CS 295:
Optimal Control and Reinforcement Learning

Winter 2020

Lecture 1: Introduction

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

Today's lecture

- Course overview and general information
- What is reinforcement learning (RL) and why study it
- Basic RL concepts

Course description for CS 295

- This course is an introduction to optimal control and reinforcement learning
- The course will consist mostly of lectures and assigned reading
- There will be assignments: reading, thinking, and some coding
- Grading based on assignments and participation
- Office hours: Fridays 9–11am, DBH 4064
- **Course announcements: piazza.com/uci/winter2020/cs295rl/home**

Course schedule (subject to updates)

Week	Tuesday	Thursday
(1) Jan 6	Introduction	Imitation learning
(2) Jan 13	Optimal control	Stochastic optimal control
(3) Jan 20	Planning	Temporal-difference methods
(4) Jan 27	Partial observability	RL with function approximation
(5) Feb 3	Policy-gradient methods	Policy-gradient methods (cont.)
(6) Feb 10	Actor–critic methods	Model-based methods
(7) Feb 17	Inverse RL	Control as inference
(8) Feb 24	Structured control	Multi-task and meta-learning
(9) Mar 2	*No lecture (Super Tuesday)*	Exploration
(10) Mar 9	RL systems	Open problems

Resources

- Sergey Levine [course]: http://rail.eecs.berkeley.edu/deeprlcourse/
- François-Lavet et al. [book]: https://www.nowpublishers.com/article/Download/MAL-071
- Bertsekas [course, 2017/19 books]: http://web.mit.edu/dimitrib/www/RLbook.html
- OpenAI [tutorial]: https://spinningup.openai.com/
- David Silver [course]: http://www0.cs.ucl.ac.uk/staff/D.Silver/web/Teaching.html
- Sutton & Barto [book]: http://www.incompleteideas.net/book/RLbook2018.pdf
- Szepesvári [book]: https://sites.ualberta.ca/~szepesva/papers/RLAlgsInMDPs.pdf

Compute resources

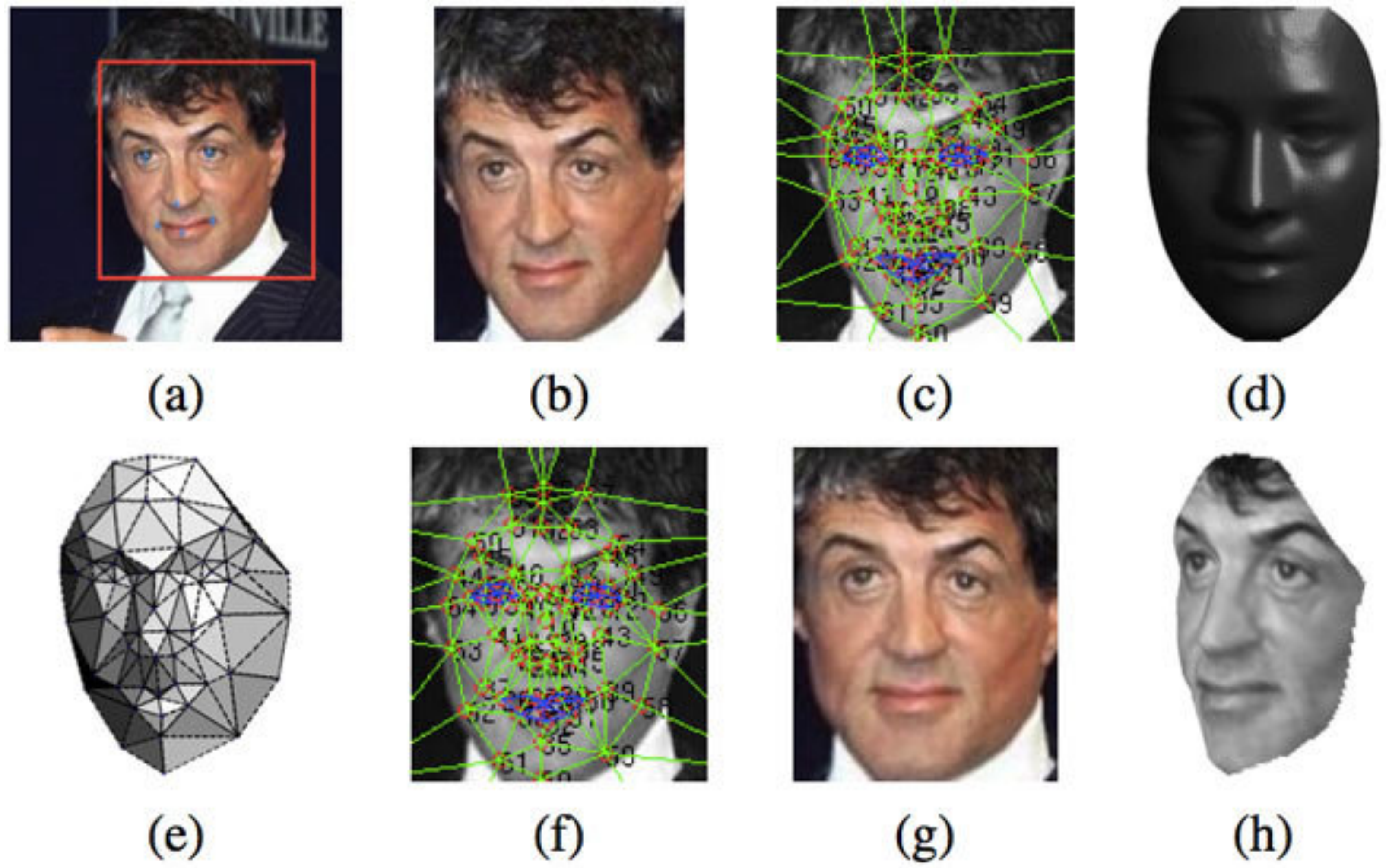
- Much of your development work can be handled by your laptop or desktop
  - Always test your code on a smaller challenge that "should" work
- When more compute resources are required:
  - Campus-wide cluster: https://hpc.oit.uci.edu/
  - Google Colab: https://colab.research.google.com/
  - We may be able to help with AWS / Google Cloud credits

What is Machine Learning

- Artificial Intelligence:
  - Can we build a machine with a property we would call "intelligence"?
- Machine Learning:
  - Can we build AI without explicitly figuring out all the details of its working?
    - Solution = problem-agnostic algorithm + problem-specific data
  - Learning = Statistics + Algorithms
  - ML = Learning + Implementation + Data

ML examples

Face recognition



Speech synthesis

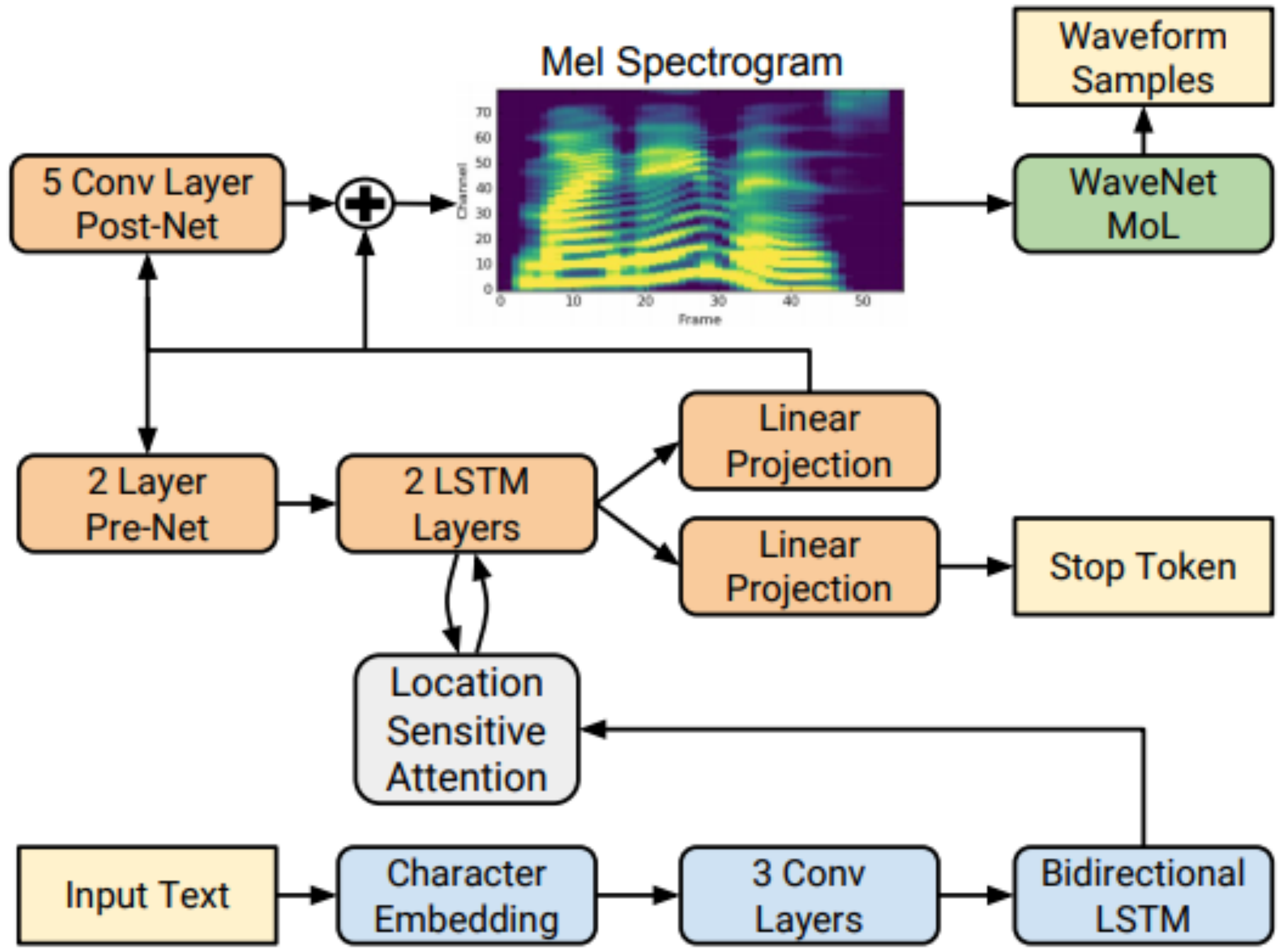
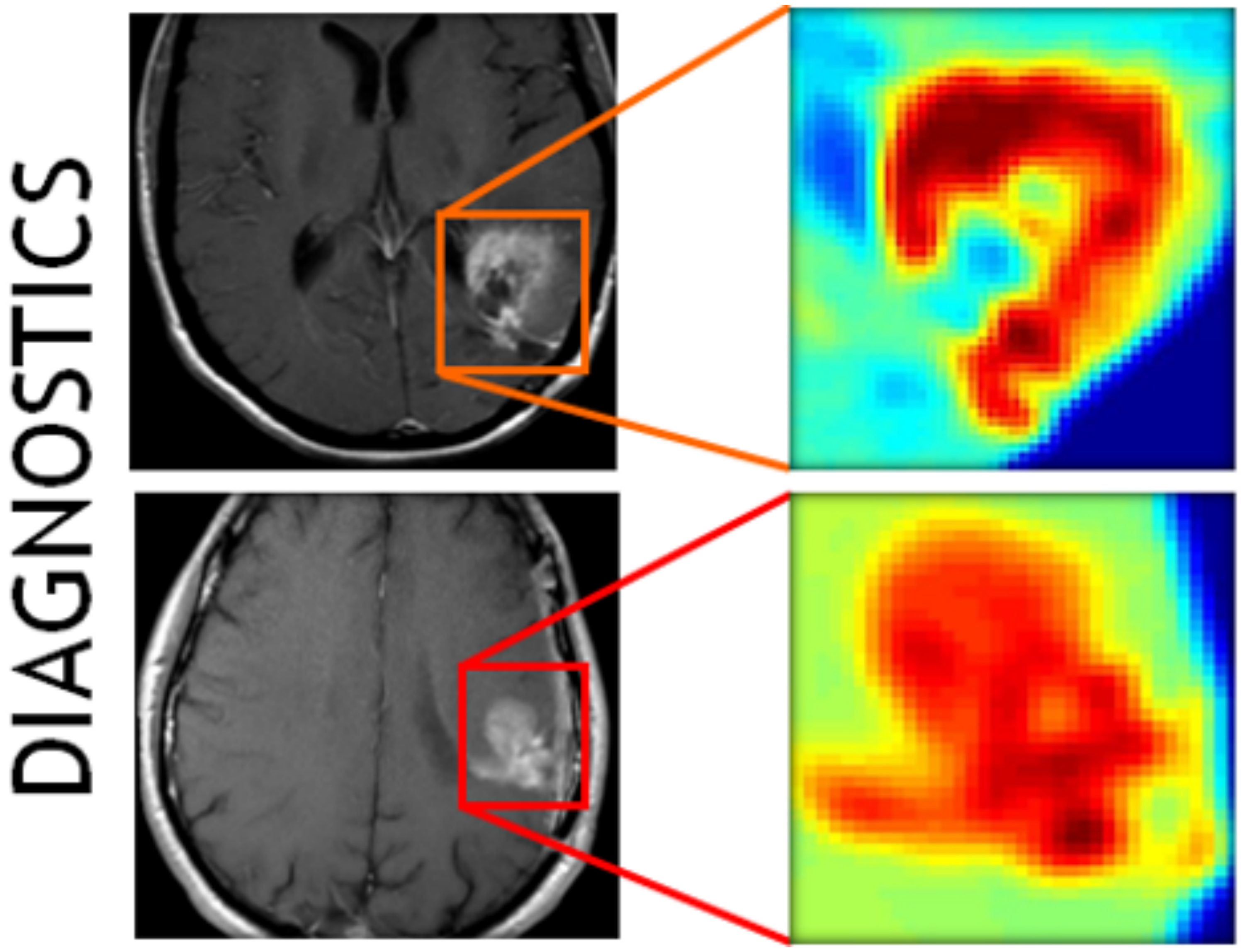


Fig. 1. Block diagram of the Tacotron 2 system architecture.

Medical diagnosis



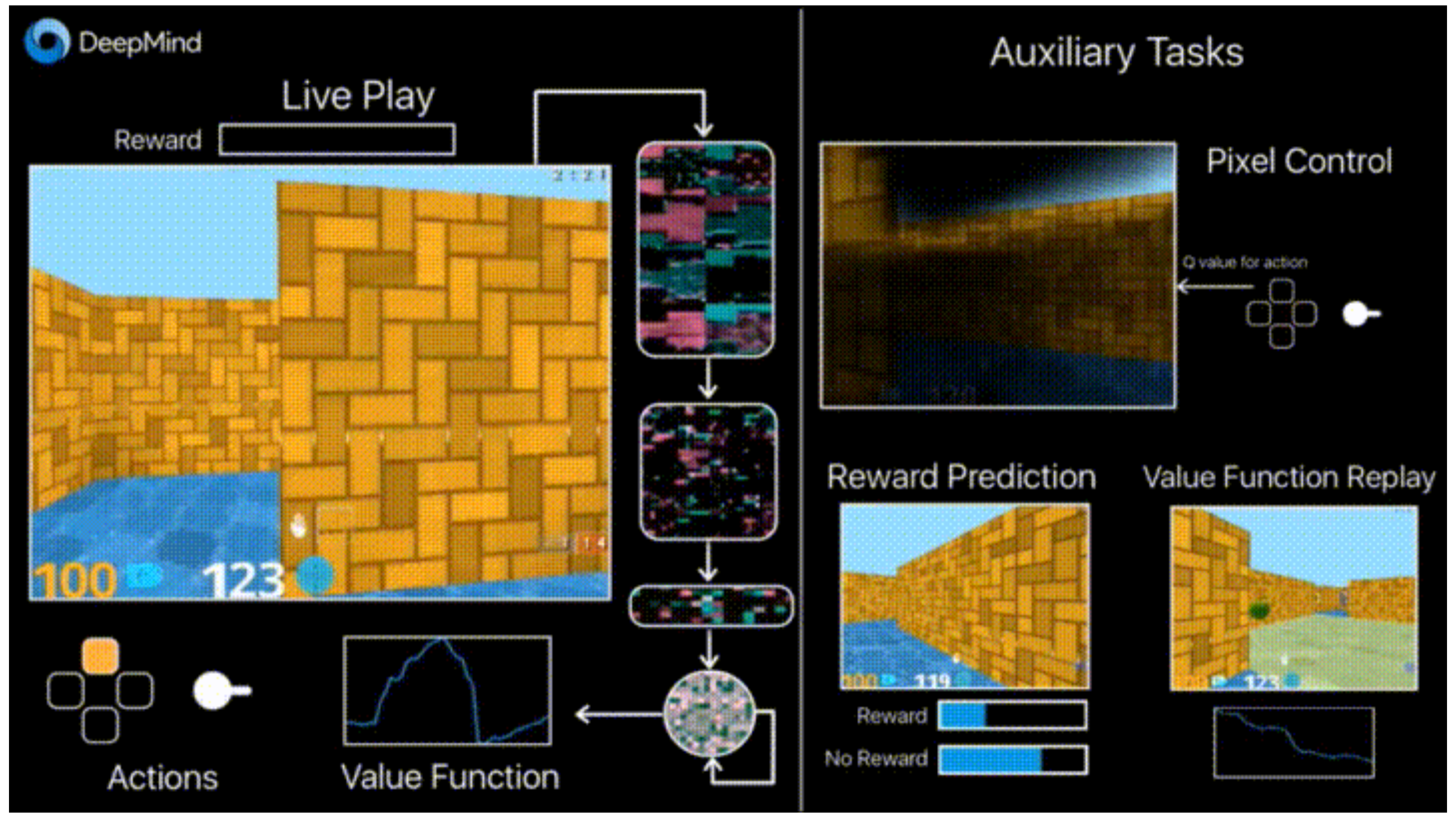
What is Reinforcement Learning

- Intelligence appears in interaction with a complex system, not in isolation
- An **agent** interact with an **environment**
- Performs **sequential** decision making:
  - Sense environment state ***s***
  - Take action ***a***
  - Repeat
- Success measured by the accumulation of reward ***r(s, a)***
  - As opposed to the "correct" action (that would be Imitation Learning)

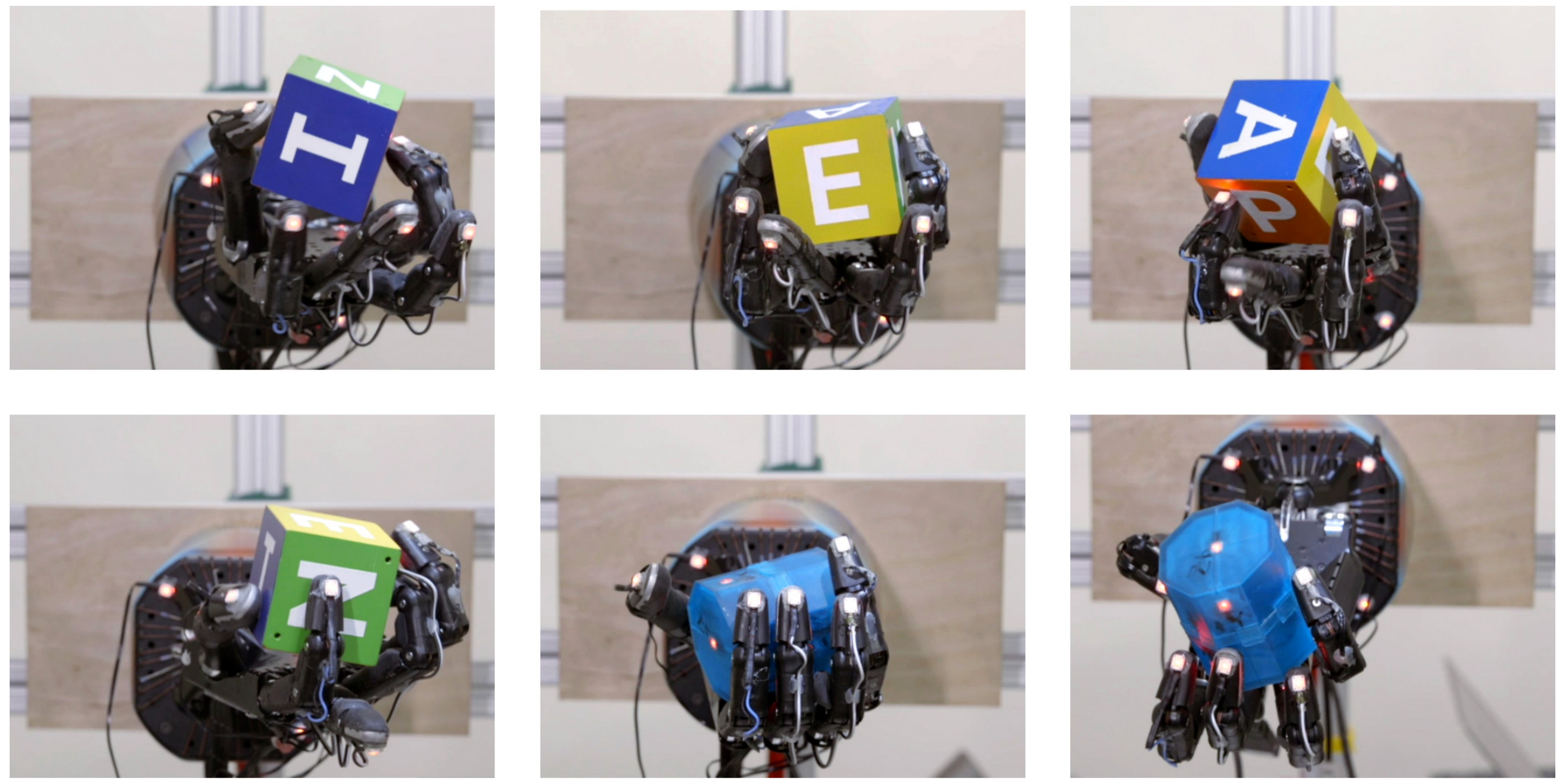
RL examples

Gameplay

Spacial navigation



Dextrous manipulation

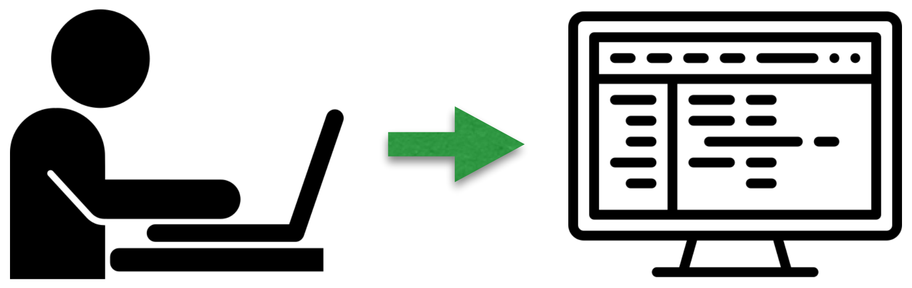
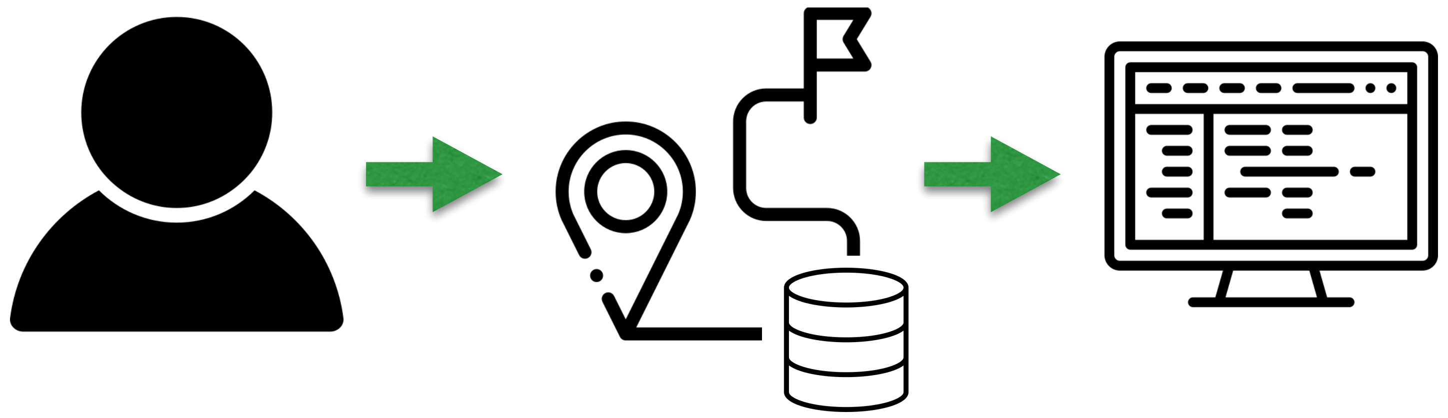
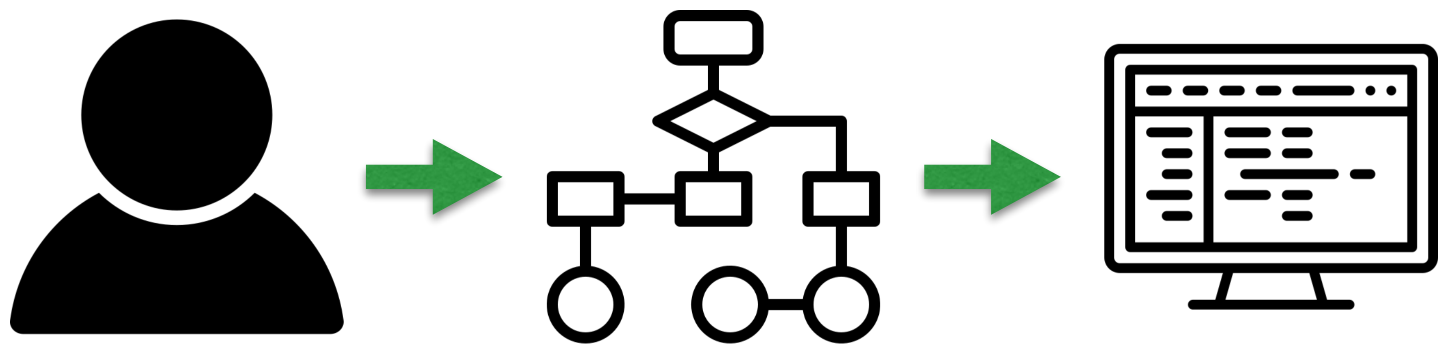
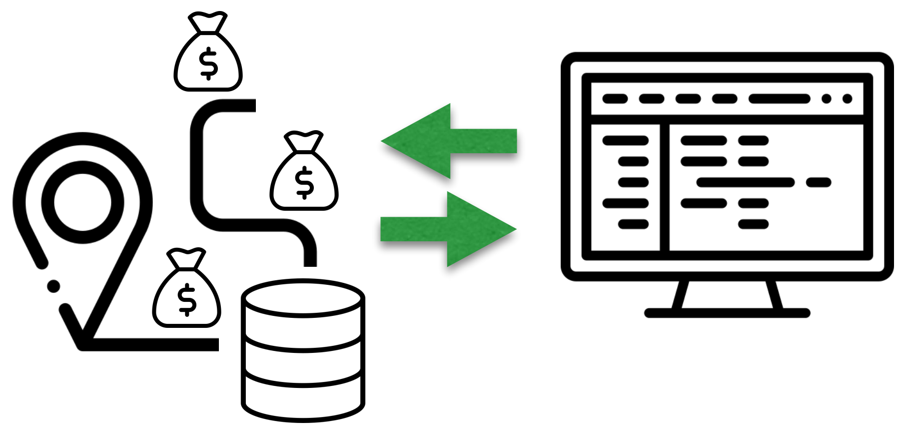


Basic RL concepts

- Dynamics $p(s_{t+1}|s_t, a_t)$
- Policy $\pi(a_t|s_t)$
- Trajectory $p(s_0, a_0, s_1, a_1, \ldots) = p(s_0) \prod_t \pi(s_t|a_t) p(s_{t+1}|s_t, a_t)$
- Return $R = \sum_t \gamma^t r(s_t, a_t) \qquad 0 \leqslant \gamma < 1$
- Value $V(s) = \mathbb{E}[R|s_0 = s]$

  $Q(s, a) = \mathbb{E}[R|s_0 = s, a_0 = a]$

Learning policies

	Explicit	Implicit
"how"	Programming 	Imitation Learning 
"what"	Specification 	Reinforcement Learning 

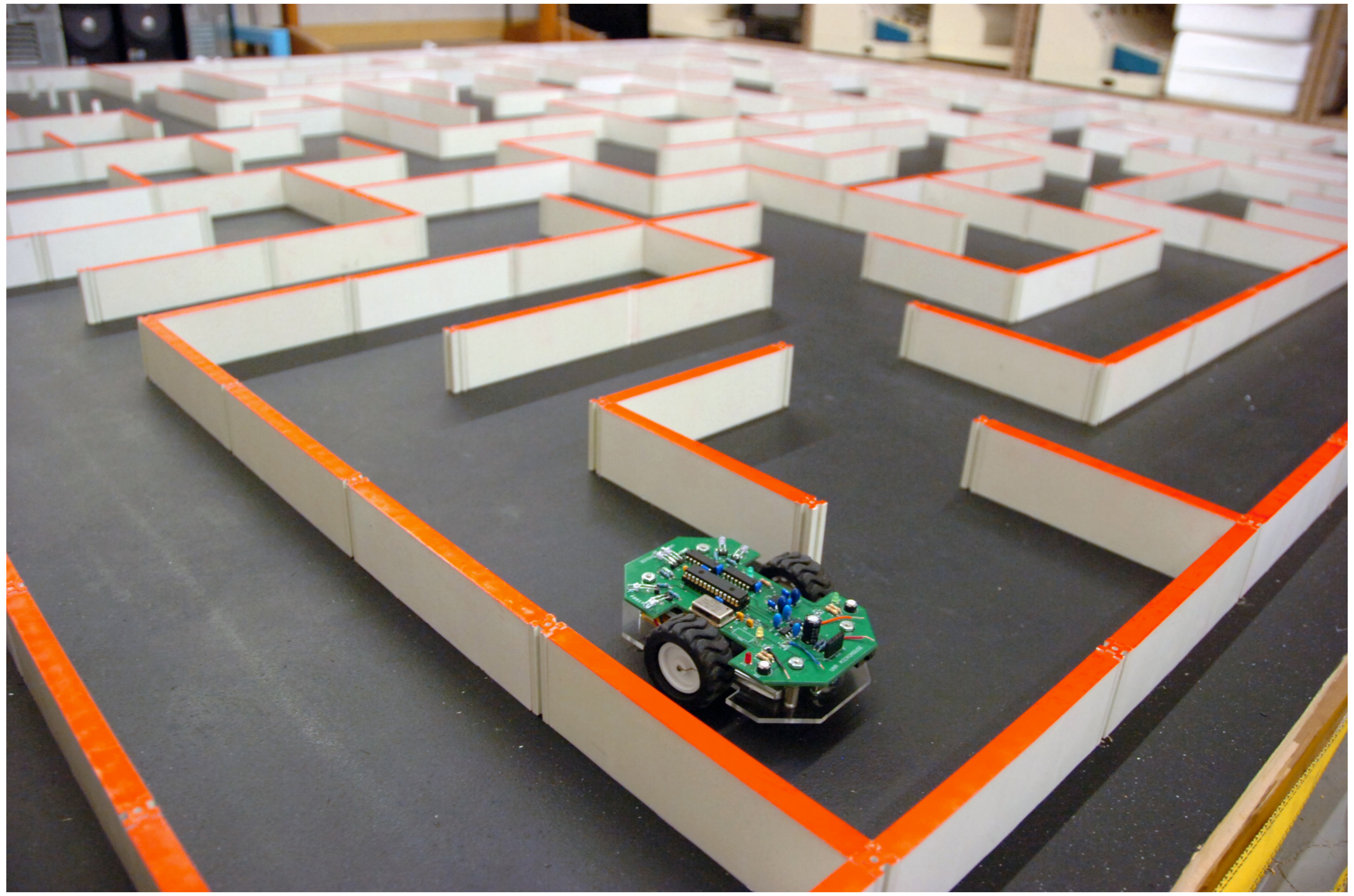
RL is ML... but special

- Test distribution of trajectories depends on the policy!
  - Cannot avoid train–test mismatch
  - To reduce it, learner interacts with the environment to collect data = exploration
  - Balanced exploration is challenging
- Policy space is strewn with local optima
  - Actions in a sequence need to be coordinated
- A good policy may require memory
  - Learning to remember is hard!

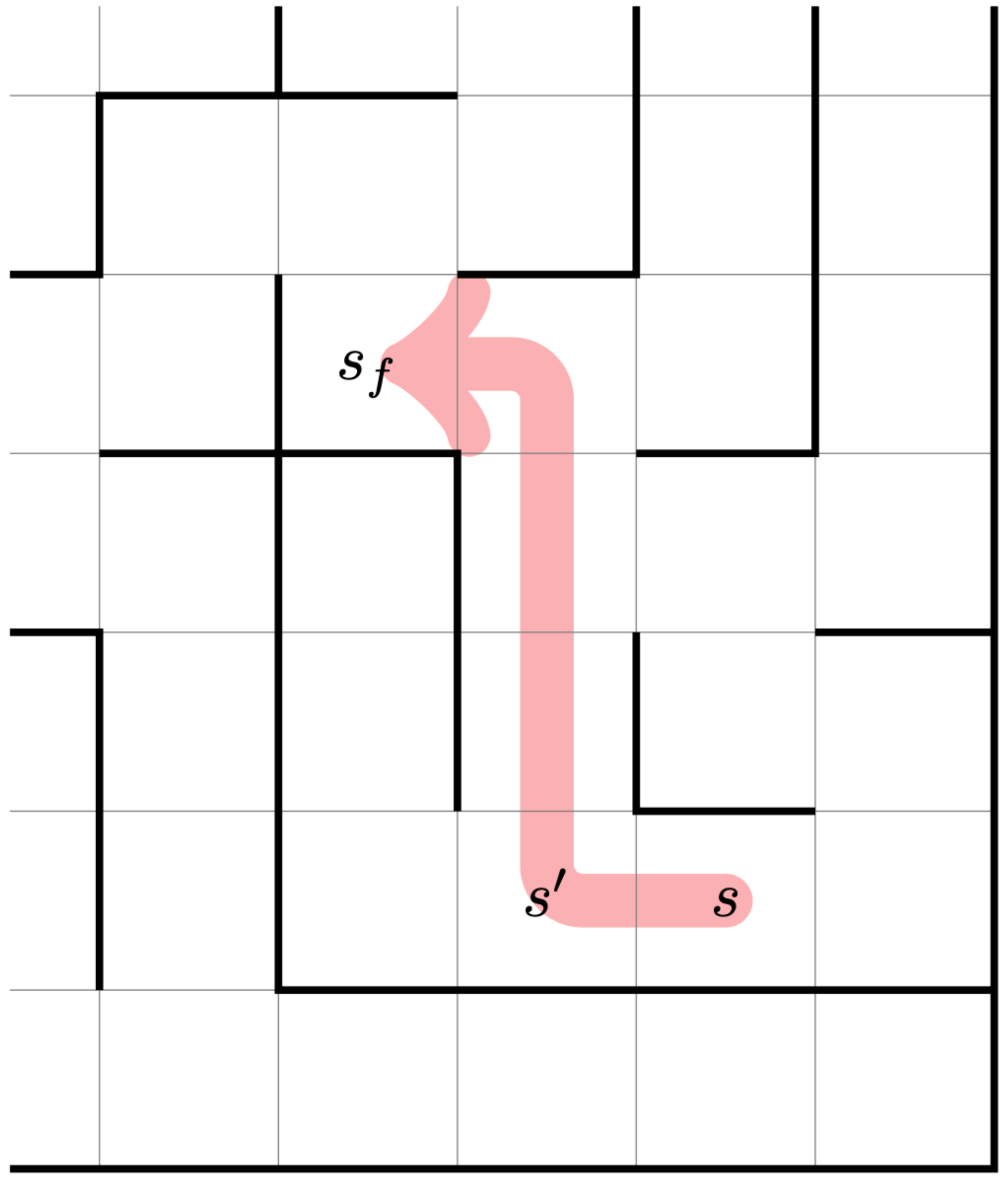
RL — the frontier

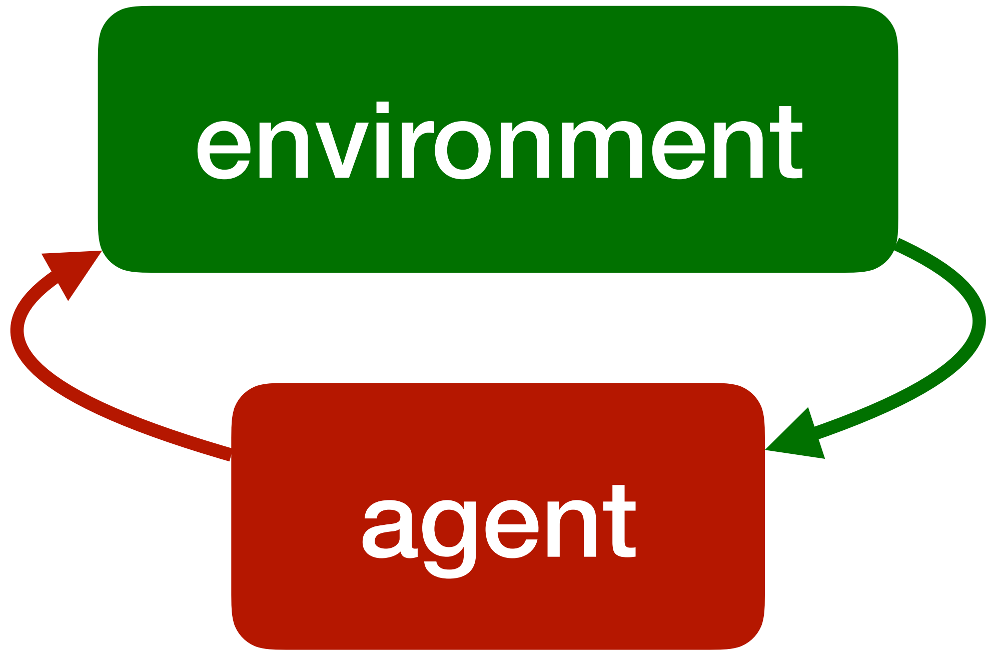
- How to perform better exploration?
- How to model / structure the agent's policy? in particular, its memory
  - Hierarchical RL
- How to jointly learn multiple tasks?
- How to learn from more kinds of data?
  - RL + imitation learning / NLP / vision / program synthesis
- How to interface with a human teacher?

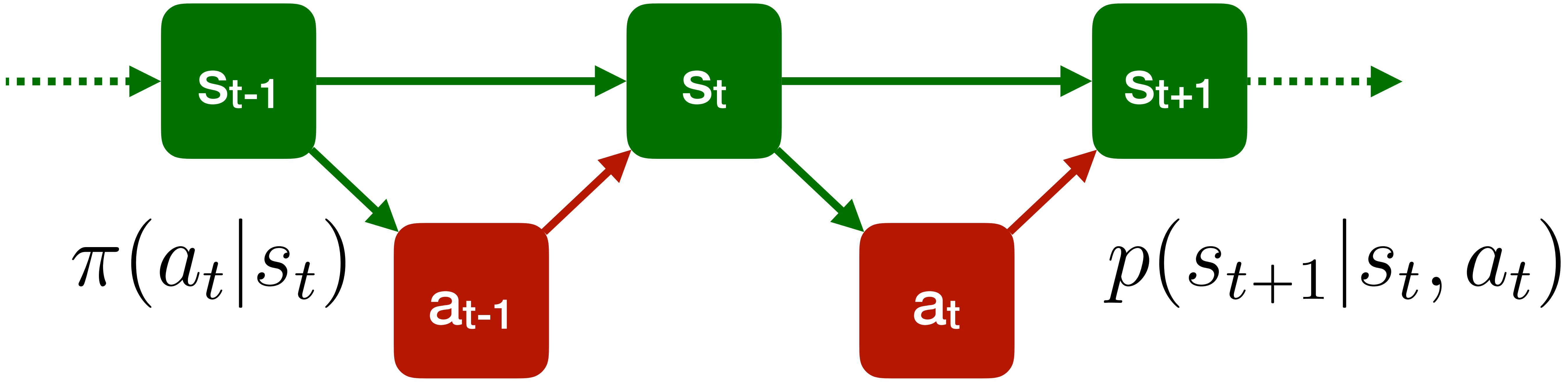
System state



System = agent + environment







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

- 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

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

Horizon classes

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

Recap

Week	Tuesday	Thursday
(1) Jan 6	Introduction	Imitation learning
(2) Jan 13	Optimal control	Stochastic optimal control
(3) Jan 20	Planning	Temporal-difference methods
(4) Jan 27	Partial observability	RL with function approximation
(5) Feb 3	Policy-gradient methods	Policy-gradient methods (cont.)
(6) Feb 10	Actor–critic methods	Model-based methods
(7) Feb 17	Inverse RL	Control as inference
(8) Feb 24	Structured control	Multi-task and meta-learning
(9) Mar 2	*No lecture (Super Tuesday)*	Exploration
(10) Mar 9	RL systems	Open problems