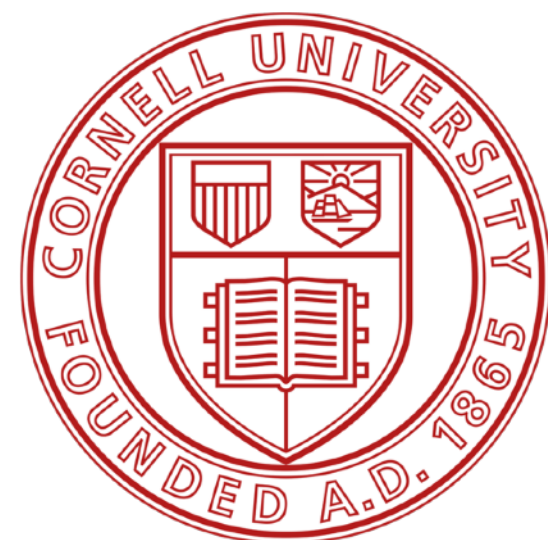


Lecture 24: Embodied vision

CS 5670: Introduction to Computer Vision



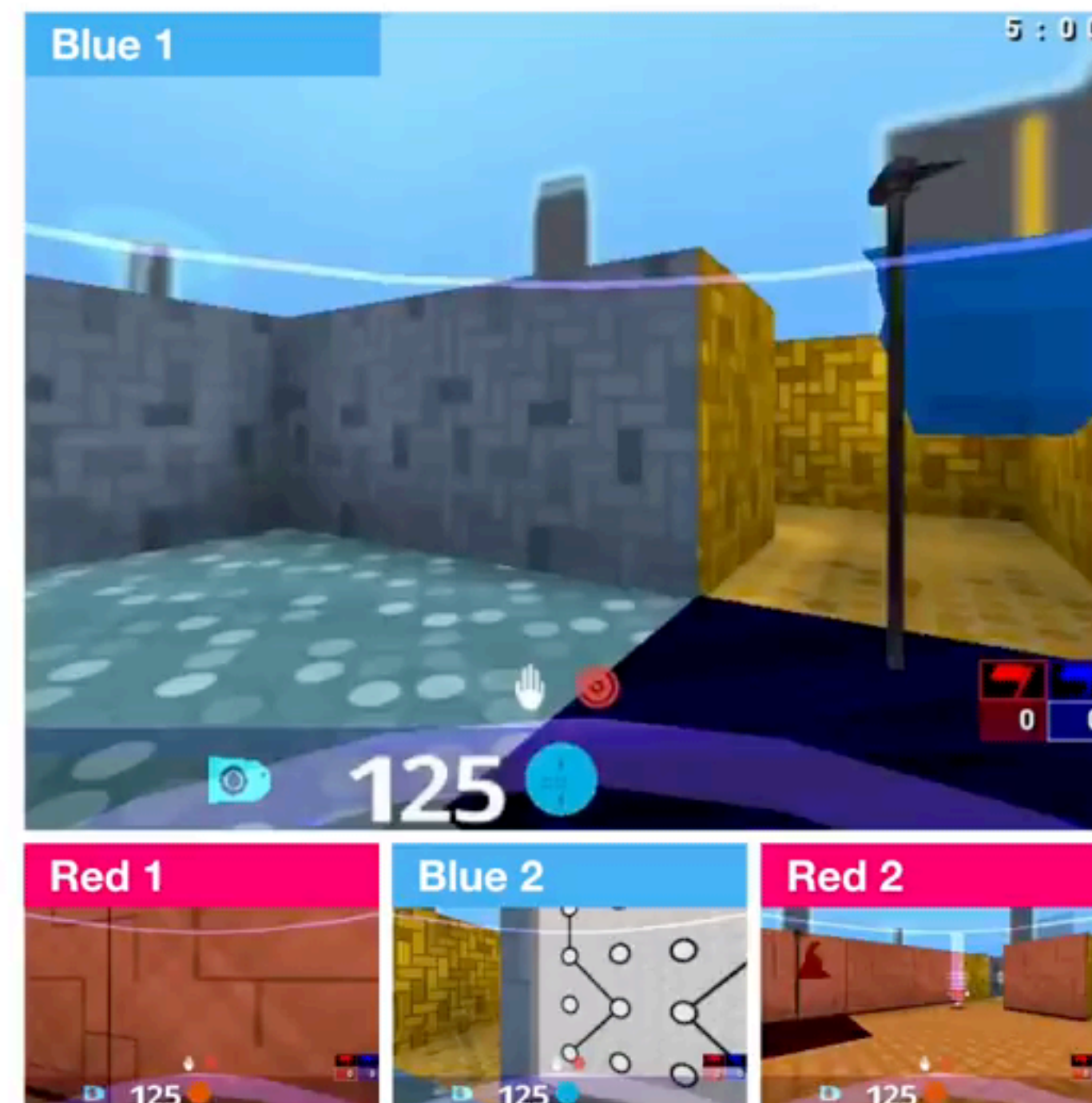
Today

- Formalisms for intelligent agents (environment, state, action, policy)
- Imitation learning
- Reinforcement learning
 - Policy gradient algorithm
 - Q-learning
- This is just a very high-level overview
 - See Sutton & Barto [<http://incompleteideas.net/book/RLbook2018.pdf>] for more.

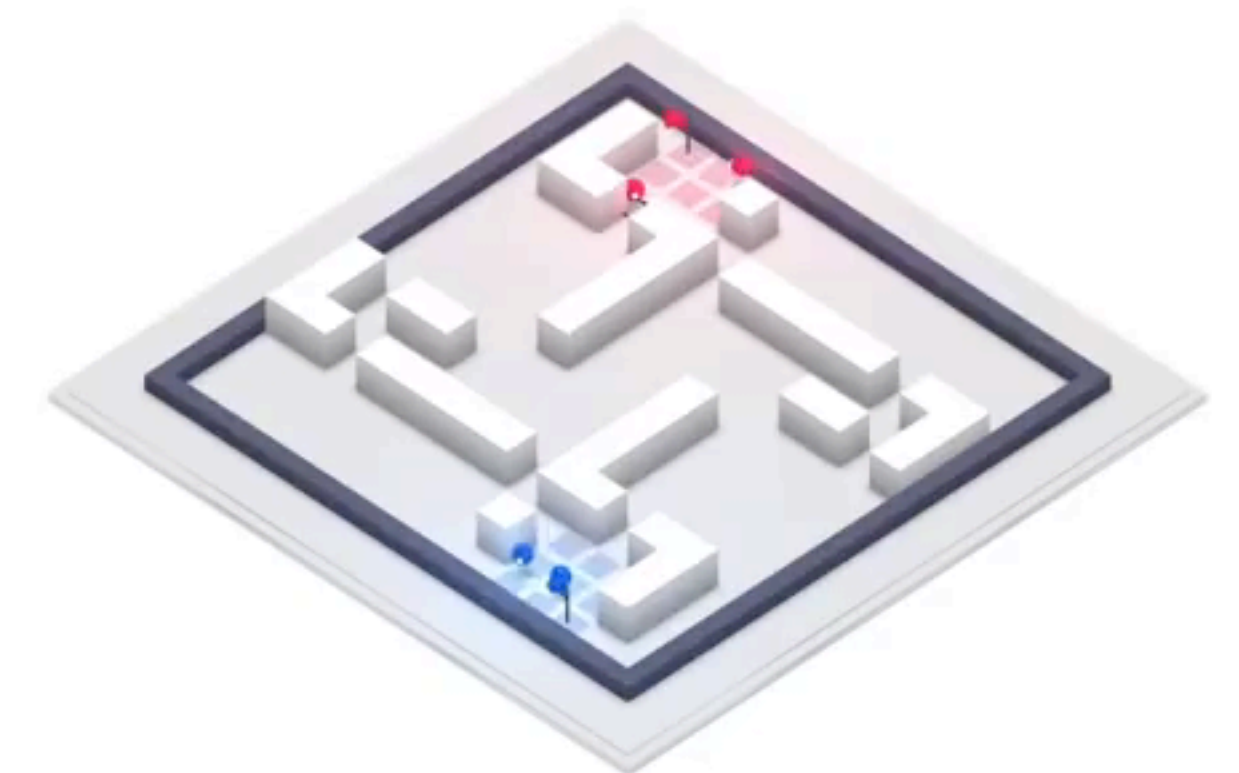


[Silver et al., 2016]

Agent observation raw pixels



[Jaderberg et al. 2018]

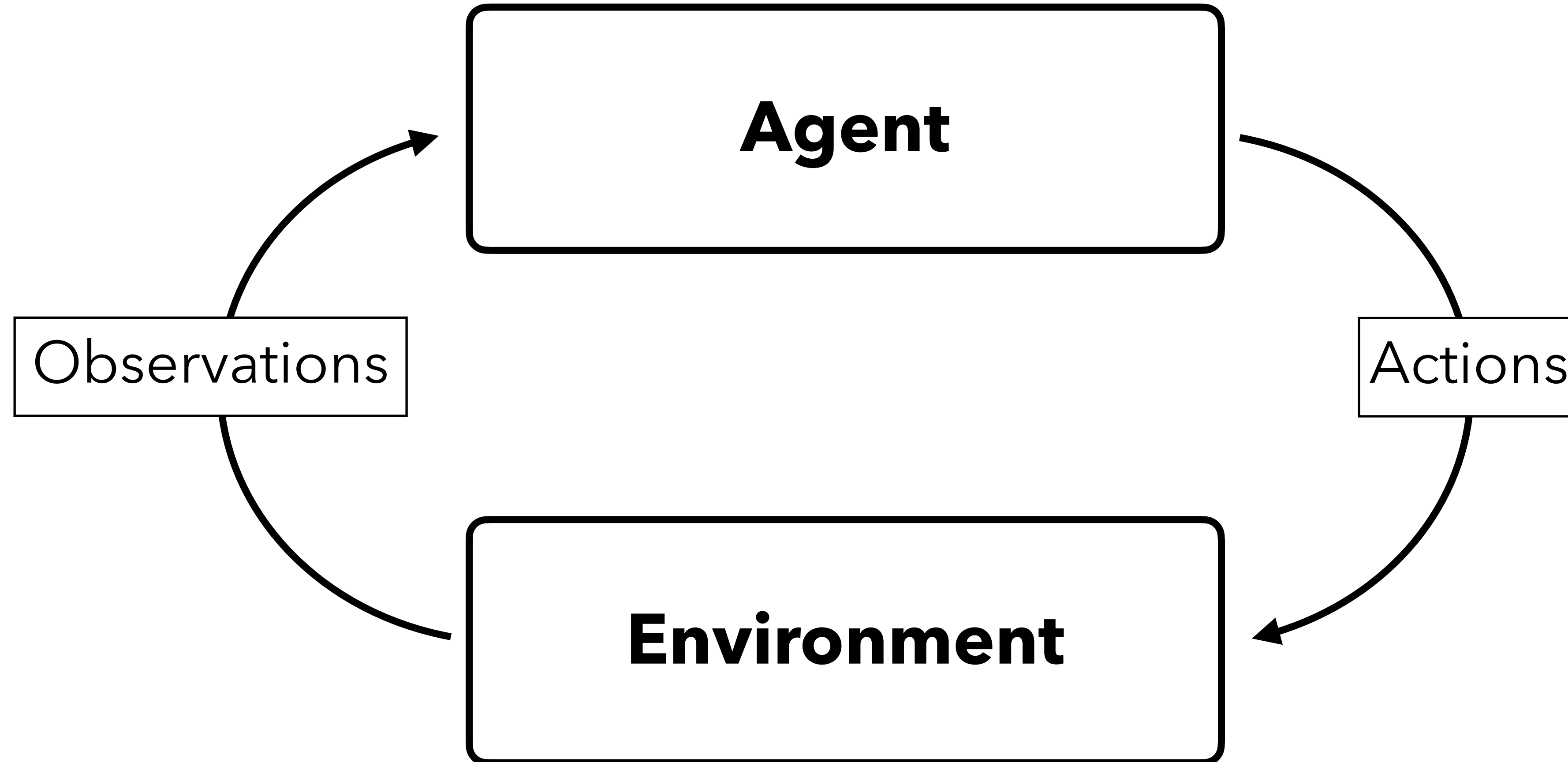


Indoor map overview

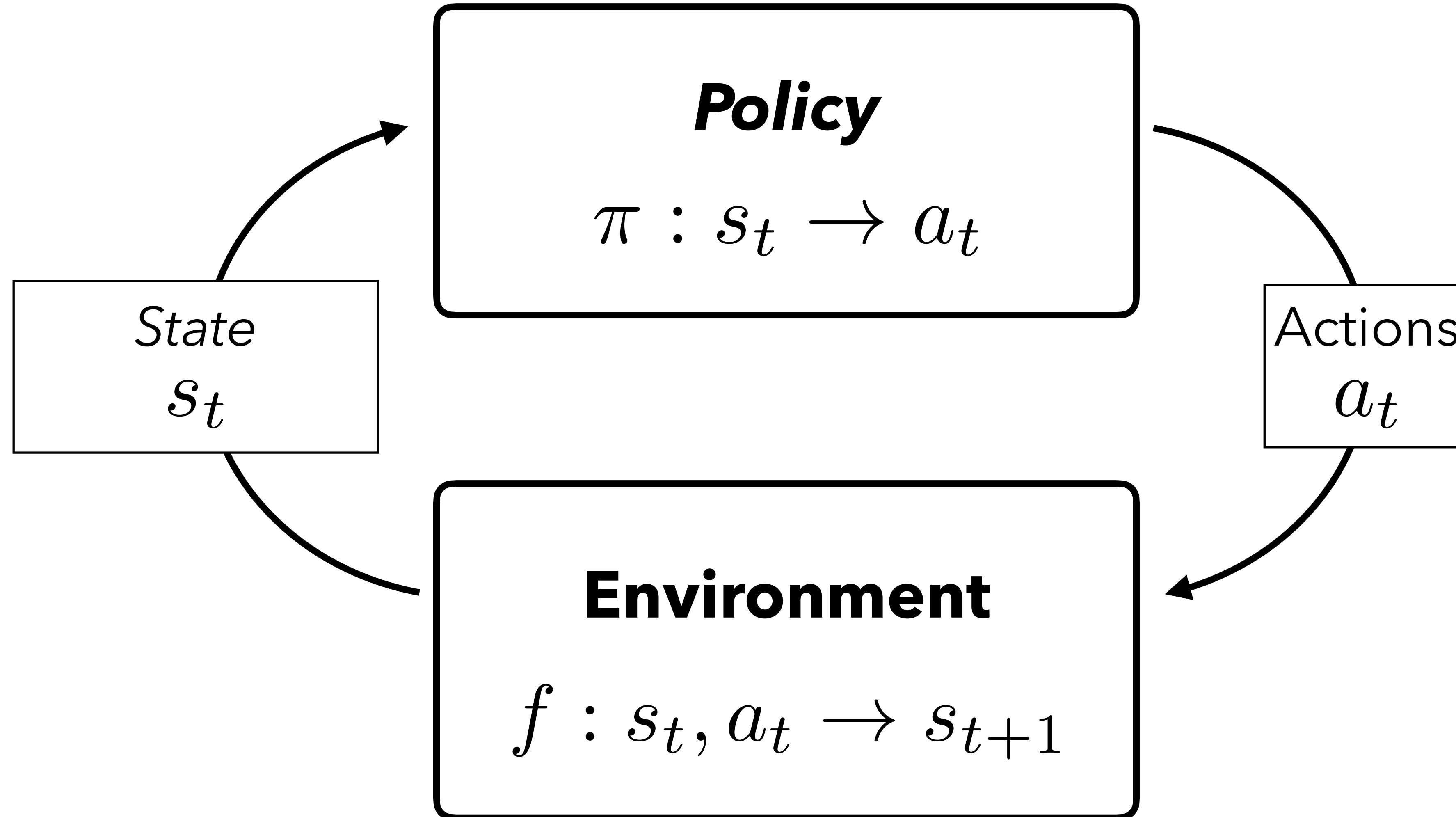
The whole purpose of visual perception, in humans, is to
make good motor decisions.

“We move in order to see and we see in order to move” – J. J. Gibson

Intelligent agents

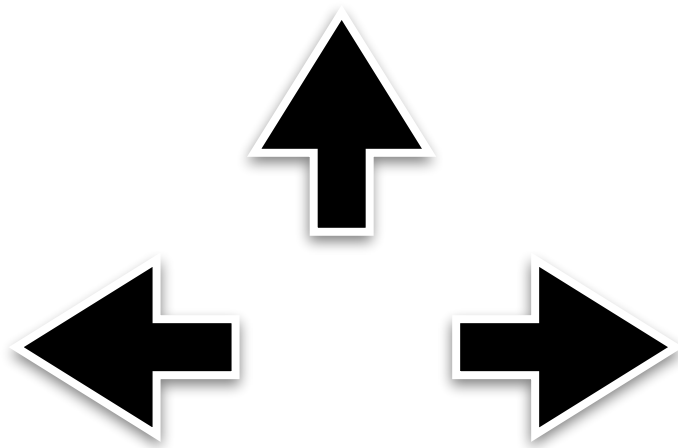
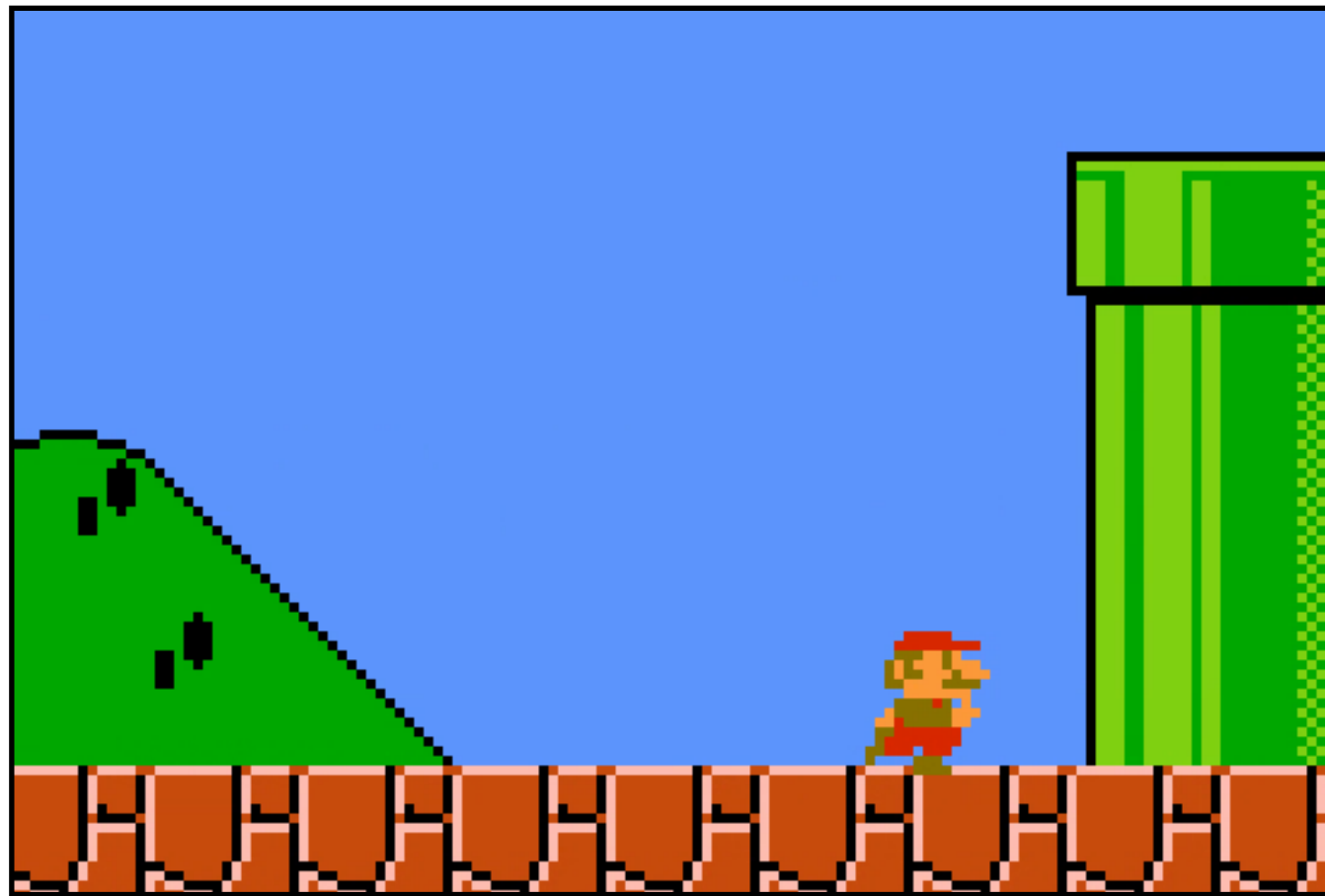


Intelligent agents

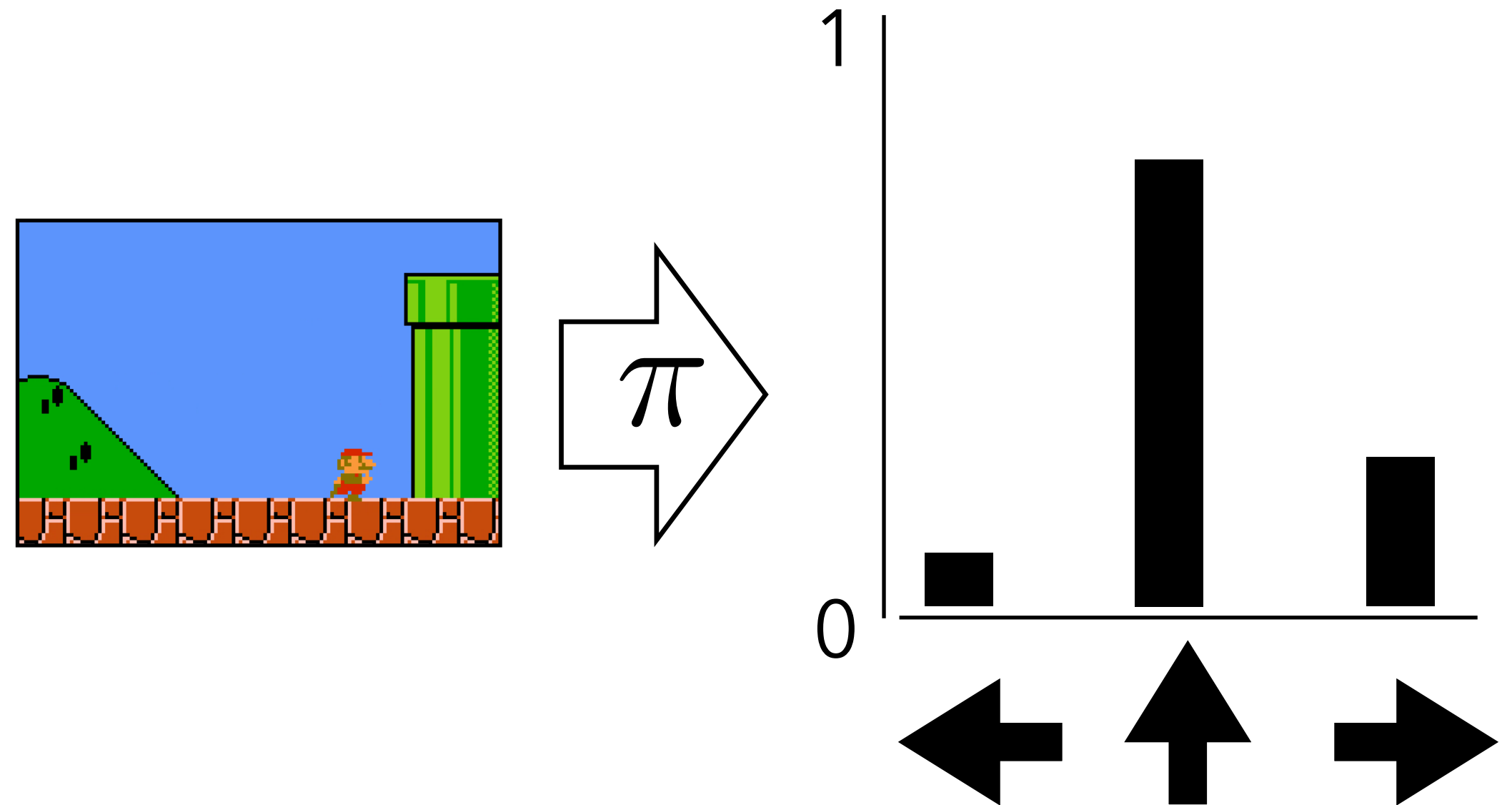


How to represent a state? How to represent policy?

state: pixels!

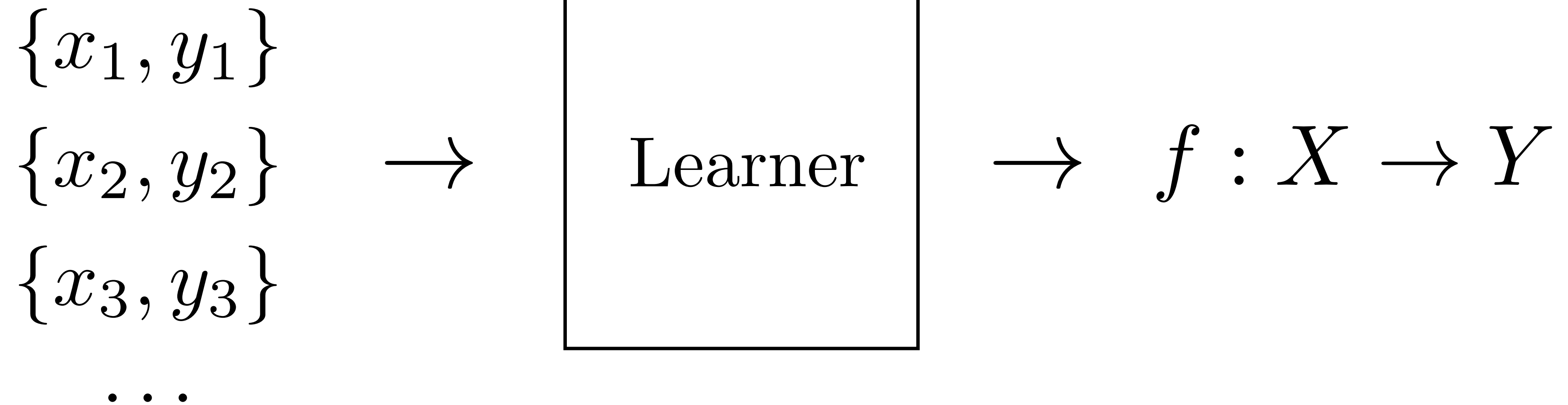


policy: action classifier



Can we use supervised learning?

Training data



$$f^* = \arg \min_{f \in \mathcal{F}} \sum_{i=1}^N \mathcal{L}(f(x_i), y_i)$$

Imitation learning

(supervised learning, applied to learn *policies*)

Training data

(from an **expert**)

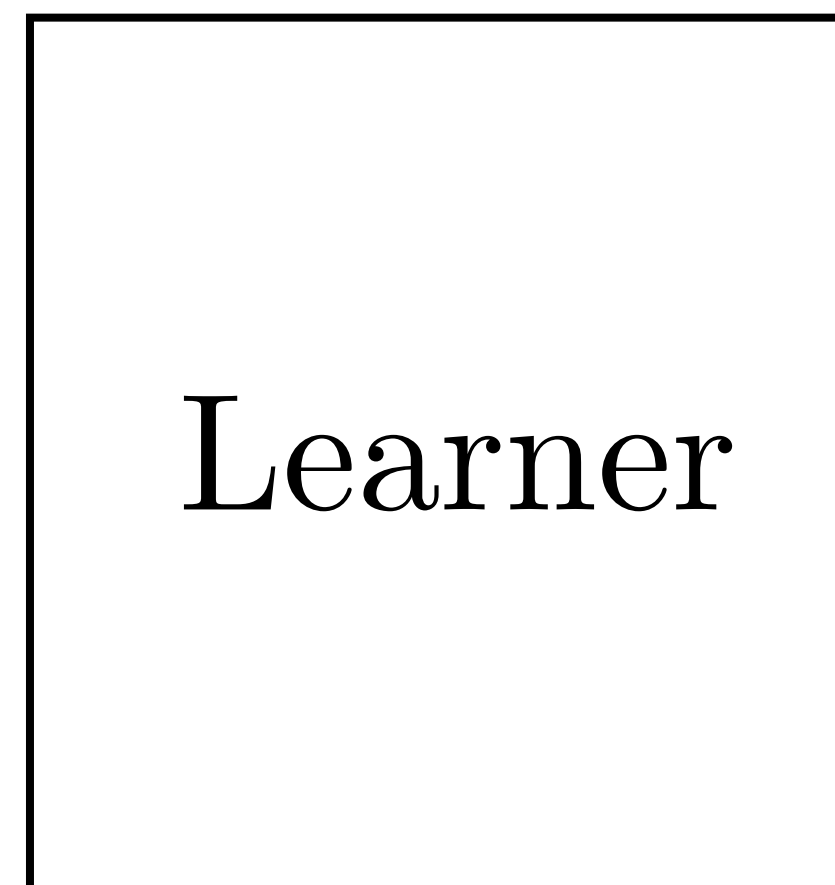
$\{s_1, a_1\}$

$\{s_2, a_2\}$

$\{s_3, a_3\}$

...

→

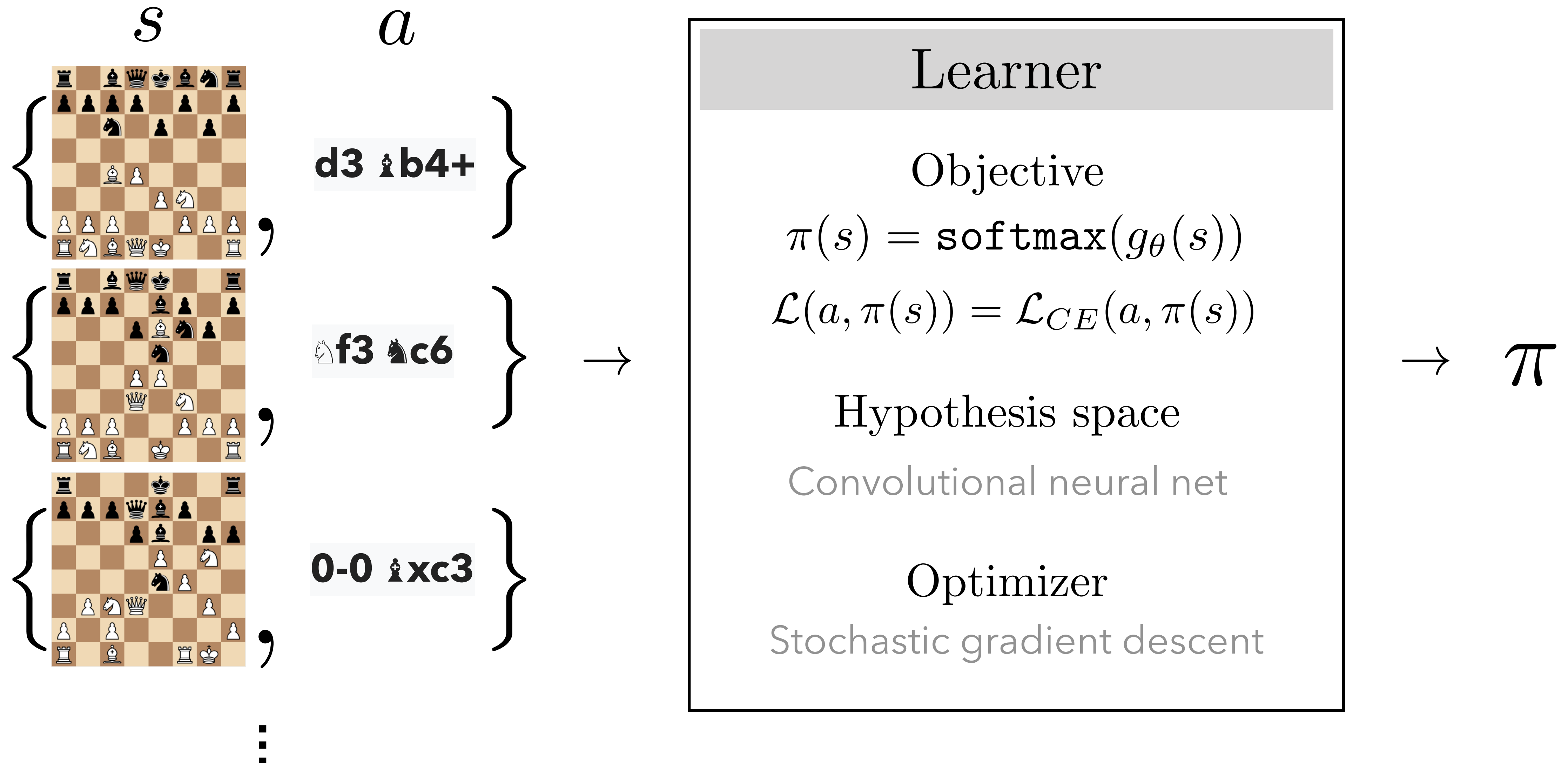


→

$\pi : s \rightarrow a$

$$\pi^* = \arg \min_{\pi \in \Pi} \sum_{i=1}^N \mathcal{L}(\pi(s_i), a_i)$$

Imitation learning

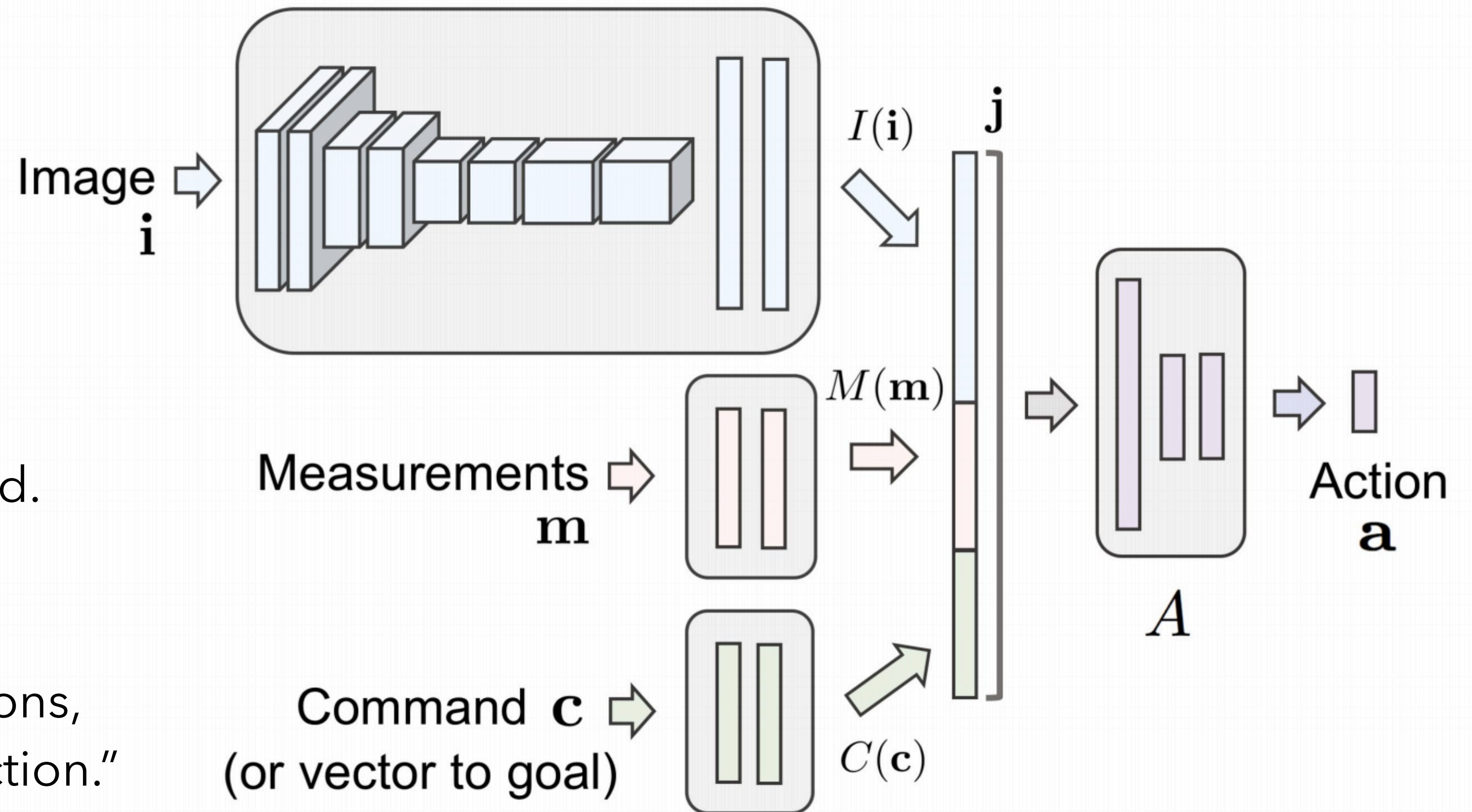


From images to actions



+ auxiliary measurements, e.g. speed.

+ a goal capturing expert's intentions,
e.g. "Turn right at the next intersection."



End-to-end Driving via Conditional Imitation Learning

Felipe Codevilla, Matthias Mueller, Alexey Dosovitskiy, Antonio Lopez, Vladlen Koltun

Submitted to ICRA 2018

FPS: 15

Speed: 0 km/h
Gear: N



4x

Exploiting other knowledge

Segmentation



Albedo



Depth



Optical flow



- Can use mid-level representations like depth, motion.
- Or do transfer learning from pretrained net

Teleoperation

ALOHA 2 🤞



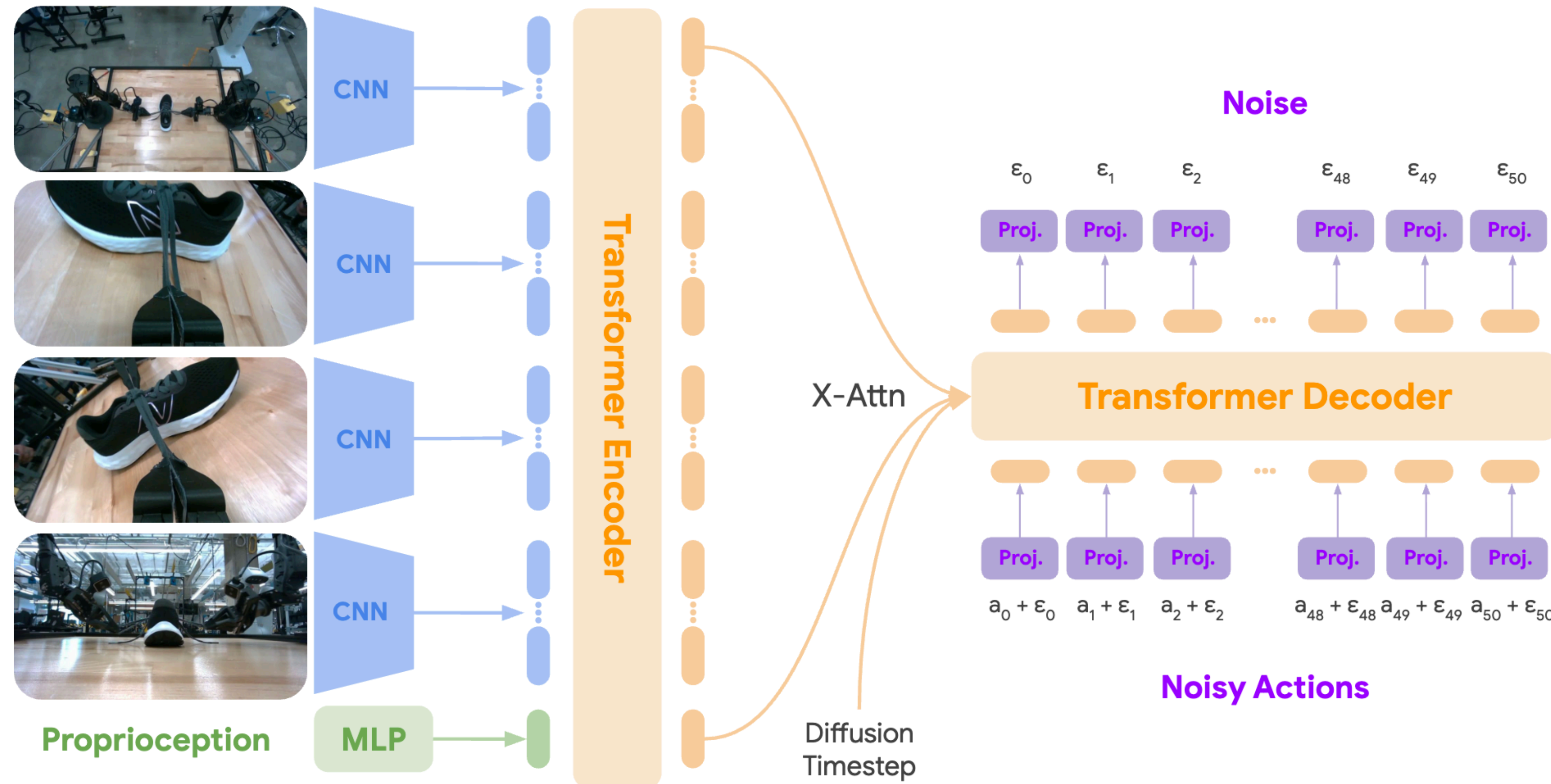
Google DeepMind

Stanford
University

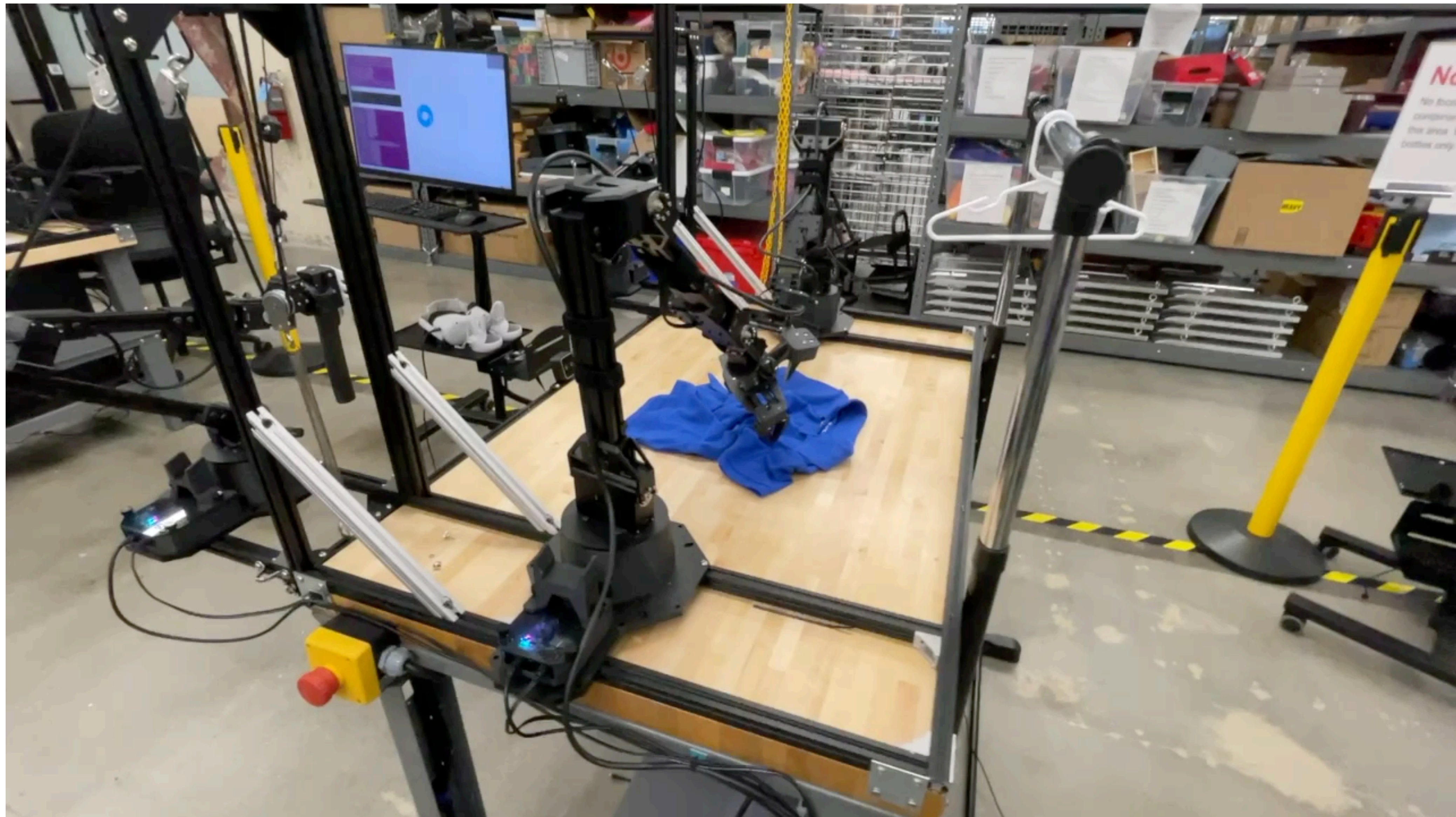
Teleoperation for manipulation tasks



Predicting actions from video



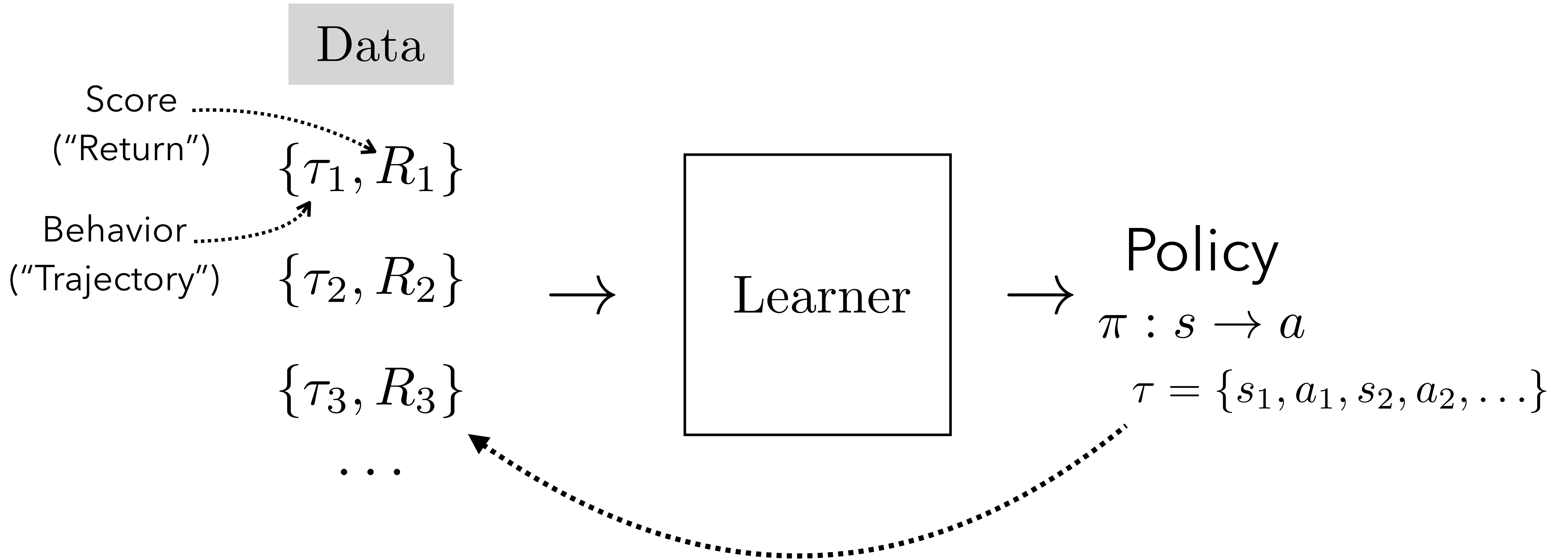
Behavior cloning for object manipulation



Behavior cloning for object manipulation

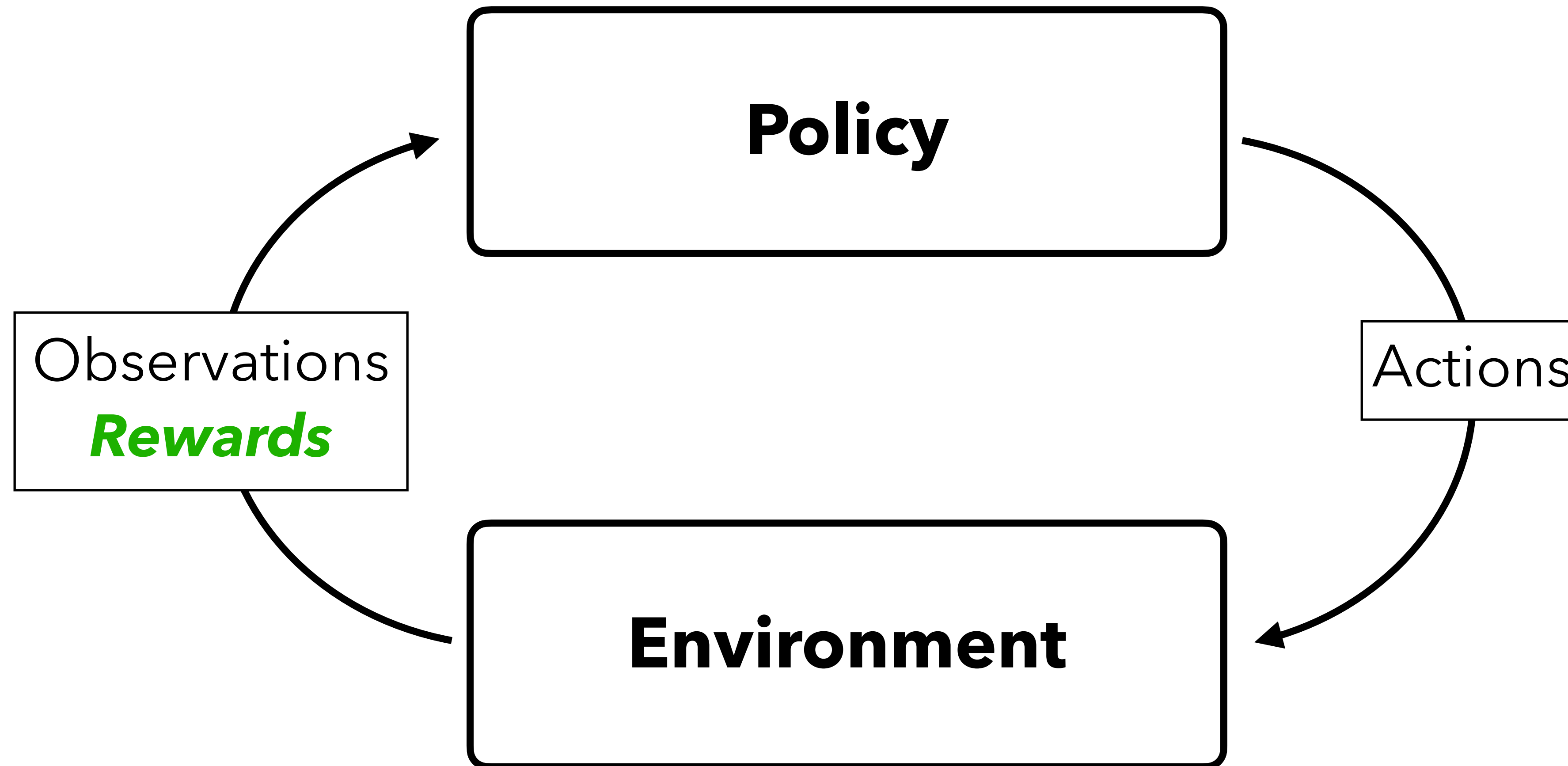


Reinforcement learning



What's a good policy? (what's the learning objective?)

Reinforcement learning



Learn a policy that takes actions that maximize **reward**

Imitation learning

Hand-curated training data

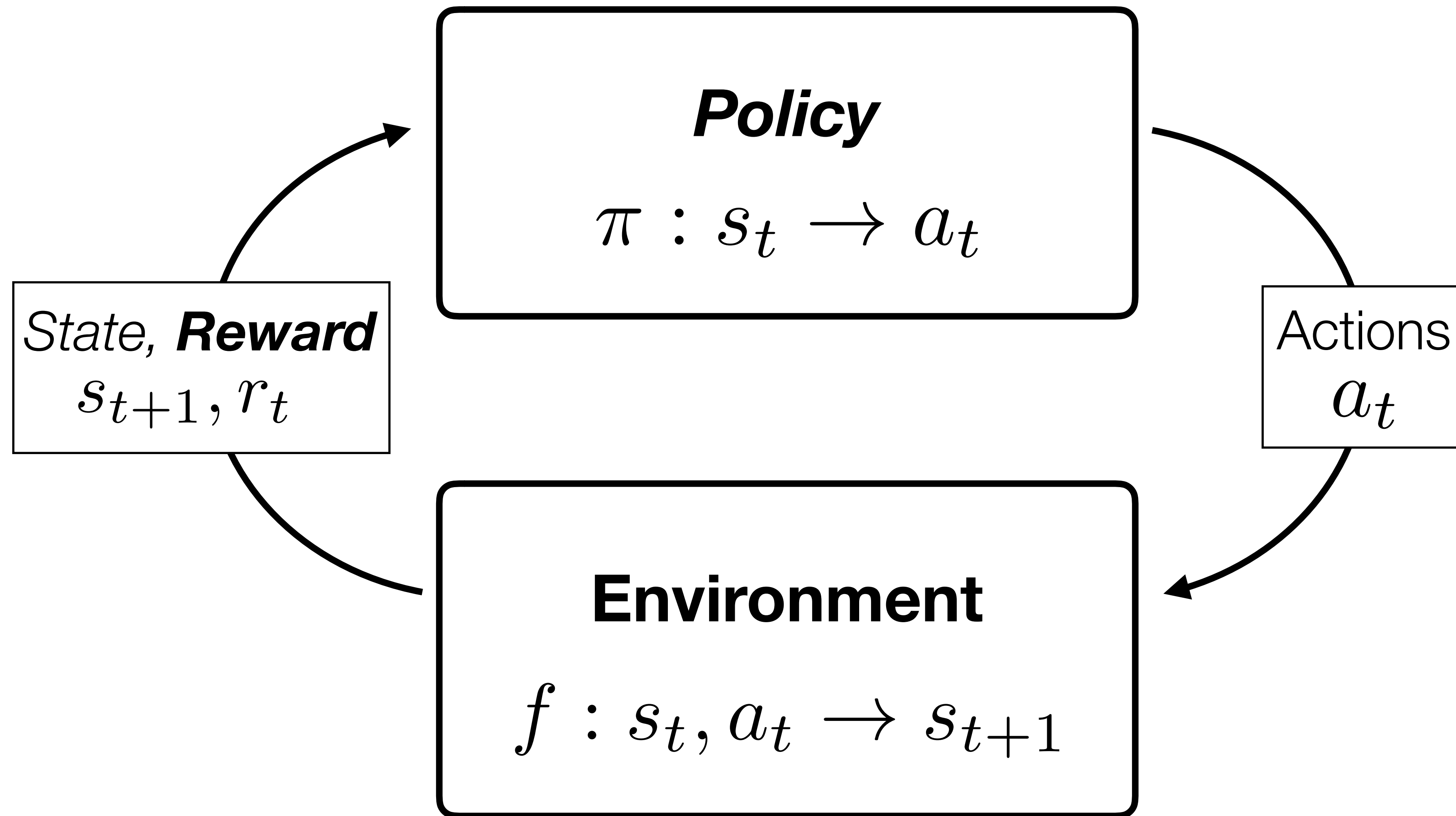
- + Instructive examples
- + Follows a curriculum
- Expensive
- Limited to teacher's knowledge

Reinforcement learning

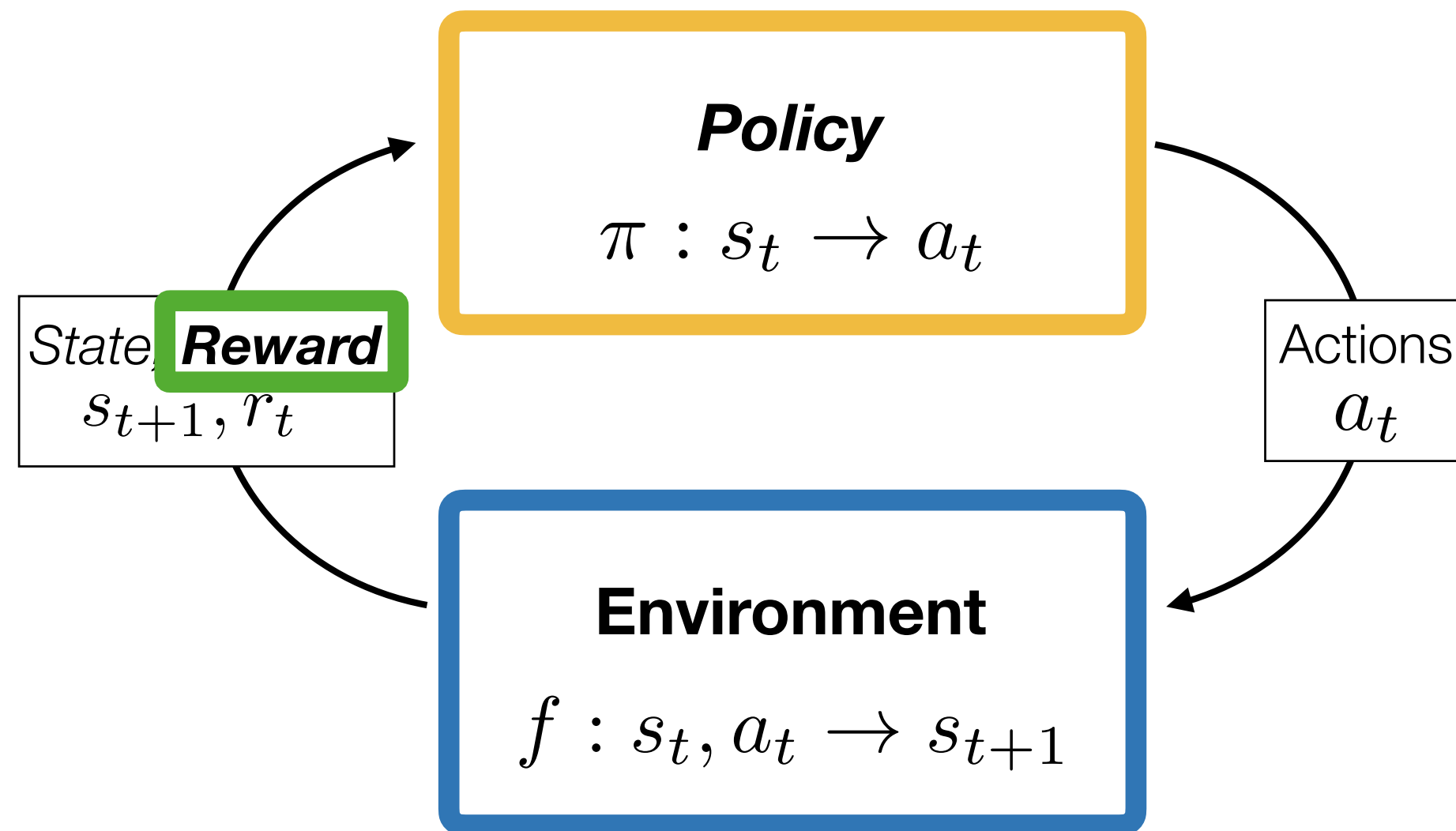
No training data, have to play around and collect the data *yourself*

- + No need for labeled data
- + Can learn things no human knows how to do
- Less instructive
- No curriculum
- Have to explore

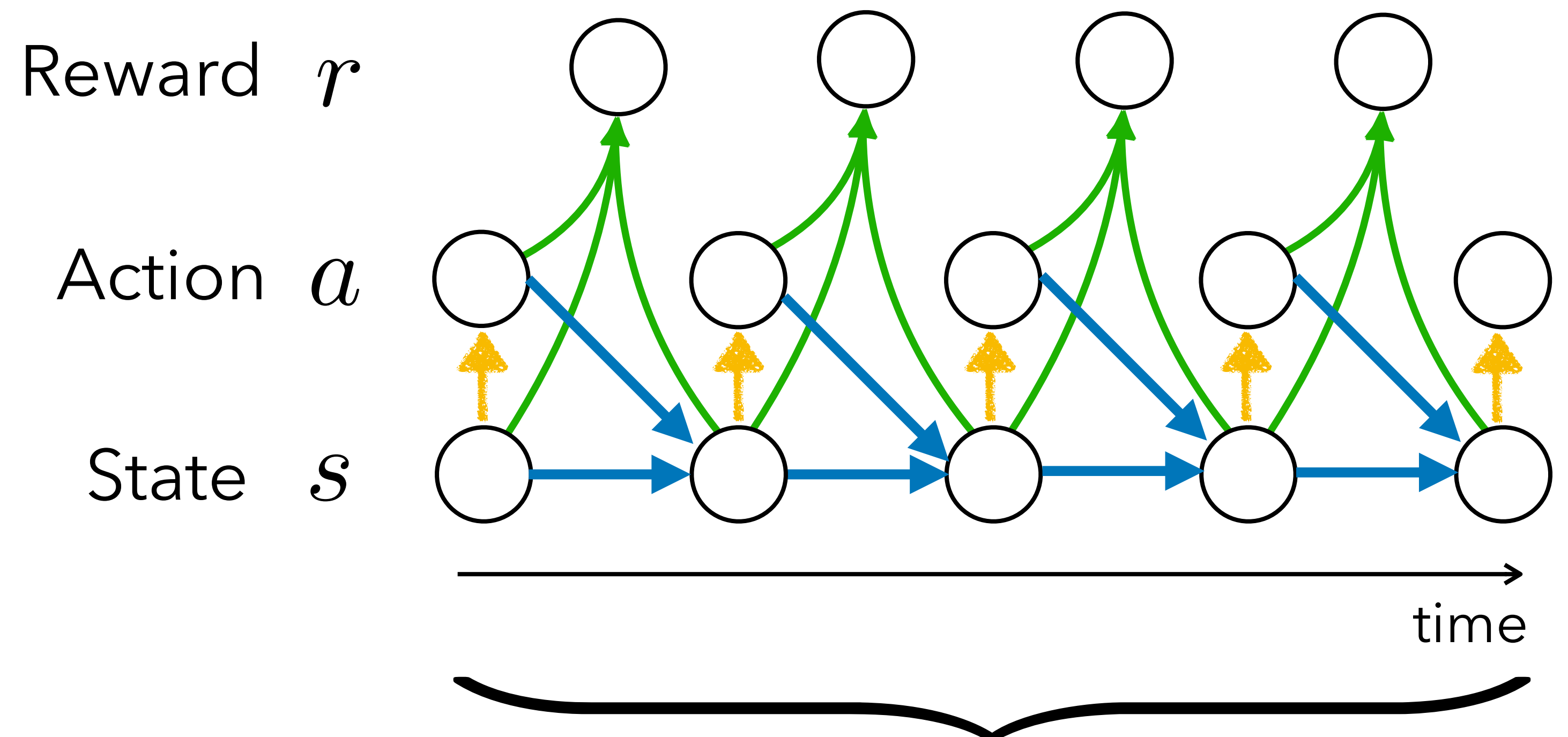
Reinforcement learning



Reinforcement learning



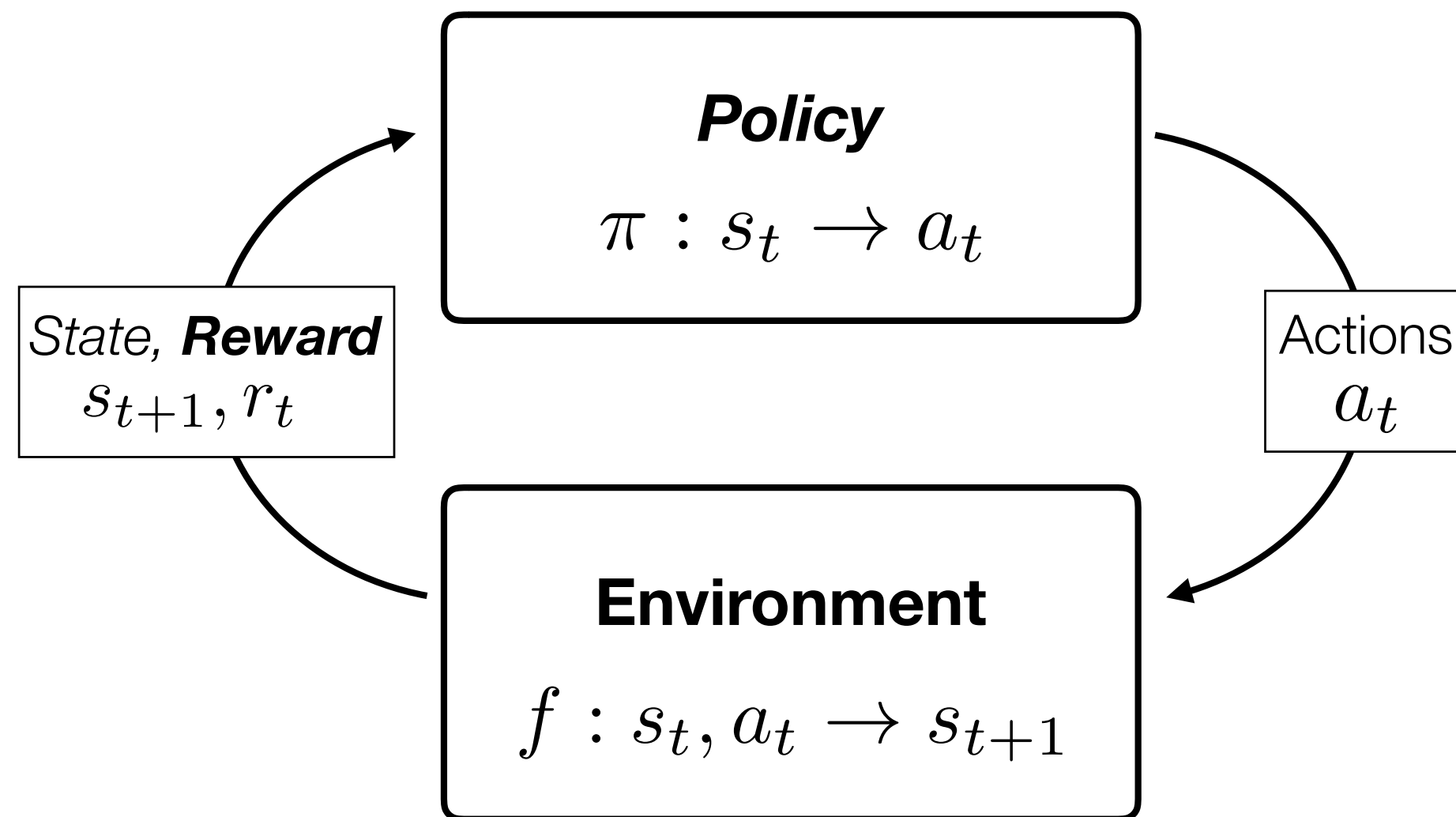
Markov decision process (MDP)



A sample from the MDP is called a **trajectory**

$$\tau = (s_0, a_0, r_0, s_1, a_1, r_1, \dots)$$

Reinforcement learning



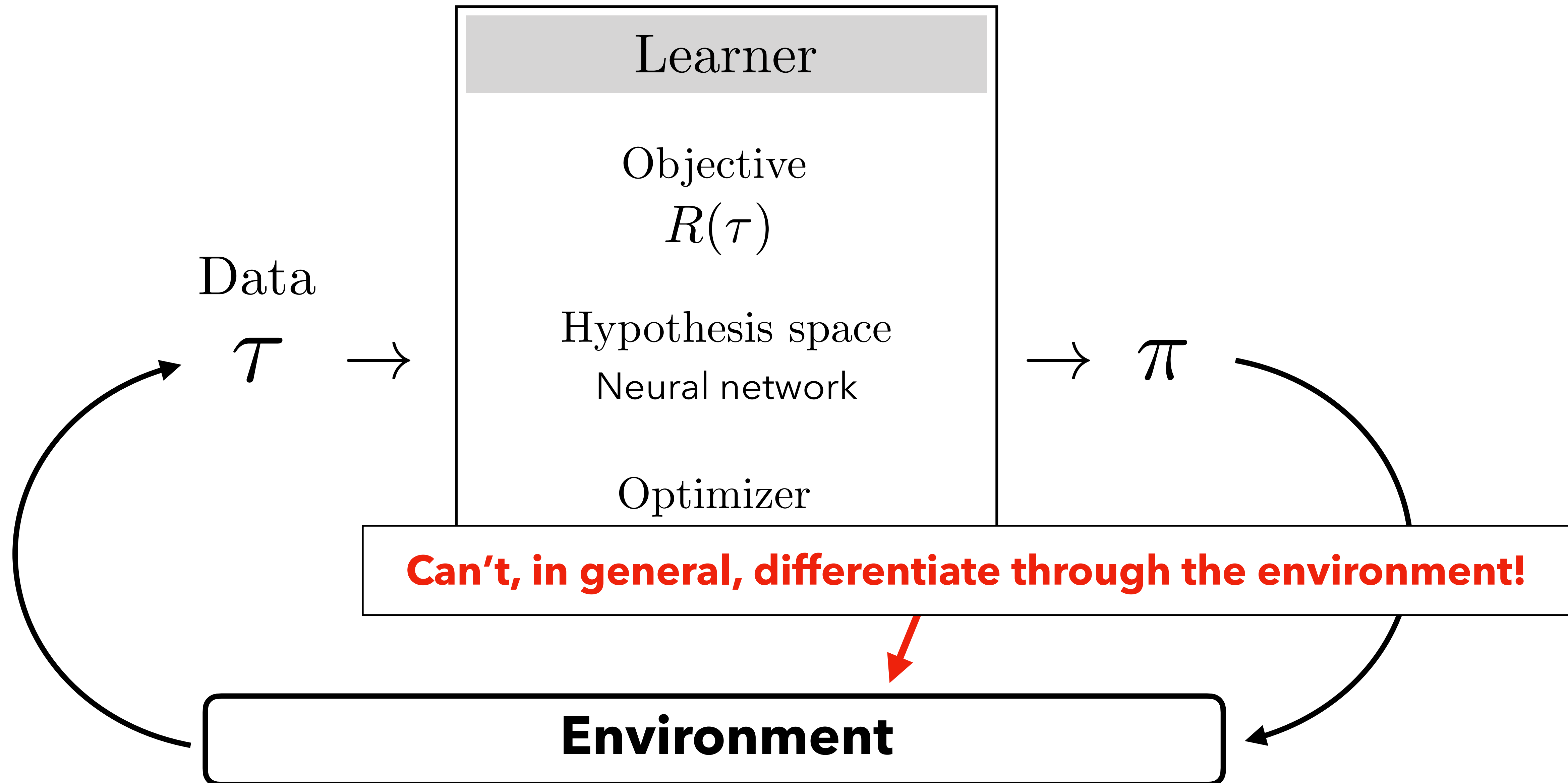
Trajectory $\tau = (s_0, a_0, r_0, s_1, a_1, r_1, \dots)$

Discounted rewards $R(\tau) = \sum_{t=0}^{\infty} \gamma^t r_t, \quad \gamma \in (0, 1)$

Learn a policy that takes actions that maximize expected reward

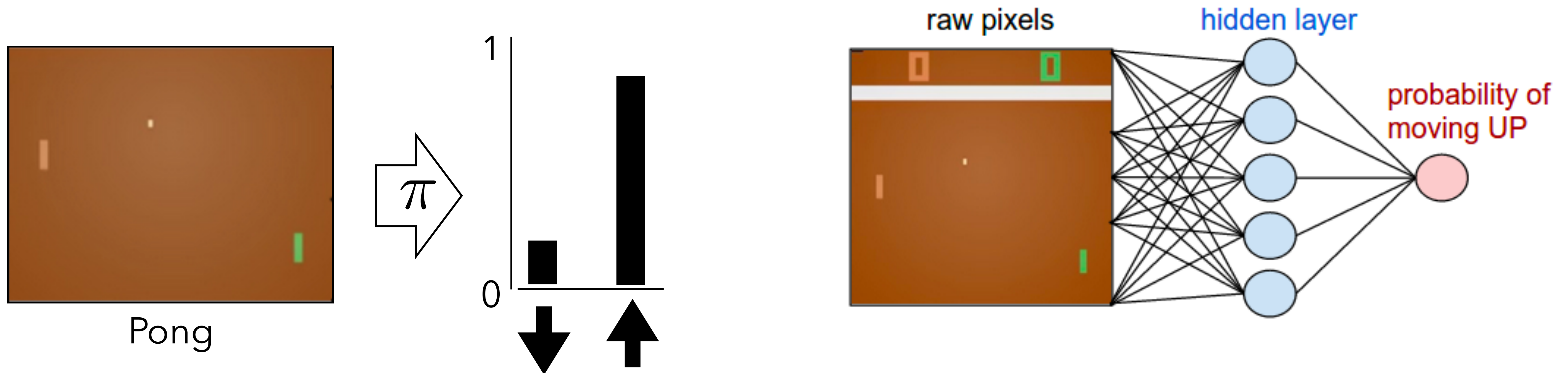
$$\pi^* = \arg \max_{\pi} \mathbb{E}_{\tau \sim \pi} [R(\tau)]$$

Reinforcement learning



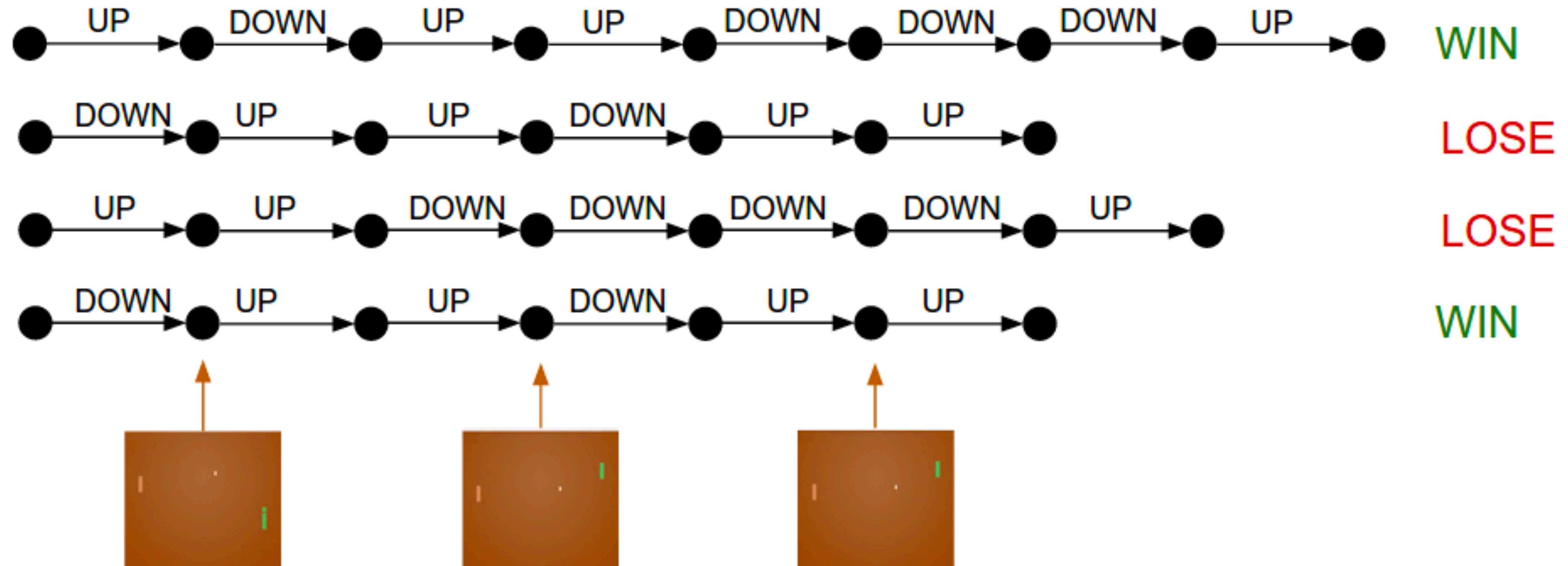
Environment is not differentiable! – How to optimize?

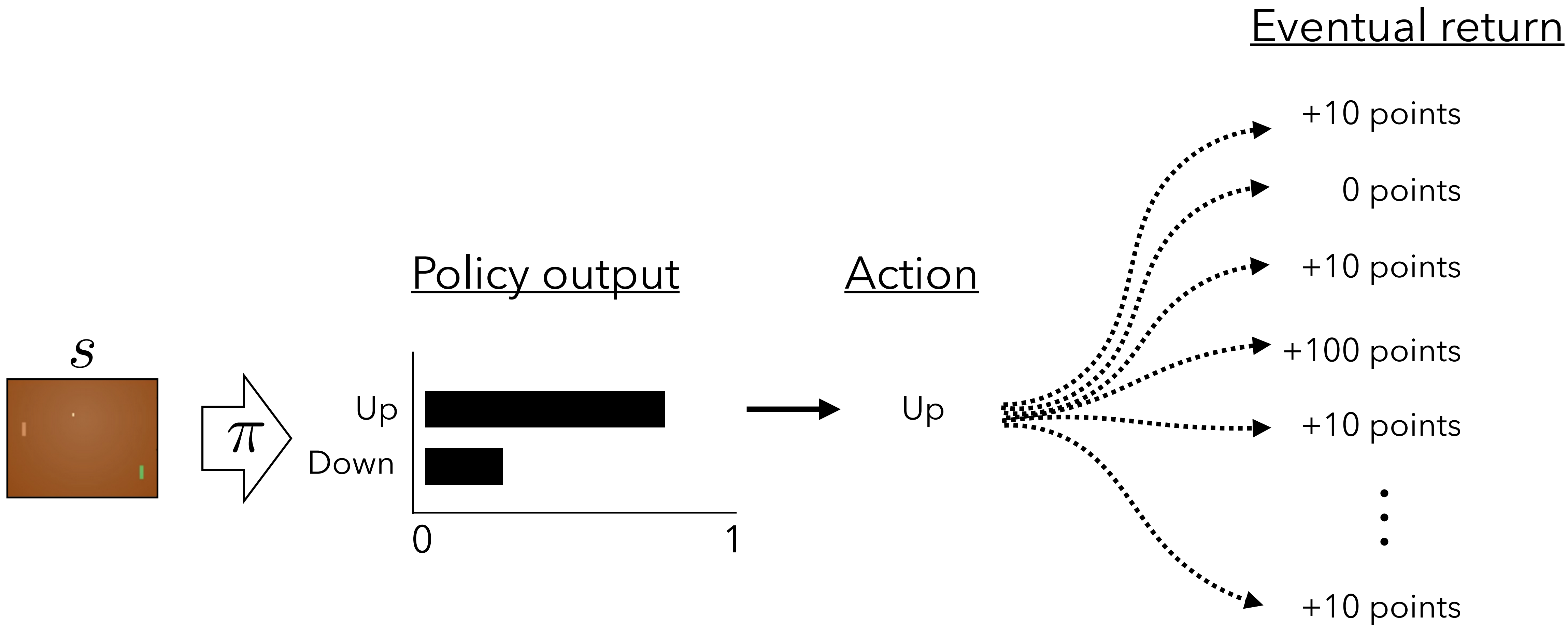
Policy gradients: Run a policy for a while. See what actions led to high rewards. Increase their probability.



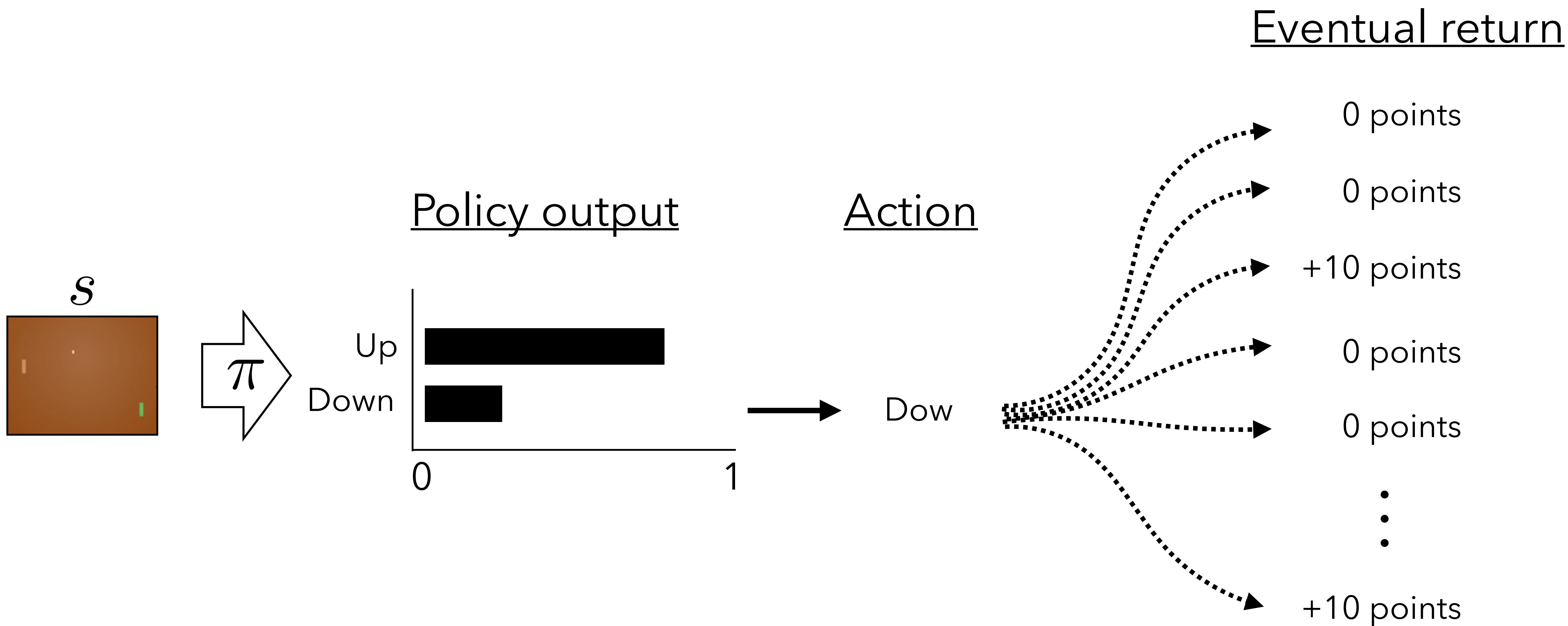
[Adapted from Andrej Karpathy: <http://karpathy.github.io/2016/05/31/rl/>]

Policy gradients: Run a policy for a while. See what actions led to high rewards. Increase their probability.





$\pi(a|s)$ = probability of choosing action a given state s



$\pi(a|s)$ = probability of choosing action a given state s

Policy gradient

Goal: take derivatives of expected reward w.r.t. the policy parameters.

$$\begin{aligned}\frac{\partial}{\partial \theta} \mathbb{E}_{\tau \sim \pi_{\theta}} [R(\tau)] &= \frac{\partial}{\partial \theta} \int_{\tau} p(\tau | \theta) R(\tau) d\tau \\ &= \int_{\tau} p(\tau | \theta) \left[\frac{\partial}{\partial \theta} \log(p(\tau | \theta)) \right] R(\tau) d\tau \\ &= \mathbb{E}_{\tau \sim \pi_{\theta}} \left[\frac{\partial}{\partial \theta} \log(p(\tau | \theta)) R(\tau) \right]\end{aligned}$$

- Do actions with high rewards more often, and low rewards less often
- This is called the REINFORCE algorithm.
 - Estimate gradients, do gradient ascent

Policy gradient

Looks a lot like stochastic gradient descent on policy:

1. Sample a rollout, e.g. play the game with current policy

$$\tau = (s_0, a_0, s_1, a_1, \dots, s_T, a_T)$$

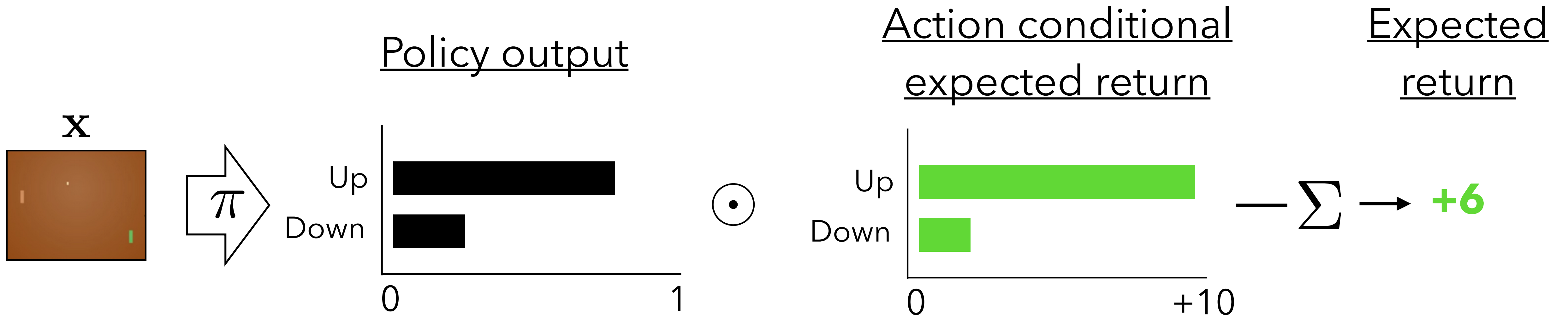
2. Compute reward, e.g. what was our game score?

$$r(\tau) = \sum_{t=0}^T R(s_t)$$

3. Do a gradient update:

$$\theta \leftarrow \theta + \alpha r(\tau) \frac{\partial}{\partial \theta} \pi_{\theta}(a_t | s_t)$$

Approximated via sampling



$$\nabla_{\theta} \mathbb{E}_{\tau \sim \pi_{\theta}} [R(\tau)] = \mathbb{E}_{\tau \sim \pi_{\theta}} [R(\tau) \nabla_{\theta} \log \pi_{\theta}] \quad \leftarrow \text{Estimate gradient using REINFORCE and do gradient descent}$$

Policy gradient

1. Start with an arbitrary initial policy.
2. **Roll out** this *stochastic* policy many times, sampling different random actions each time.
3. Update your policy to place higher probability on actions that led to higher returns.

Mathematically, this approximates gradient ascent on policy parameters, so as to maximize reward.

Policy gradient

- What happens in a rollout? Recall we're taking a step: $\mathbb{E}_{\tau \sim \pi_{\theta}} \left[\frac{\partial}{\partial \theta} \log(p(\tau|\theta)) R(\tau) \right]$

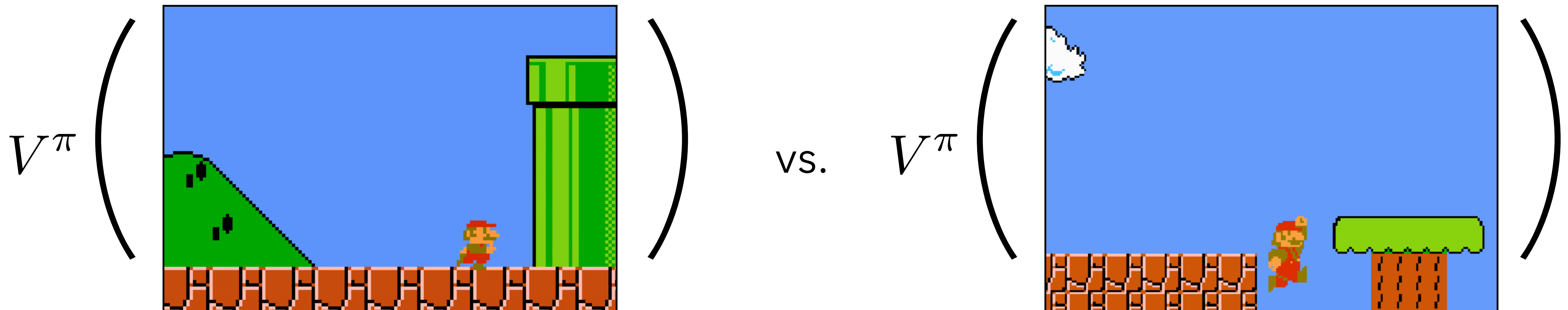
$$\frac{\partial}{\partial \theta} \log(p(\tau|\theta)) = \sum_{t=0}^T \frac{\partial}{\partial \theta} \log \pi_{\theta}(a_t|s_t)$$

- *All* actions become more likely if the reward is high.
- Doesn't do *credit assignment*.

How good is a state?

Value function: expected future reward from starting in s .

$$V^{\pi}(s) = \mathbb{E} \left[\sum_{t \geq 0} \gamma^t r_t \mid s_0 = s, \pi \right]$$



- One advantage is credit assignment: we know which state/action was useful.
- Sometimes more sample efficient, and updates have less variance.

How good is a state-action pair?

- Could we *learn* the value function and use it to choose actions?
 - We need more than that. You'd also need to know the dynamics, i.e. what state you'd end up with if you took each action.
- Instead, learn **action-value function** (or **Q function**).

$$Q(s, a) = \mathbb{E} \left[\sum_{t \geq 0} R_t \mid s_0 = s, a_t = a \right]$$

- Optimal action for a state: $\operatorname{argmax}_a Q(s, a)$

Finding a good Q function

- Good Q function should satisfy a recurrence relation called the Optimal Bellman Equation:

Quality of state/action pair Where will I end up? What if I take the *very best* next action?

$$Q^*(s, a) = r(s, a) + \gamma \mathbb{E}_{p(s'|s, a)} \left[\max_{a'} Q^*(s_{t+1}, a') \mid s_t = s, a_t = a \right]$$

Finding a good Q function

- Measuring the Bellman error for Q:

$$r(s_t, a_t) + \gamma \max_a Q(s_{t+1}, a) - Q(s_t, a_t)$$

- Approximate Q with a neural net $Q(s, a; \theta)$. For each episode i :

1. Do the policy induced by Q and get a trajectory:

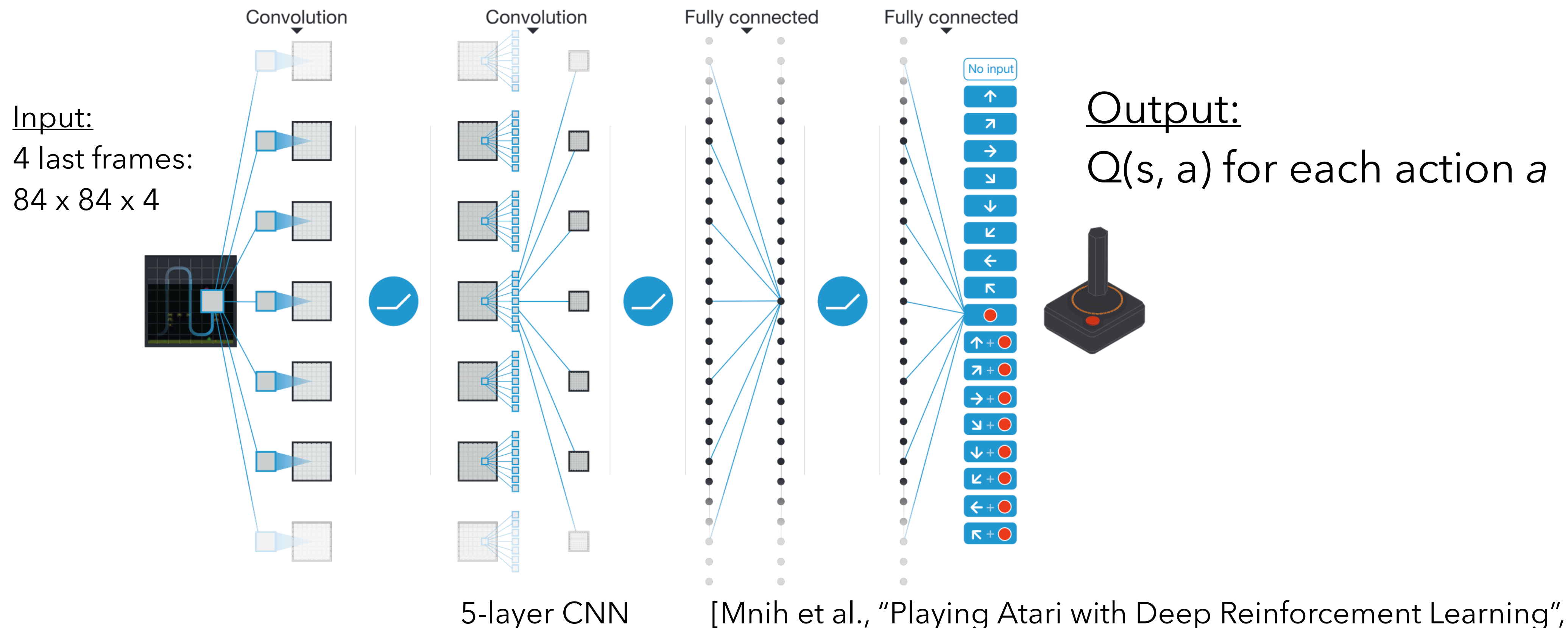
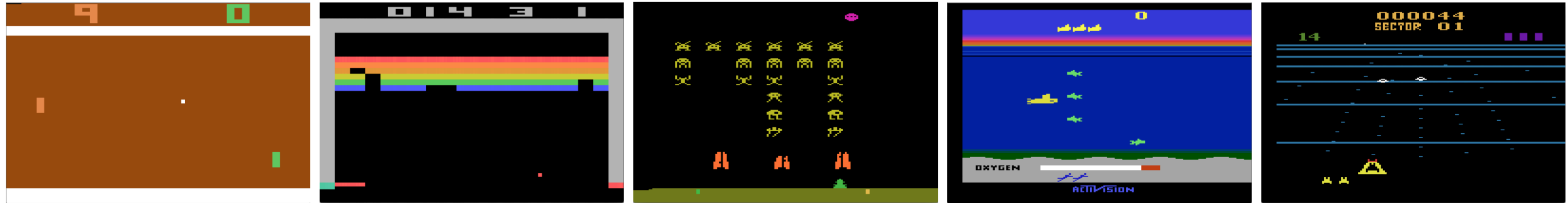
$$\tau = (s_0, a_0, r_0, s_1, a_1, r_1, \dots)$$

2. Update the parameters using backprop, minimizing approximation error:

$$t_i = r(s_t, a_t) + \gamma \max_a Q(s_{t+1}, a; \theta_{i-1})$$

$$L(\theta_i) = (t_i - Q(s_t, a_t; \theta_i))^2$$

Playing Atari games with deep Q-learning



Playing Atari games

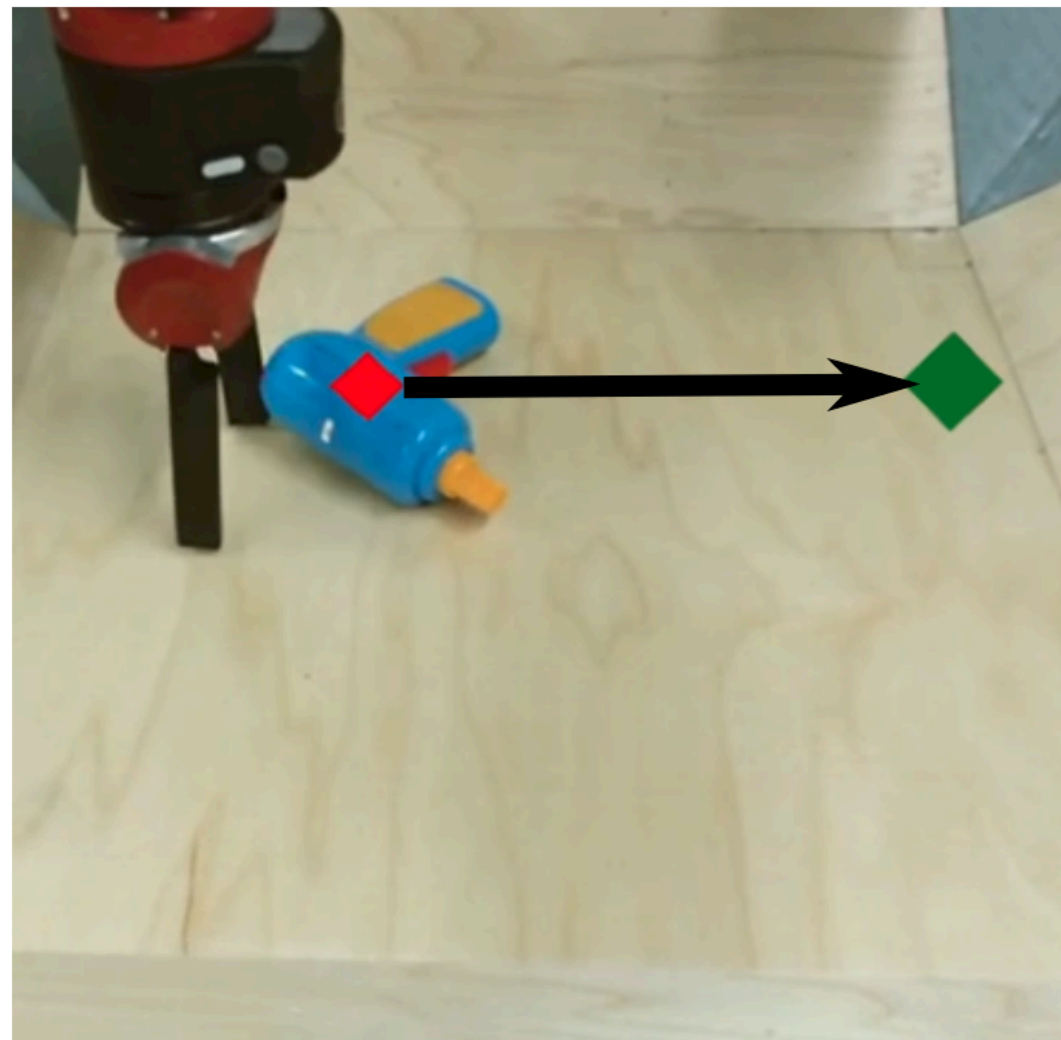
100 Training Episodes

Model-based control

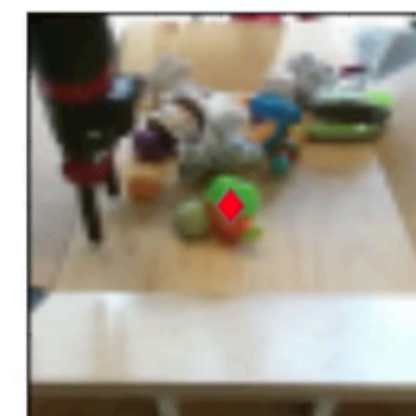
- Learn the **dynamics** of the environment: $p(s_{t+1} \mid s_t, a_t)$
- Where do I end up in the future if I perform this action?
- If states are images, we want to **predict the future** after you do an action

Model-based control

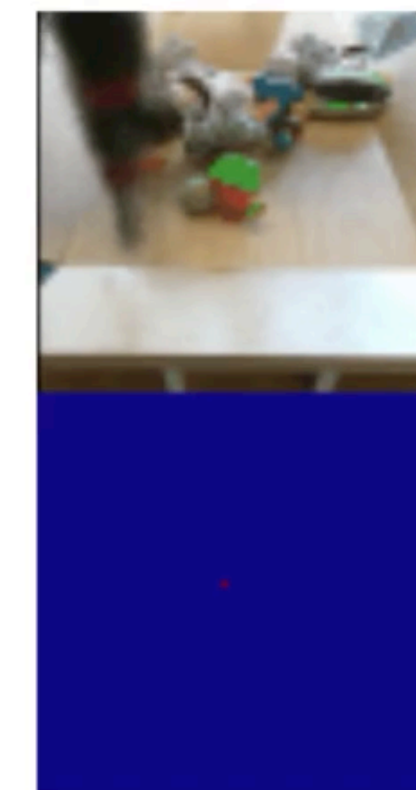
- Learn the **dynamics** of the environment: $p(s_{t+1} \mid s_t, a_t)$
- Where do I end up in the future if I perform this action?
- If states are images, we want to **predict the future** after you do an action



Pushing task



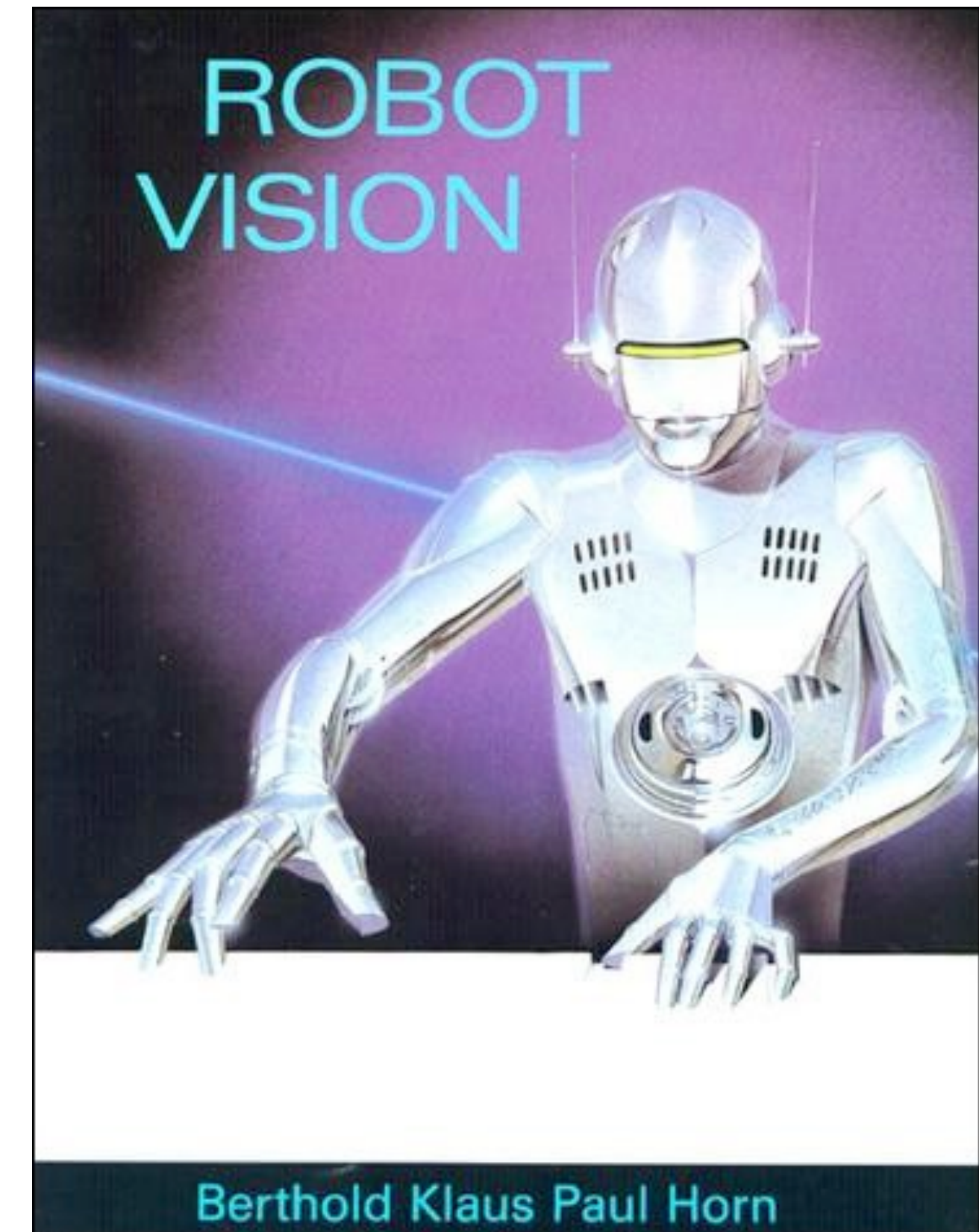
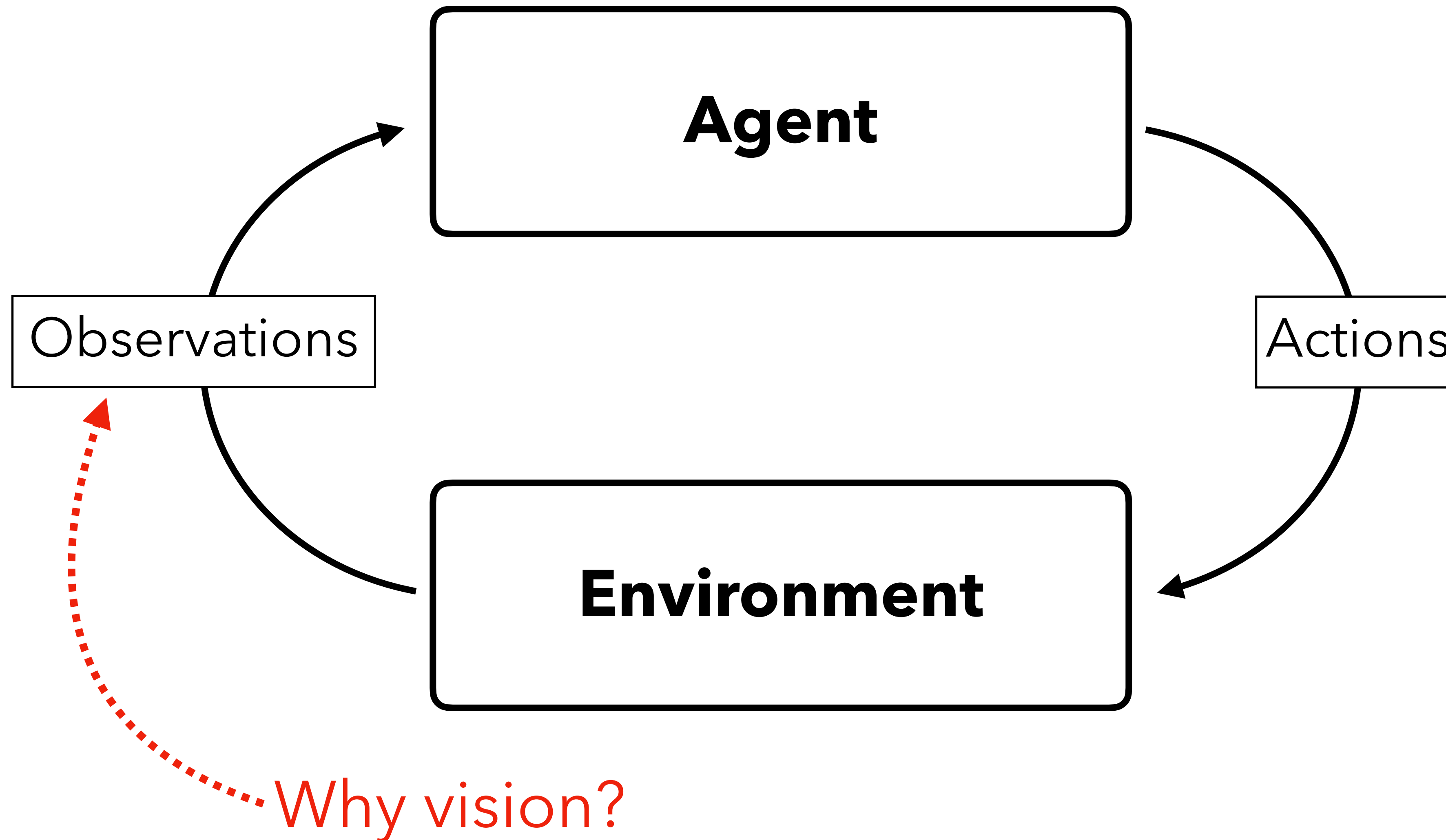
Designated Pixel ♦



SNA (Ours)

Video prediction

Intelligent agents



Why vision?

1. Human-like intelligence (and animal-like), often relies heavily on vision

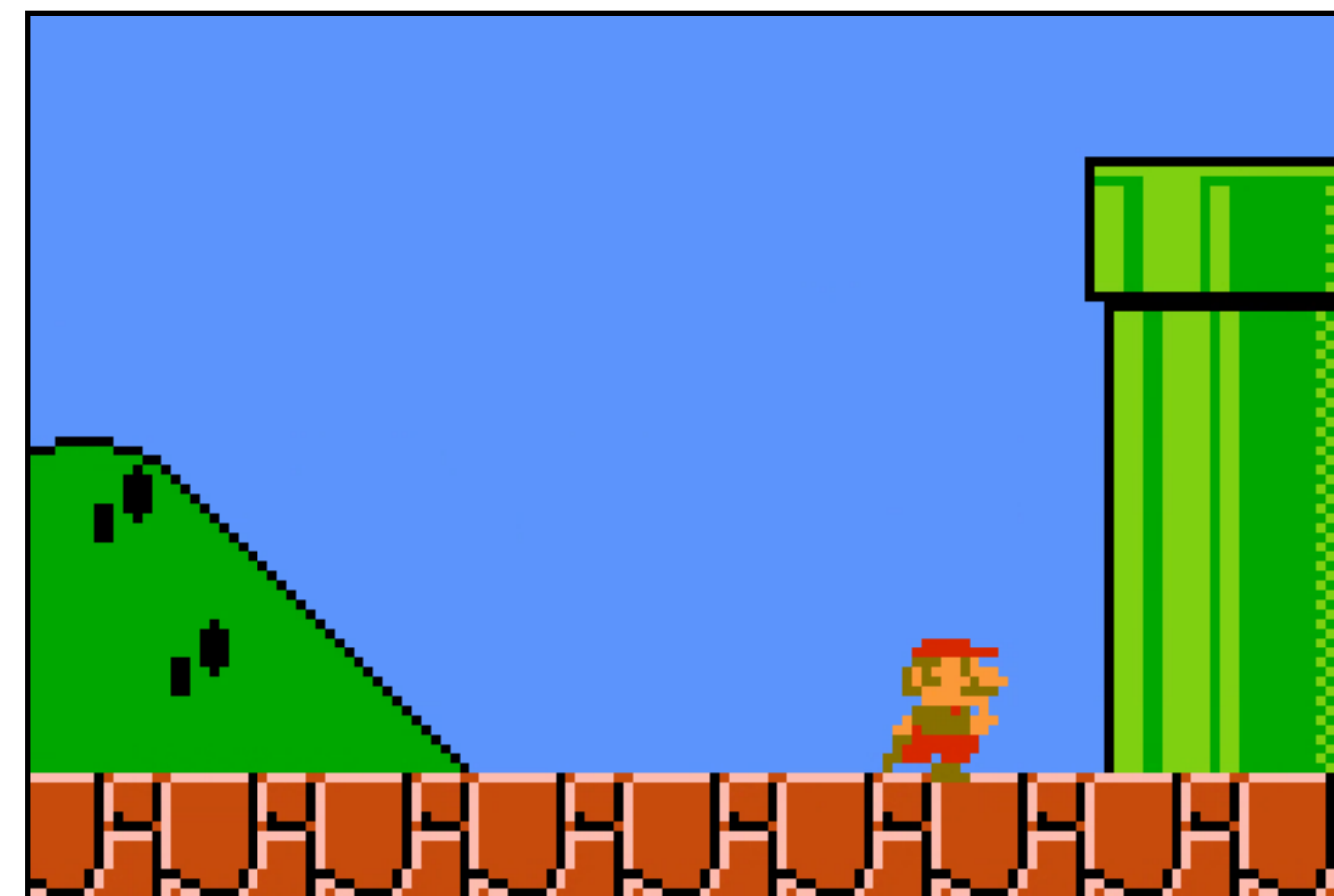
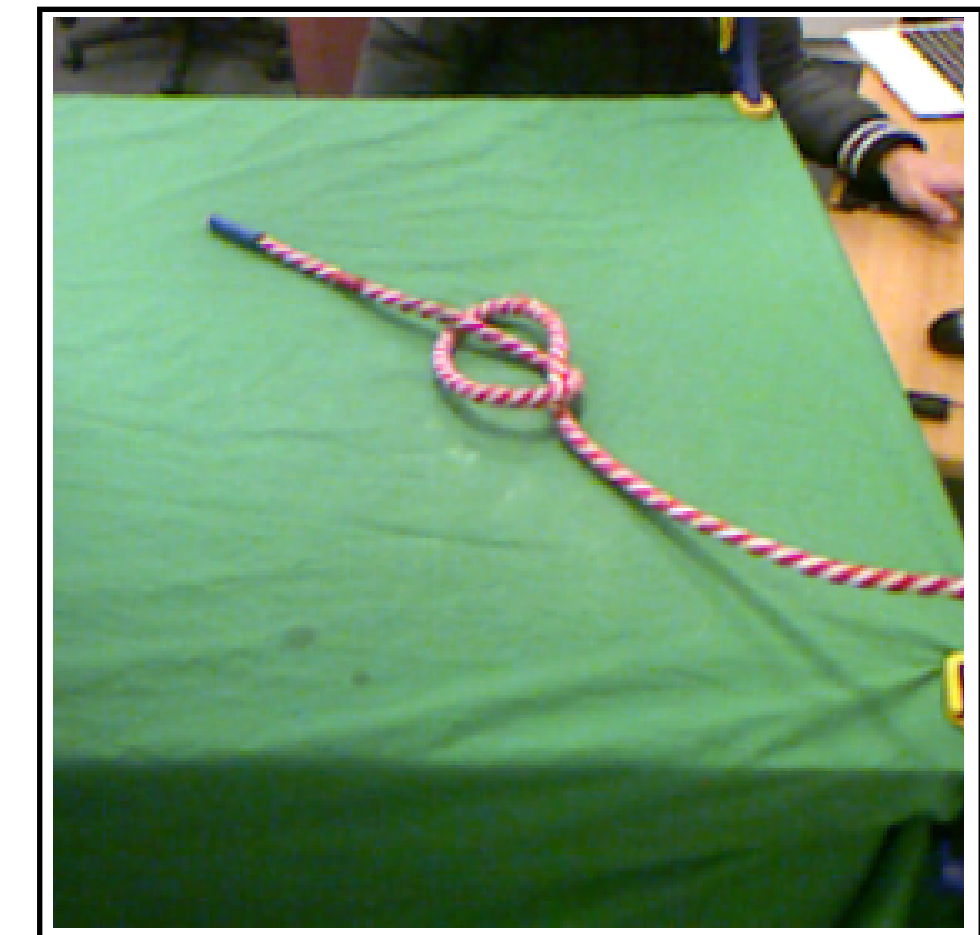
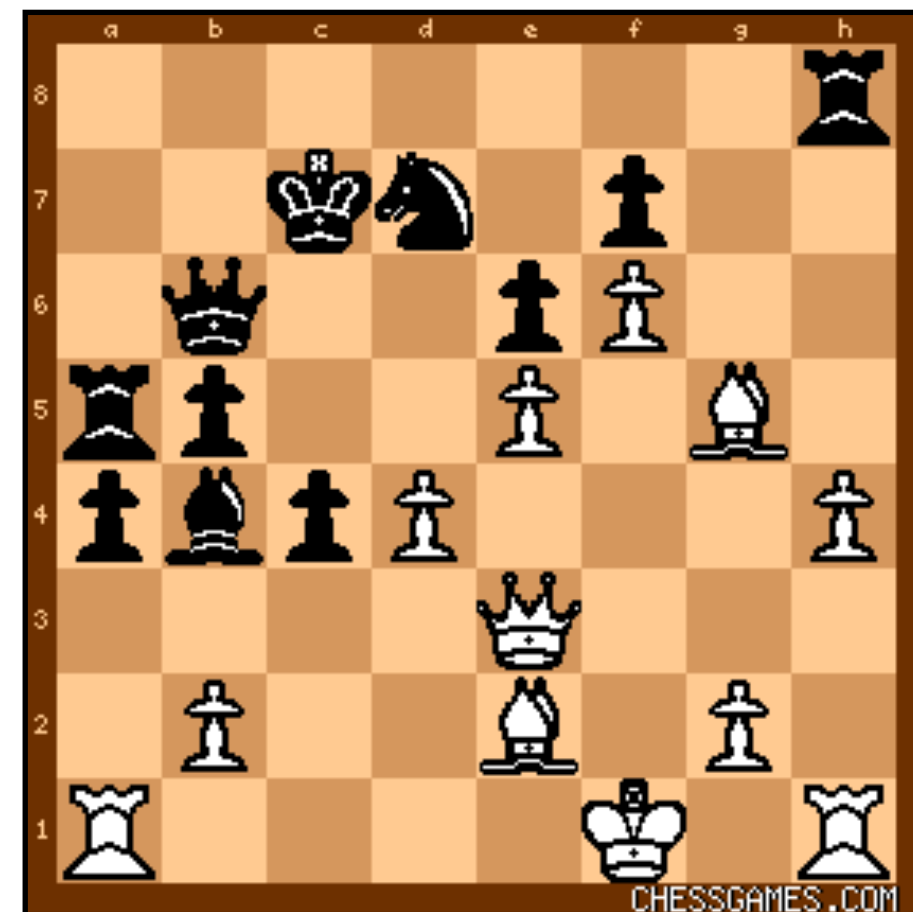
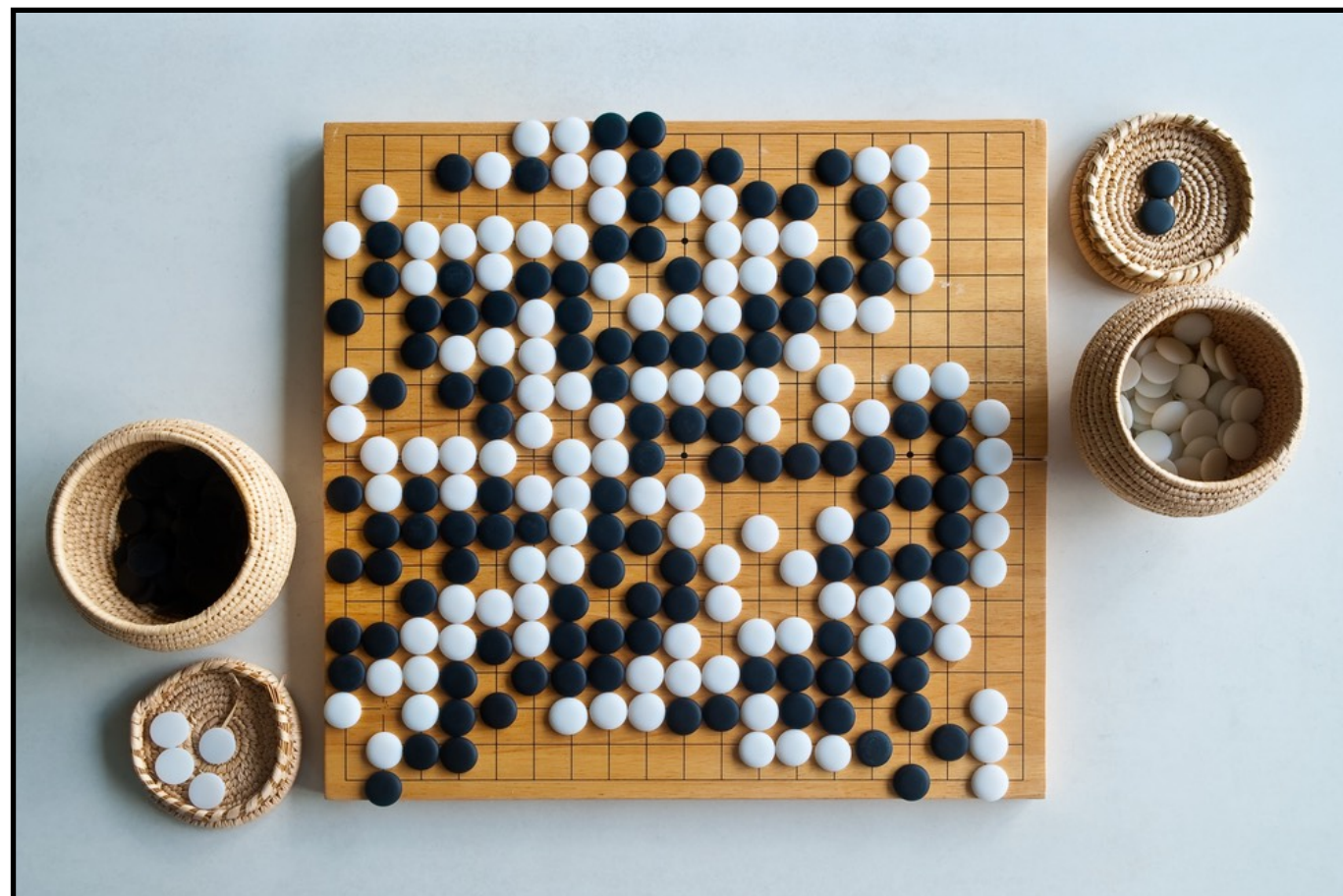


(credit: Johannes Burge)

We already know it works well!

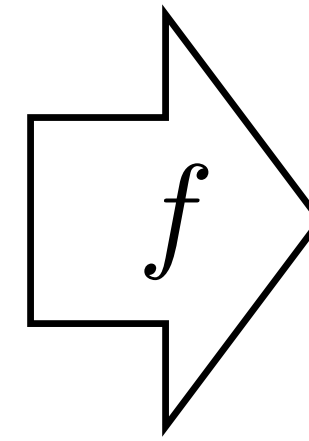
Why vision?

2. Universal interface



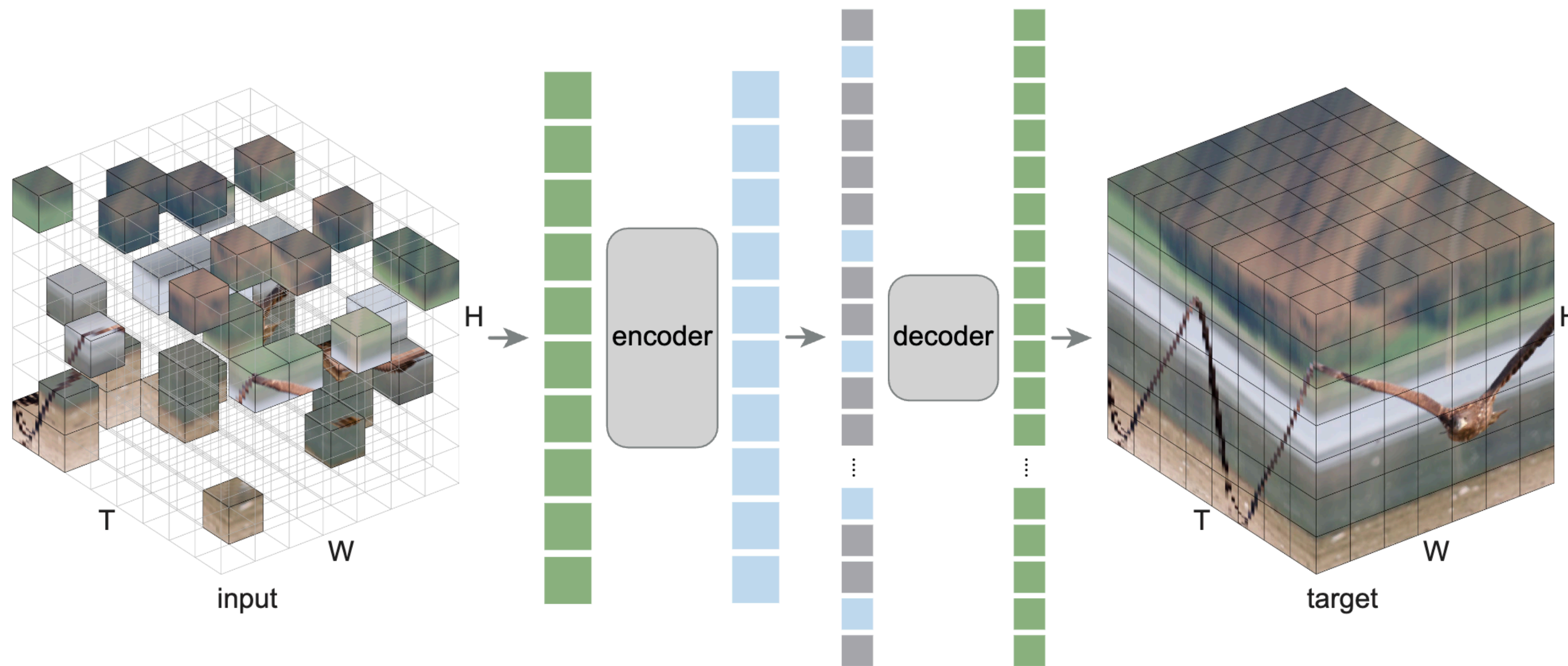
Why vision?

3. Structured representations



Why vision?

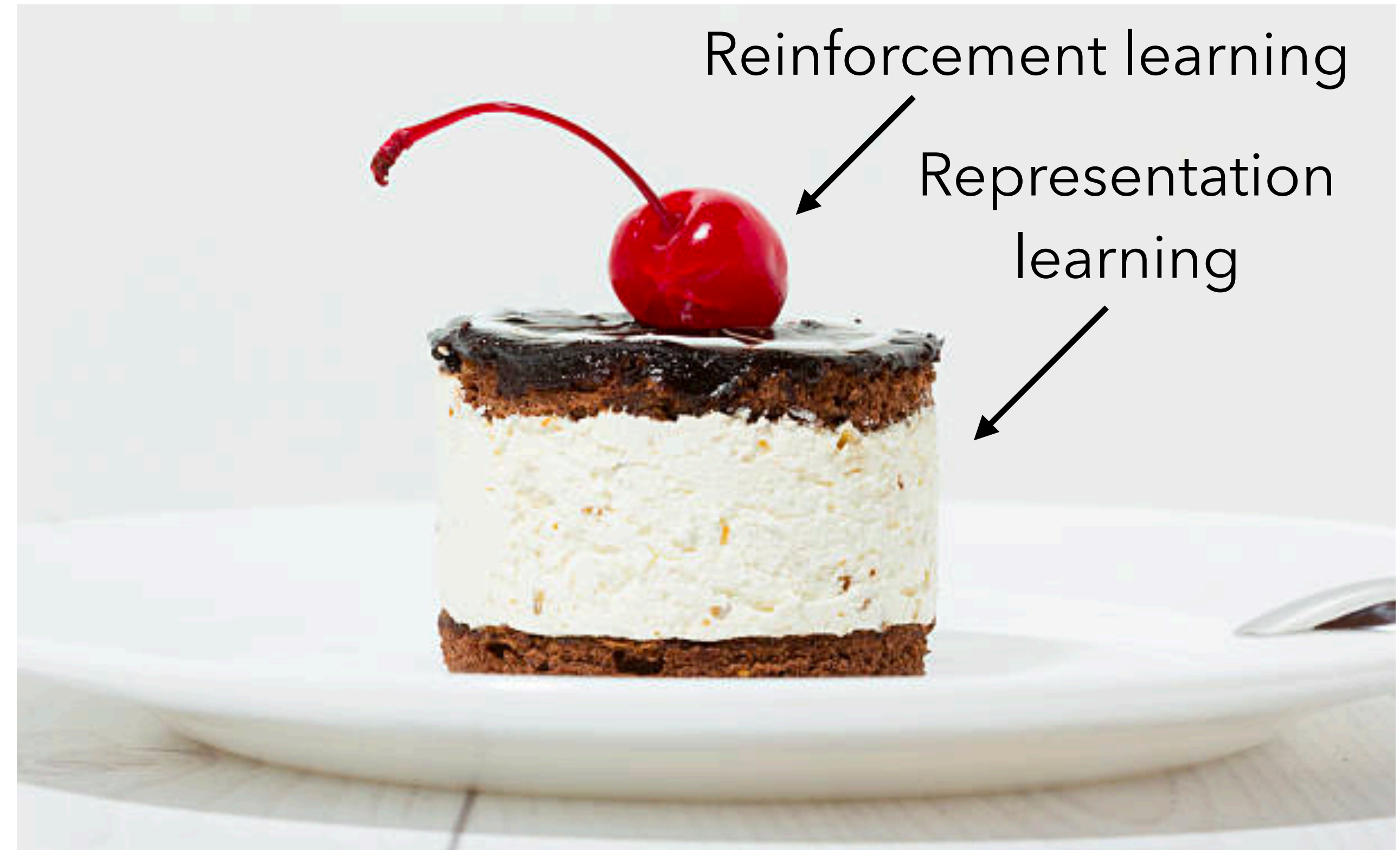
4. Lots of data (e.g., from the internet) for representation learning



[Feichtenhofer et al., "Masked Autoencoders As Spatiotemporal Learners", 2022]

Importance of perception

A hypothesis: if vision can give us a good representation/model of the world, then planning and control should be easy.



Yann LeCun's "cake"

Next class: detecting image manipulations