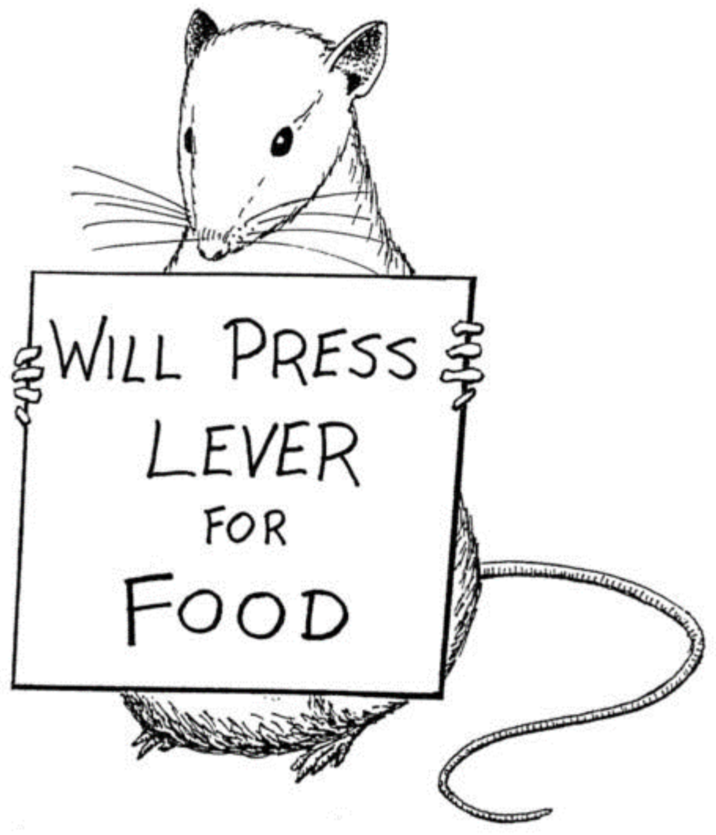
CS 277: Control and Reinforcement Learning

Winter 2024

Lecture 1: Introduction

Roy Fox
Department of Computer Science
School of Information and Computer Sciences
University of California, Irvine



Today's lecture

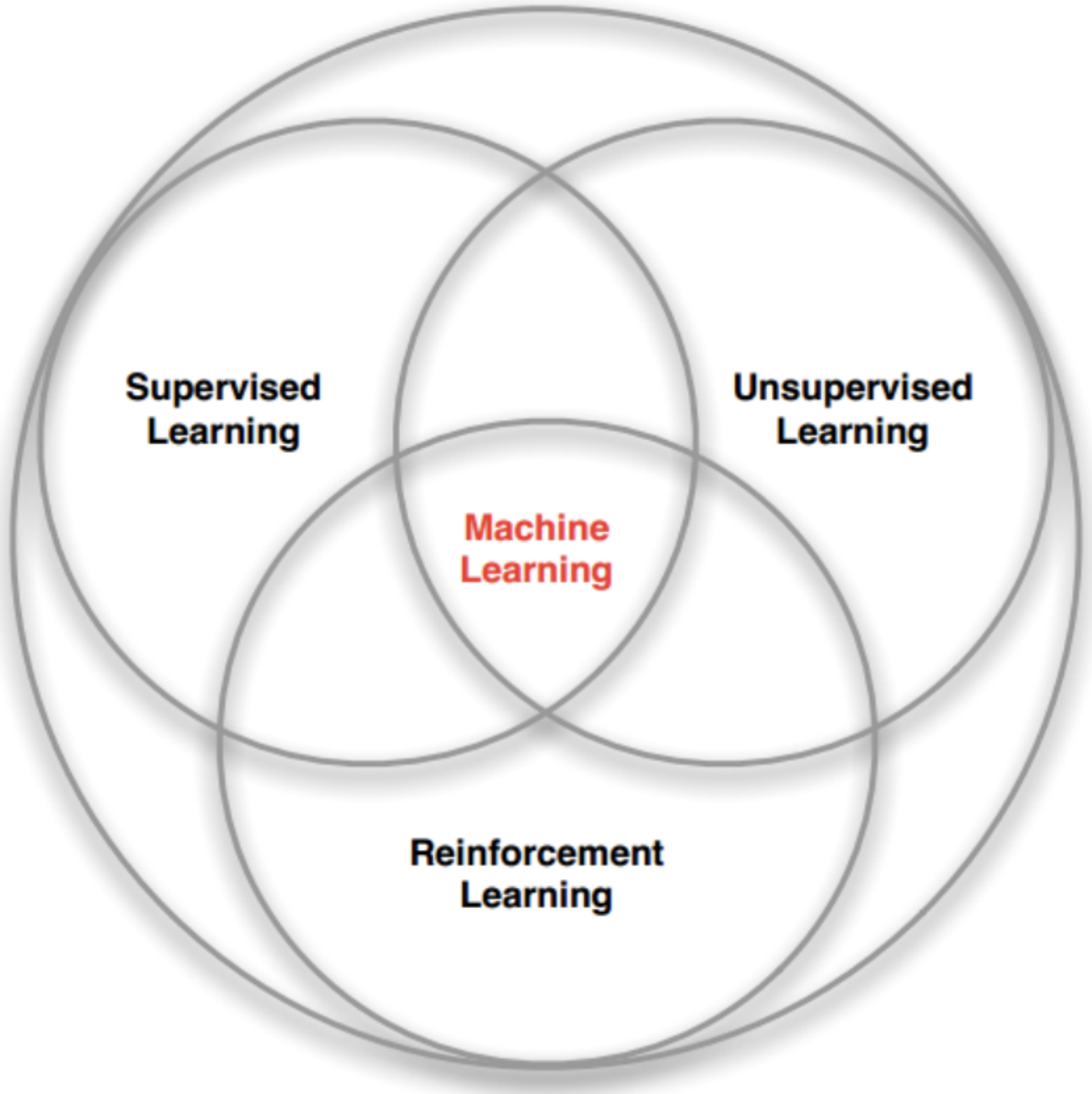
What is reinforcement learning?

Course logistics

Why is RL interesting?

RL ⊆ control learning ⊆ ML

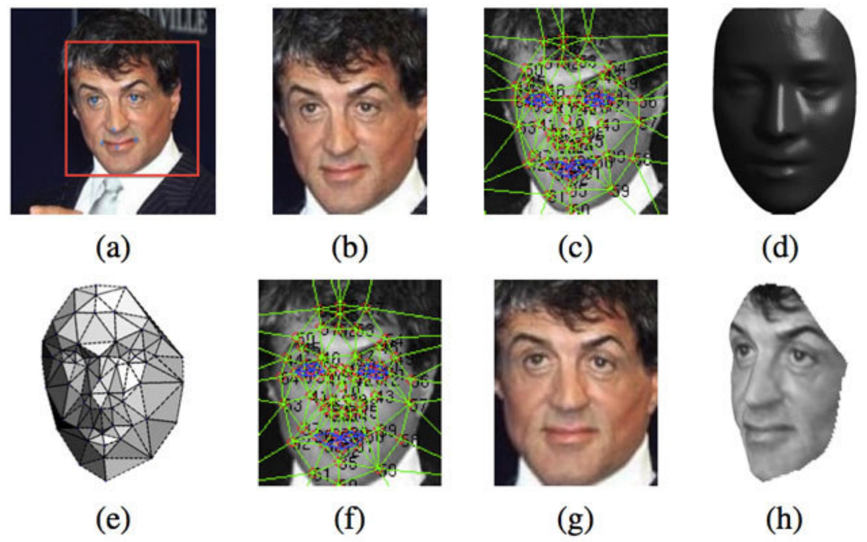
- Reinforcement Learning = learning from reinforcement (rewards)
  - But it came to encompass many settings of learning to control
  - Distinguished by sequential decision making and learning
- Many consider RL a separate ML paradigm, but it can involve:
  - Supervised learning
  - Unsupervised learning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

Face recognition



Speech synthesis

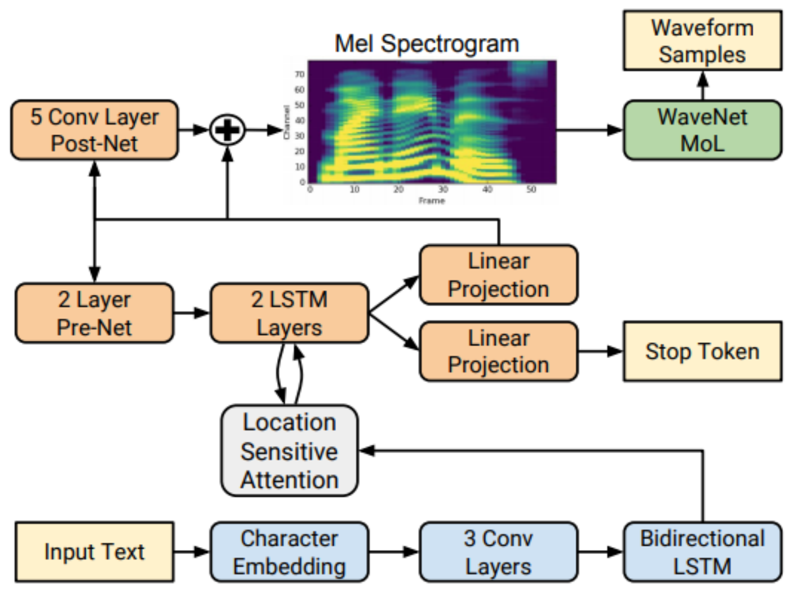
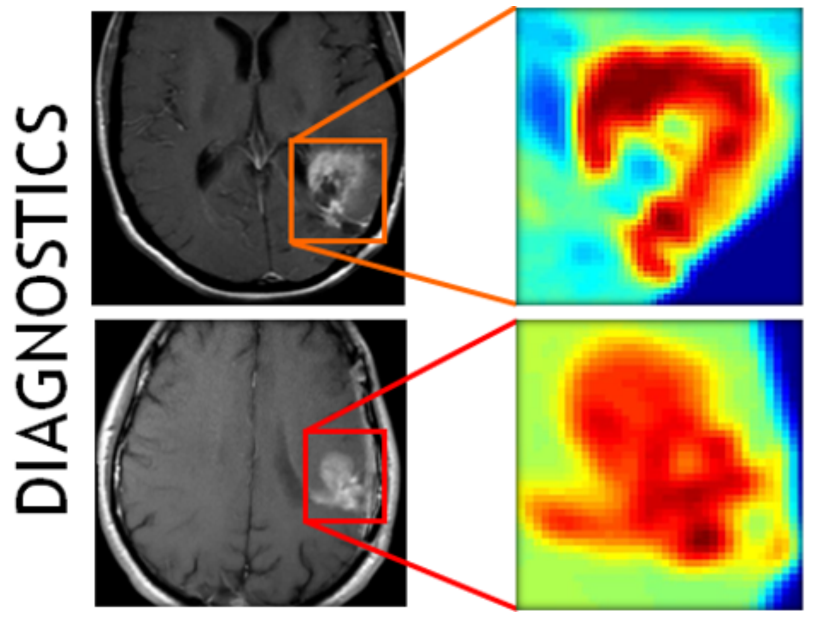


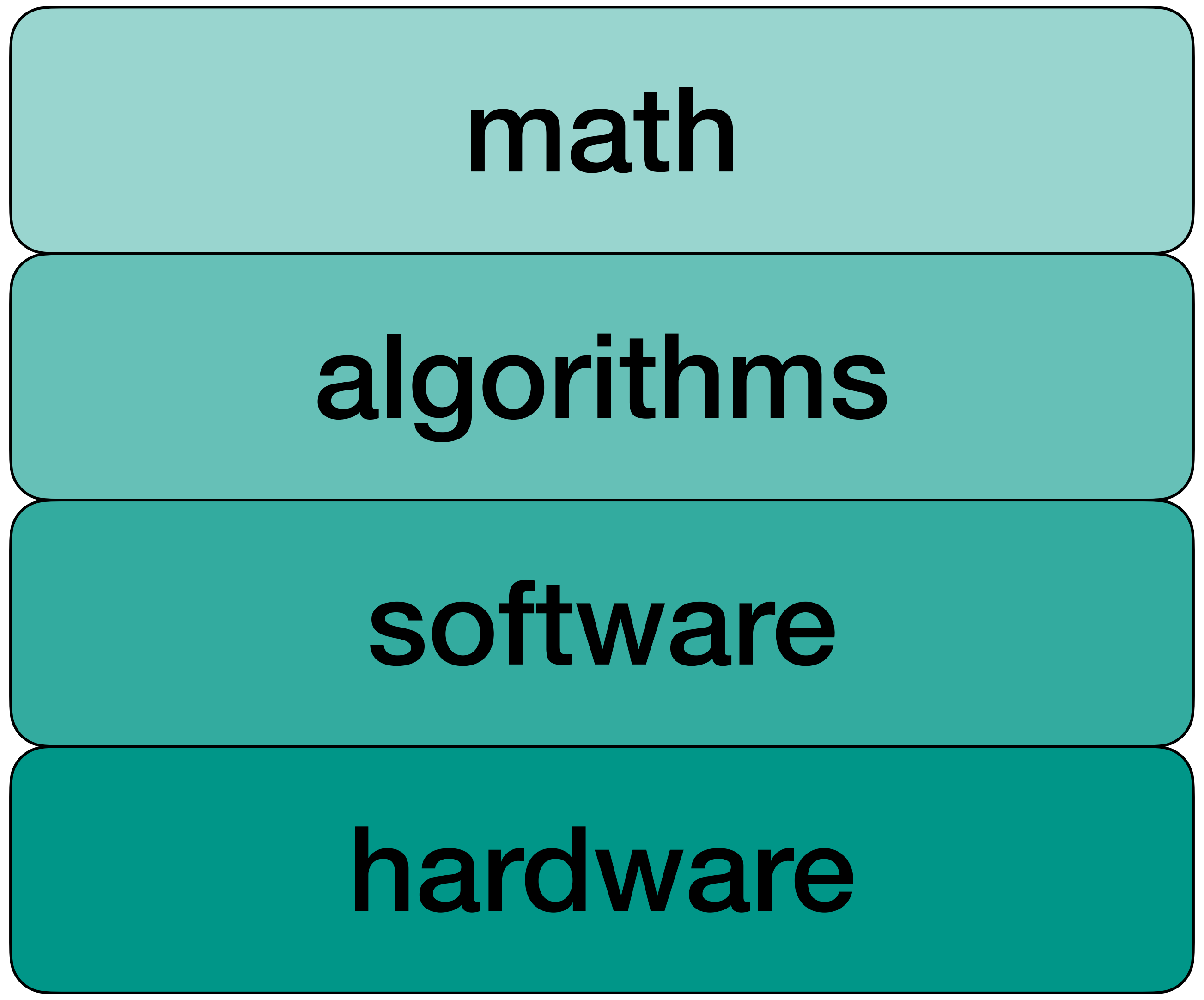
Fig. 1. Block diagram of the Tacotron 2 system architecture.

Medical diagnosis



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

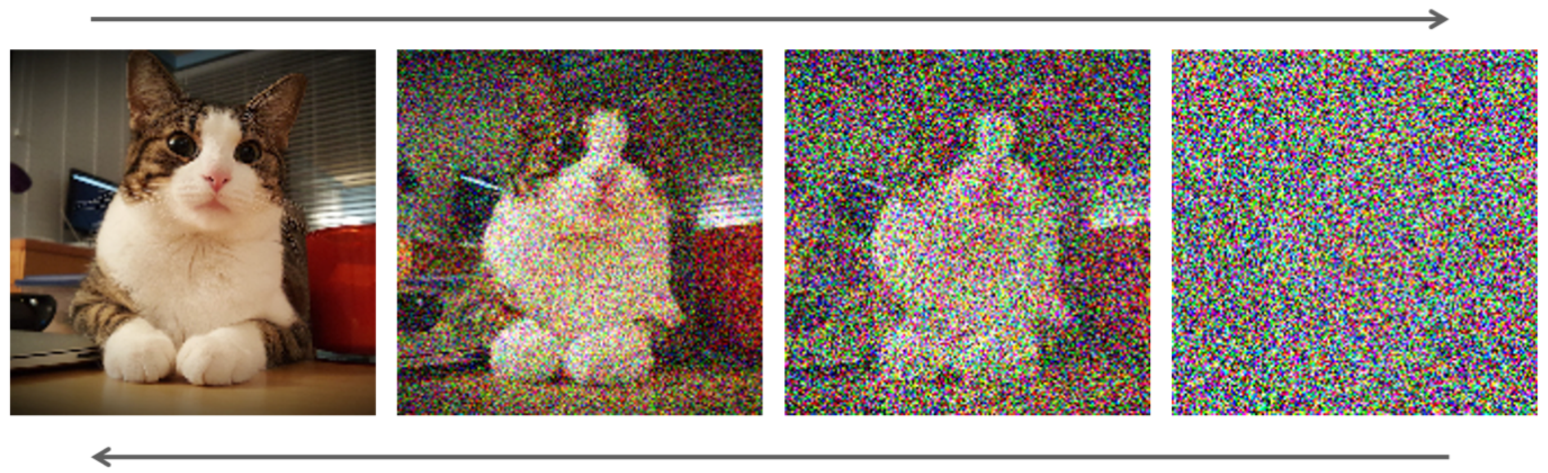
The ML stack



- 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

ML success stories

Image generation



Language generation

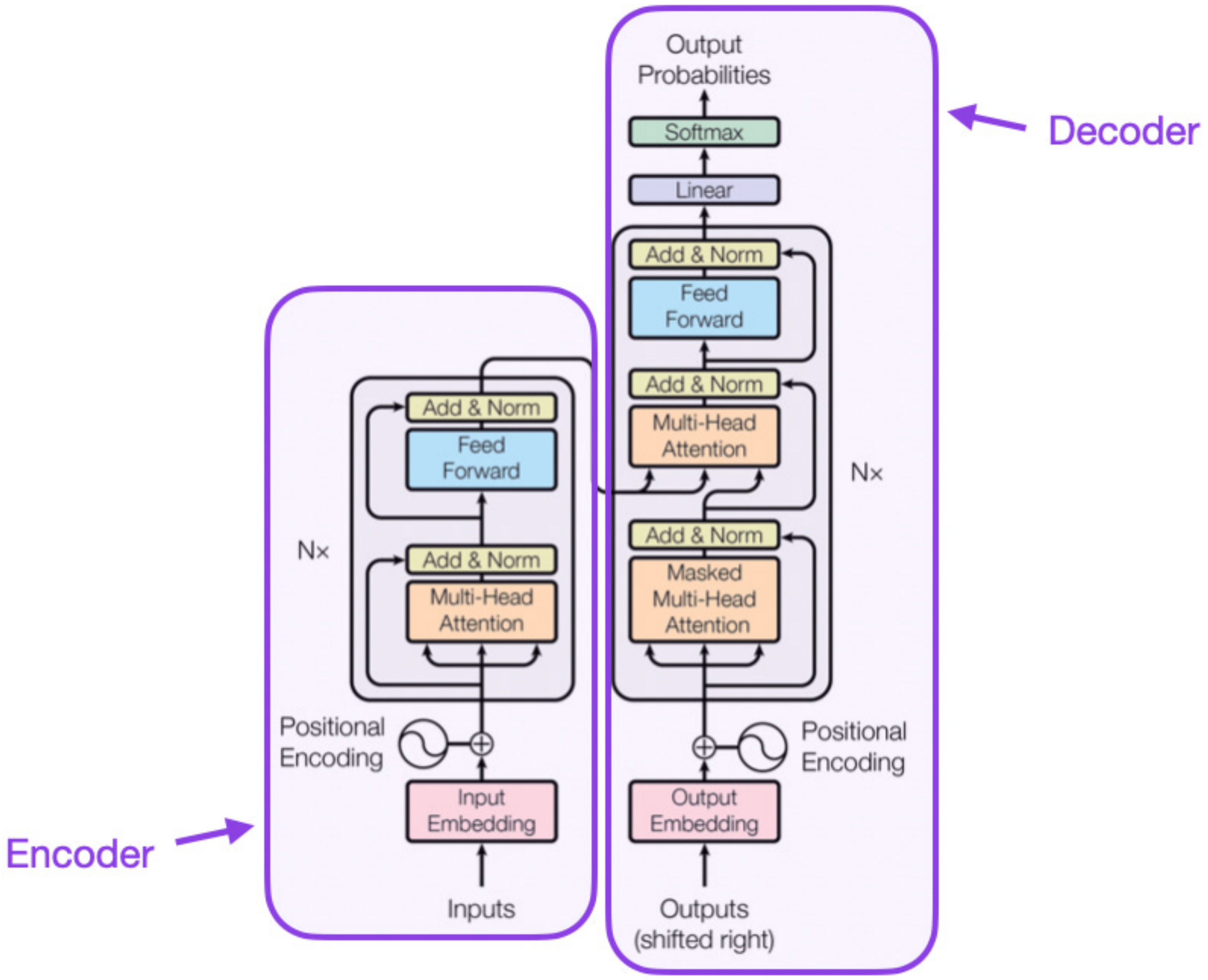
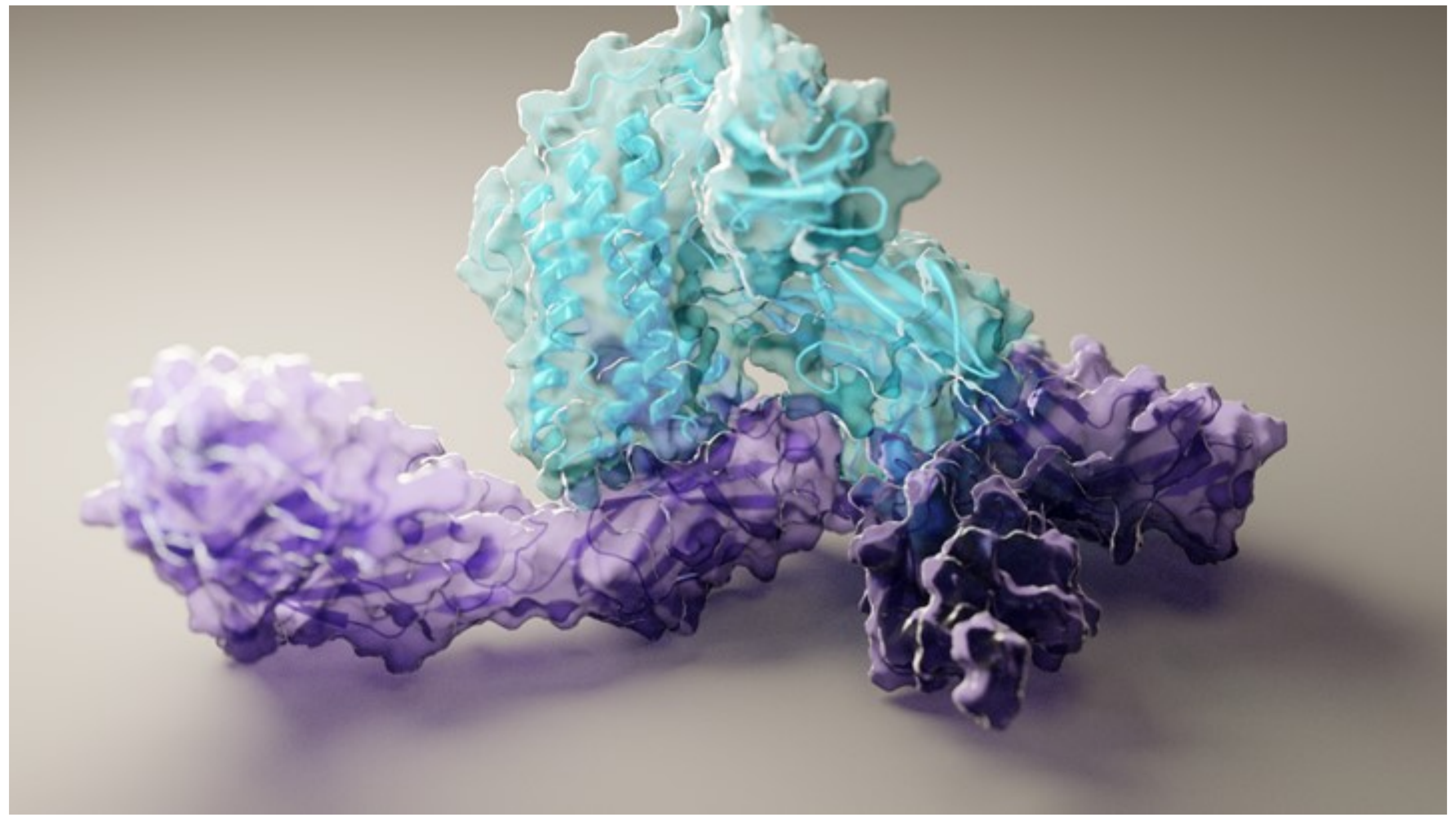


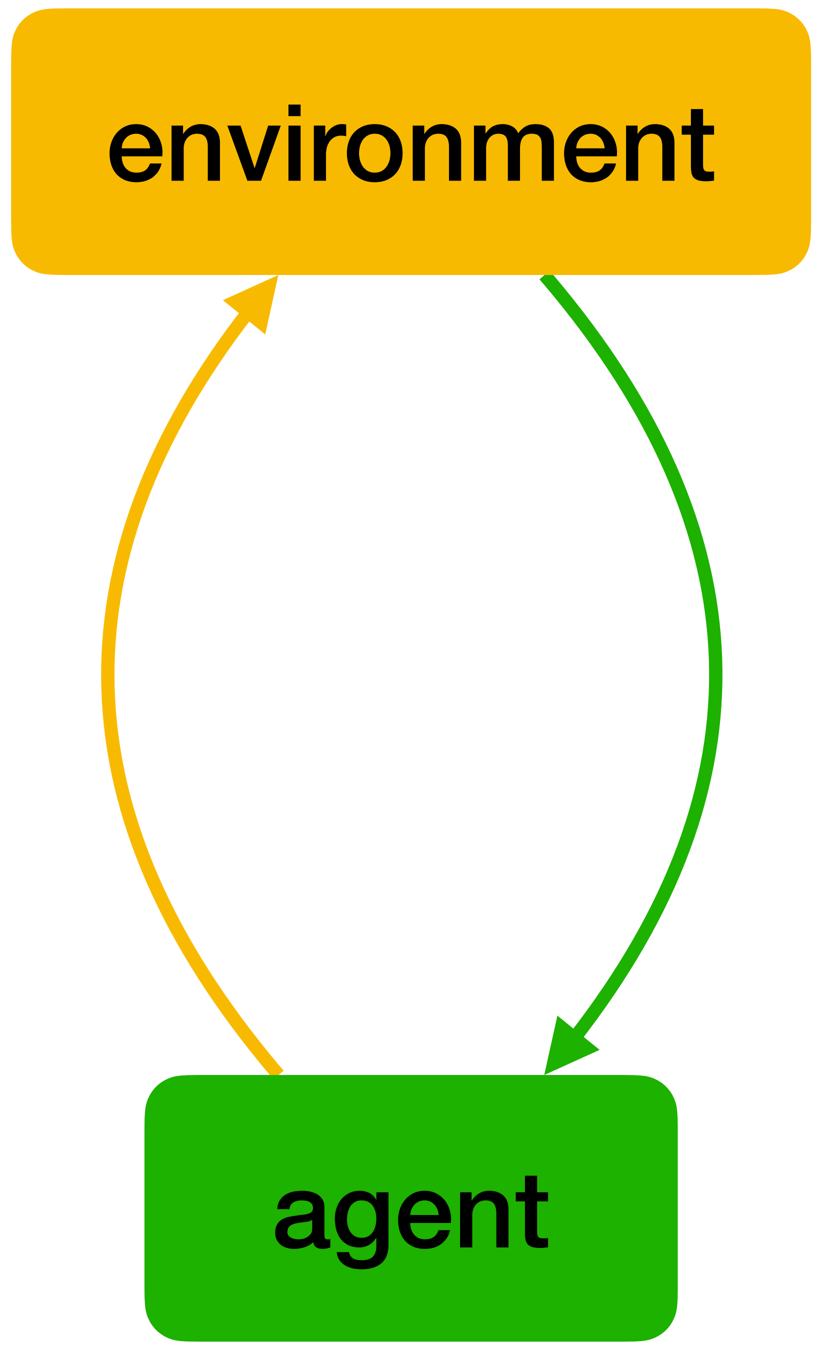
Figure 1: The Transformer - model architecture.

Protein folding



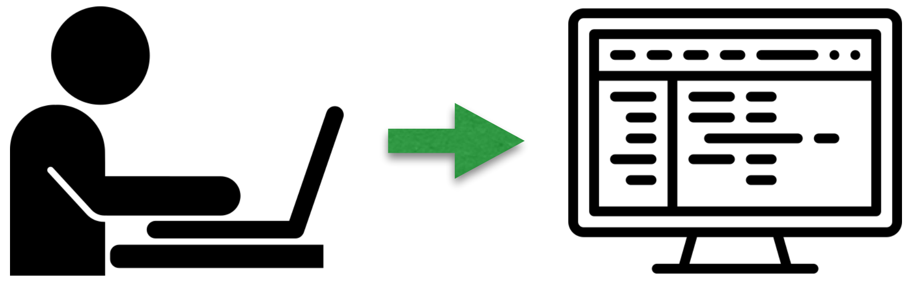
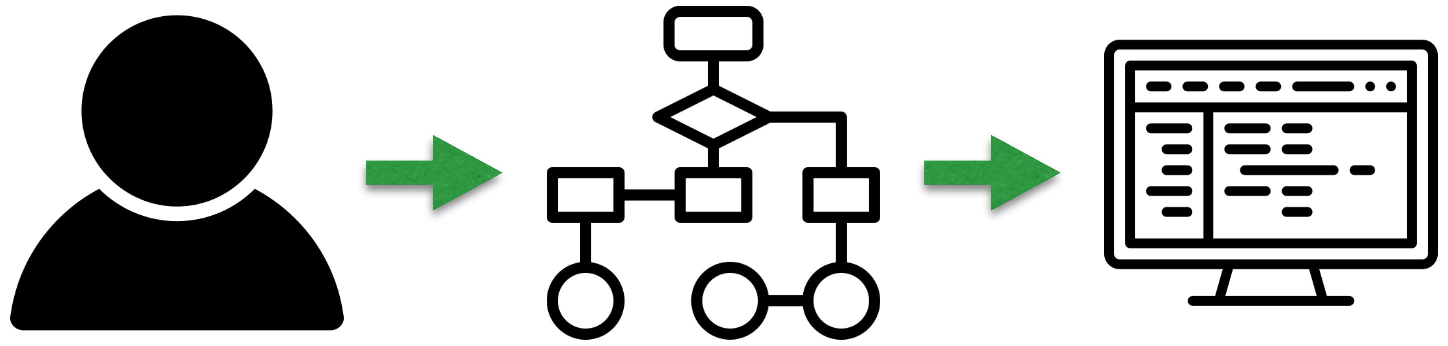
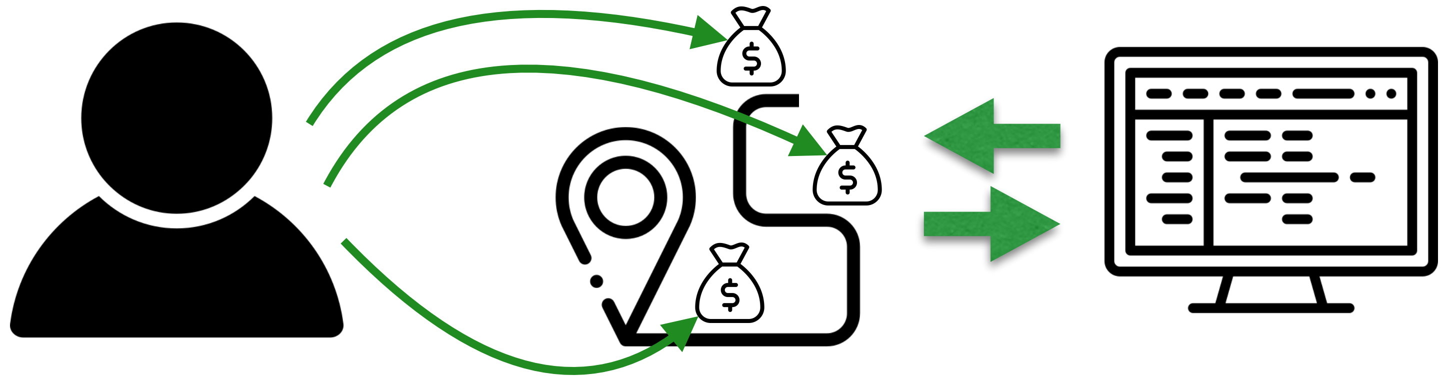
What is control learning (CL)?

- 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



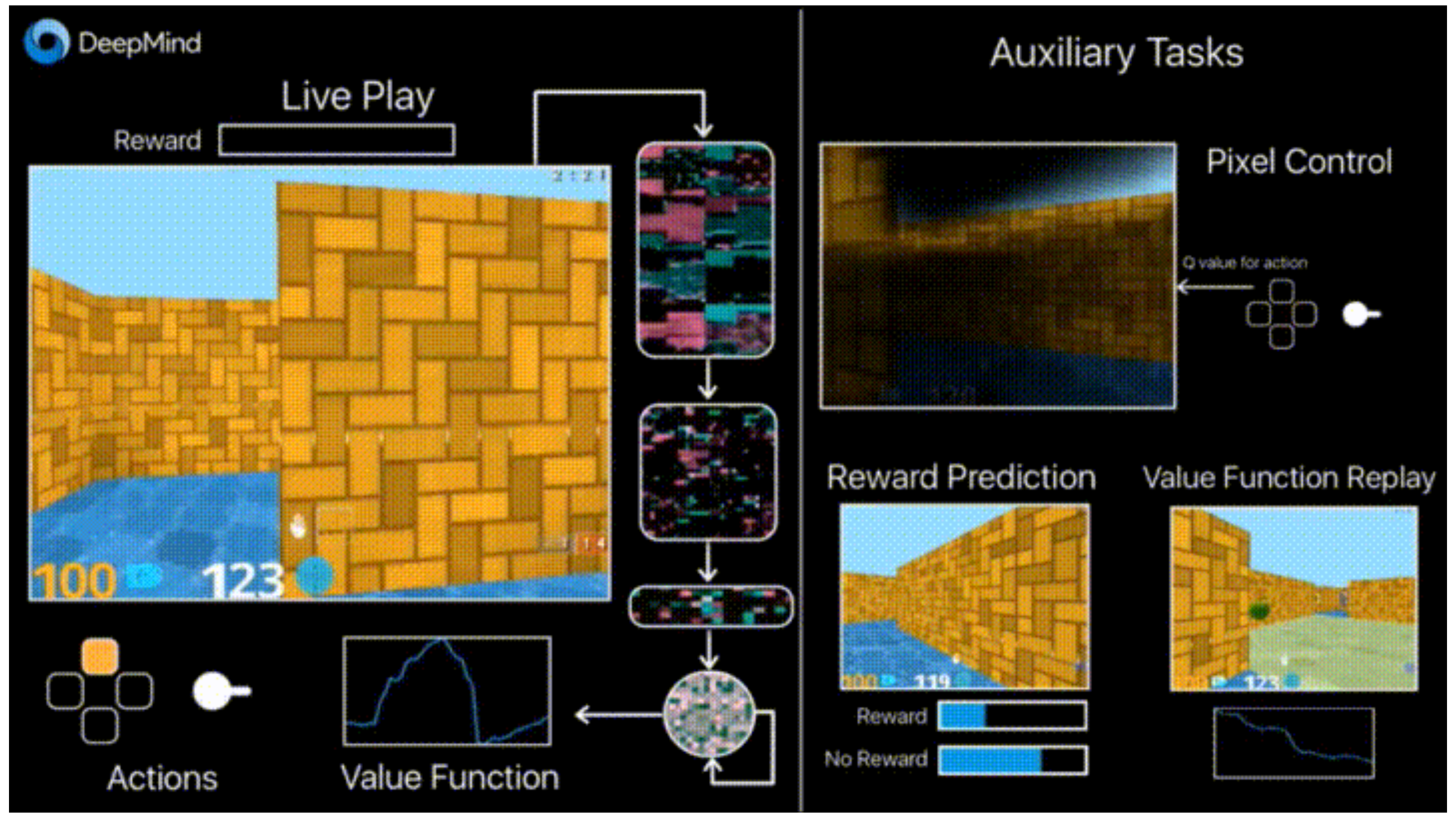
- Success can be measured by matching good actions — imitation learning (IL)
  - Or by accumulating high rewards $r(s, a)$ — reinforcement learning (RL)

Control preference elicitation

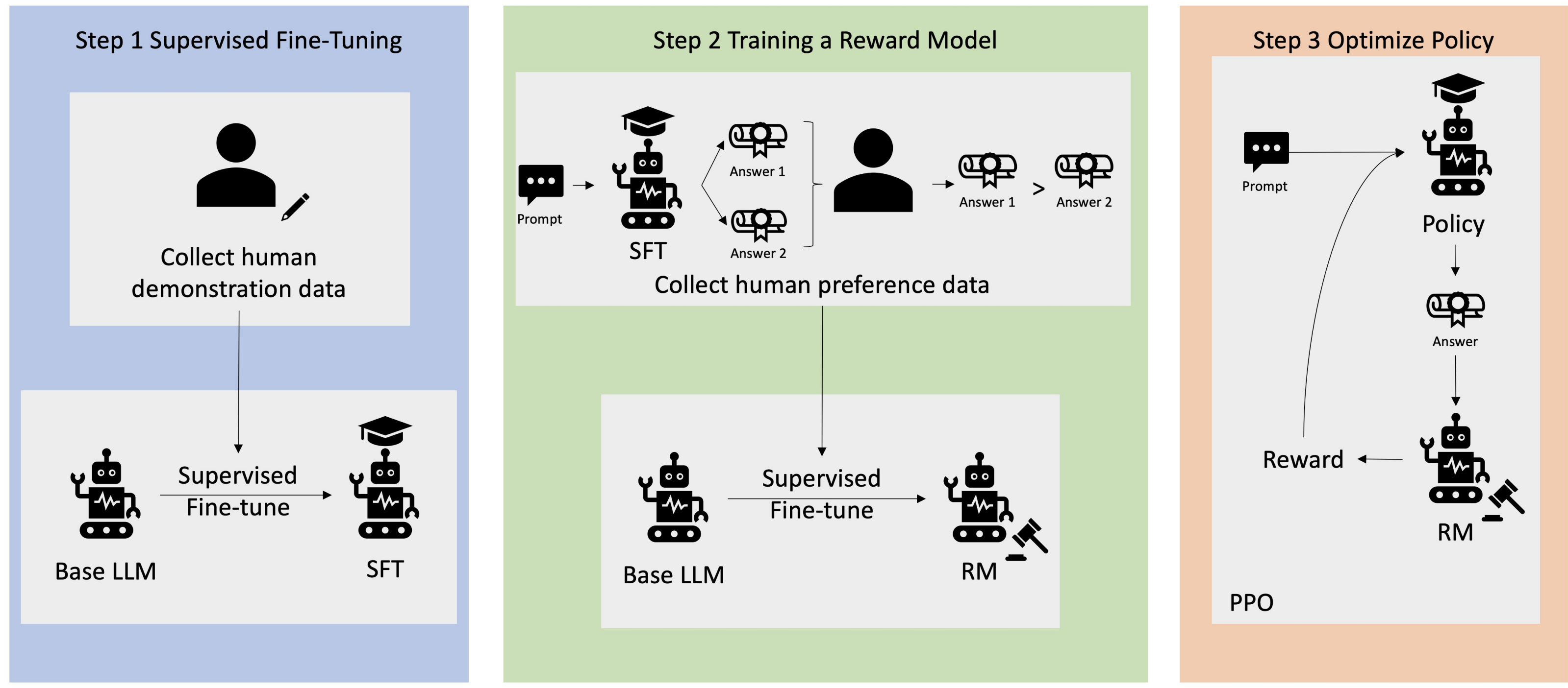
	Explicit	Implicit
"how"	**Programming** 	**Imitation Learning** 
"what"	**Instruction Following** 	**Reinforcement Learning** 

RL success stories

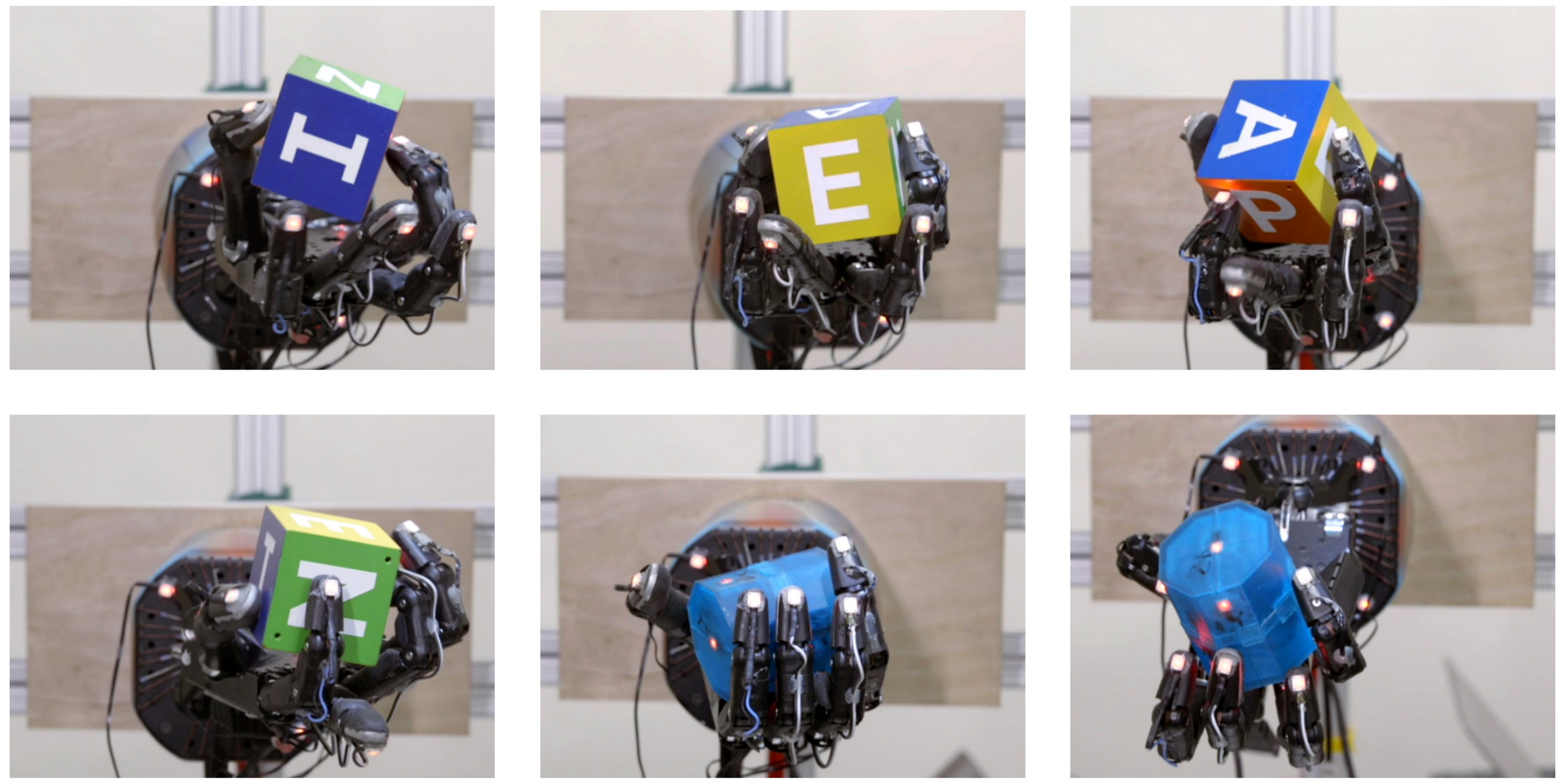
Spatial navigation



Generator fine-tuning



Dextrous manipulation



RL 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

Why is RL interesting?

Course logistics: general

- Course website: https://royf.org/crs/CS277/W24
  - Schedule; recordings; exercises; resources
- Forum: https://edstem.org/us/courses/50593
  - Announcement; discussions
- Office hours: https://calendar.app.google/2Nn38LMZqH3FWkUp9
  - Welcome to schedule 15-min slots; individually or with classmates; 4-hour notice
- TA: Armin Karamzade
  - Office hours: https://calendar.app.google/dernobKRE38Gzxis7

Course logistics: lectures and discussions

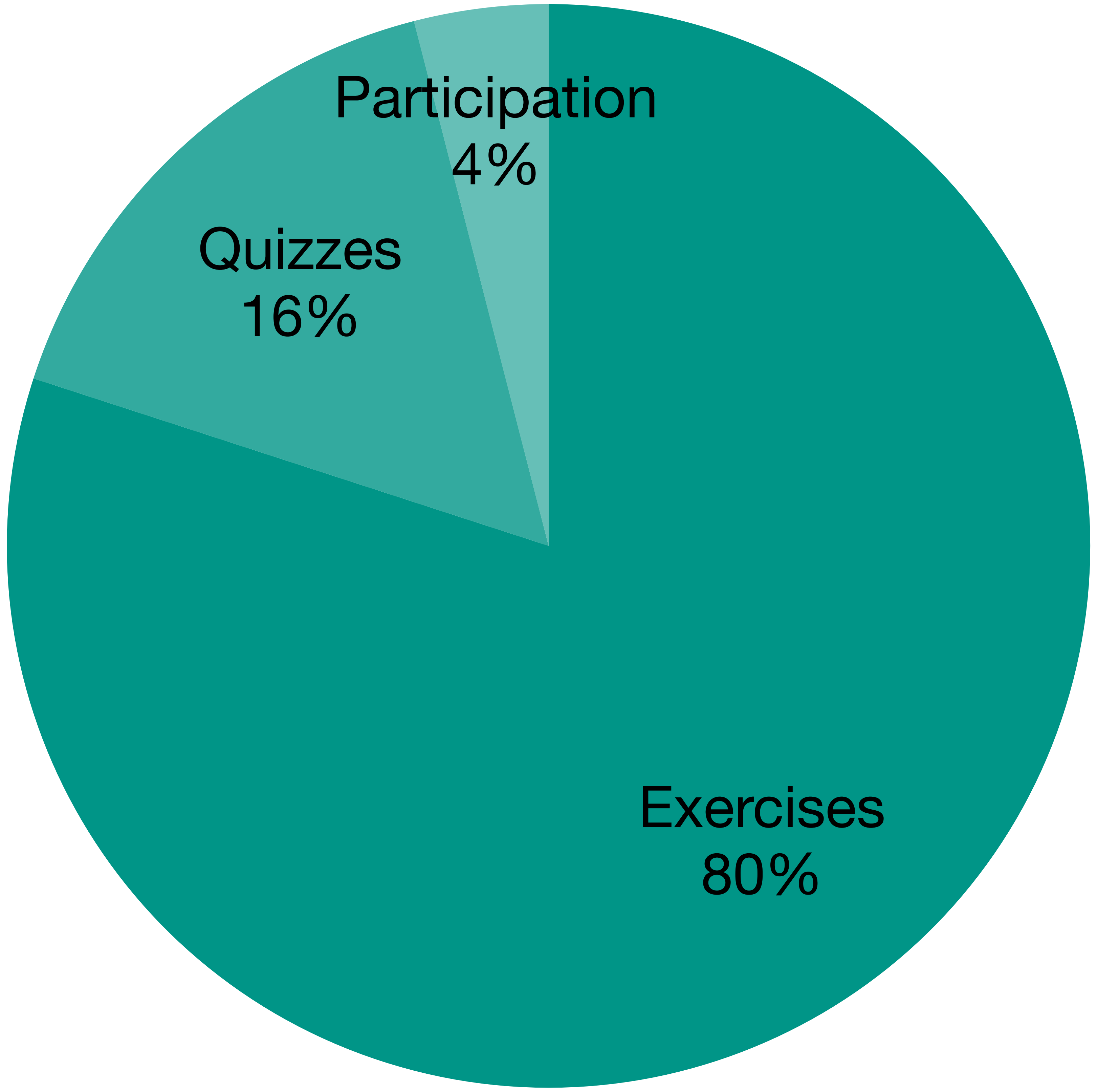
- Lectures
  - When: Tuesdays and Thursdays, 2–3:20pm
  - Where: DBH 1200
  - Recorded when possible, uploaded to the course website
  - Attendance is optional but recommended
- Class discussions
  - Reviewing quizzes and exercises following deadline
  - Recaps, deep dives, freeform discussions

Course logistics: quizzes and exercises

- Quizzes
  - Weekly, about that week's topics; deadlines the following Monday
  - Discussed the following Tuesday in class
- Exercises
  - Roughly every other week; deadlines typically Friday
  - Understand RL concepts; apply RL techniques in Python
  - Discussed the following Tuesday in class
- Submission: https://www.gradescope.com/courses/688814

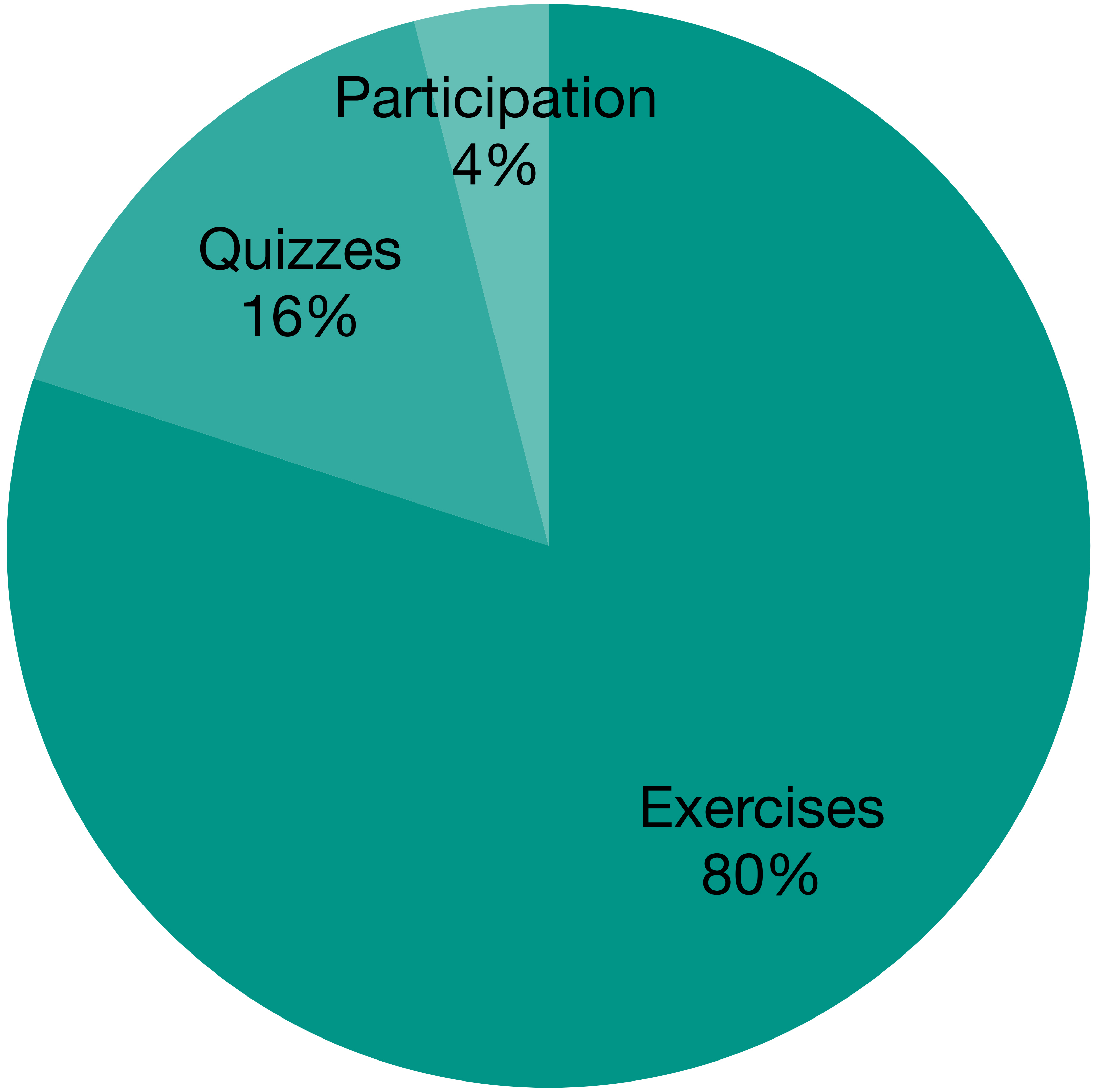
Grading policy: exercises

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



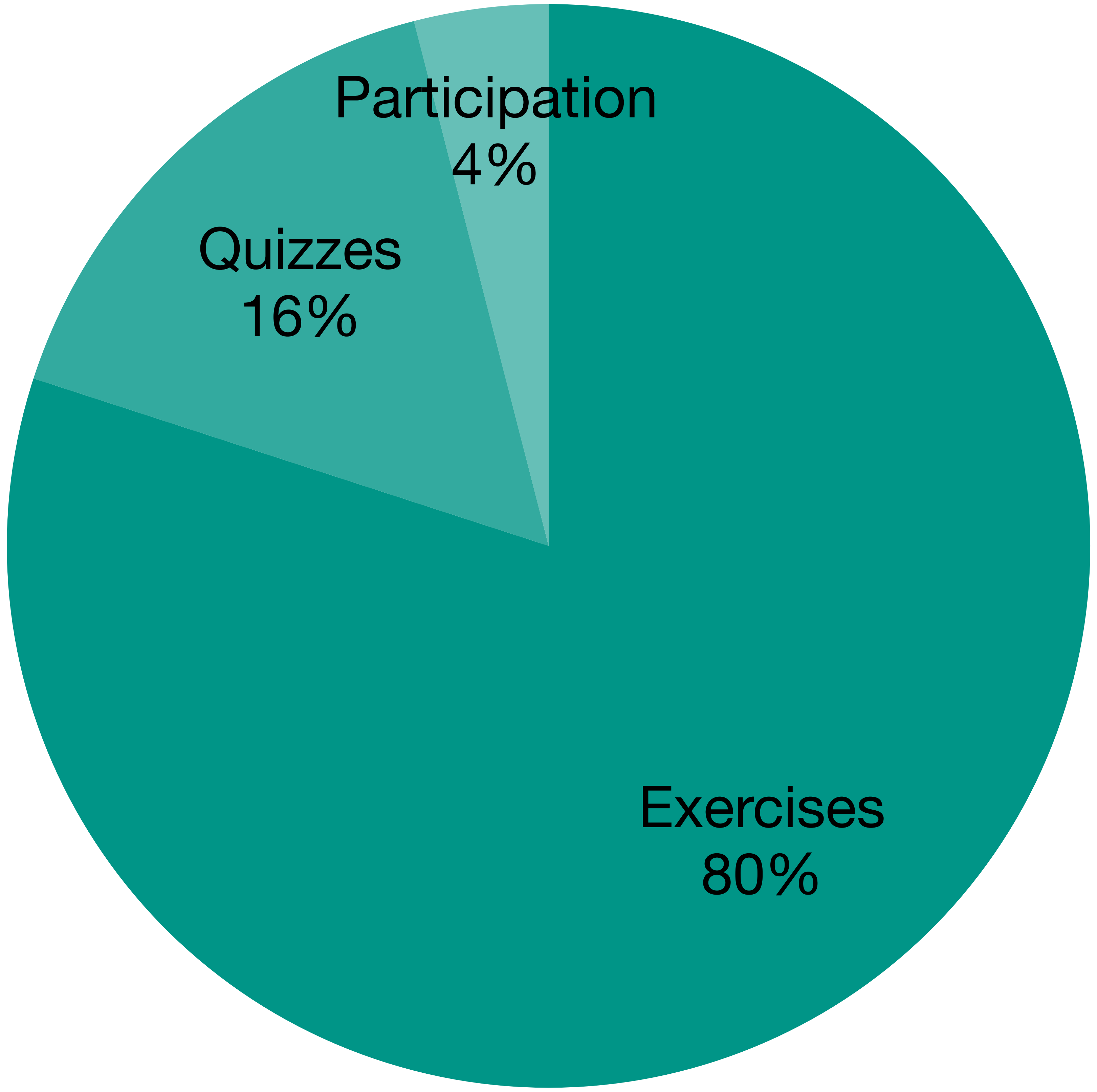
Grading policy: quizzes

- Review the week's topics, think about them a bit
- ≥6 quizzes, up to 16% total + 2% bonus
  - Half the score for submitting a complete quiz
  - Half the score for doing better than random guess
- No late submission



Grading policy: participation

- Class, office hours, or forum participation: 2%
  - Ask questions if you have any
  - Answer quiz or forum questions if you can
  - Share thoughtful comments
  - Post relevant useful links
  - Be <u>on-topic</u> (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
  - There'll be a lot going on, and nobody understands everything immediately
  - If you walk away with a good general understanding of the basics — we win!
- Help your friends and get help — from me too — but never cheat!

Today's lecture

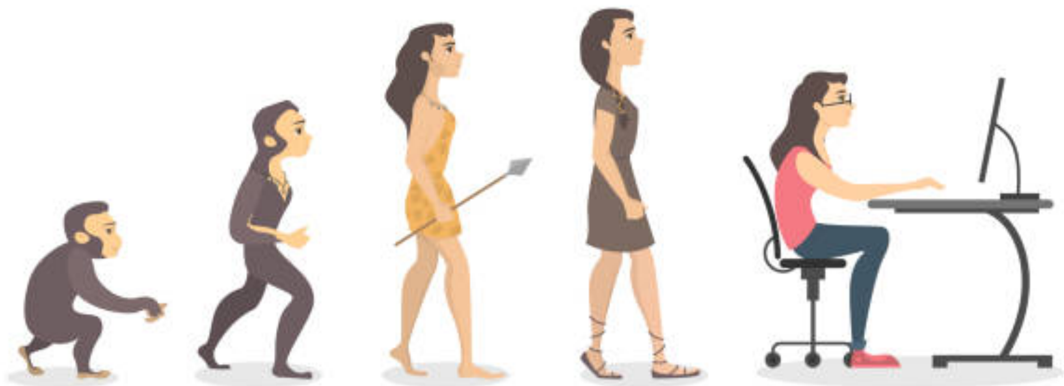
What is reinforcement learning?

Course logistics

Why is RL interesting?

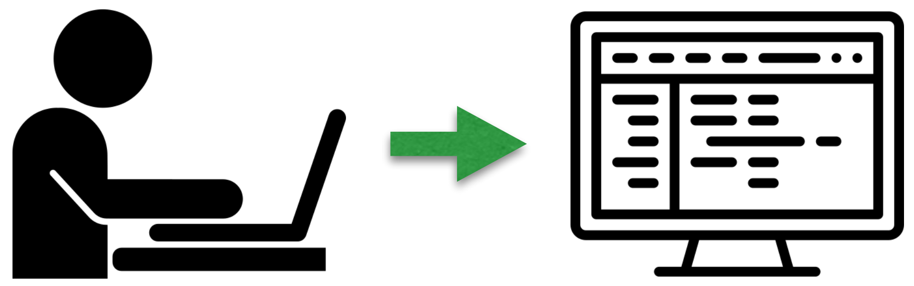
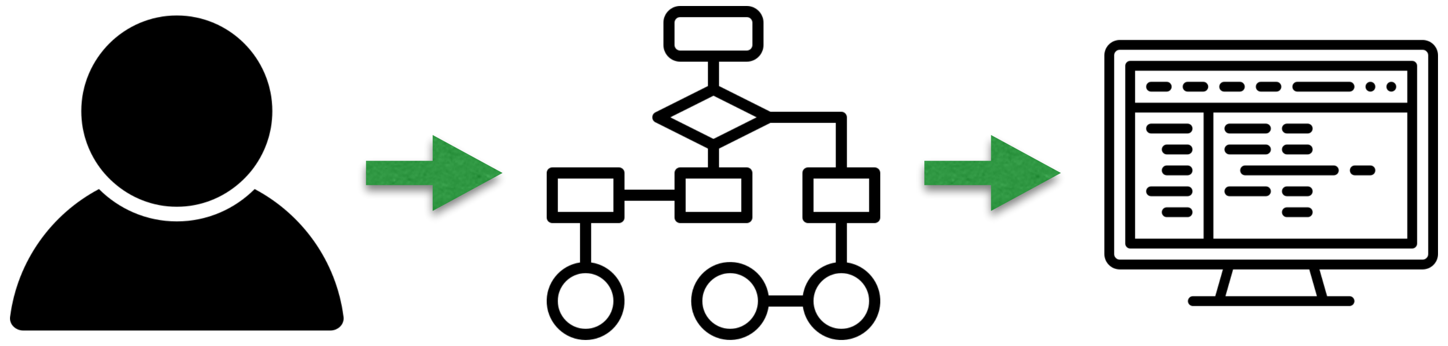
Why is RL powerful?

- Many (all?) problems can be formulated as control
  - But consider: is it sequential? multi-agent? a more specific structure?
- Active + online = very little supervision
  - Even incidental, like in evolution! Supervisor can be “surprised”



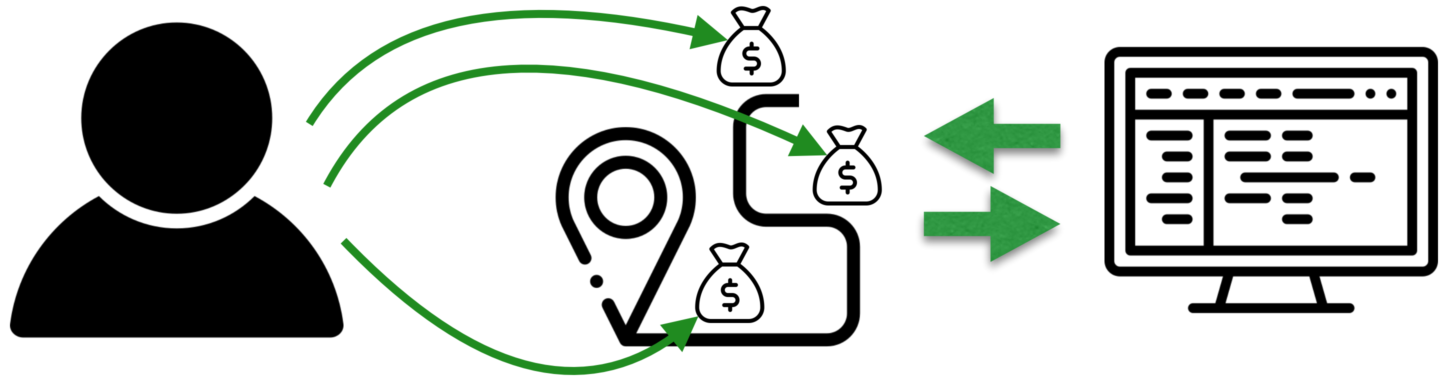
- More general CL: incorporate stronger supervision
  - Supervisor burden is a tradeoff between data amount ↔ informativeness

Control preference elicitation

	Explicit	Implicit
"how"	**Programming** 	**Imitation Learning** 
"what"	**Instruction Following** 	**Reinforcement Learning** 

How is RL different?

online

	Provided correct output	Provided feedback
Teacher selects input	**IL from demonstrations** 	**Offline RL** 
Learner selects input	**IL from corrections** 	**RL** 

active

What would “solving” RL look like?

modularity?

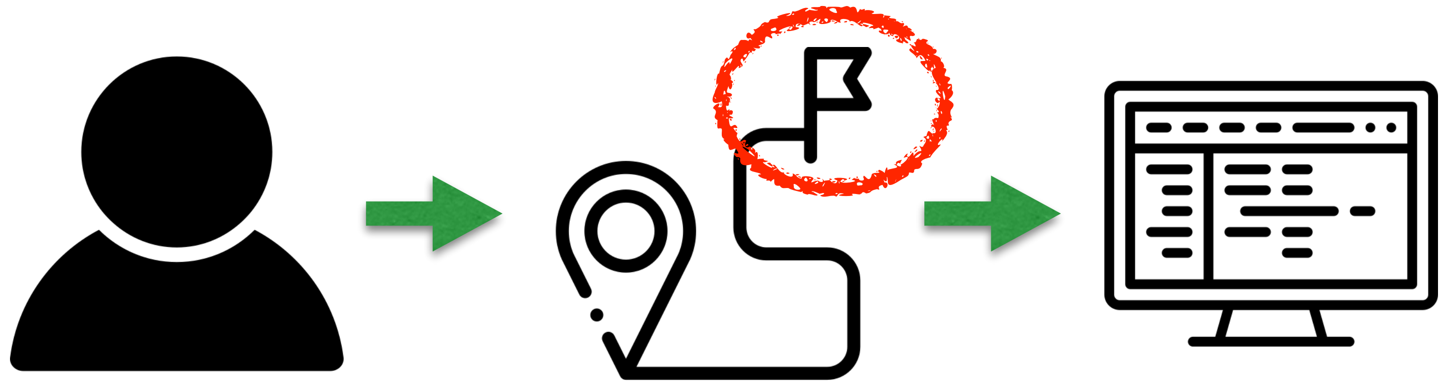
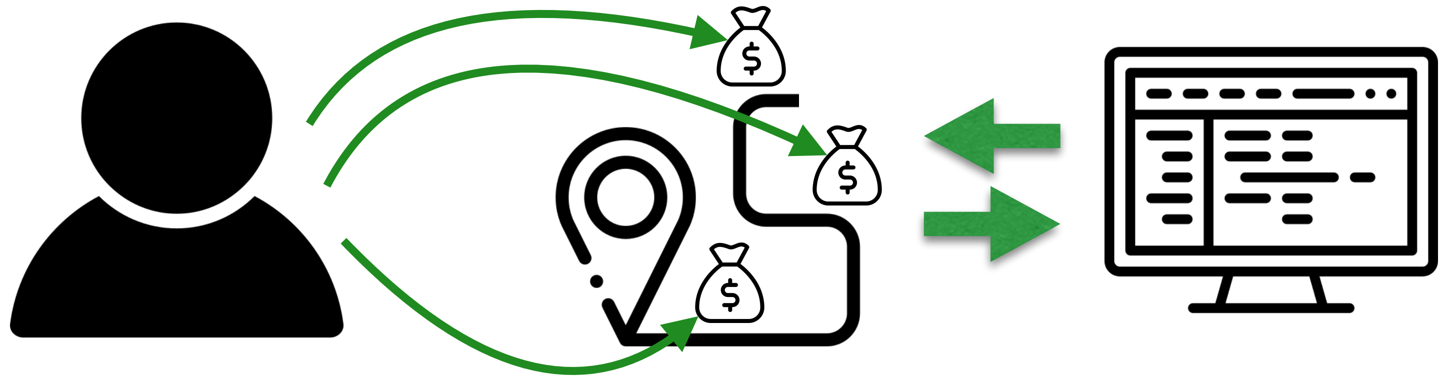
Foundation model | Continual learning

- Foundation model?
  - Large model
  - Huge amount of data
  - Centrally trained
  - Fine-tuned, built into pipelines
- Continual learning?
  - Flexible model
  - Ad-hoc data
  - Distributed learning
  - Mixed supervision, shared learning

The last ML frontier?

Why is RL hard?

- It's all about the data: amount and informativeness

	Provided correct output	Provided feedback
Teacher selects input	**IL from demonstrations**  **expert, train–test mismatch**	**Offline RL**  **extreme train–test mismatch**
Learner selects input	**IL from corrections**  **hard to give** **exploration**	**RL**  **weak signal, exploration**

Today's lecture

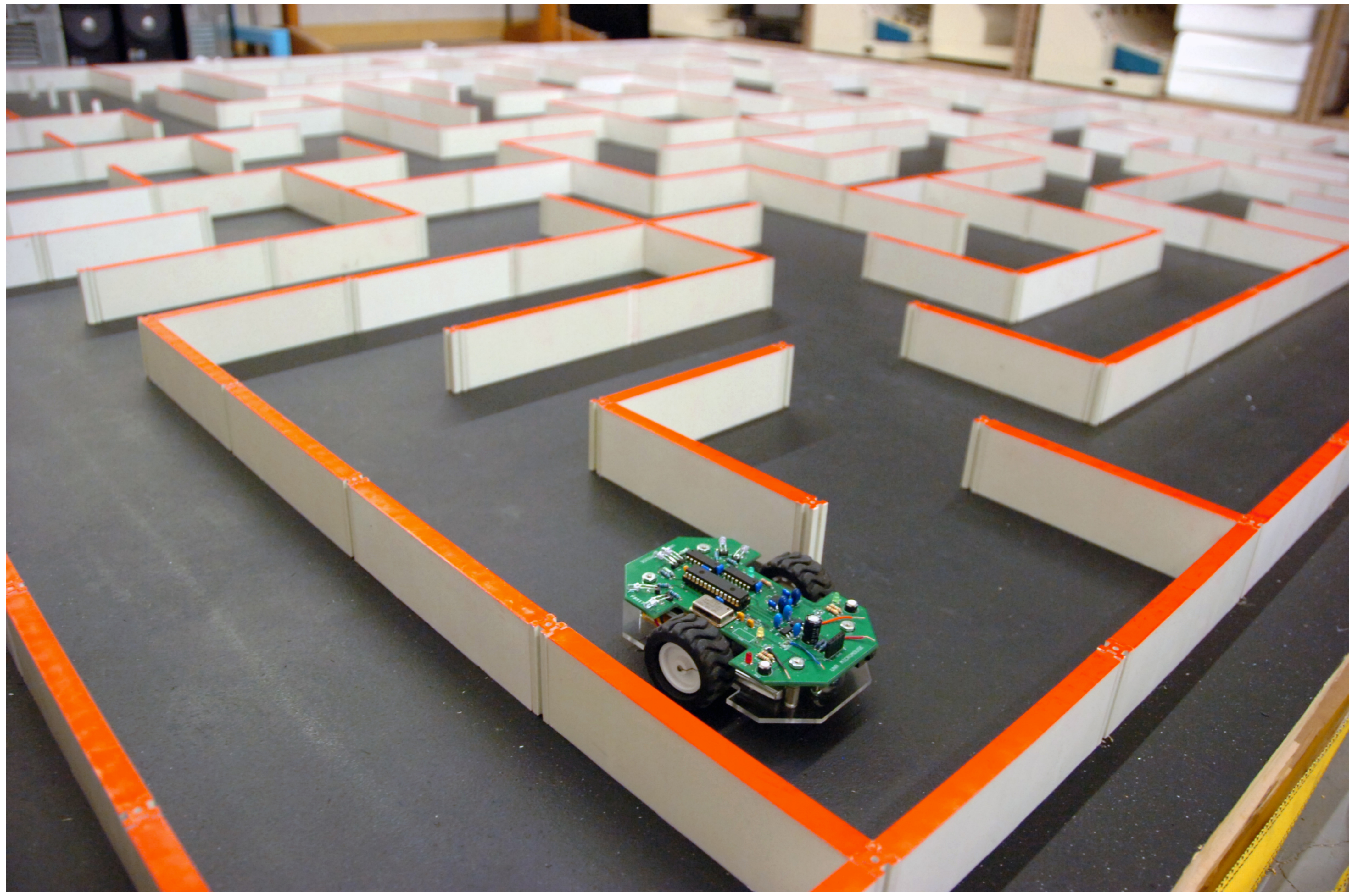
What is reinforcement learning?

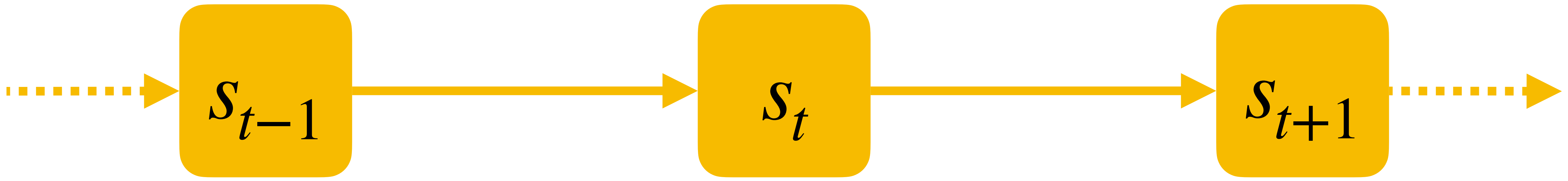
Course logistics

Why is RL interesting?

Basic RL concepts

System state





System state

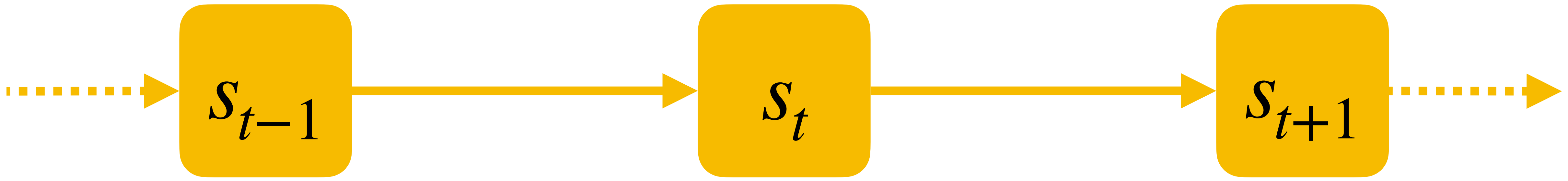
- Markov property: the future is independent of the past, given the present

$$p(s_{t+1}, s_{t+2}, \ldots \mid s_0, s_2, \ldots, s_t) = p(s_{t+1}, s_{t+2}, \ldots \mid s_t)$$

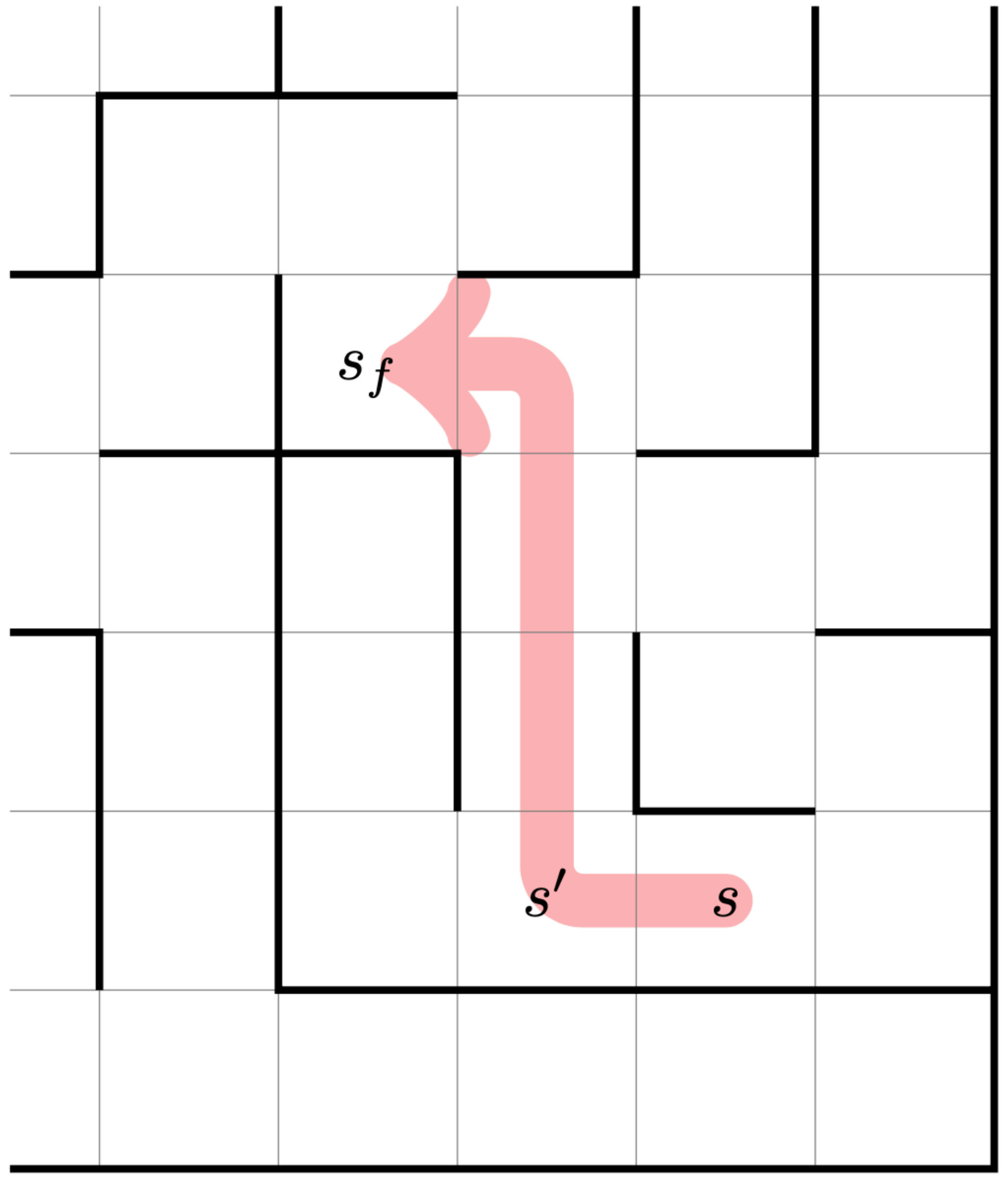
- State = all relevant information from history

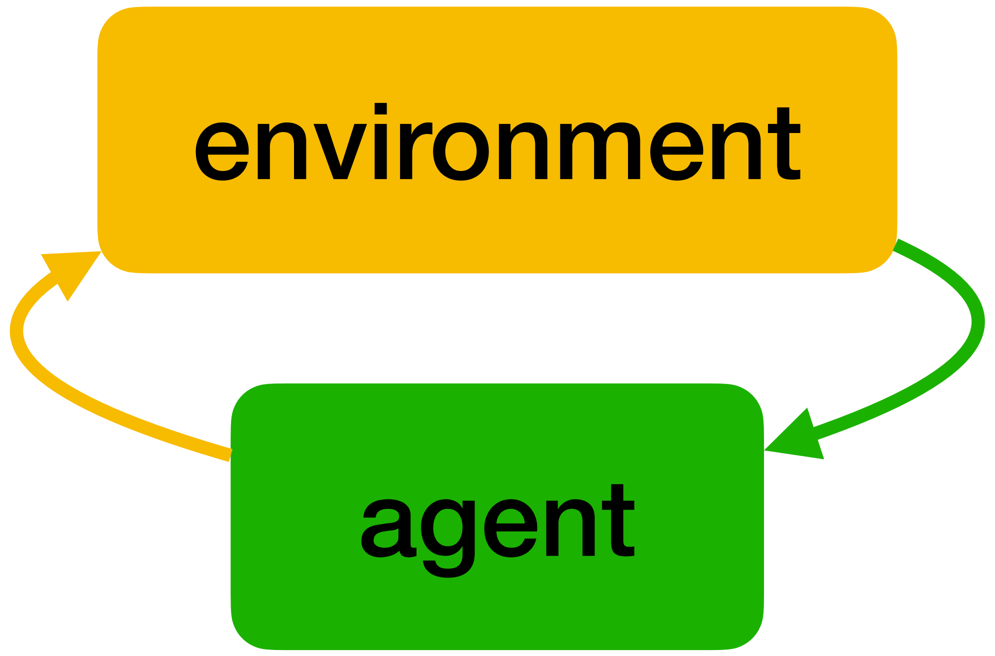
  for future!

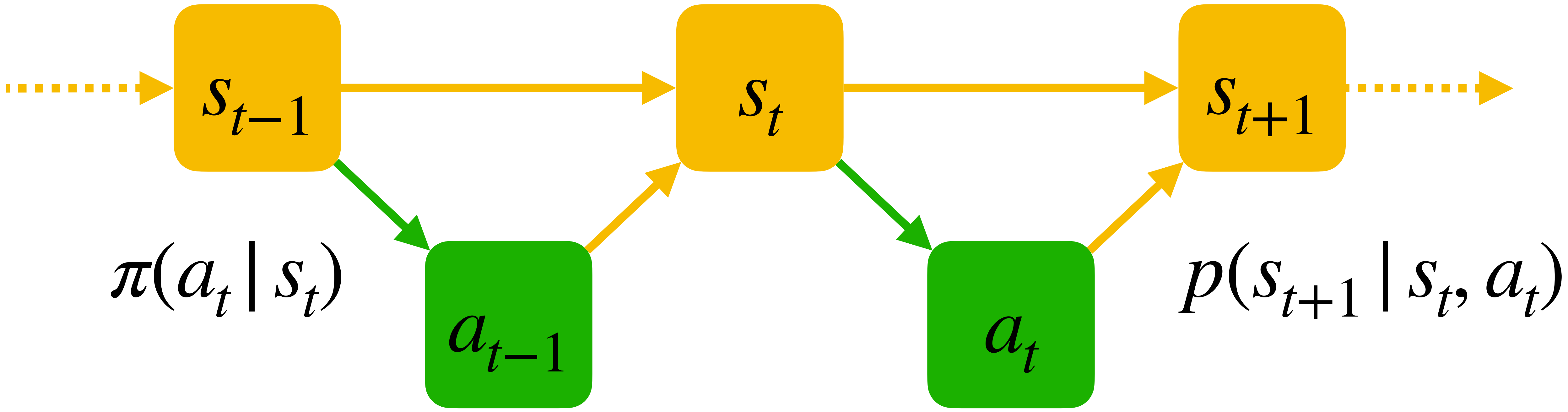
  - Given s_t, the history $h = (s_0, \ldots, s_t)$ and the future $(s_{t+1}, s_{t+2}, \ldots)$ are independent



System = agent + environment

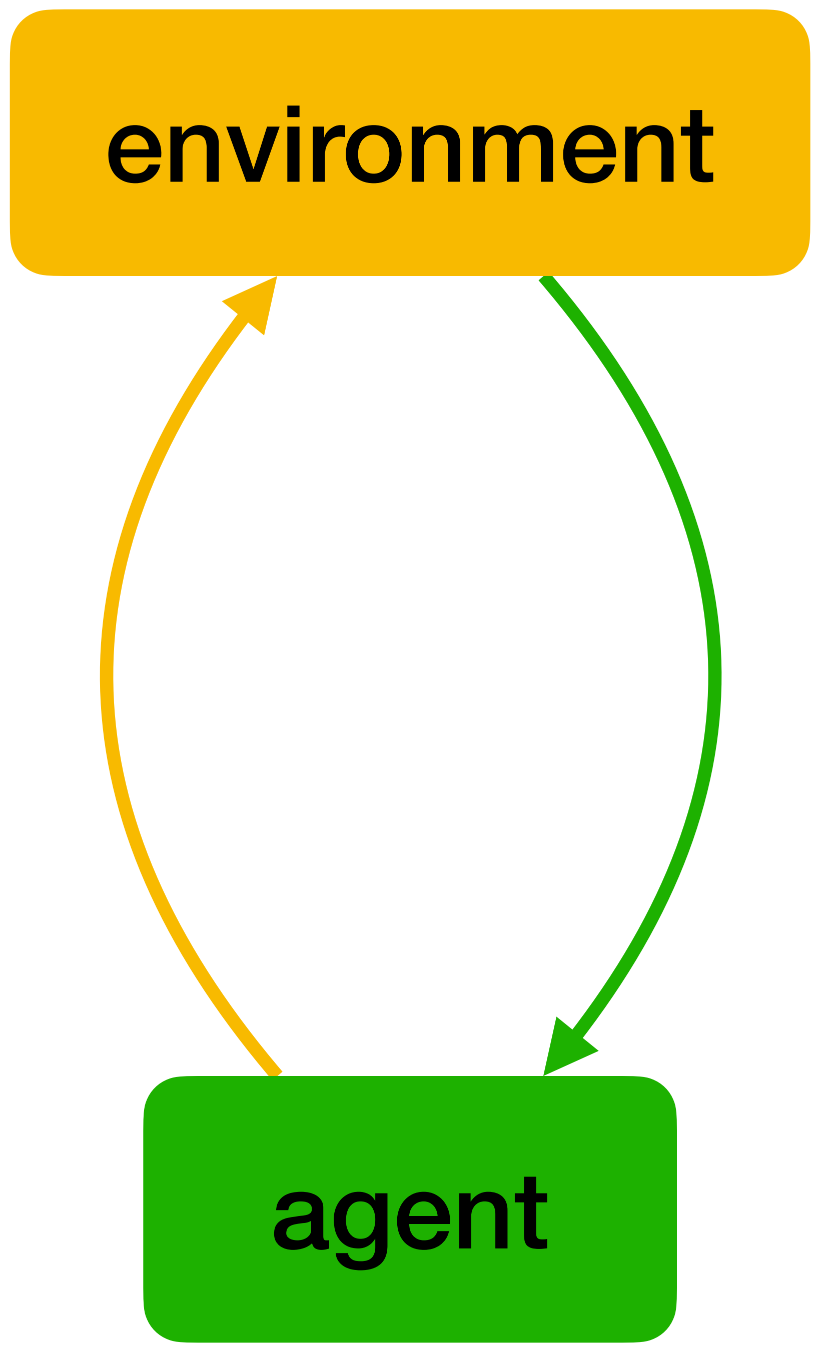






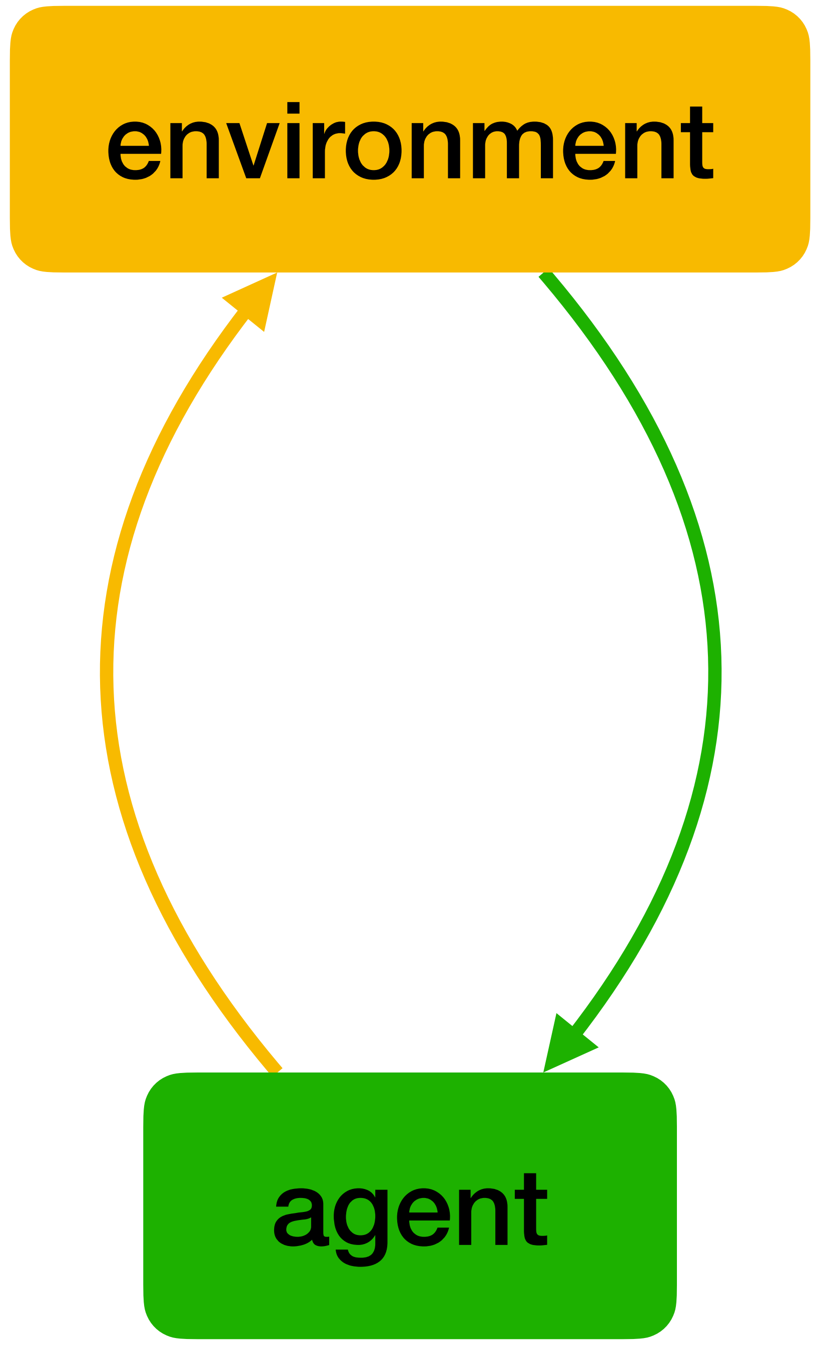
Markov Decision Process (MDP)

- Model of environment
  - $\mathcal{S}$ = set of states
  - $\mathcal{A}$ = set of actions
  - $p(s' \mid s, a)$ = 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 $\pi(a \mid s)$ = 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: $s_t \to (t, s_t)$



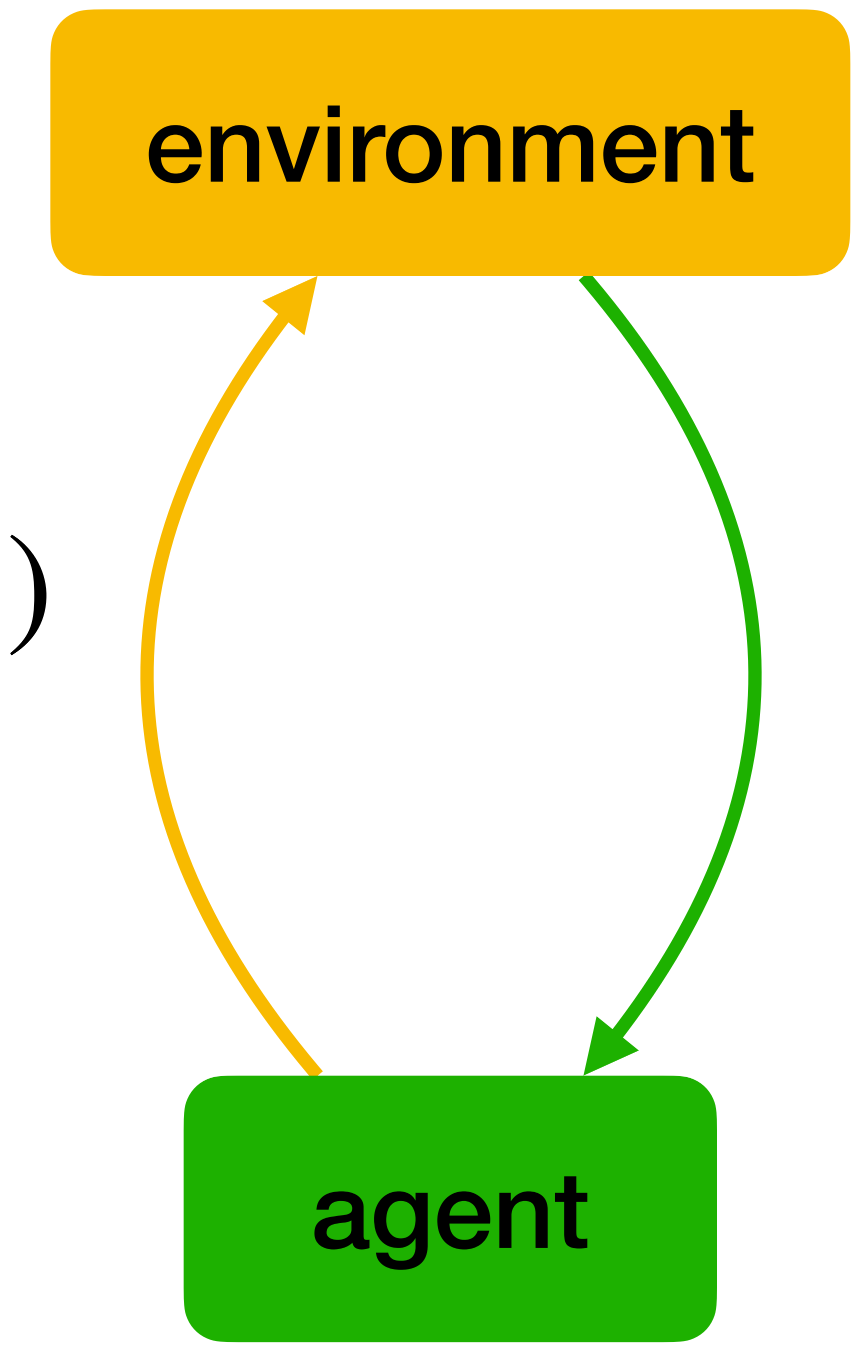
Trajectories

- The agent's behavior iteratively uses (rolls out) the policy

- Trajectory: $\xi = (s_0, a_0, s_2, a_2, \ldots, s_T)$

- MDP + policy induce distribution over trajectories

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



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

Learning from rewards

- Providing demonstrations is hard
  - Particularly for learner-generated trajectories

as in online learning

- Can the teacher just score learner actions?
  - 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 π

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: $R(\xi) = \lim_{T\to\infty} \frac{1}{T} \sum_{t=0}^{T-1} r(s_t, a_t)$
- Discounted: $R(\xi) = \sum_{t=0}^{\infty} \gamma^t r(s_t, a_t) \qquad 0 \le \gamma < 1$
- Episodic: $R(\xi) = \sum_{t=0}^{T-1} r(s_t, a_t)$ s.t. $s_T = s_f$

Basic RL concepts

- State: $s \in \mathcal{S}$; action: $a \in \mathcal{A}$; 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

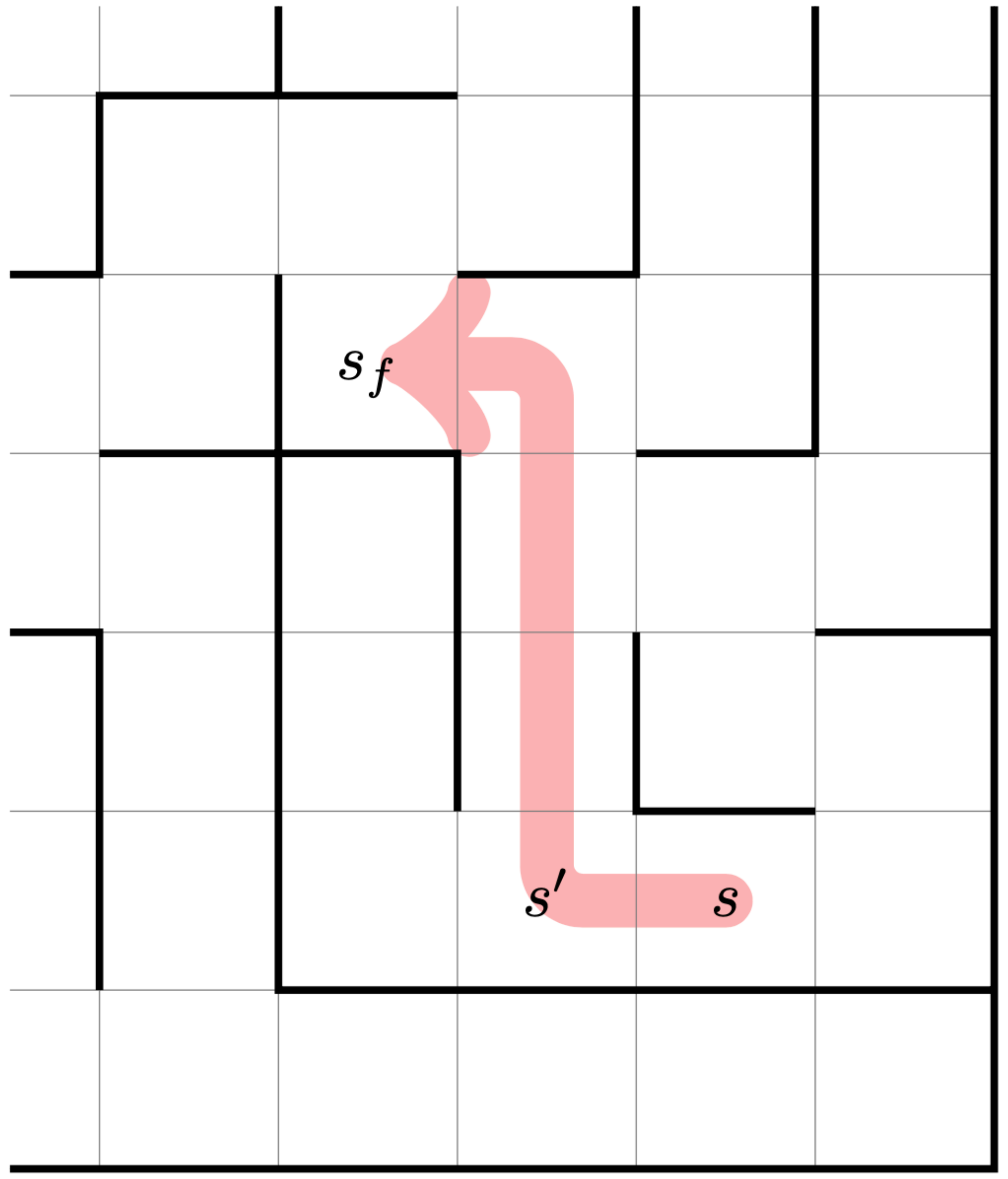
- Trajectory: $p_\pi(\xi = s_0, a_0, s_1, a_1, \ldots) = p(s_0) \prod_t \pi(a_t | s_t) p(s_{t+1} | s_t, a_t)$

- Return: $R(\xi) = \sum_t \gamma^t r(s_t, a_t) \qquad 0 \leq \gamma < 1$

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

Algorithm 1 Bellman-Ford

$V(s_f) \leftarrow 0$
$V(s) \leftarrow \infty \qquad \forall s \in S \backslash \{s_f\}$
for ℓ from 1 to $|S| - 1$ **do**
$\quad V(s) \leftarrow \min_{a \in A}\{1 + V(f(s, a))\} \qquad \forall s \in S \backslash \{s_f\}$

Logistics

logistics

- Follow announcements and discussions on ed
- See website for schedule, recordings, resources, etc.

exercises

- Quiz 1 due next Monday
- Exercise 1 to be published soon, due next Friday