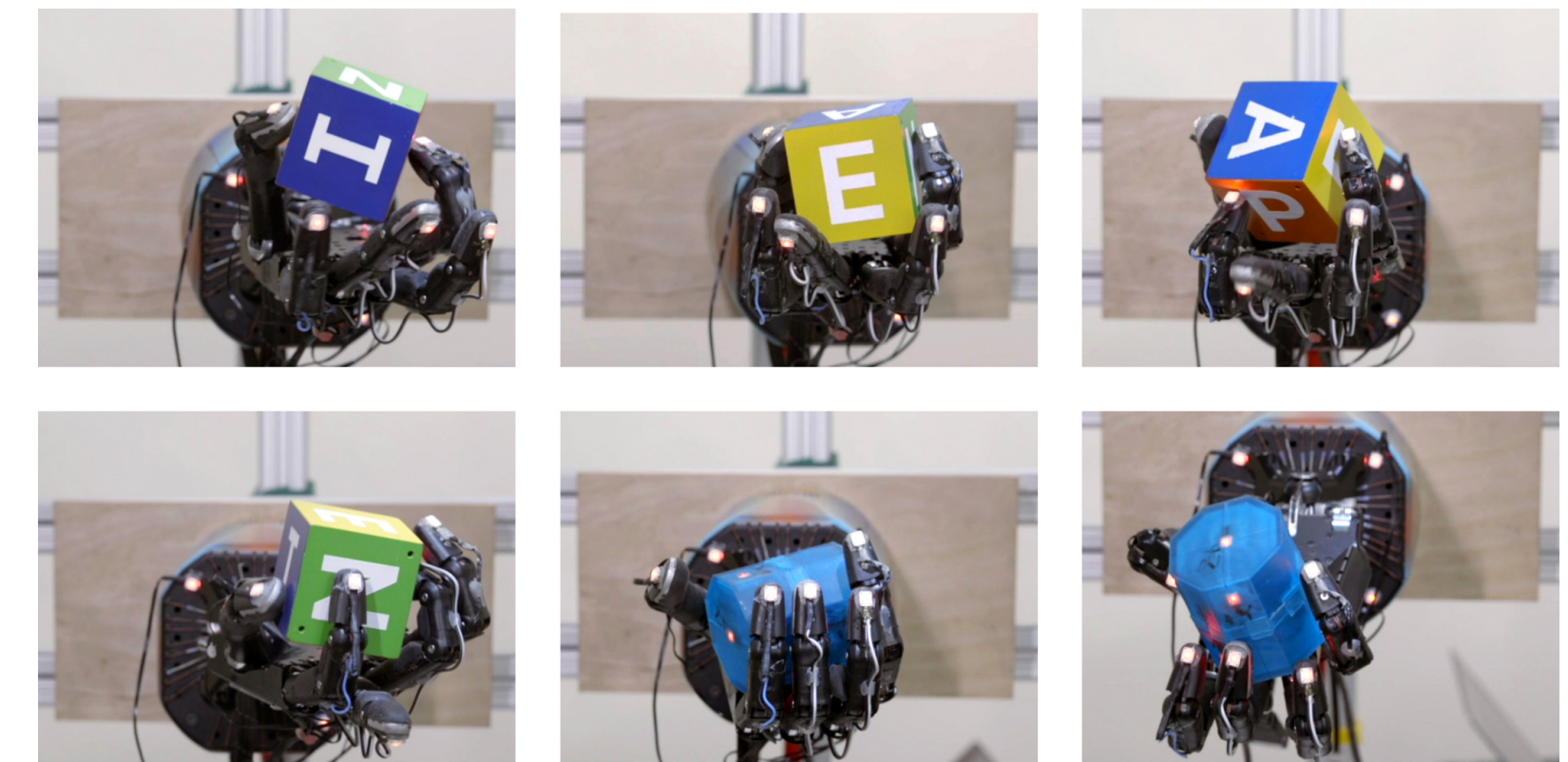
## Gameplay



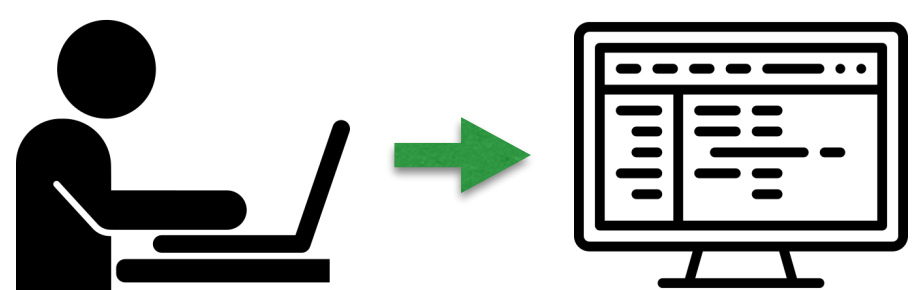

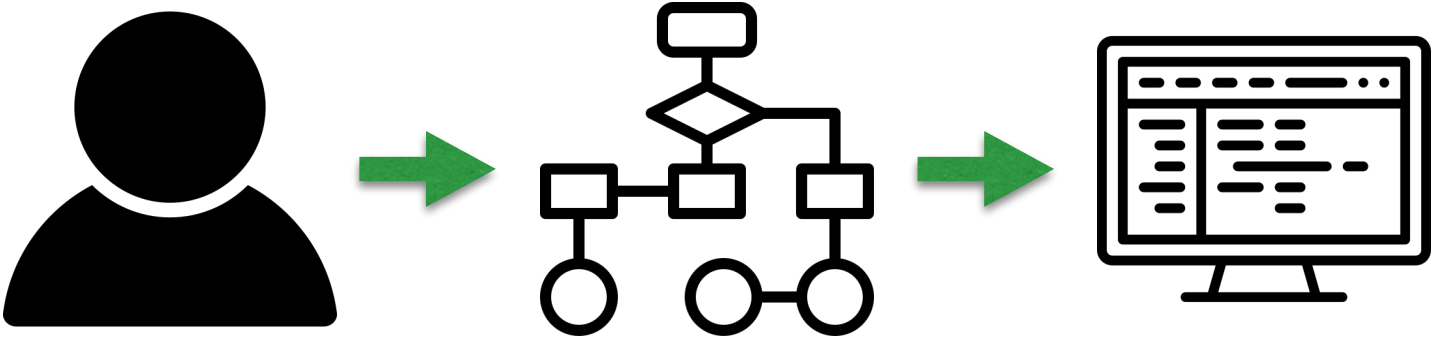
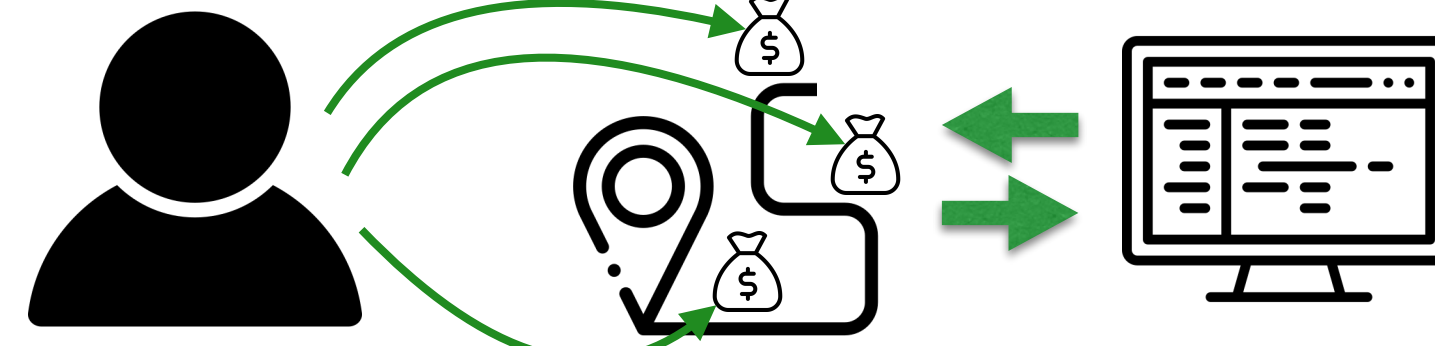
## Spatial navigation



## Dextrous manipulation



# Control preference elicitation

	Explicit	Implicit
"how"	<p><b>Programming</b></p> 	<p><b>Imitation Learning</b></p> 
"what"	<p><b>Instruction Following</b></p> 	<p><b>Reinforcement Learning</b></p> 



# Control learning is ML... but special

- In RL, unlike supervised, no ground truth, only feedback (**online learning**)
- **Exploration** = the learner collects data by interaction
  - The agent decides on which states to train (**active learning**) — and test!
  - Cannot avoid some train–test mismatch
- **Sequential decision making** need to be coordinated
  - Optimization space is strewn with **local optima**
- A good policy may require **memory**
  - Agent state is **latent** → combine control and inference



# Today's lecture

---

What is reinforcement learning?

**Course logistics**

Basic RL concepts

# Course logistics: general

- **Course website:** <https://royf.org/crs/W22/CS277>
  - Schedule; recordings; assignments; resources
- **Forum:** <https://edstem.org/us/courses/16527>
  - Announcement; discussions
- **Office hours:** <https://calendly.com/royfox/office-hours>
  - Welcome to schedule 15-min slots; individually or with classmates; 4 hour notice
- **TA:** Tiancheng Xu
  - Office hours: <https://calendly.com/tianchex/15min>



# Course logistics: lectures and discussions

---

- Lecture videos
  - ▶ Uploaded to the course website
  - ▶ Please watch before the class discussion for each topic
- Class discussions (optional)
  - ▶ When: Tuesdays and Thursdays, 11am–12:20
  - ▶ Where: <https://uci.zoom.us/j/96005379683>
  - ▶ Recorded and uploaded to the course website

# Course logistics: quizzes and assignments

---

- Quizzes

- ▶ Weekly, about that week's topics; Friday deadlines
- ▶ Discussed the following Tuesday in class

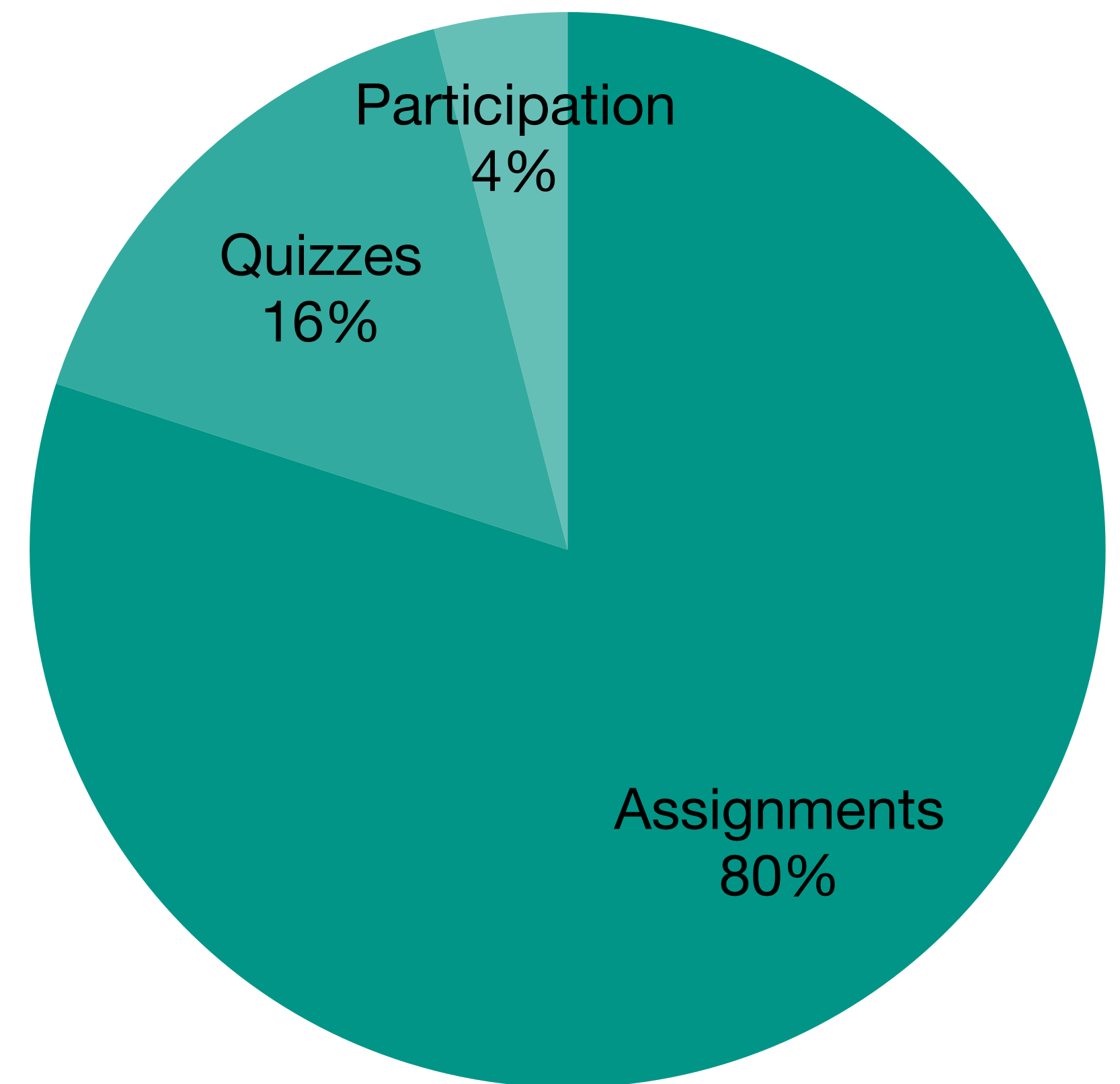
- Assignments

- ▶ Roughly every other week
- ▶ Understand RL theory; apply RL techniques in Python
- ▶ Discussed the following Thursday in class

- Submission: <https://www.gradescope.com/courses/342903>

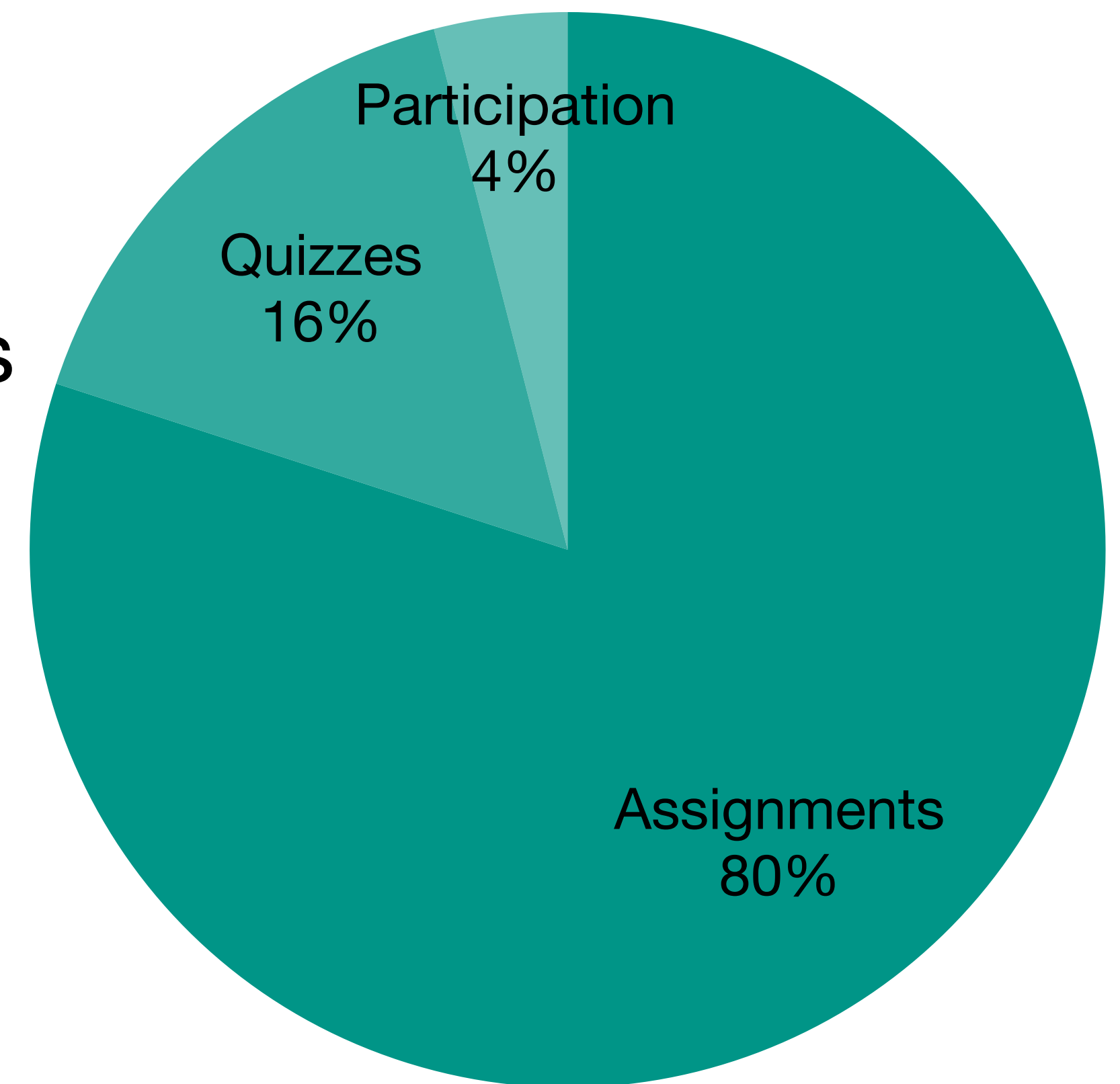
# Grading policy: assignments

- Show your math, code, and results
- Encouraged to discuss with me or classmates
  - But solve yourself
- 4 best of 5 assignments count for 20% each
- 5% bonus for scoring at least 50% on all 5
- Late submission: 5 grace days total



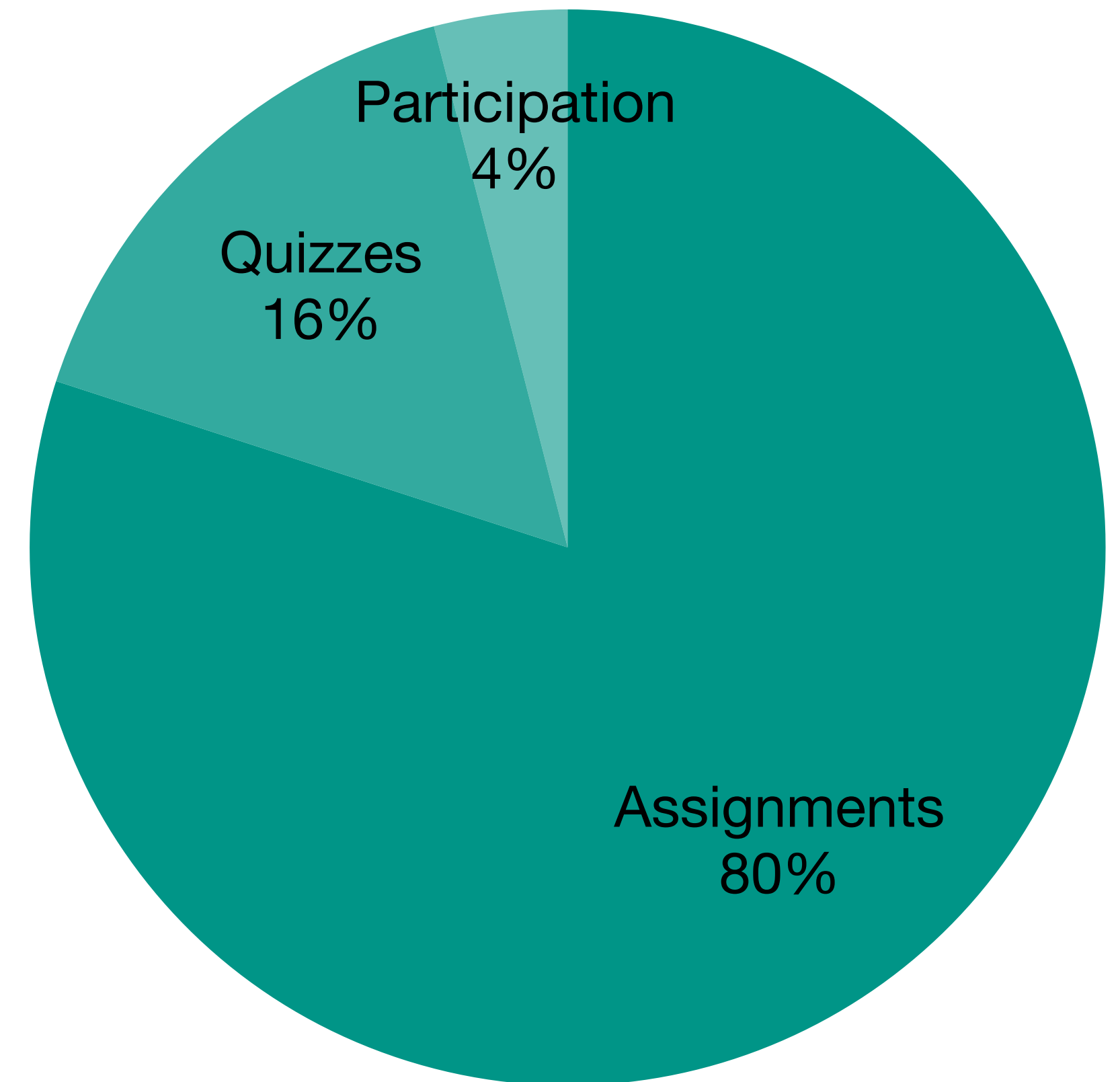
# Grading policy: quizzes

- Review the week's topics, think about them a bit
- 9 quizzes, 2% each, up to 16% total + 2% bonus
  - 1% for submitting a complete quiz
  - 1% for being not too much off (“passing”)
- Late submission: up to 2 Monday submissions



# Grading policy: participation

- Forum participation: 2%
  - ▶ Ask questions if you have any
  - ▶ Answer questions if you can
  - ▶ Share thoughtful comments
  - ▶ Post relevant useful links
  - ▶ Be on-topic (excluding administrative)
- Course evaluations: 2%





# What will it take to do well?

---

- We'll rely heavily on math: probability theory, linear algebra, calculus
  - I'm here to help, but solid background expected
- You'll need to code well in Python
- Some ideas are challenging — ask early what you don't fully understand
- Help your friends and get help — from me too! — but never cheat



# Today's lecture

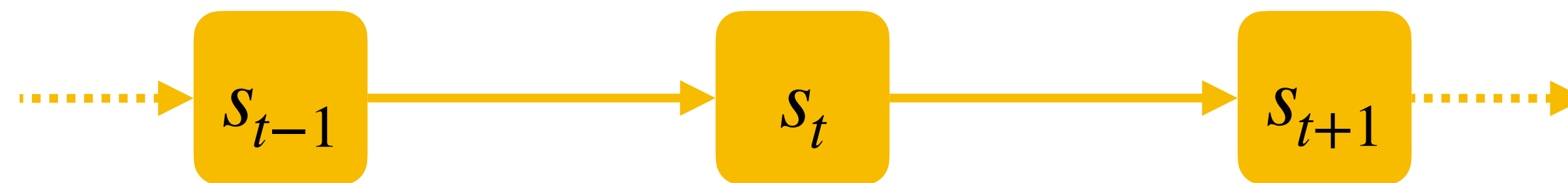
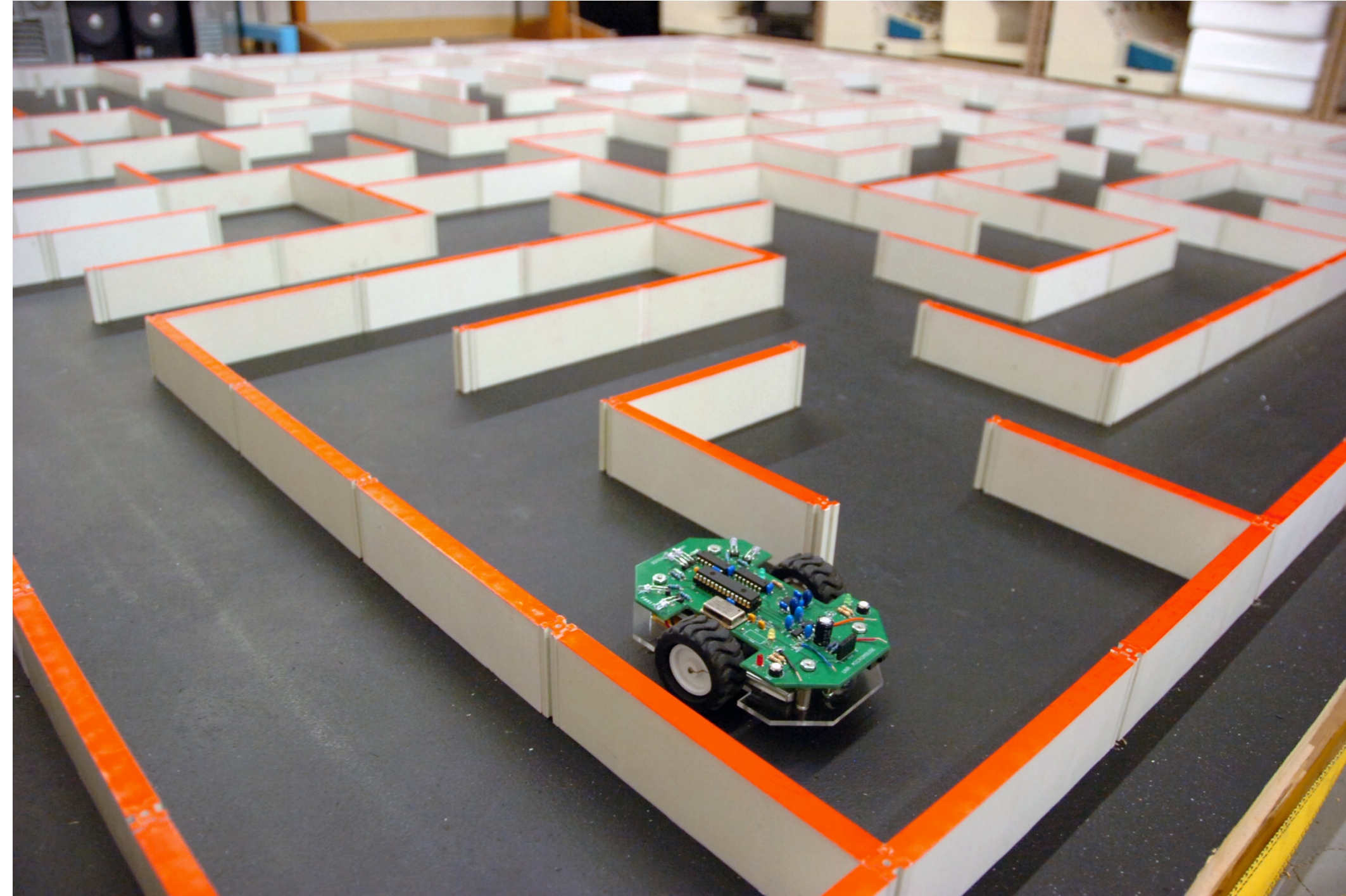
---

What is reinforcement learning?

Course logistics

**Basic RL concepts**

# System state



# System state

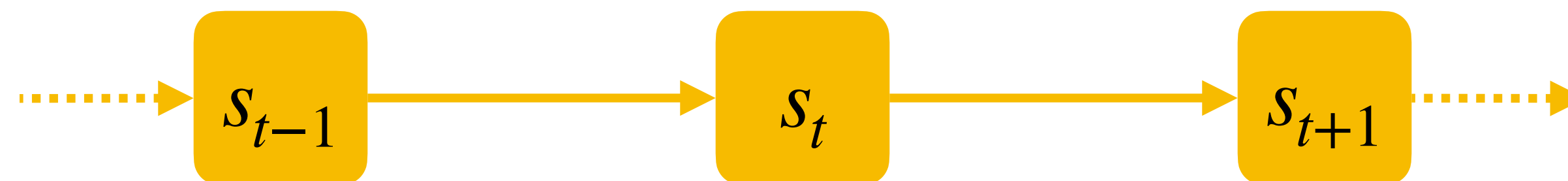
- **Markov property**: the future is independent of the past, given the present

$$p(s_{t+1}, s_{t+2}, \dots | s_0, s_1, \dots, s_t) = p(s_{t+1}, s_{t+2}, \dots | s_t)$$

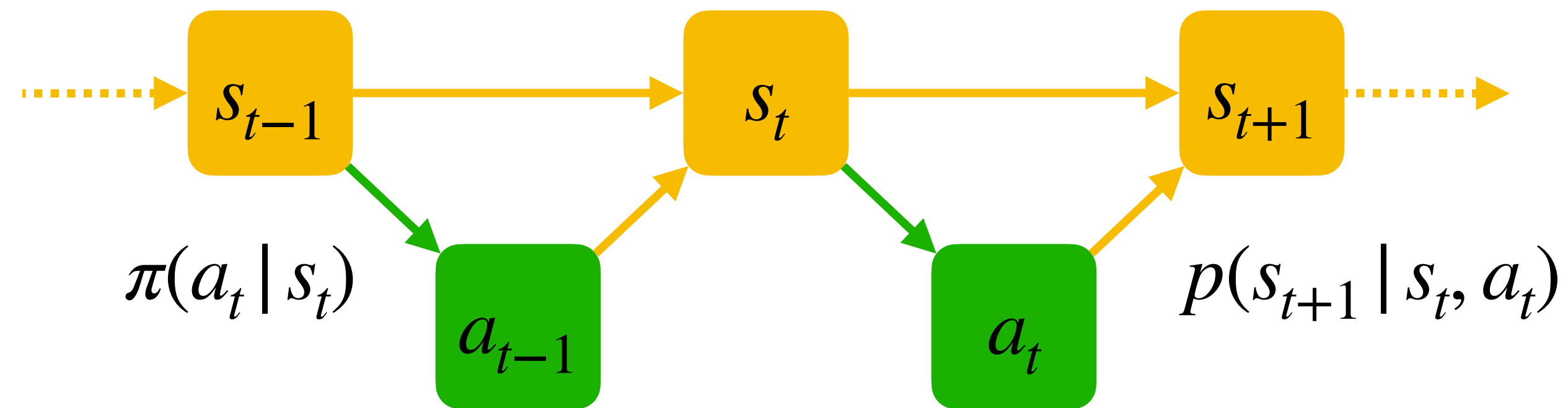
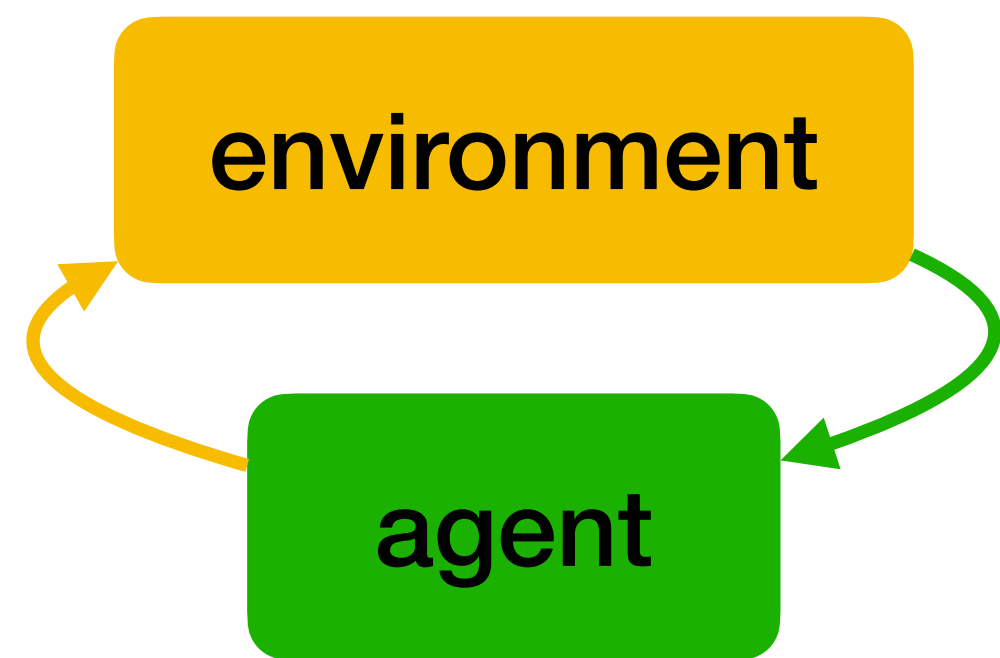
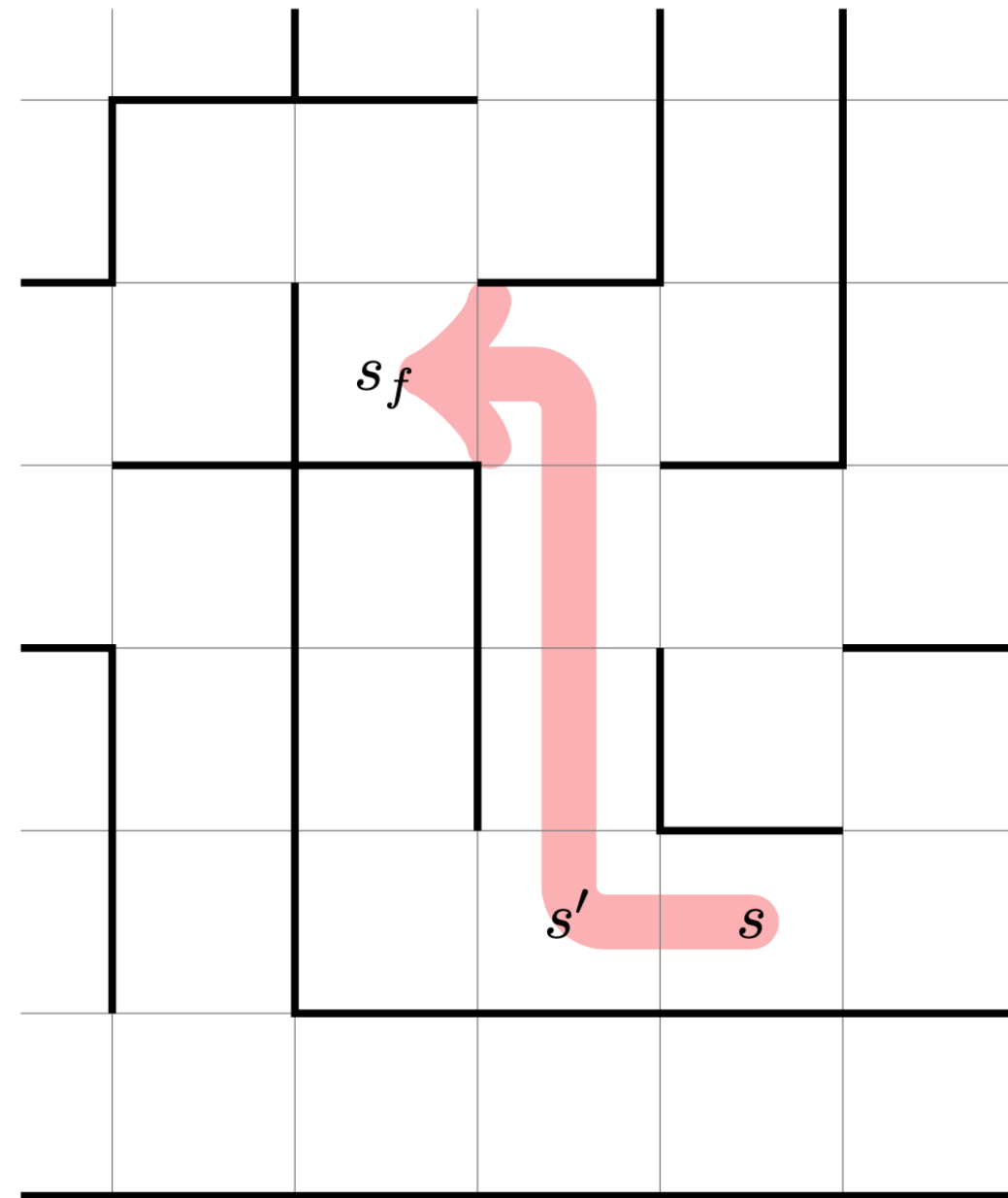
- **State** = all relevant information from history

↙  
**for future!**

- ▶ Given  $s_t$ , the **history**  $h = (s_0, \dots, s_t)$  and the **future**  $(s_{t+1}, s_{t+2}, \dots)$  are independent

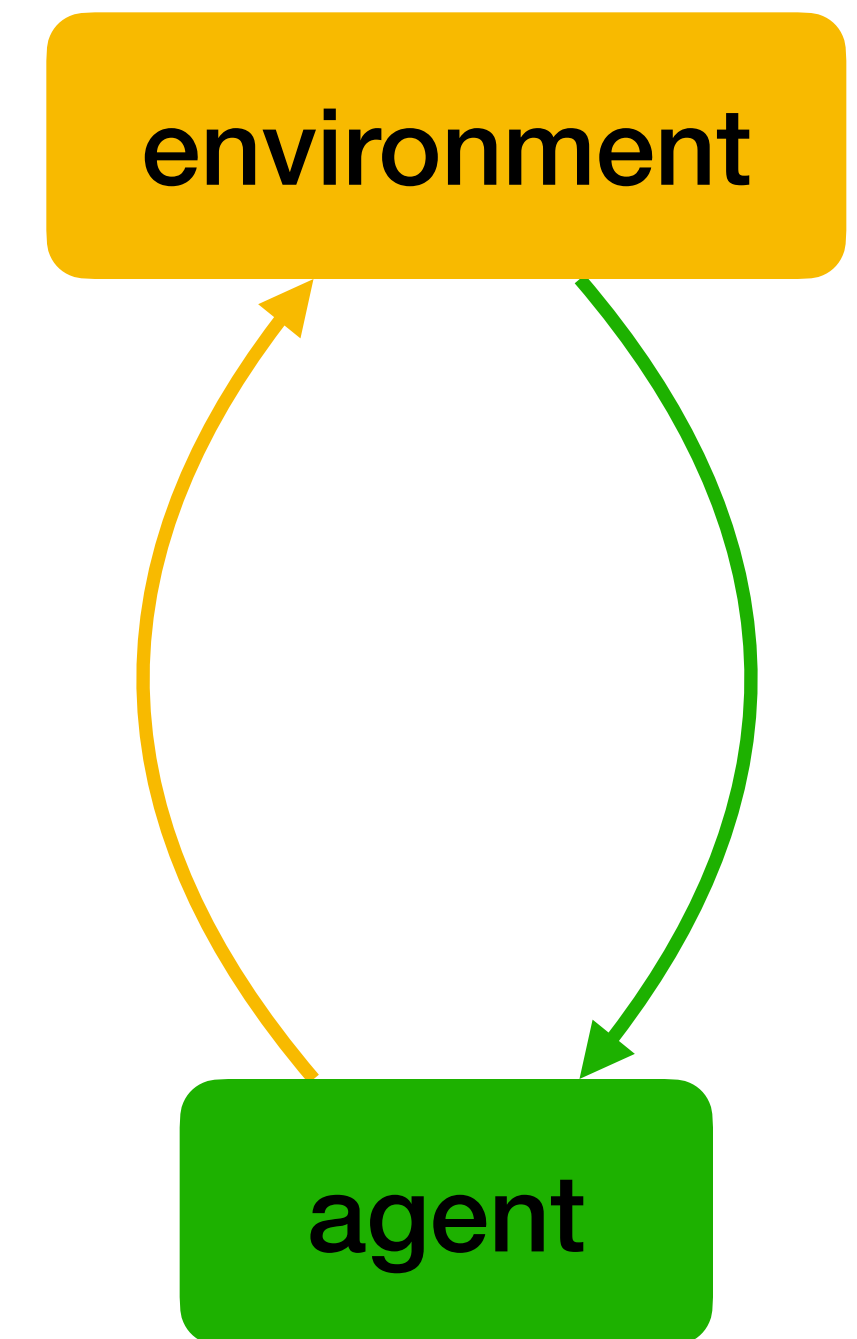


# System = agent + environment



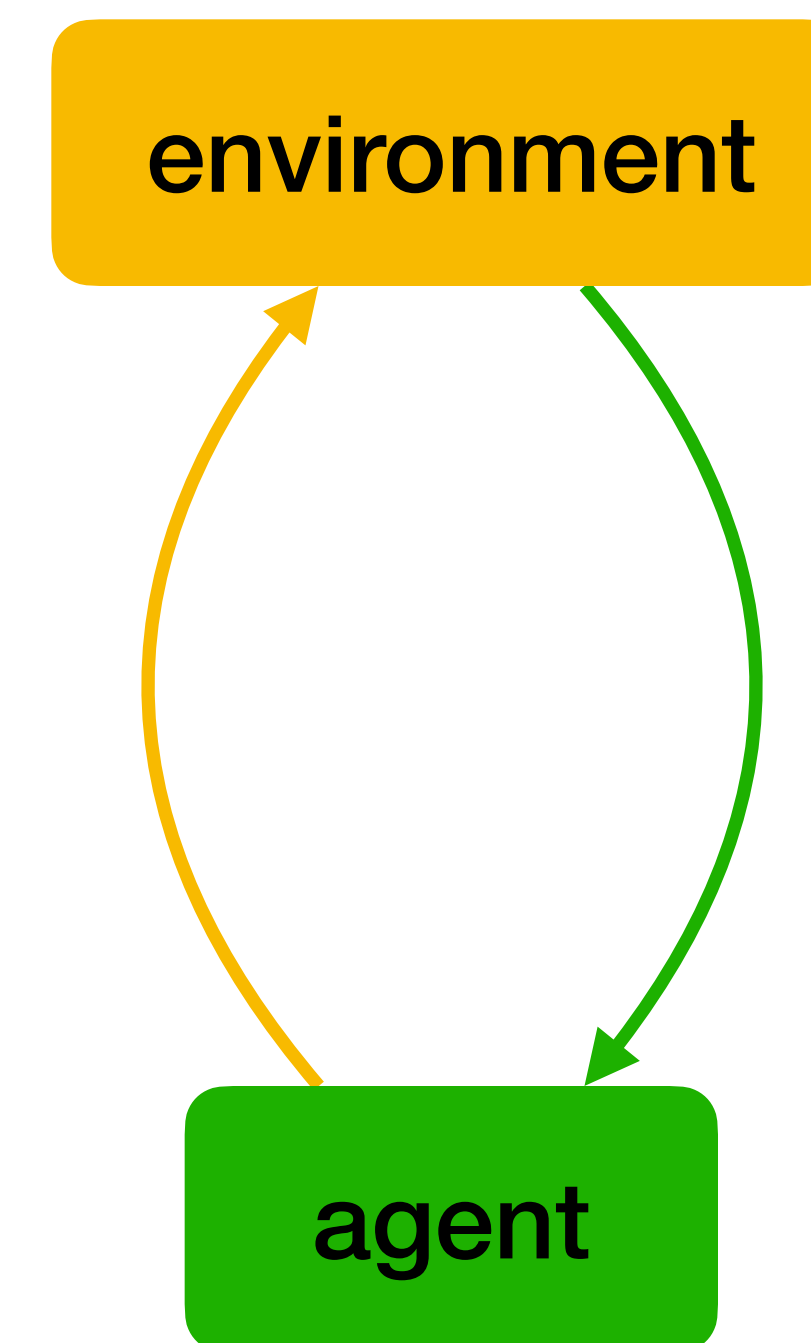
# Markov Decision Process (MDP)

- Model of environment
  - $S$  = set of **states**
  - $A$  = set of **actions**
  - $p(s' | s, a)$  = state **transition** probability
    - Probability that  $s_{t+1} = s'$ , if  $s_t = s$  and  $a_t = a$



# Agent policy

- “Model” of agent decision-making
  - ▶ **policy**  $\pi(a | s)$  = probability of taking action  $a_t = a$  in state  $s_t = s$
  - ▶ In MDP, action  $a_t$  only depends on current state  $s_t$ :
    - **Markov property** =  $s_t$  is all that matters in history
    - **Causality** = cannot depend on the future
  - ▶ Should the policy **depend on time**?  $\pi_t : s_t \mapsto a_t$ 
    - Sometimes; can add  $t$  as feature:  $s_t \rightarrow (t, s_t)$

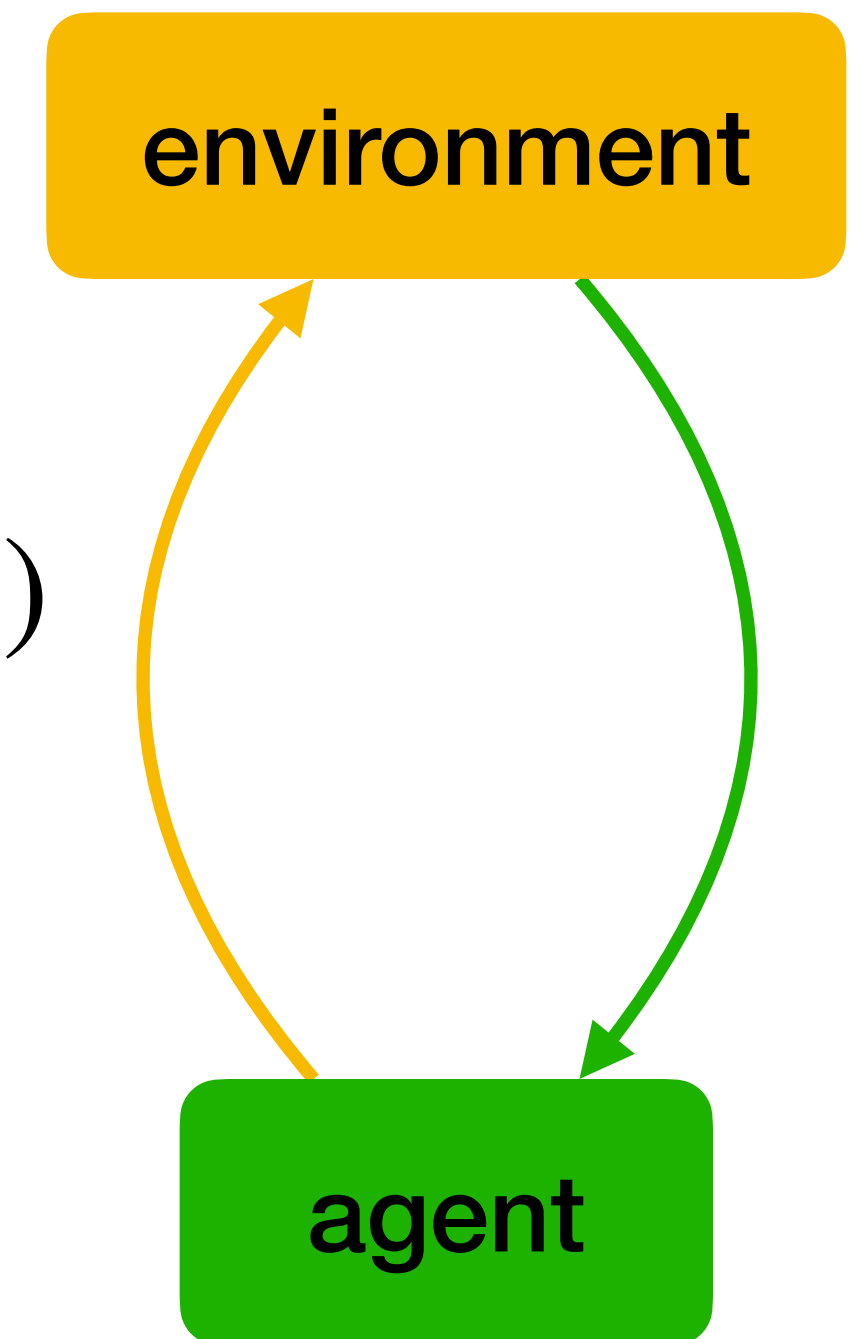


# Trajectories

- The agent's behavior iteratively uses (**rolls out**) the policy
- **Trajectory**:  $\xi = (s_0, a_0, s_1, a_1, \dots, s_T)$
- MDP + policy induce **distribution over trajectories**


$$\begin{aligned} p_{\pi}(\xi) &= p(s_0)\pi(a_0 | s_0)p(s_1 | s_0, a_0)\cdots\pi(a_{T-1} | s_{T-1})p(s_T | s_{T-1}, a_{T-1}) \\ &= p(s_0) \prod_{t=0}^{T-1} \pi(a_t | s_t)p(s_{t+1} | s_t, a_t) \end{aligned}$$

- **Imitation learning**: learn from dataset of expert demonstrations
  - **Supervised learning** of  $\pi(a | s)$  from “labeled” states  $(s_t, a_t)$





# Learning from rewards

- Providing demonstrations is hard
  - Particularly for learner-generated trajectories
- Can the teacher just **score** learner actions?  **as in online learning**
  - **Reward**:  $r(s, a) \in \mathbb{R}$
- High reward is positive **reinforcement** for this behavior (in this state)
  - Much closer to how natural agents learn
  - Designing and **programming**  $r$  often easier than programming / demonstrating  $\pi$

# Actions have long-term consequences

---

- Tradeoff: **short-term rewards** vs. **long-term returns** (accumulated rewards)
  - ▶ Fly drone: **slow down** to **avoid crash**?
  - ▶ Games: **slowly** build **strength**? block opponent? all out attack?
  - ▶ Stock trading: **sell now** or wait for **growth**?
  - ▶ Infrastructure control: **reduce power output** to **prevent blackout**?
  - ▶ Life: **invest** in college, obey **laws**, get started **early** on course project
- Forward thinking and planning are hallmarks of **intelligence**

# Returns

- **Return** = total reward =  $R = \sum_{t \geq 0} \gamma^t r(s_t, a_t)$ 
  - Summarize reward sequence  $r_t = r(s_t, a_t)$  as single number to be **maximized**
- **Discount factor**  $\gamma \in [0, 1]$ 
  - Higher **weight** to short-term rewards (and costs) than long-term
  - Good mathematical properties:
    - Assures **convergence**, simplifies algorithms, reduces variance
- Vaguely economically motivated (inflation)

# Horizon classes

• Finite:  $R(\xi) = \sum_{t=0}^{T-1} r(s_t, a_t)$

• Infinite:  $R(\xi) = \lim_{T \rightarrow \infty} \frac{1}{T} \sum_{t=0}^{T-1} r(s_t, a_t)$

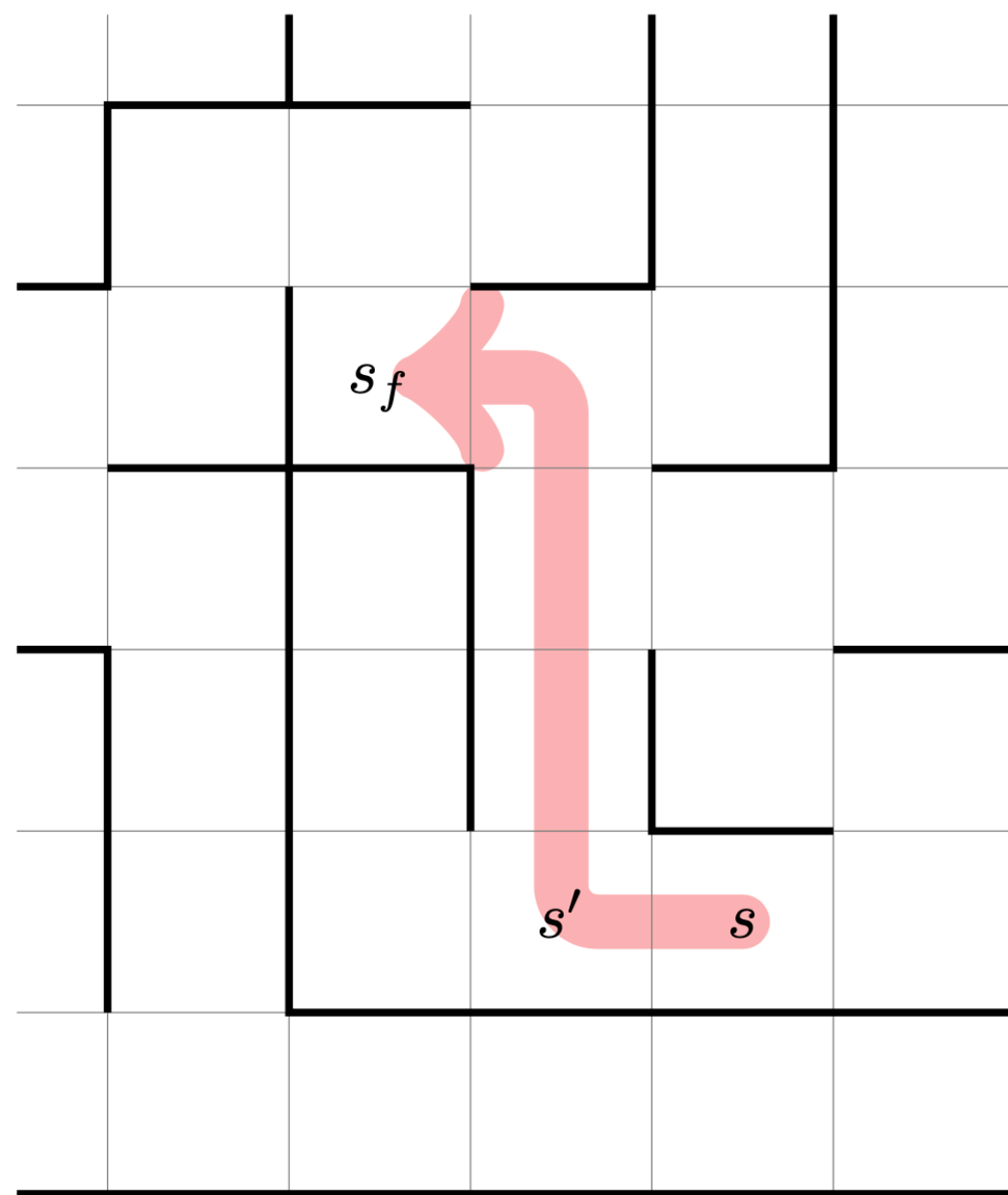
• Discounted:  $R(\xi) = \sum_{t=0}^{\infty} \gamma^t r(s_t, a_t) \quad 0 \leq \gamma < 1$

• Episodic:  $R(\xi) = \sum_{t=0}^{T-1} r(s_t, a_t) \quad \text{s.t. } s_T = s_f$

# Basic RL concepts

- **State:**  $s \in S$ ; **action:**  $a \in A$ ; **reward:**  $r(s, a) \in \mathbb{R}$
- **Dynamics:**  $p(s_{t+1} | s_t, a_t)$  for stochastic;  $s_{t+1} = f(s_t, a_t)$  for deterministic
- **Policy:**  $\pi(a_t | s_t)$  for stochastic;  $a_t = \pi(s_t)$  for deterministic
- **Trajectory:**  $p_\pi(\xi = s_0, a_0, s_1, a_1, \dots) = p(s_0) \prod_t \pi(a_t | s_t) p(s_{t+1} | s_t, a_t)$
- **Return:**  $R(\xi) = \sum_t \gamma^t r(s_t, a_t) \quad 0 \leq \gamma < 1$
- **Value:**  $V(s) = \mathbb{E}_{\xi \sim p_\pi}[R | s_0 = s]$   
 $Q(s, a) = \mathbb{E}_{\xi \sim p_\pi}[R | s_0 = s, a_0 = a]$

# Special case: shortest path



- **Deterministic dynamics:** in state  $s$ , take action  $a$  to get to state  $s' = f(s, a)$ 
  - Example above:  $s' = f(s, a_{\text{left}})$
- **Reward:**  $(-1)$  in each step (until the goal  $s_f$  is reached)

# Shortest path: optimality principle

- **Proposition:** if  $\xi$  is a shortest path from  $s$  to  $s_f$  that goes through  $s'$ , then a suffix of  $\xi$  is a shortest path from  $s'$  to  $s_f$
- **Proof:** otherwise, let  $\xi'$  be a shorter path from  $s'$  to  $s_f$ , then take  $s \xrightarrow{\xi} s' \xrightarrow{\xi'} s_f$
- It follows that for all  $s \neq s_f$  
$$V(s) = \min_a (1 + V(f(s, a)))$$
- The optimal policy is 
$$\pi(s) = \arg \min_a (1 + V(f(s, a)))$$

---

## Algorithm 1 Bellman-Ford

---

$$V(s_f) \leftarrow 0$$

$$V(s) \leftarrow \infty \quad \forall s \in S \setminus \{s_f\}$$

**for**  $\ell$  from 1 to  $|S| - 1$  **do**

$$V(s) \leftarrow \min_{a \in A} \{1 + V(f(s, a))\} \quad \forall s \in S \setminus \{s_f\}$$

---

# Logistics

---

logistics

- Follow announcements and discussions on [ed](#)
- See [website](#) for schedule, recordings, resources, etc.

assignments

- Quiz 1 due [this Friday](#)
- Assignment 1 to be published soon