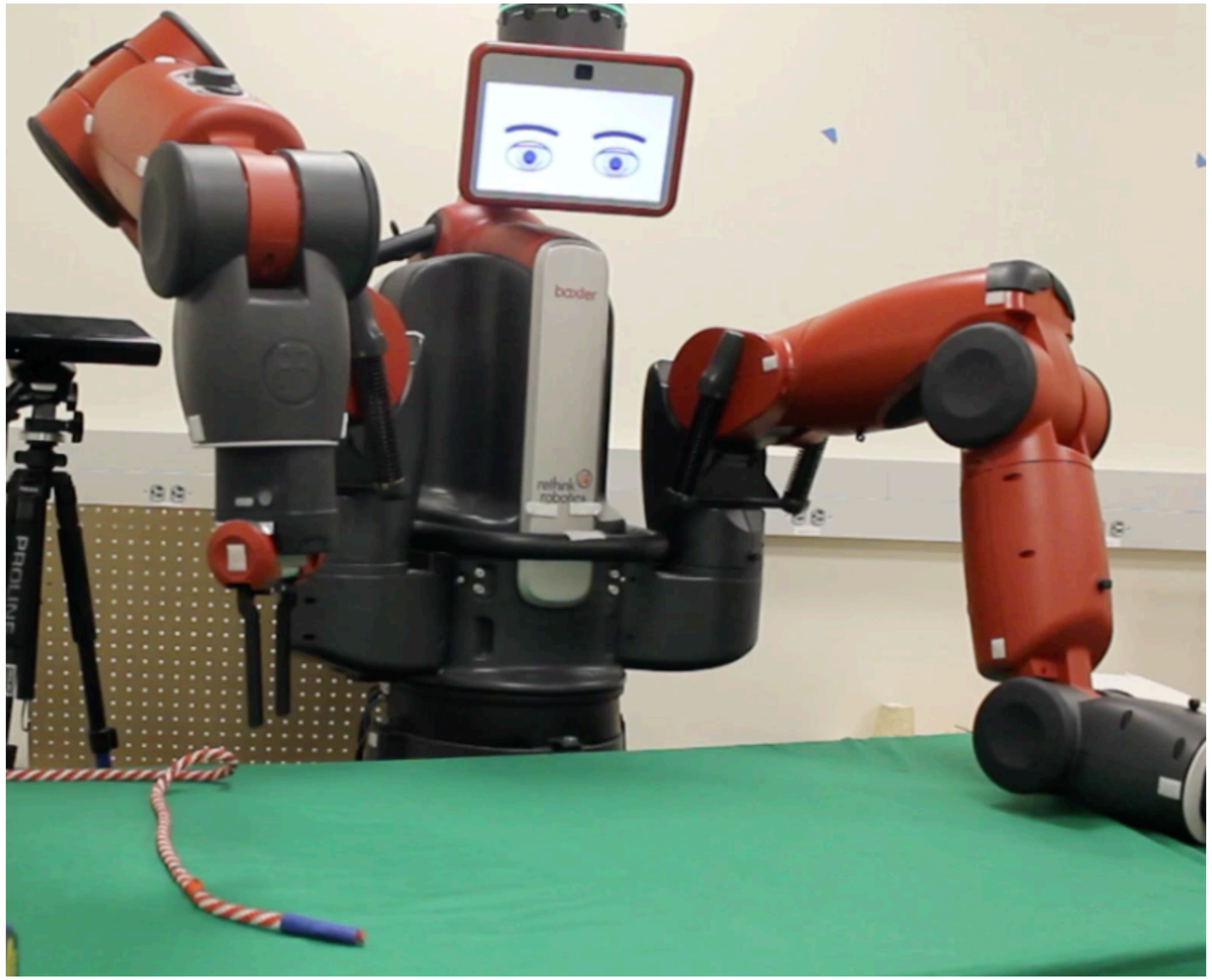


Vision for embodied agents

Bill Freeman, Antonio Torralba, Phillip Isola
6.819 / 6.869

Announcements

- Pset 7 due this Thursday
- Be working on final project!
  - Presentations 12/6 - 12/11 — Sign up online
  - Writeups due 12/12

OpenAI Spinning Up

[https://spinningup.openai.com/en/latest/spinningup/rl_intro.html]

- Intro to reinforcement learning (RL)
- Code
- Curated list of papers
- Advice on how to do research in RL

[Another good blog: http://karpathy.github.io/2016/05/31/rl/]

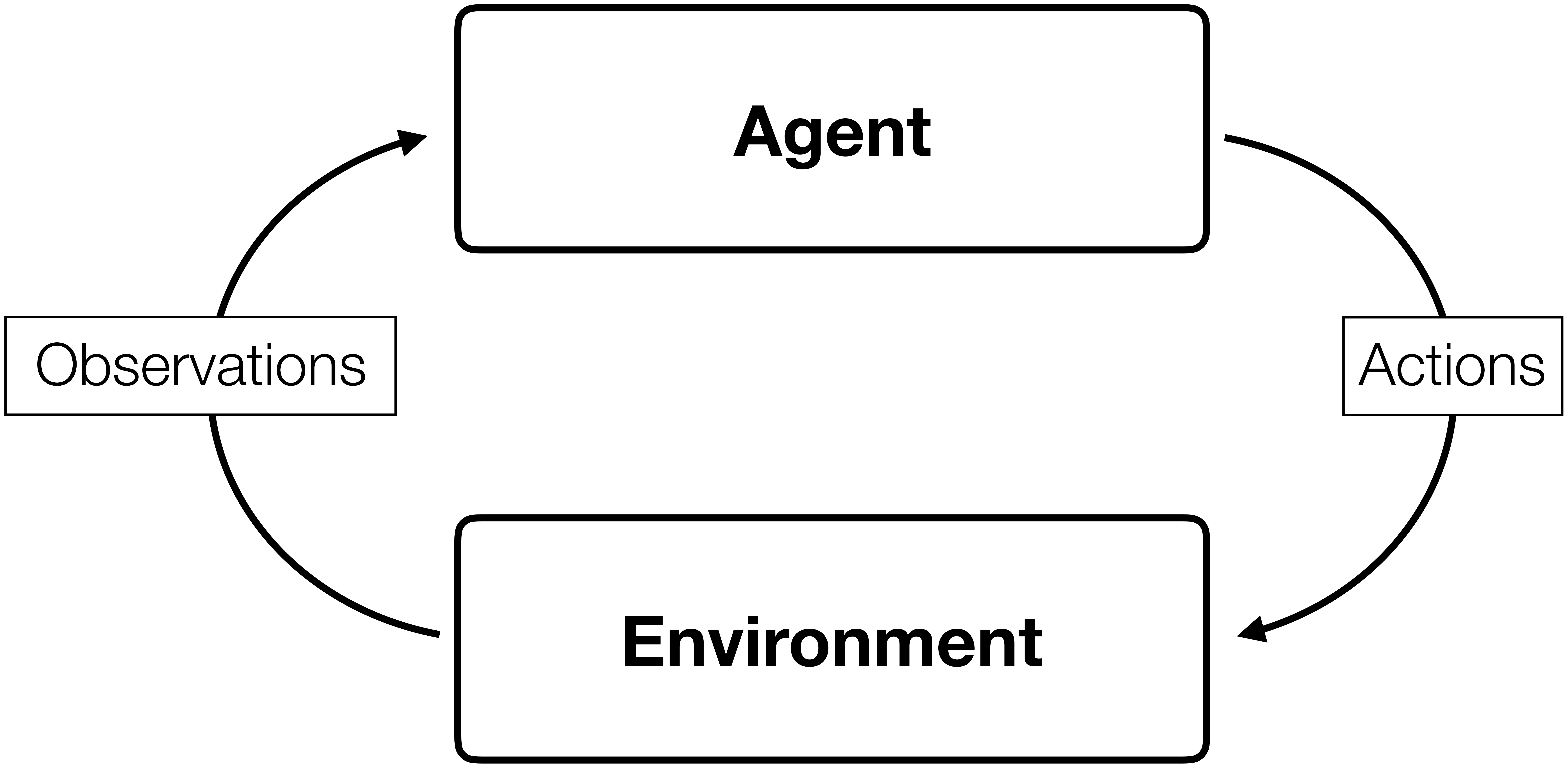
[Silver et al., 2016]

[Jaderberg et al. 2018]

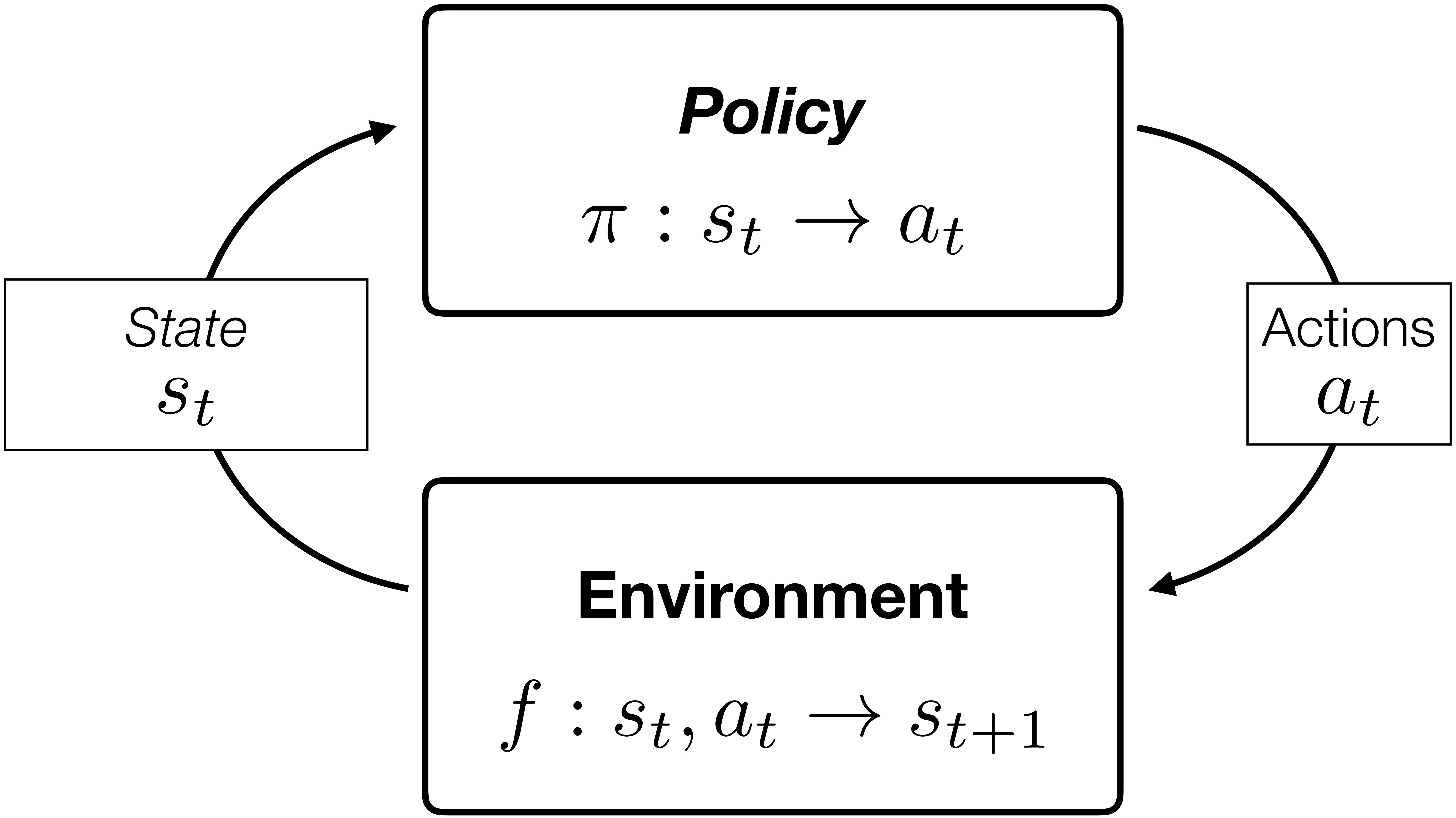
The whole purpose of visual perception, in humans, is to make good motor decisions.

We are **sensorimotor** systems.

Intelligent agents



Intelligent agents



Recipe for deep learning in a new domain

1. Transform your data into numbers (e.g., a vector)

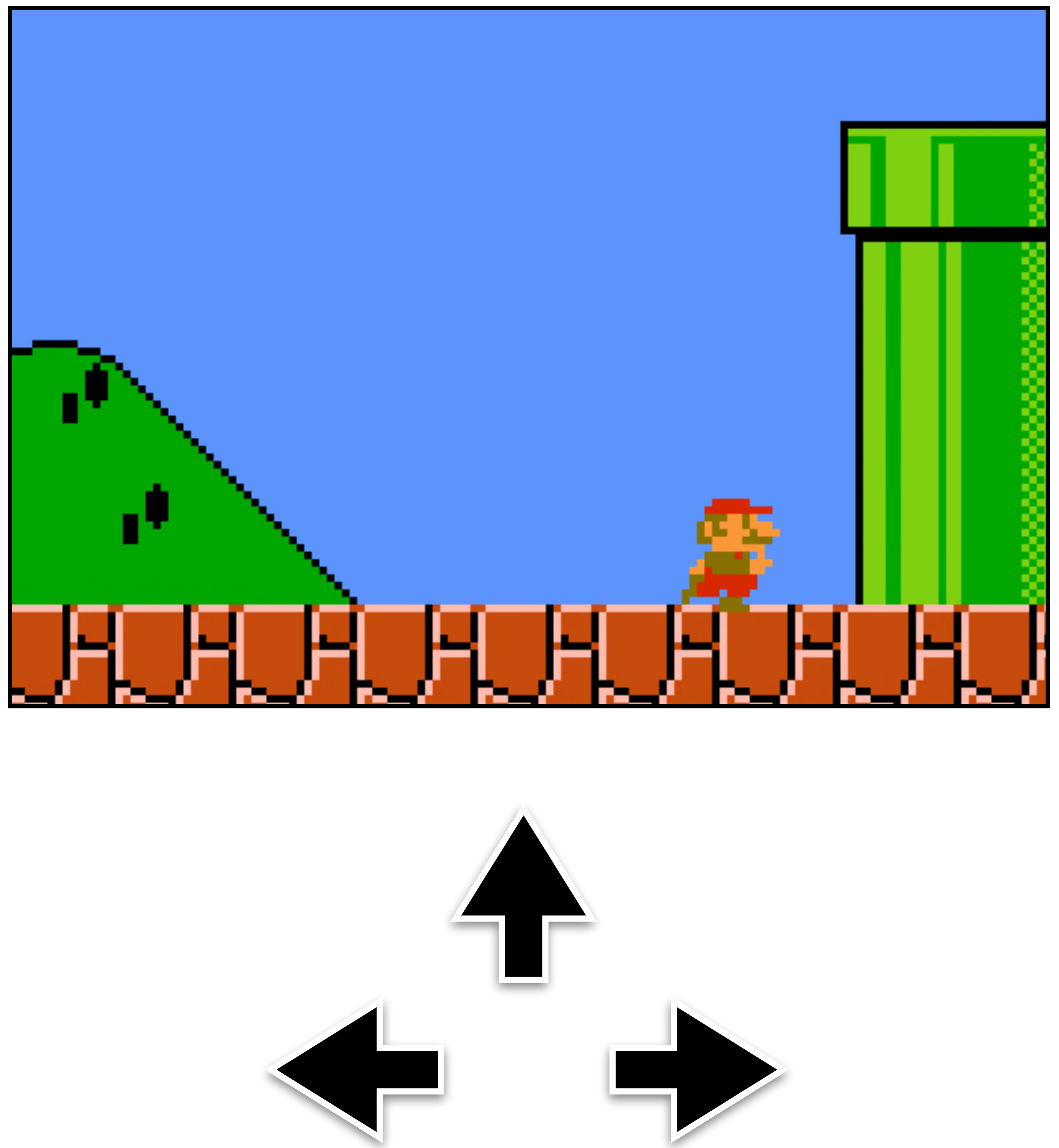
2. Transform your goal into an equation (objective function)

3. #1 and #2 specify the “learning problem”

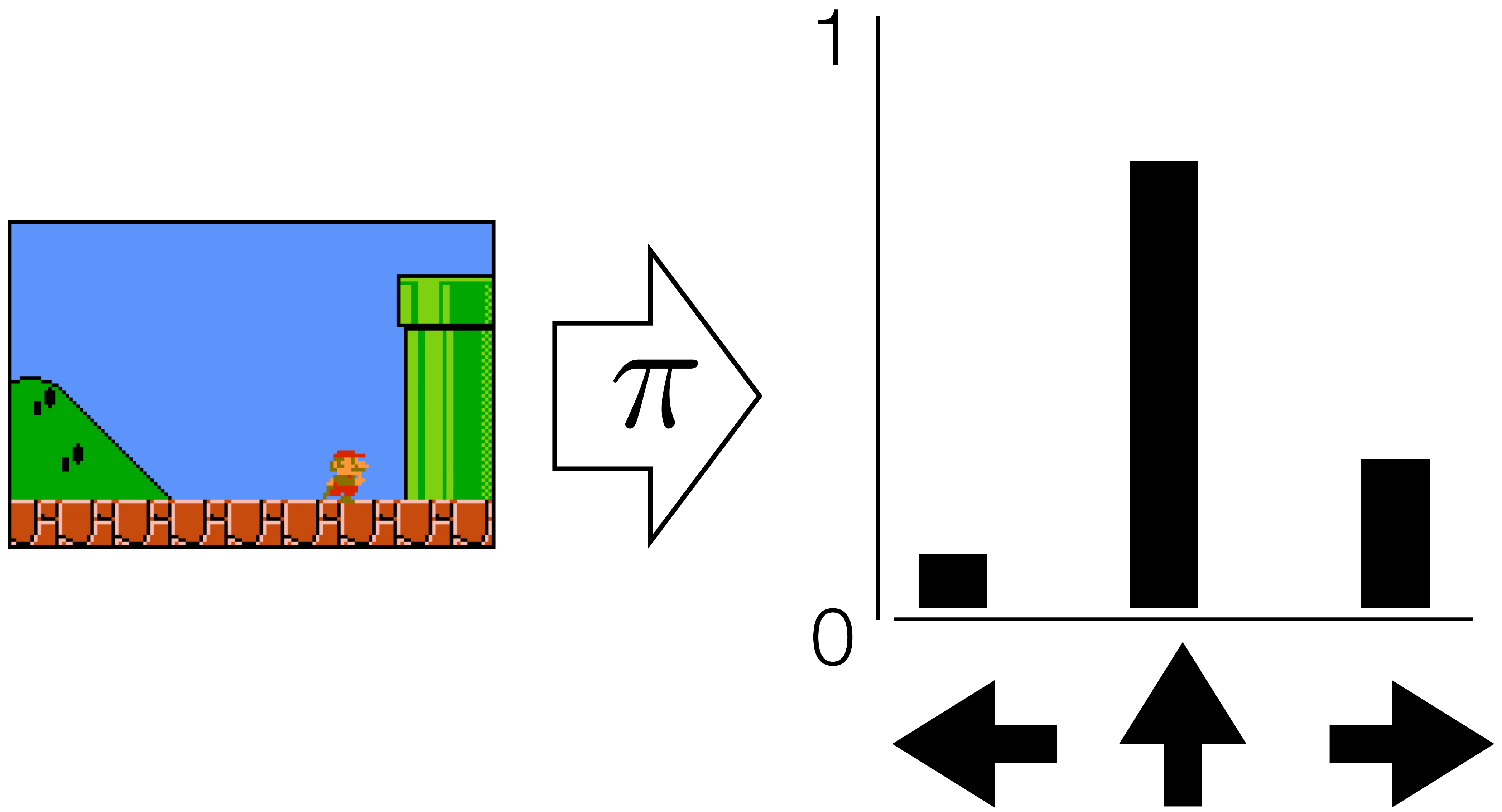
4. Use a generic optimizer (SGD) and an appropriate architecture (e.g., CNN or RNN) to solve the learning problem

How to represent a state? How to represent policy?

state: pixels!



policy: action classifier



Learning from examples

(aka **supervised learning**)

Training data

$$\begin{matrix} \{x_1, y_1\} \\ \{x_2, y_2\} \\ \{x_3, y_3\} \\ \dots \end{matrix} \quad \rightarrow \quad \boxed{\text{Learner}} \quad \rightarrow \quad f : X \rightarrow Y$$

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

Imitation learning

(still just **supervised learning**, applied to learn *policies*)

Training data

$$\{s_1, a_1\}$$
$$\{s_2, a_2\} \quad \rightarrow \quad \boxed{\text{Learner}} \quad \rightarrow \quad \pi : s \rightarrow a$$
$$\{s_3, a_3\}$$
$$\ldots$$

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

Imitation learning

s a

$\Big\{$  , d3 ♝b4+ $\Big\}$

$\Big\{$  , ♘f3 ♞c6 $\Big\}$

$\Big\{$  , 0-0 ♝xc3 $\Big\}$

$\vdots$

$\rightarrow$

Learner

Objective

$$\pi(s) = \texttt{softmax}(g_\theta(s))$$

$$\mathcal{L}(a, \pi(s)) = H(a, \pi(s))$$

Hypothesis space

Convolutional neural net

Optimizer

Stochastic gradient descent

$\rightarrow$ π

Learning without examples

(includes **unsupervised learning** and **reinforcement learning**)

Data

$$\{x_1\}$$
$$\{x_2\} \quad \rightarrow \quad \boxed{\text{Learner}} \quad \rightarrow \quad ?$$
$$\{x_3\}$$
$$\cdots$$

Unsupervised Representation Learning

Data

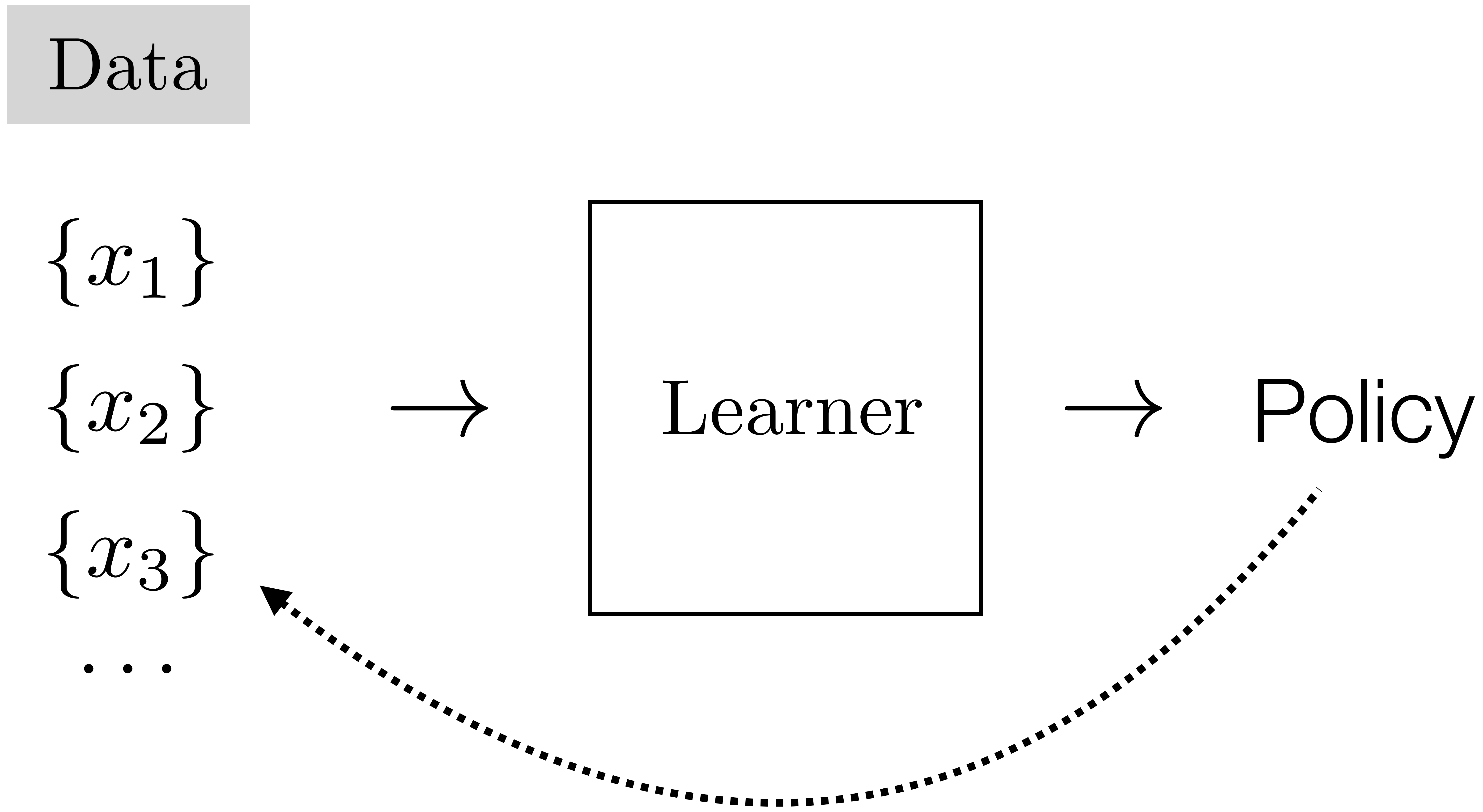
$\{x_1\}$

$\{x_2\}$ $\rightarrow$ Learner $\rightarrow$ Representations

$\{x_3\}$

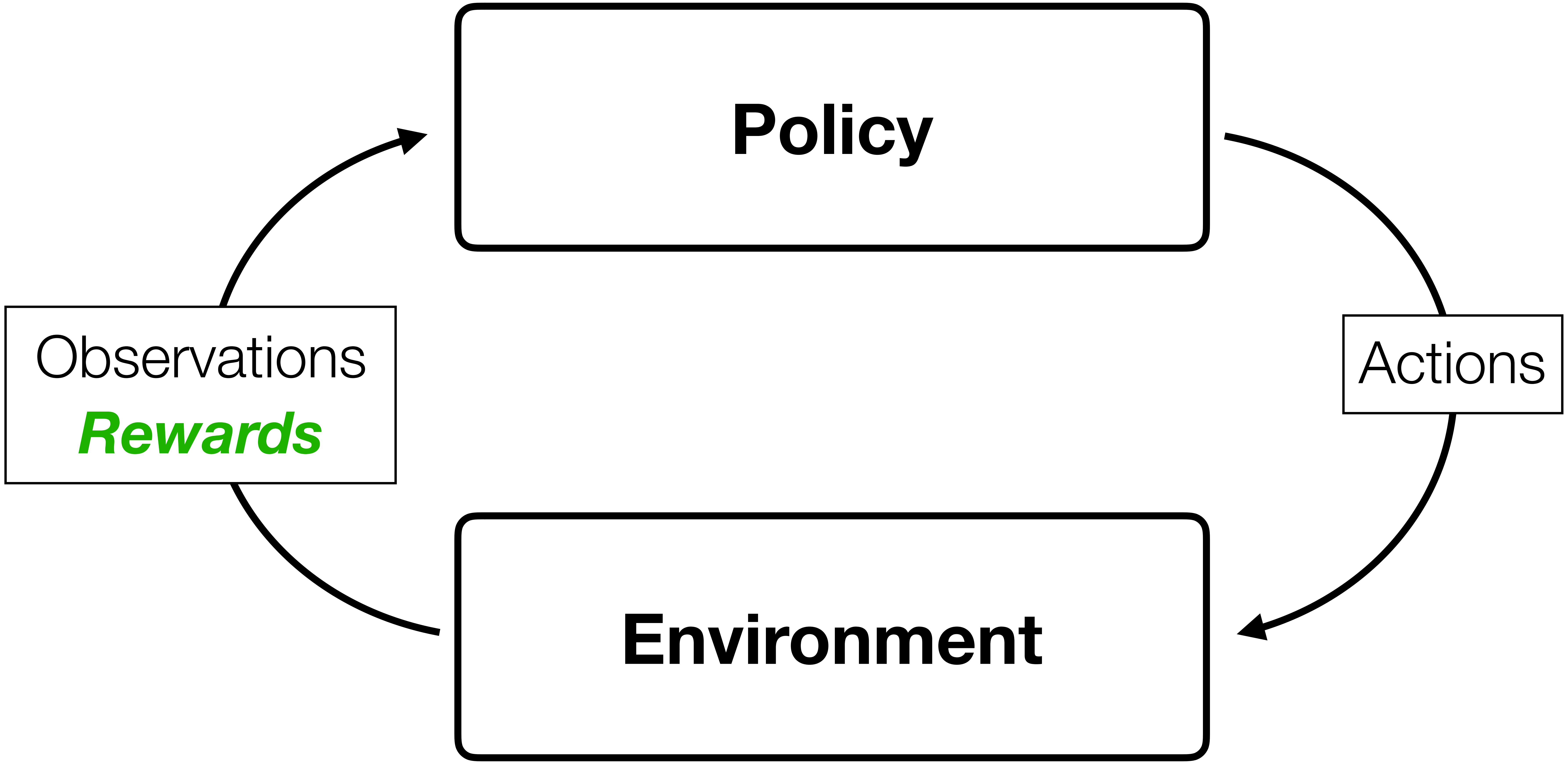
$\dots$

Reinforcement learning



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

Reinforcement learning

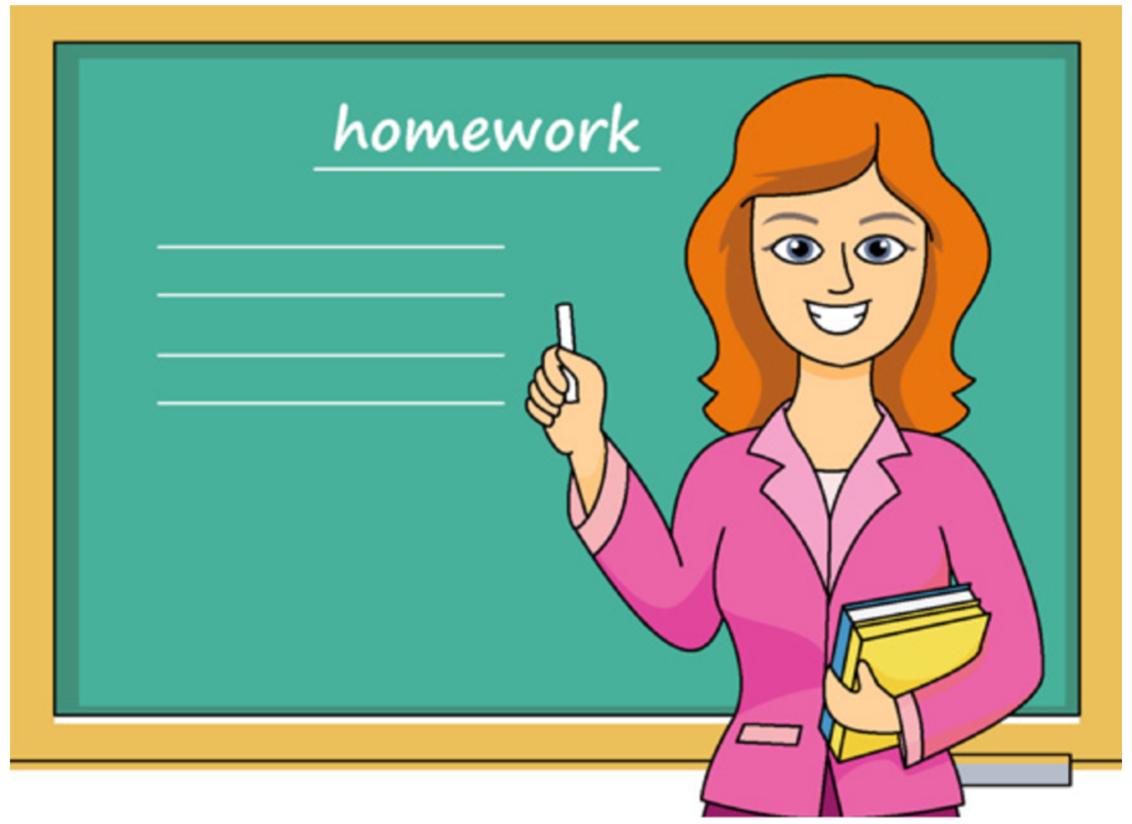


Learn a policy that takes actions that maximize **reward**

Imitation learning

<u>Hand-curated</u> training data

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

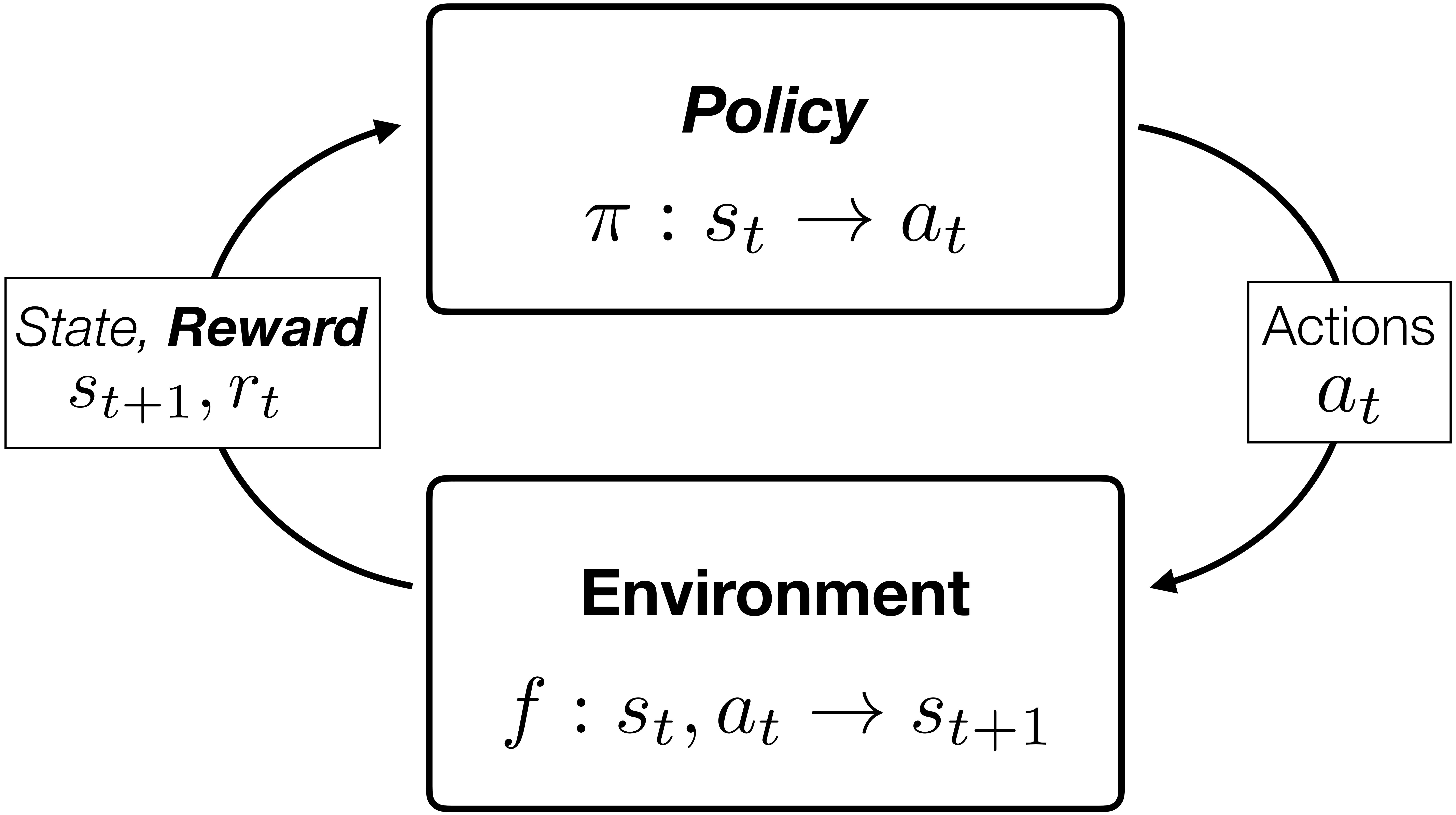


Reinforcement learning

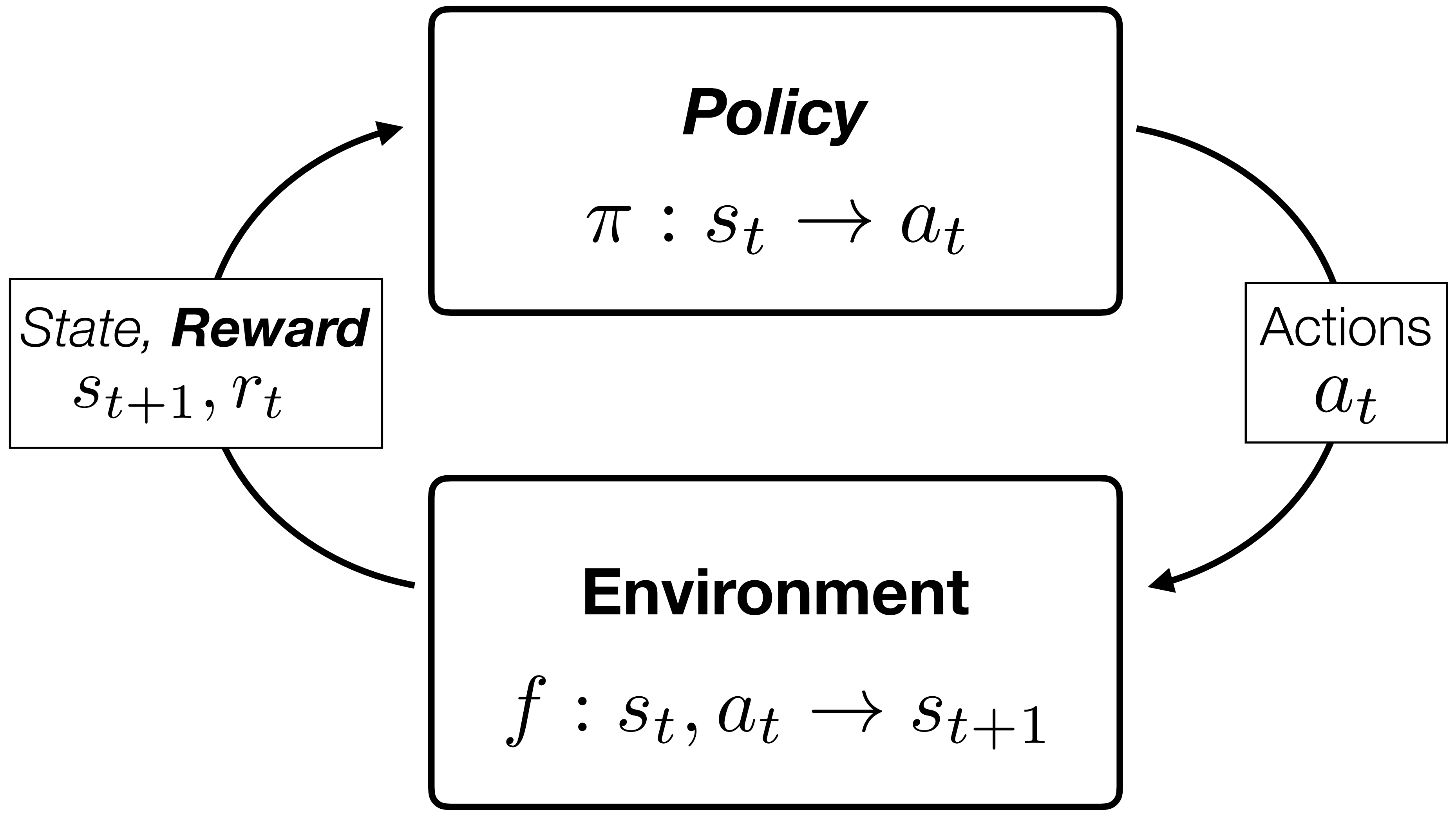
<u>No</u> training data, have to play around and collect the data *yourself*

+ 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



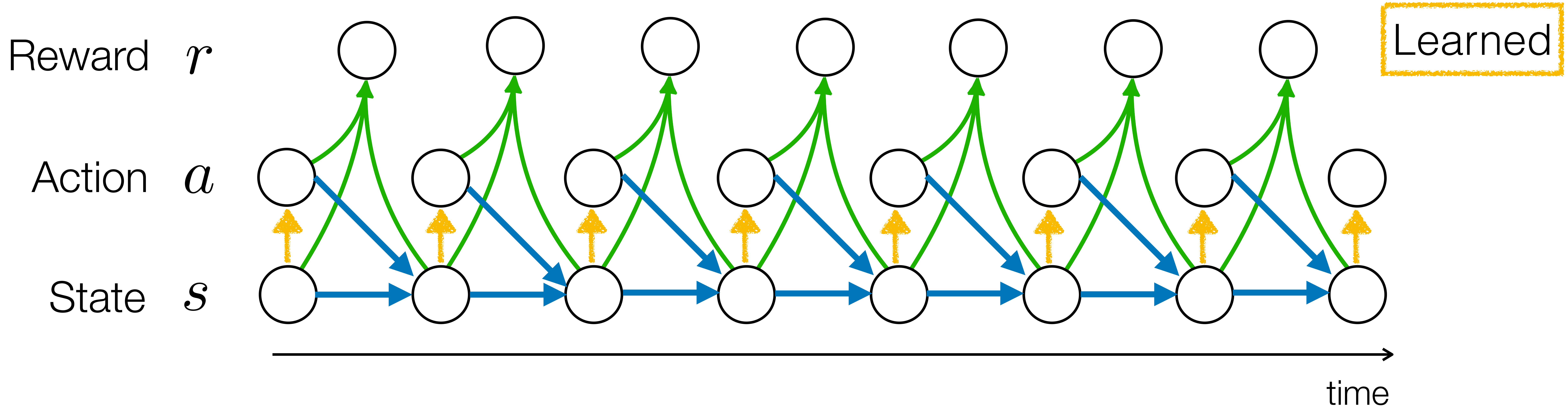
Trajectory $\tau = (s_0, a_0, r_0, s_1, a_1, r_1, \ldots)$

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

Learn a policy that takes actions that maximize reward

$$\pi^* = \arg\max_{\pi} R(\tau)$$

Reinforcement learning



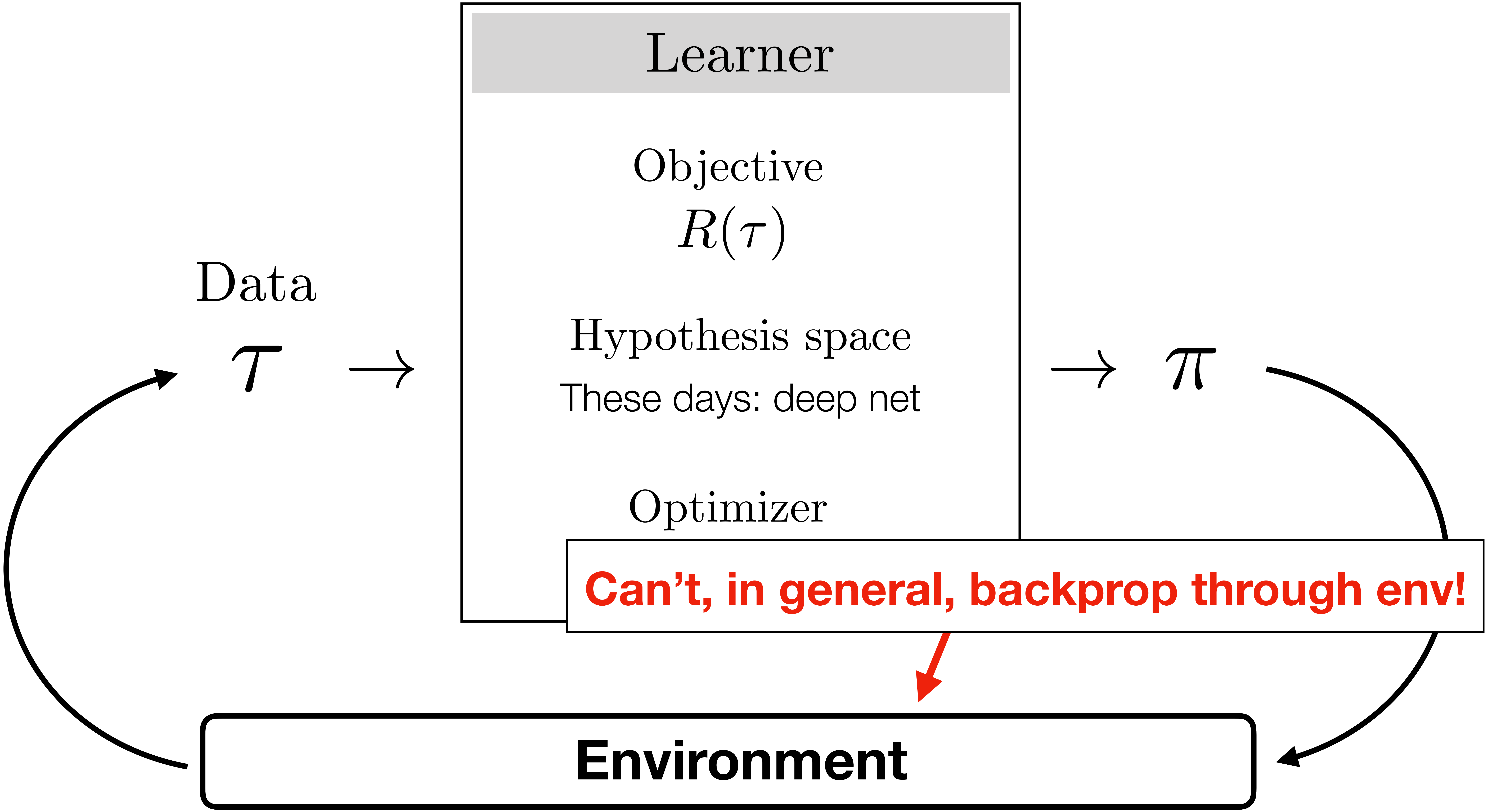
Policy

Environment (State transition function)

Reward

This is called a **Markov decision process (MDP)**

Reinforcement learning

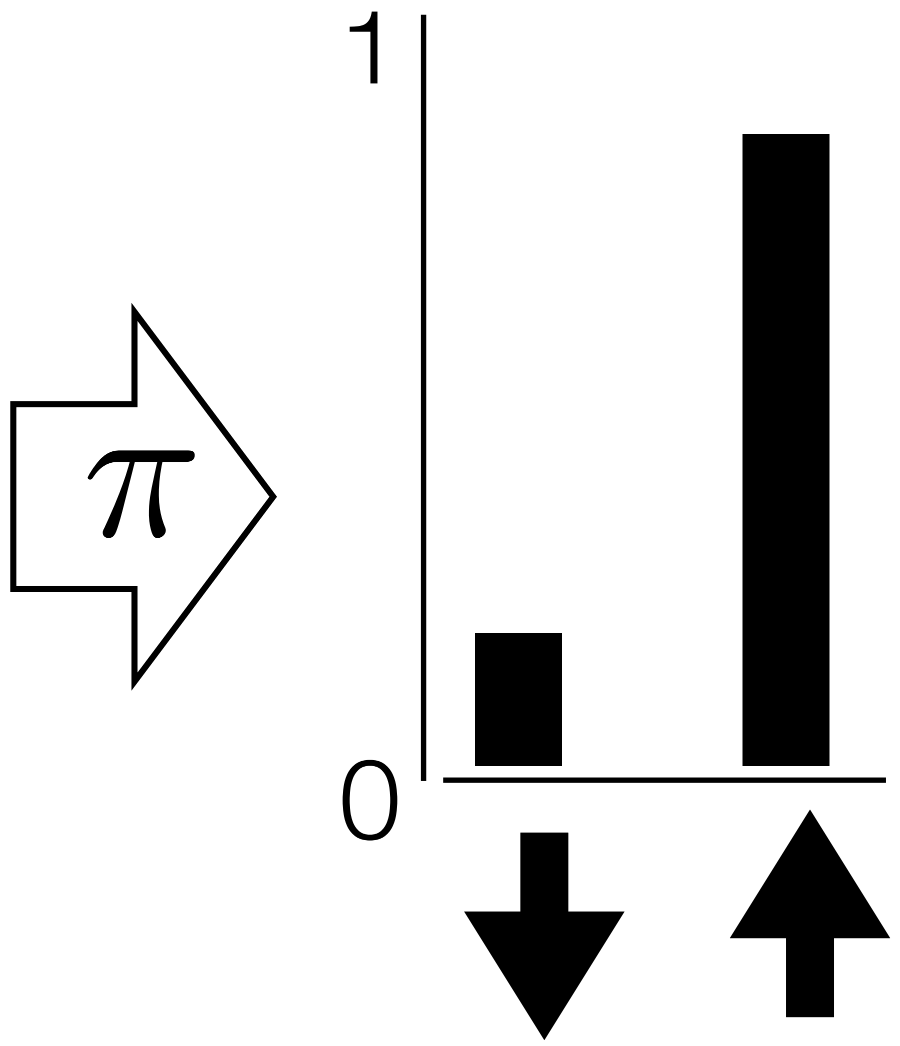


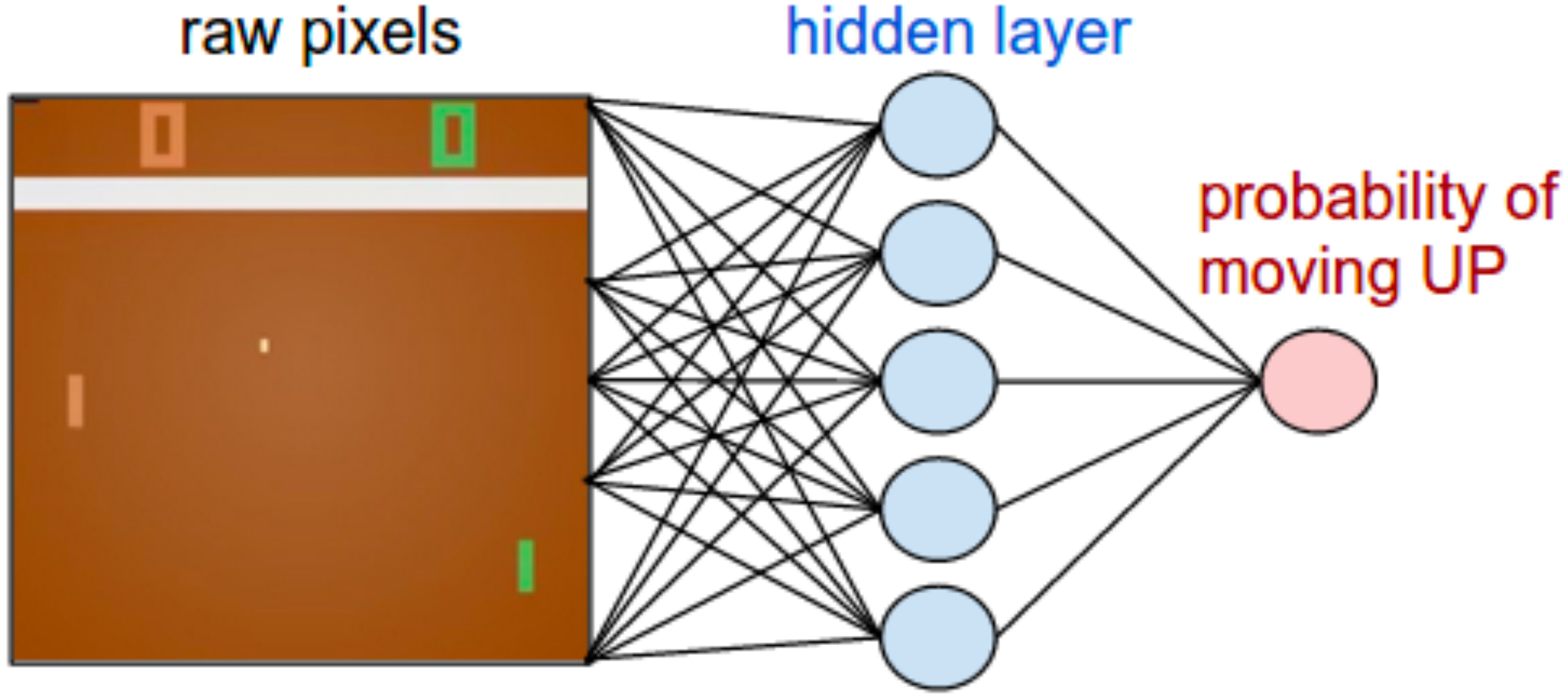
Environment is not differentiable! — How to optimize?

Idea #1 (trial and error):

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

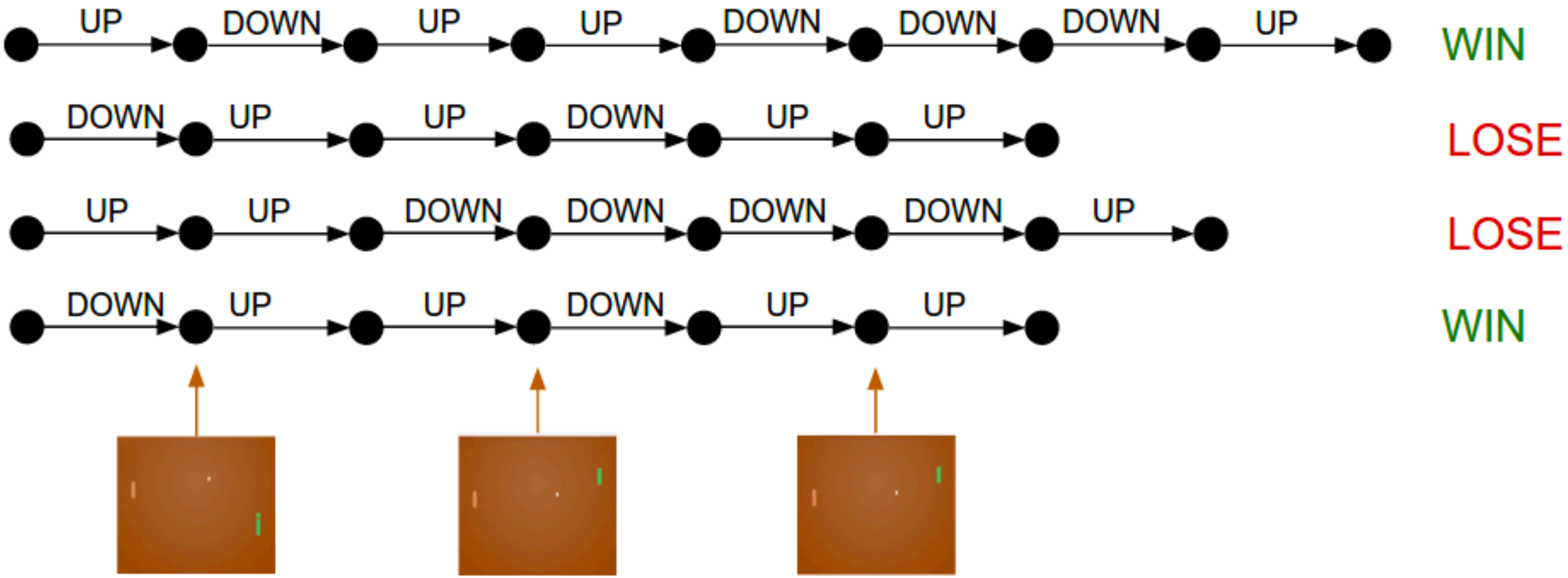
Pong



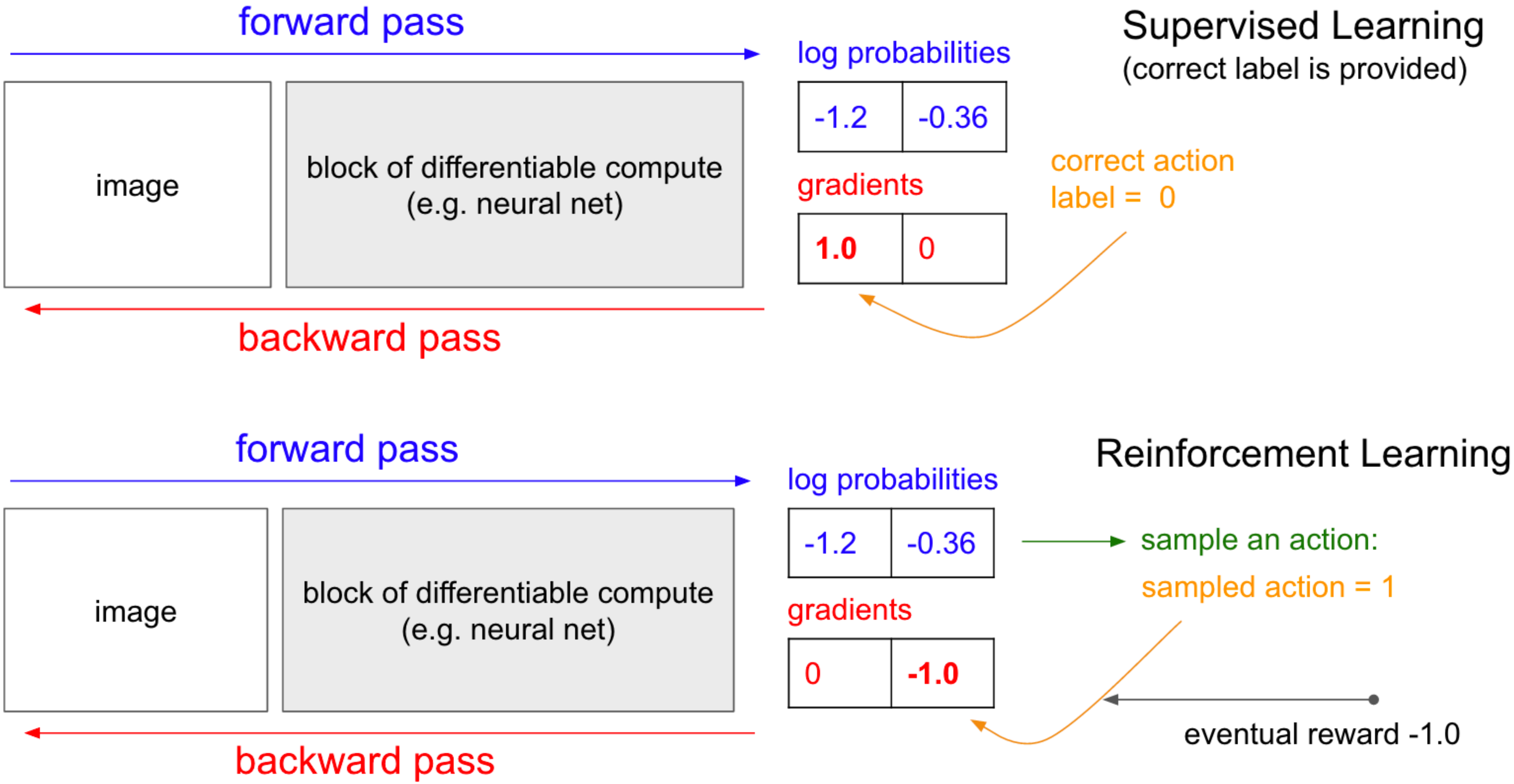


[Adapted from Andrej Karpathy: http://karpathy.github.io/2016/05/31/rl/]

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



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



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

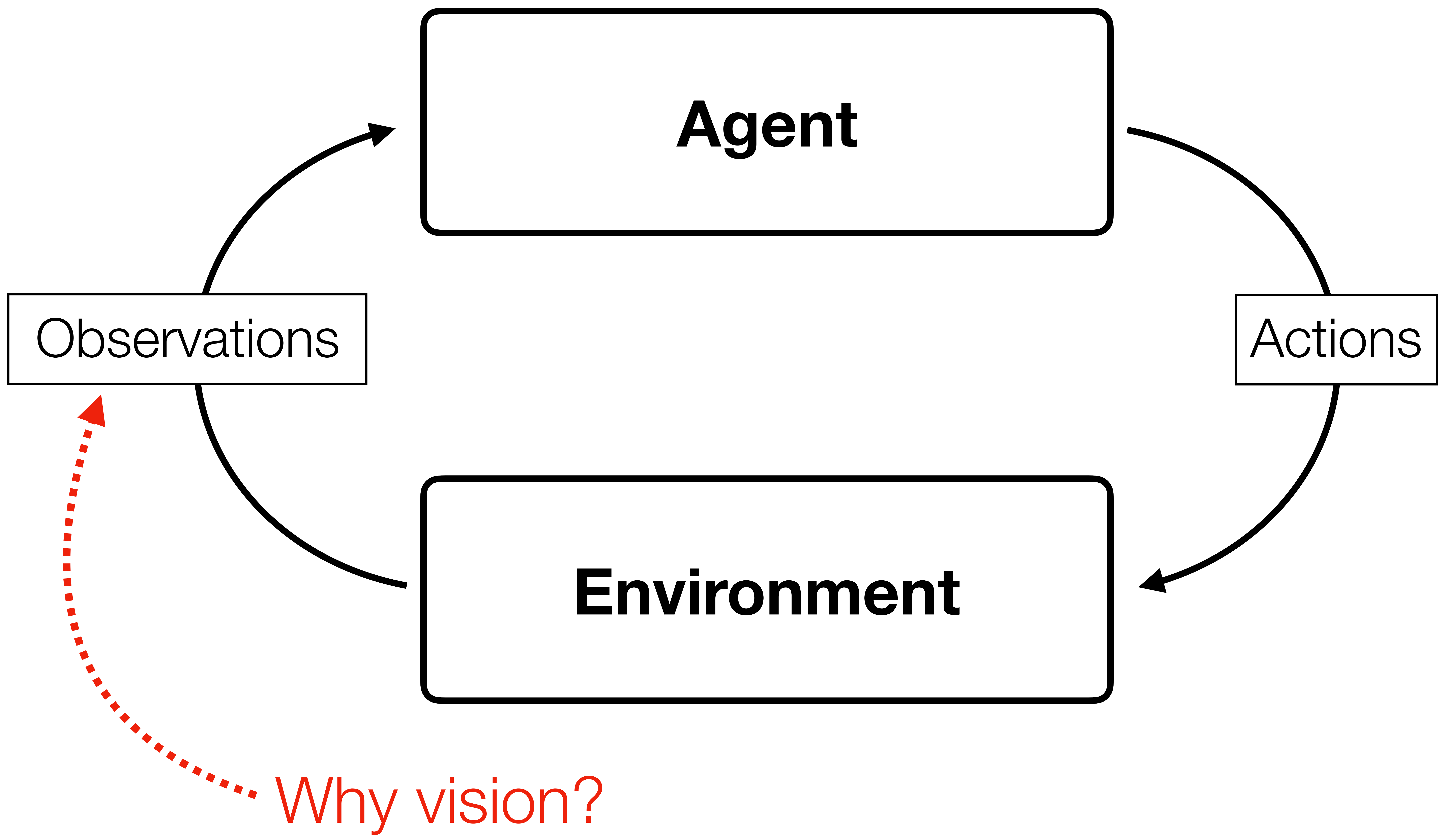
Environment is not differentiable! — How to optimize?

Policy gradients

1. Start with an arbitrary initial policy
2. **Rollout** this *stochastic* policy a bunch of 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.

Intelligent agents



Why vision?

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

(credit: Johannes Burge)

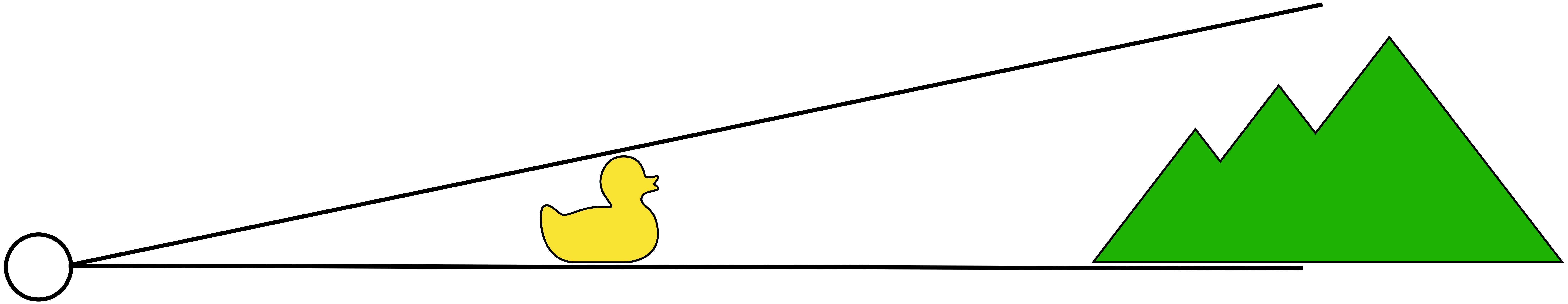
>30% of the human cortex?

http://www.kyb.tuebingen.mpg.de/research/dep/lo/visual-perception.html

[See *Animal Eyes* by Michael Land and Dan Nilsson]

Why vision?

2. Eyes are good sensors



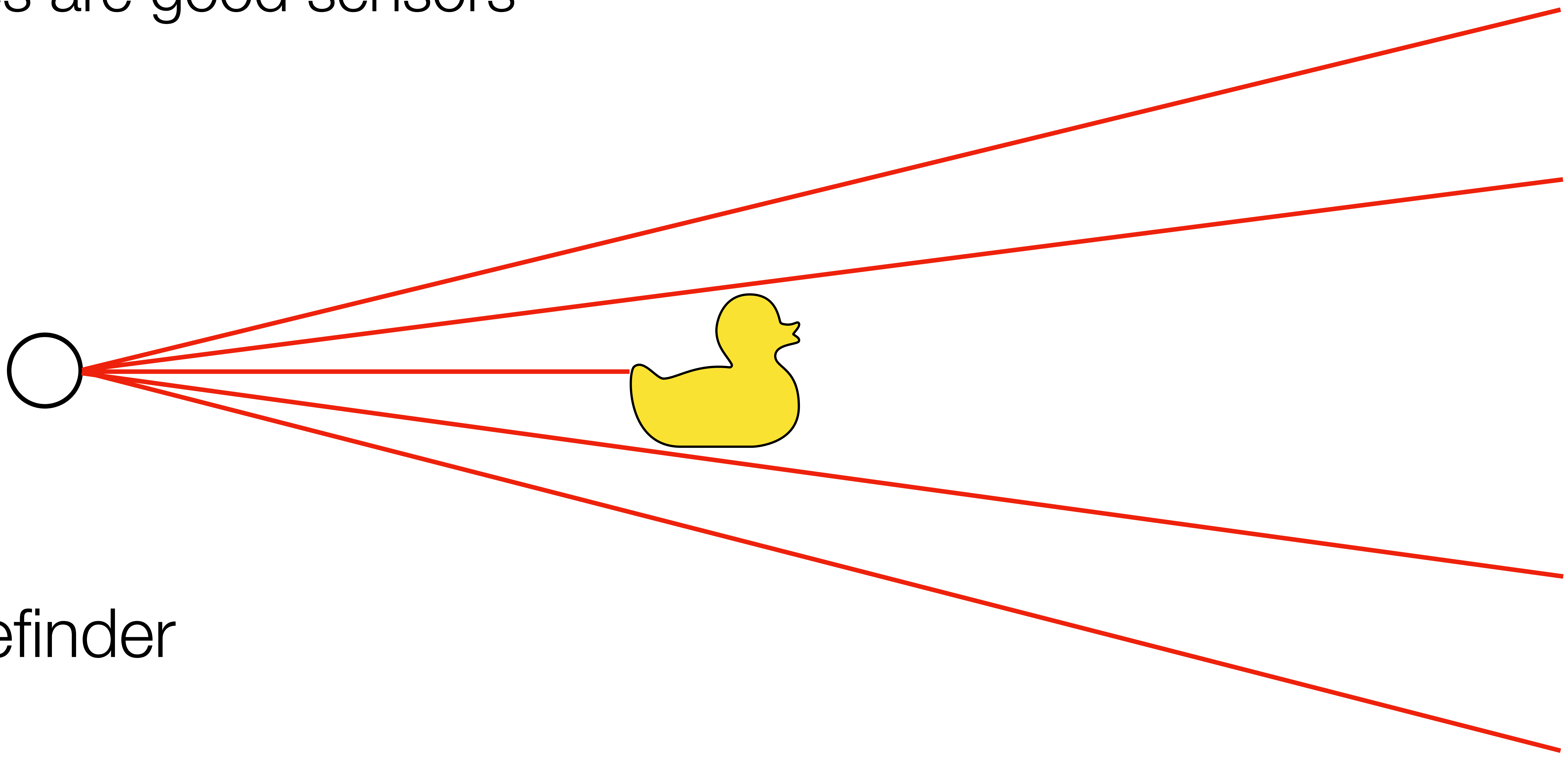
Farther away things look smaller

Get details on stuff that we can immediately interact with,
rough summary of more distant context

Why vision?

2. Eyes are good sensors

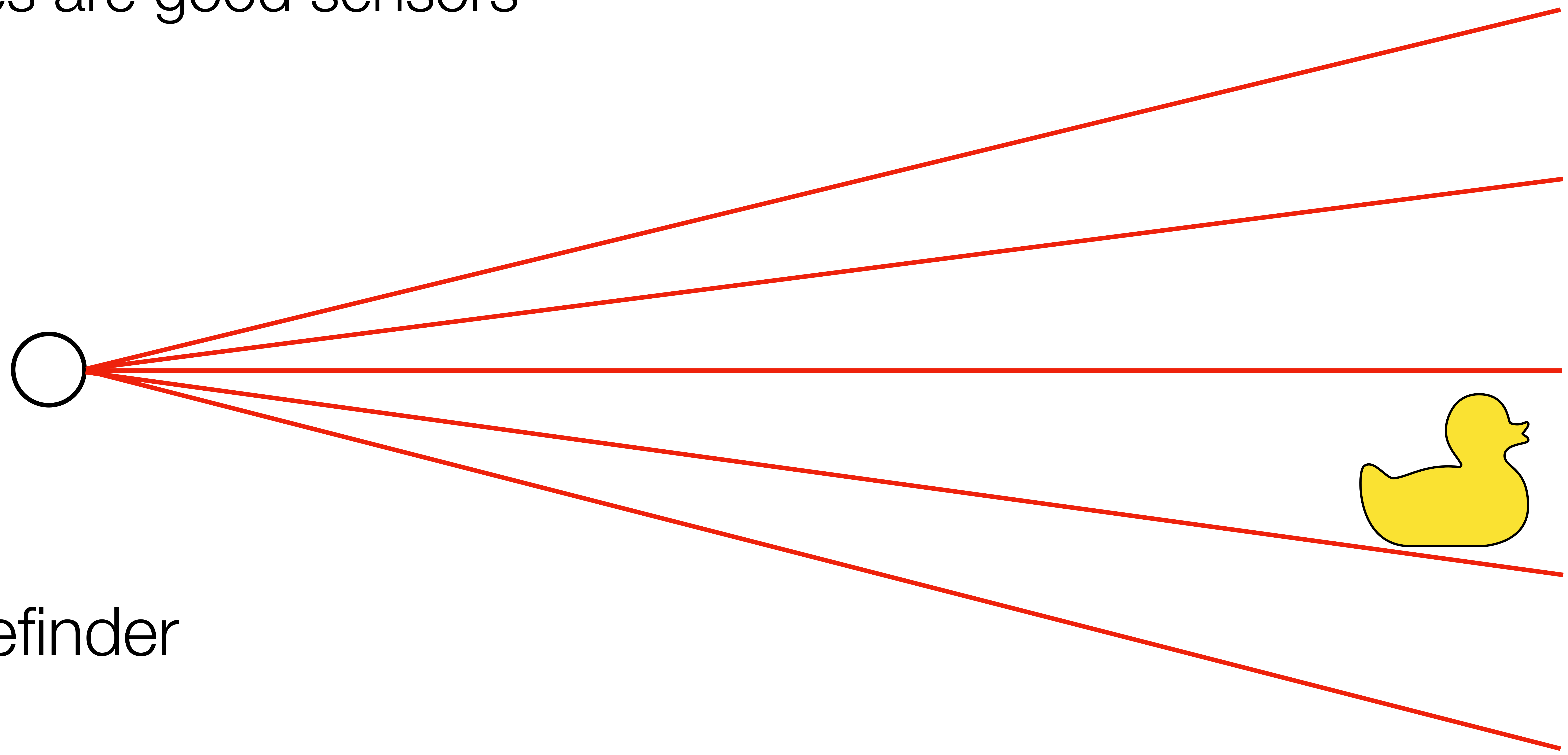
Laser rangefinder



Why vision?

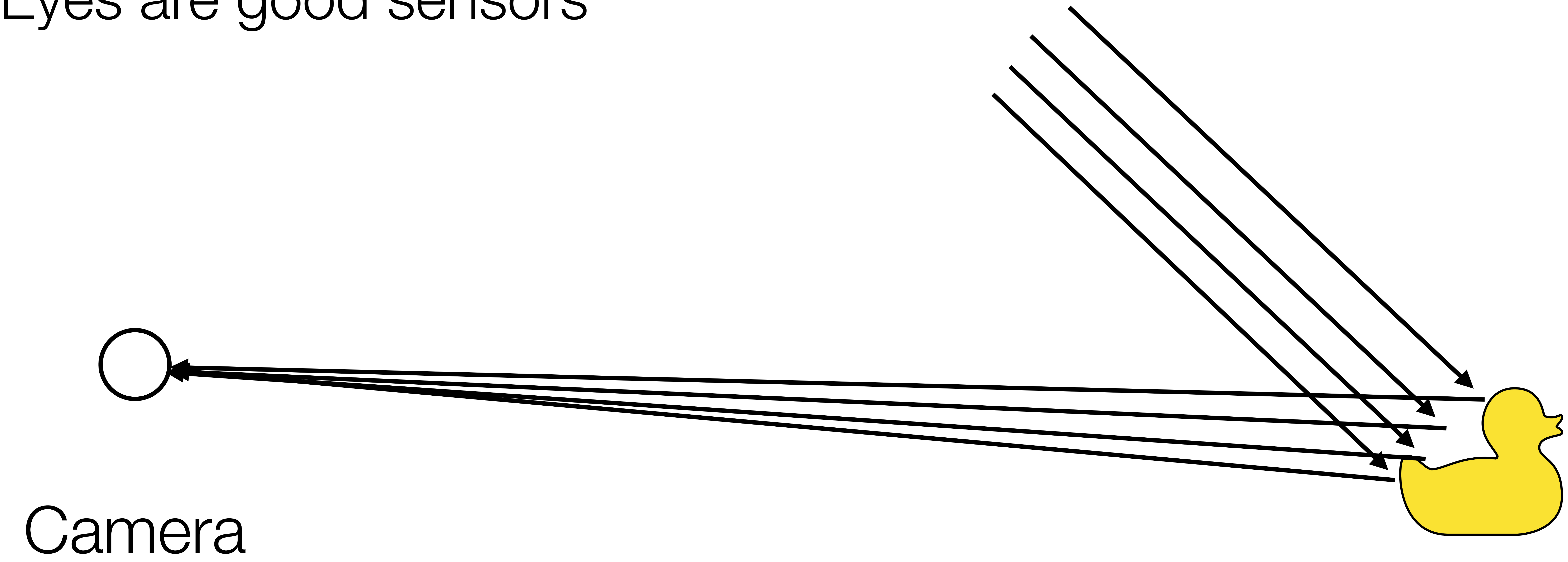
2. Eyes are good sensors

Laser rangefinder



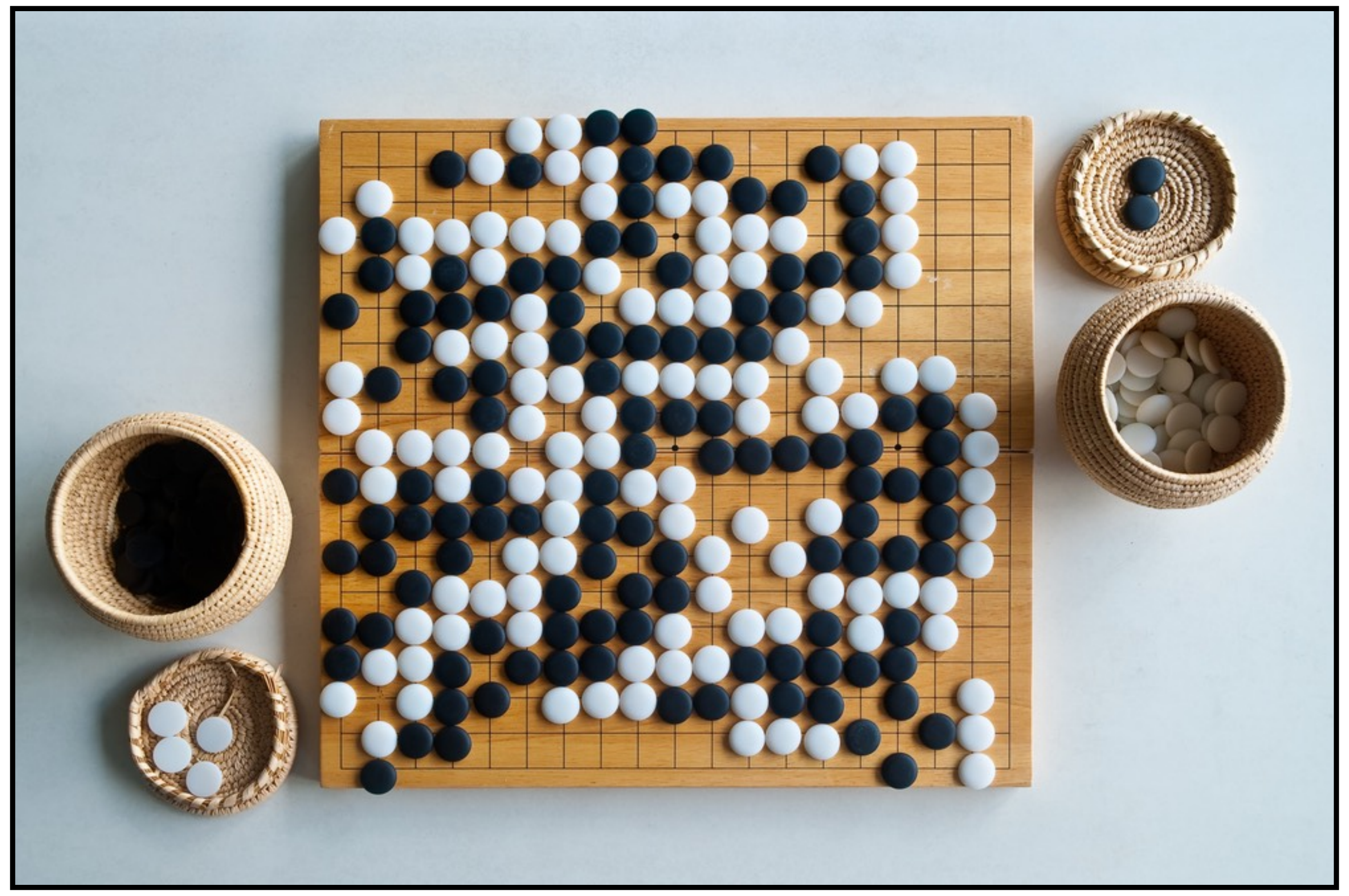
Why vision?

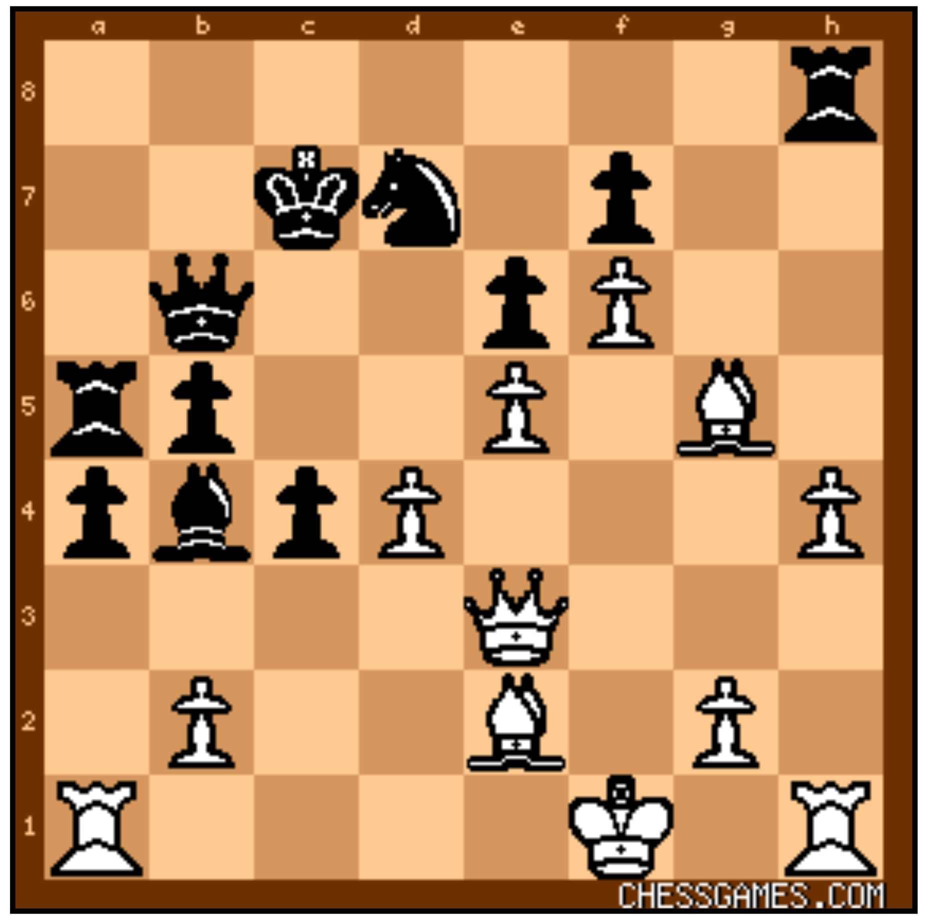
2. Eyes are good sensors

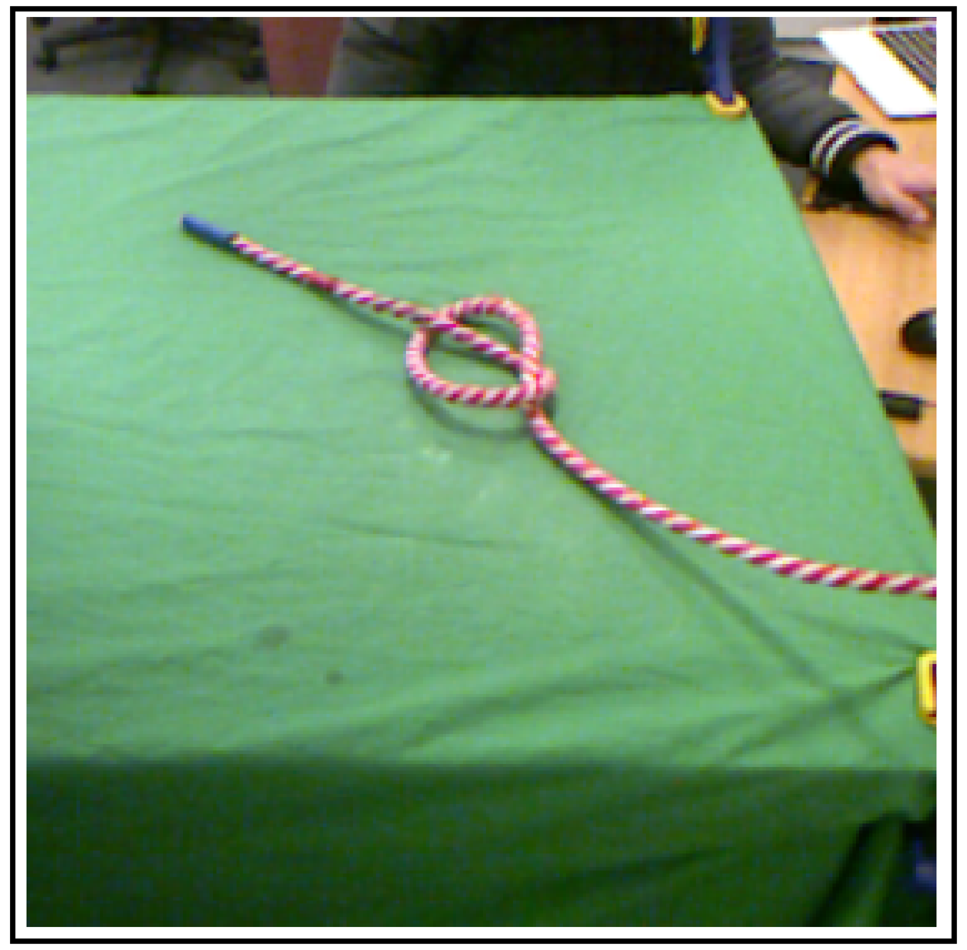


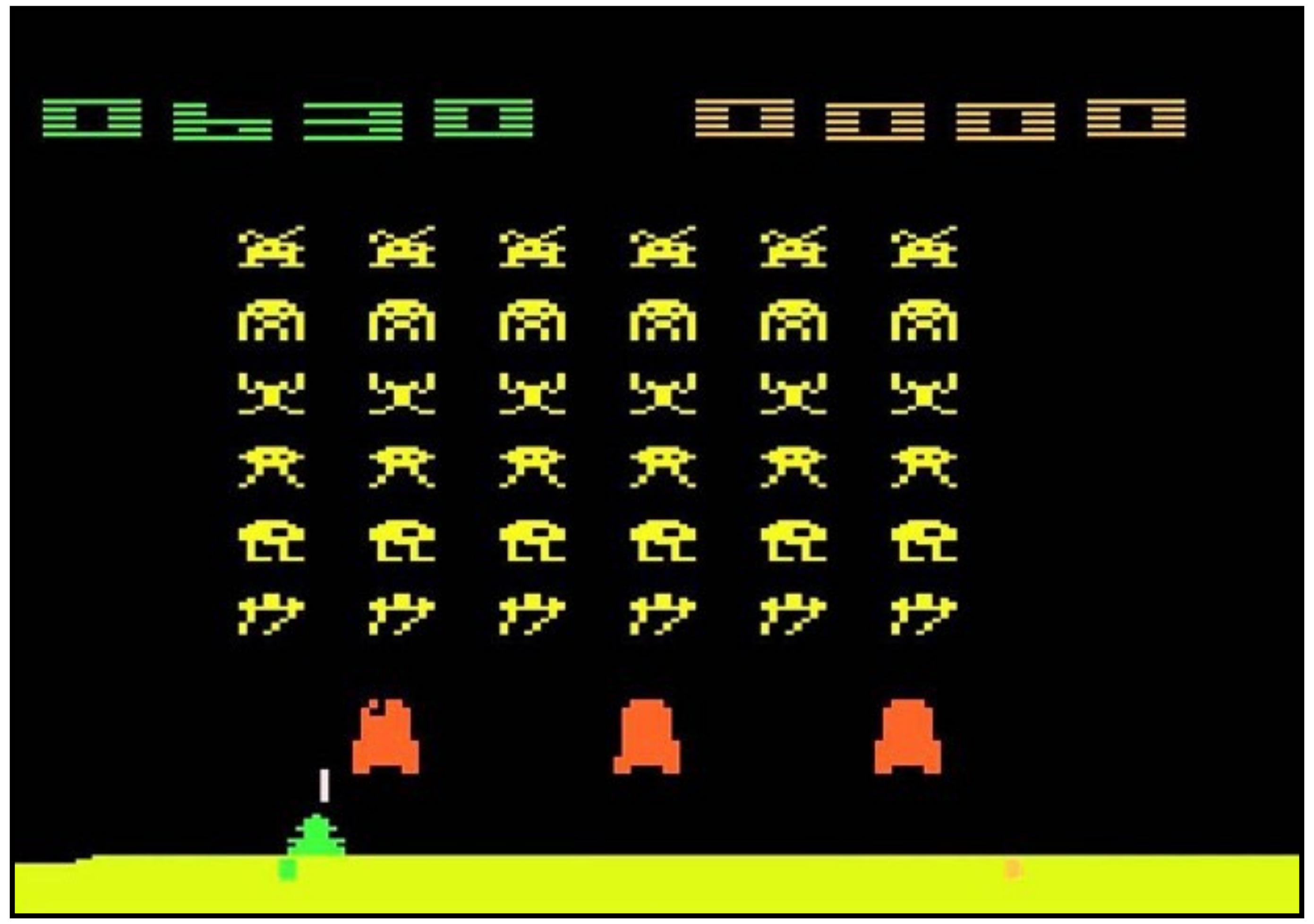
Why vision?

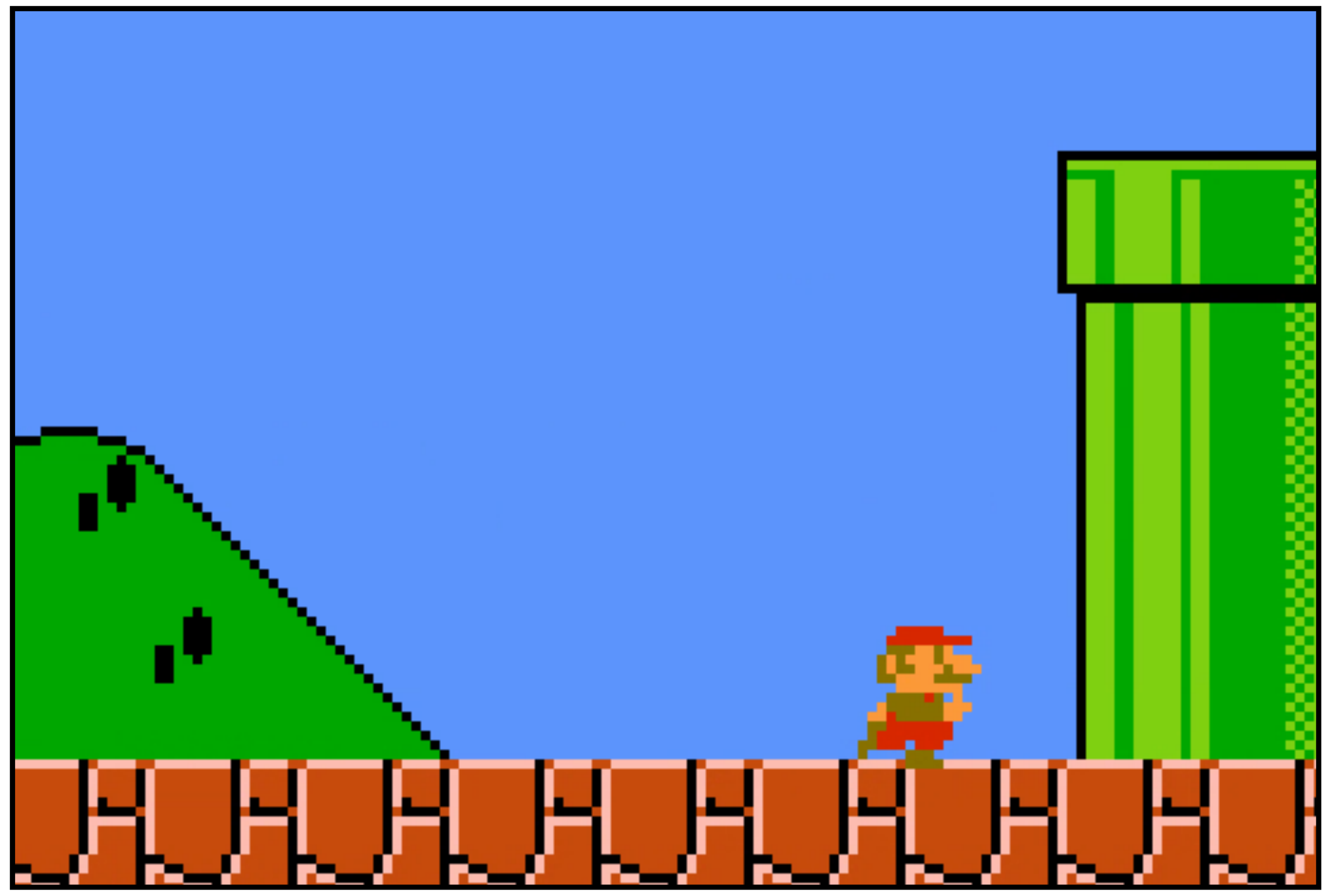
2. Universal interface





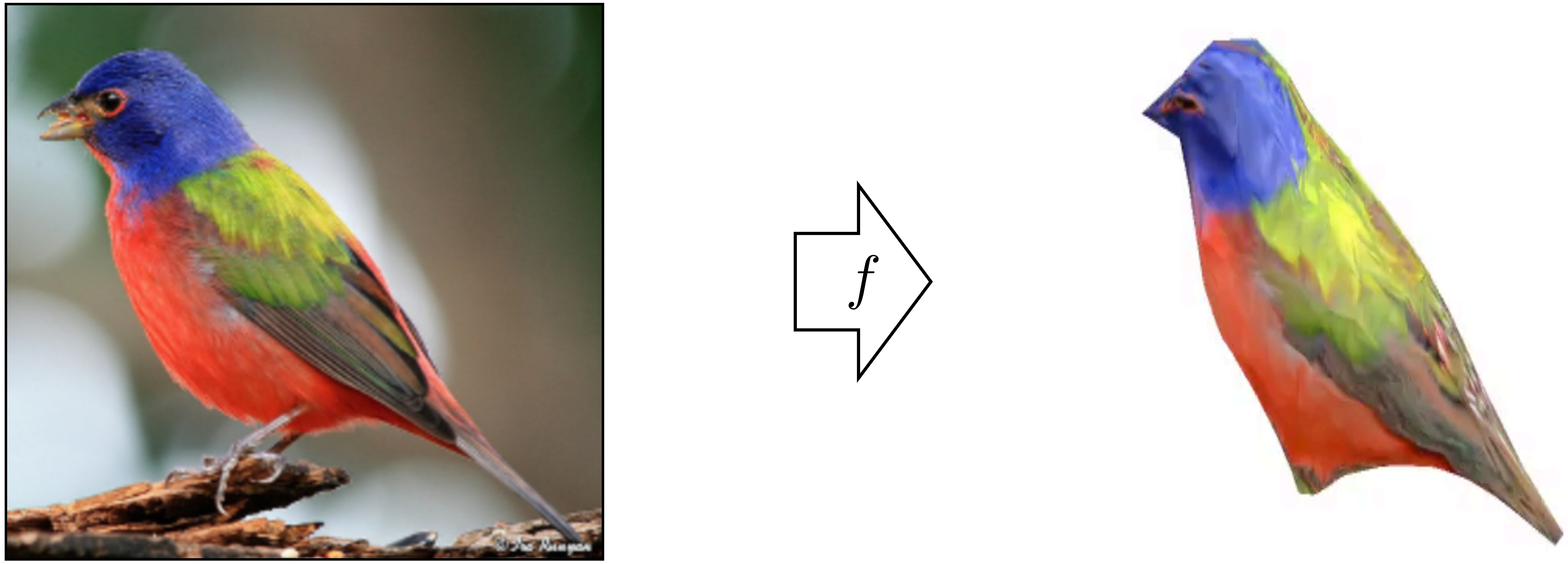






Why vision?

4. The brain’s *model building* system

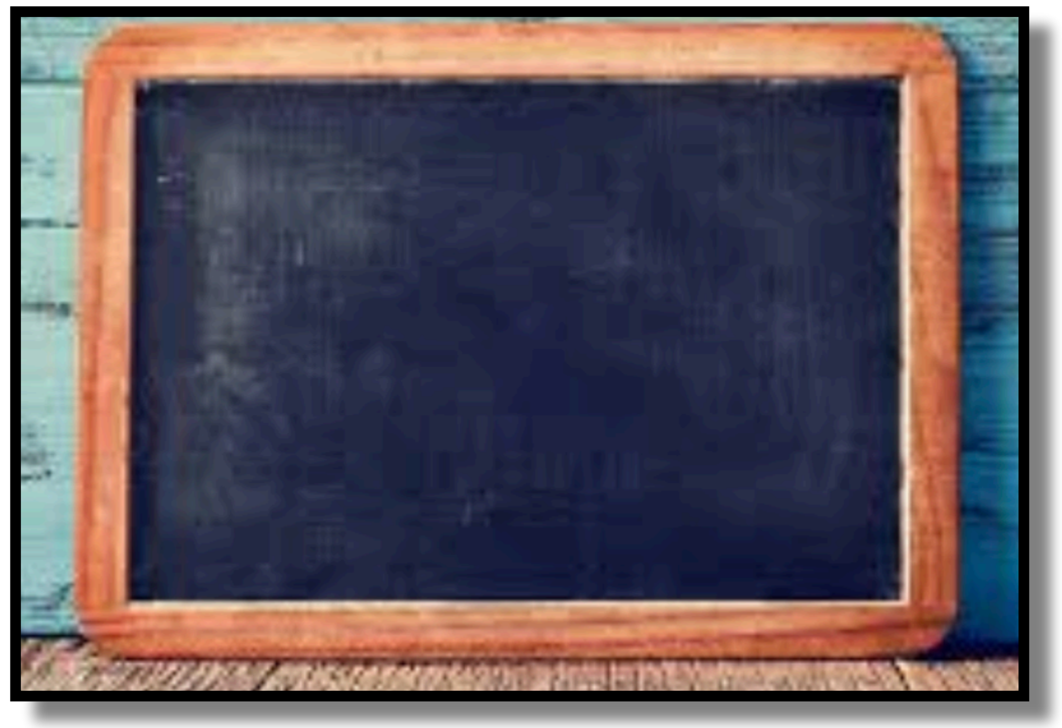


[Kanazawa, Tulsiani, et al., ECCV 2018]

Model-based intelligence

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

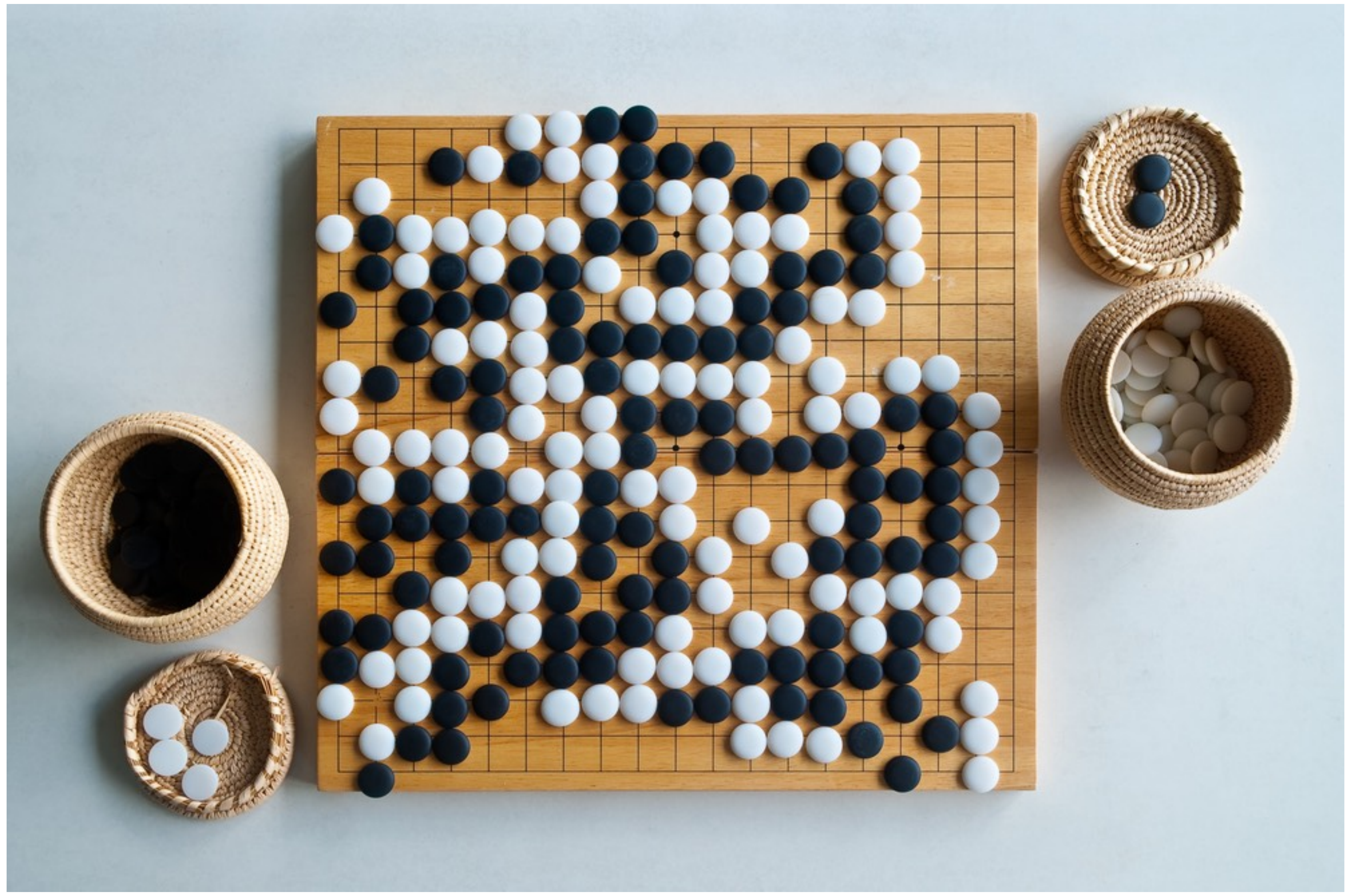
Yann LeCun's cake



ATARI Games

~10-50 million interactions!

AlphaGo



21 million games!

[Slide adapted from Pulkit Agrawal]