Microsoft

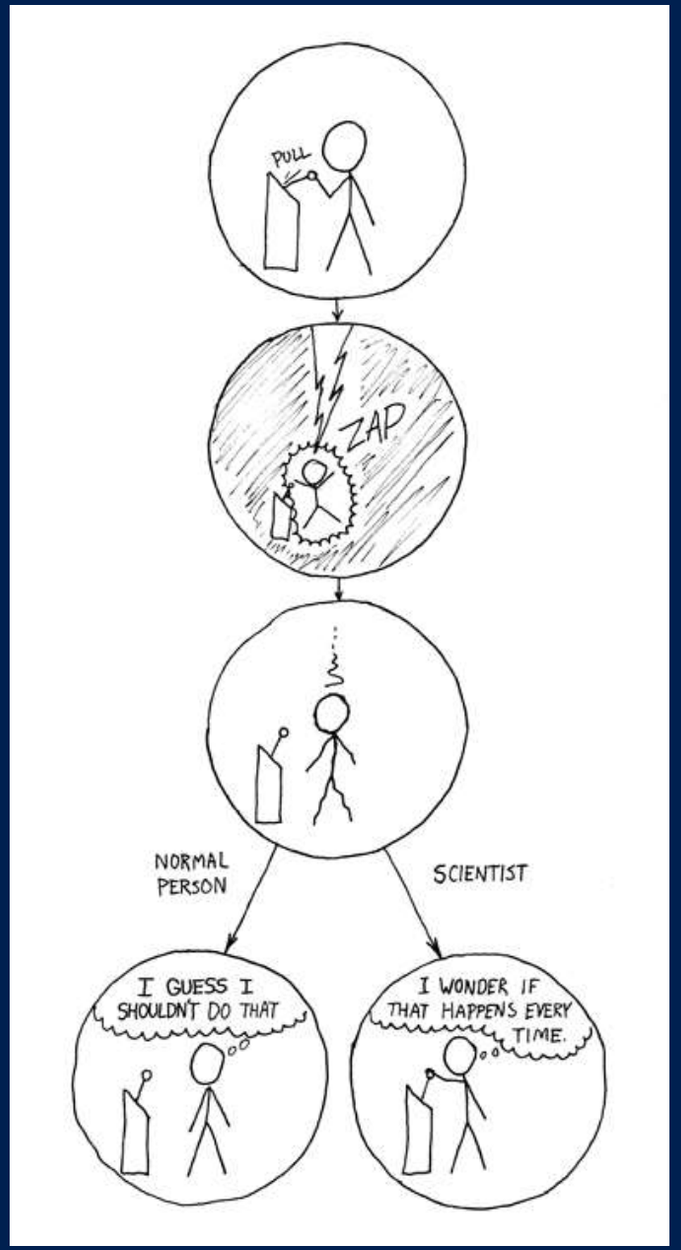
Introduction to

Reinforcement Learning

Sorin Pește

Technology Solutions Professional, Data & AI

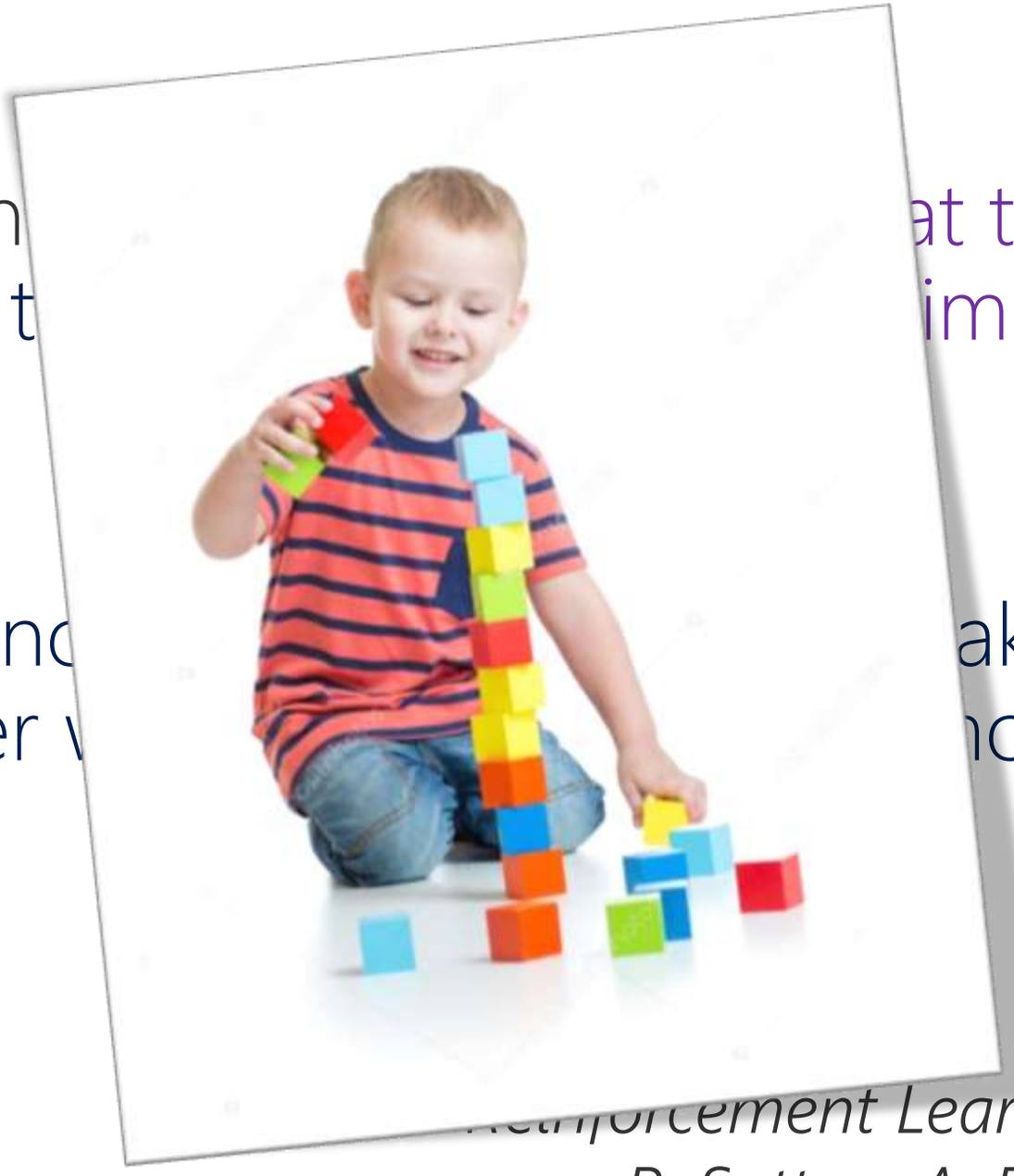
Microsoft



source: xkcd.com

"Reinforcemen at to do - how to map situations t imize a numerical

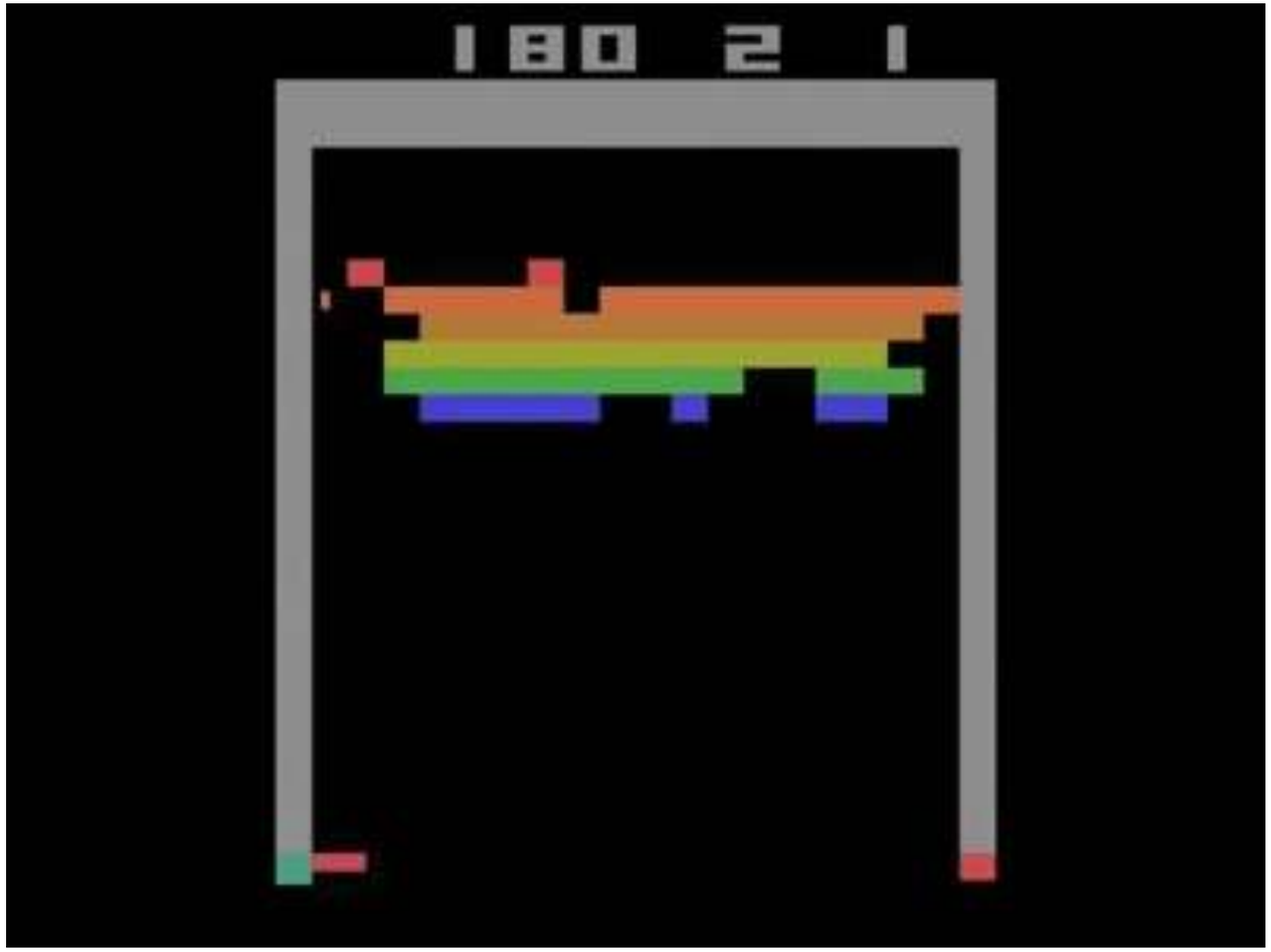
The learner is nc ake, but instead must discover v ost reward by



Reinforcement Learning: An Introduction
R. Sutton, A. Barto, MIT Press, 1998

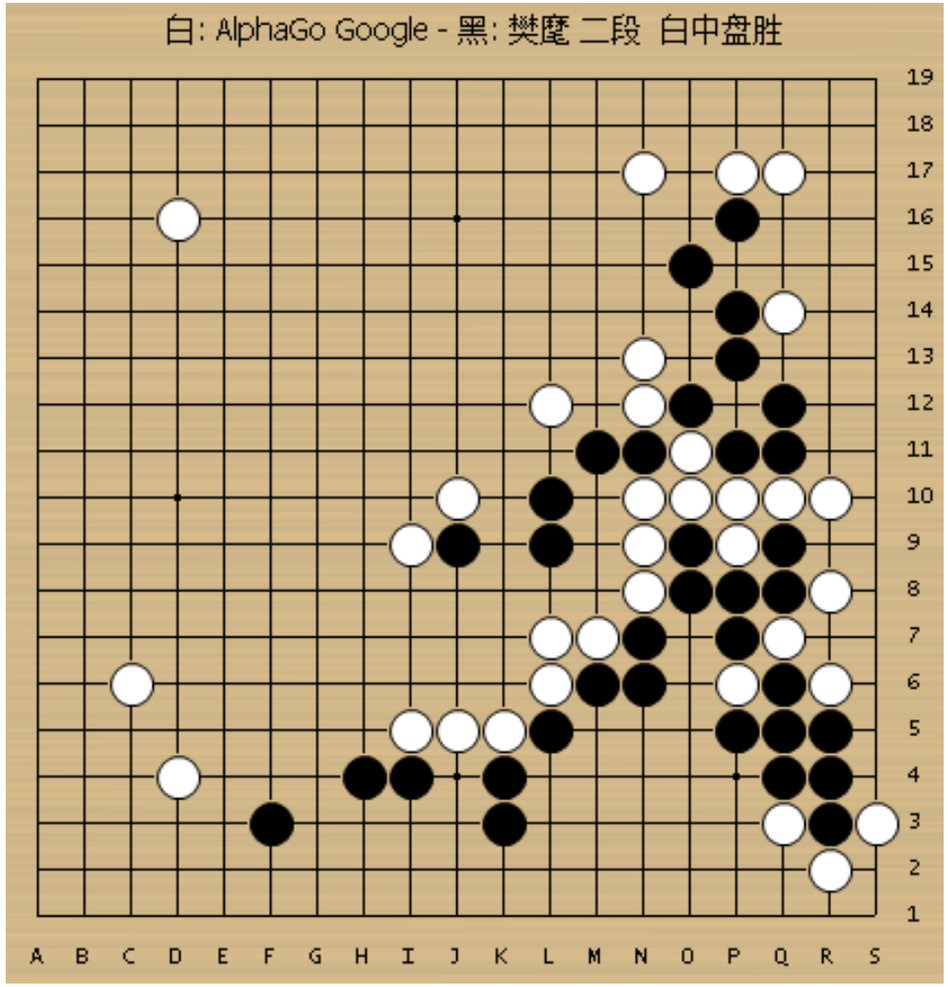
Applications

- *Playing Atari with Deep Reinforcement Learning* (Mnih et al, 2013)



Applications

- *Mastering the game of Go without human knowledge* (Silver, 2017)



Applications

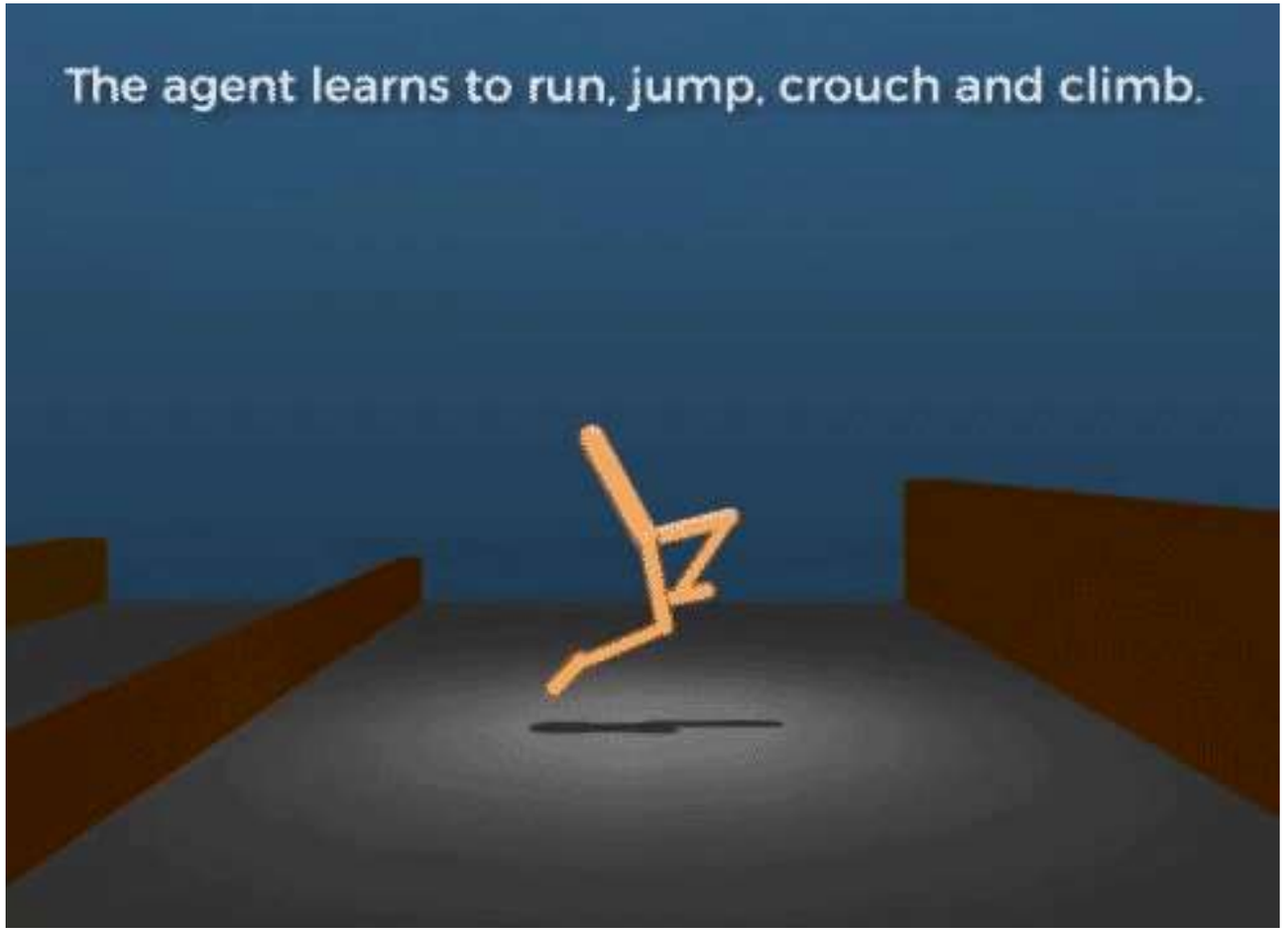
COMPUTERS

AI beats the world's best gamers after just two weeks of learning

Rich Haridy | August 14th, 2017

Applications

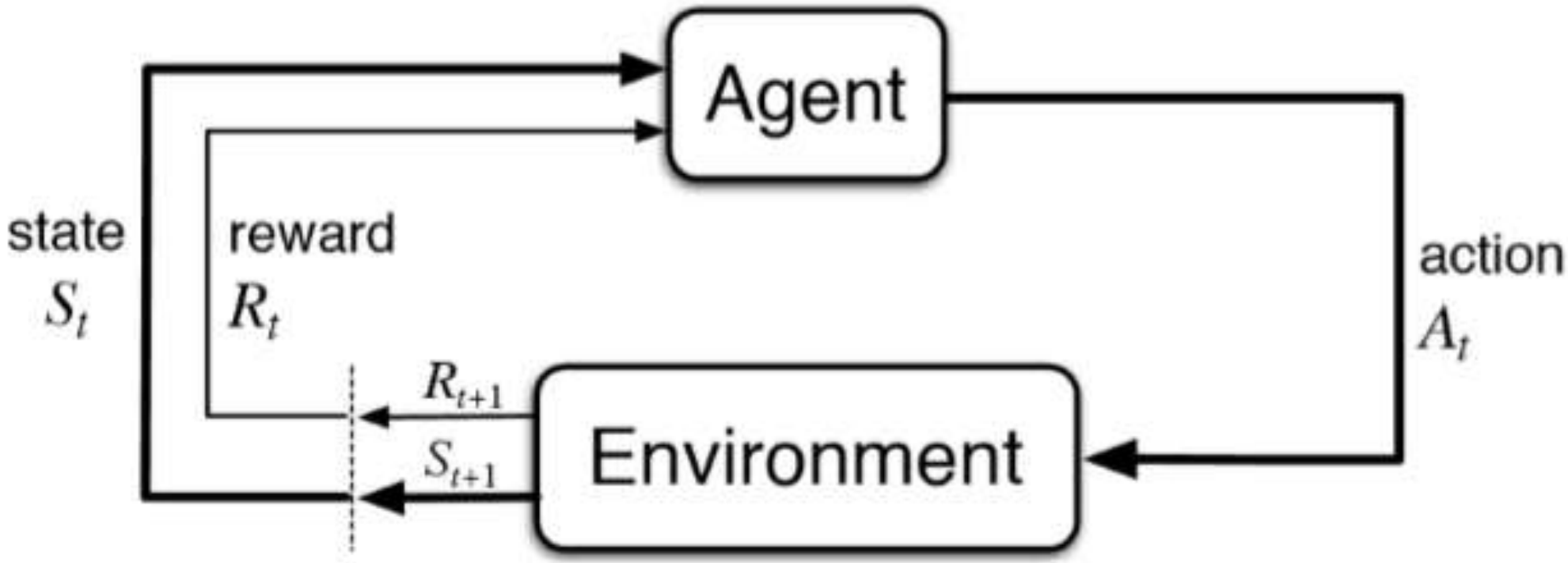
- *Emergence of Locomotion Behaviors in Rich Environments* (Heess, 2017)



Applications

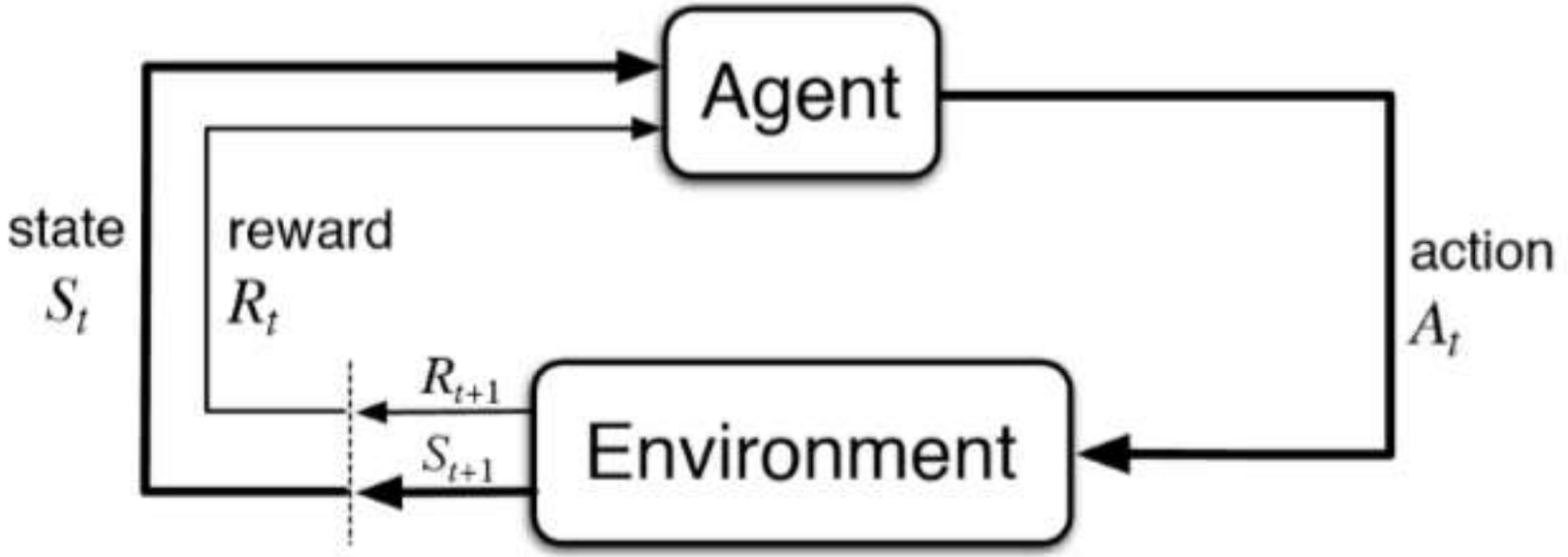
- AI-controlled Sailplane (Microsoft Research)

Agent and Environment



Environment

- Defines the world that the agent interacts with
- Basic loop:
  - Accepts Actions from the agent
  - Produces States, Rewards for the agent to sense and process

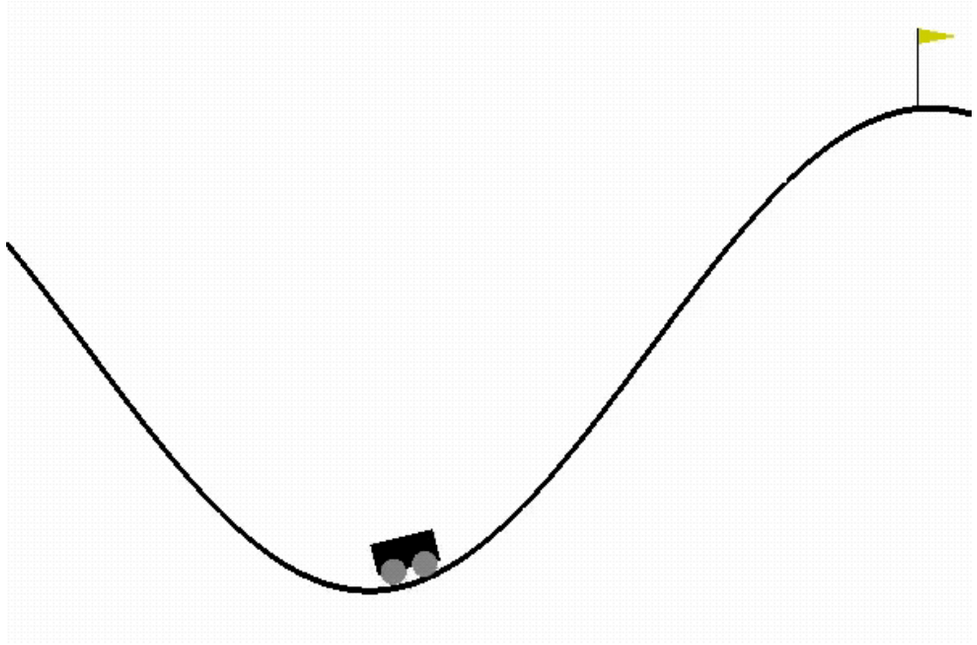


OpenAI Gym

https://gym.openai.com/

OpenAI Gym

Classic Control Problems

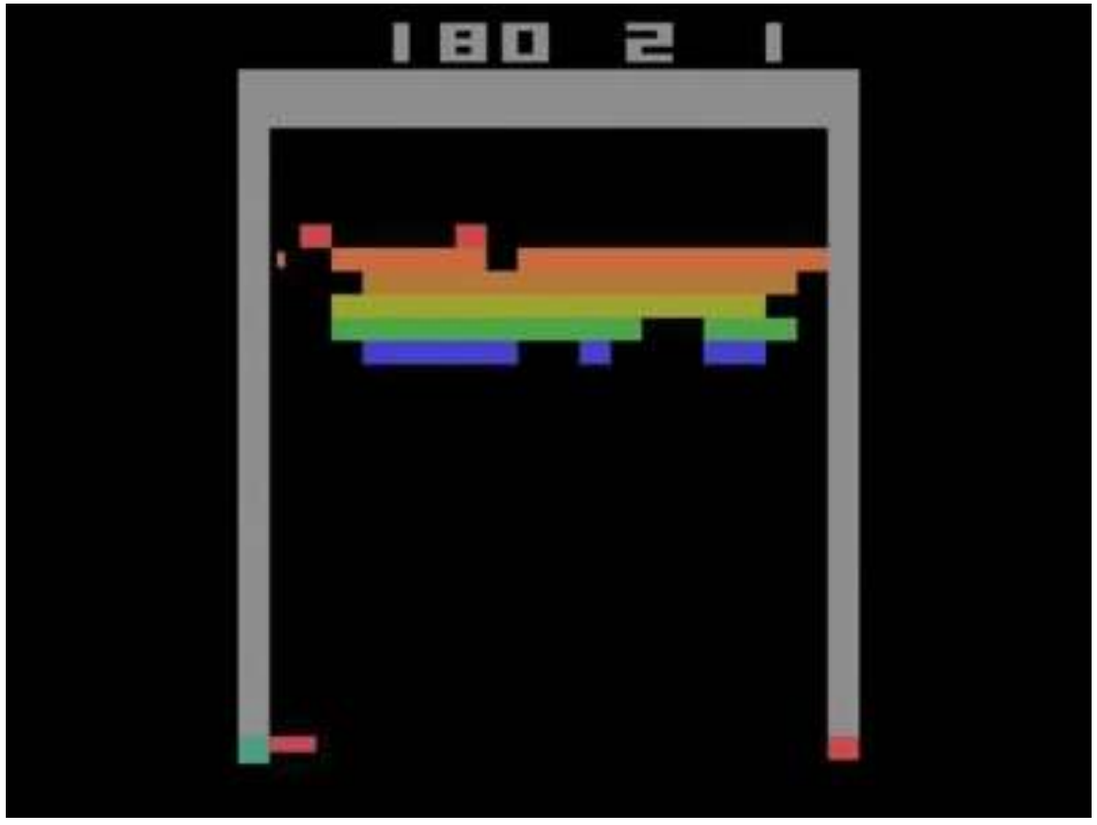


MountainCar-v0

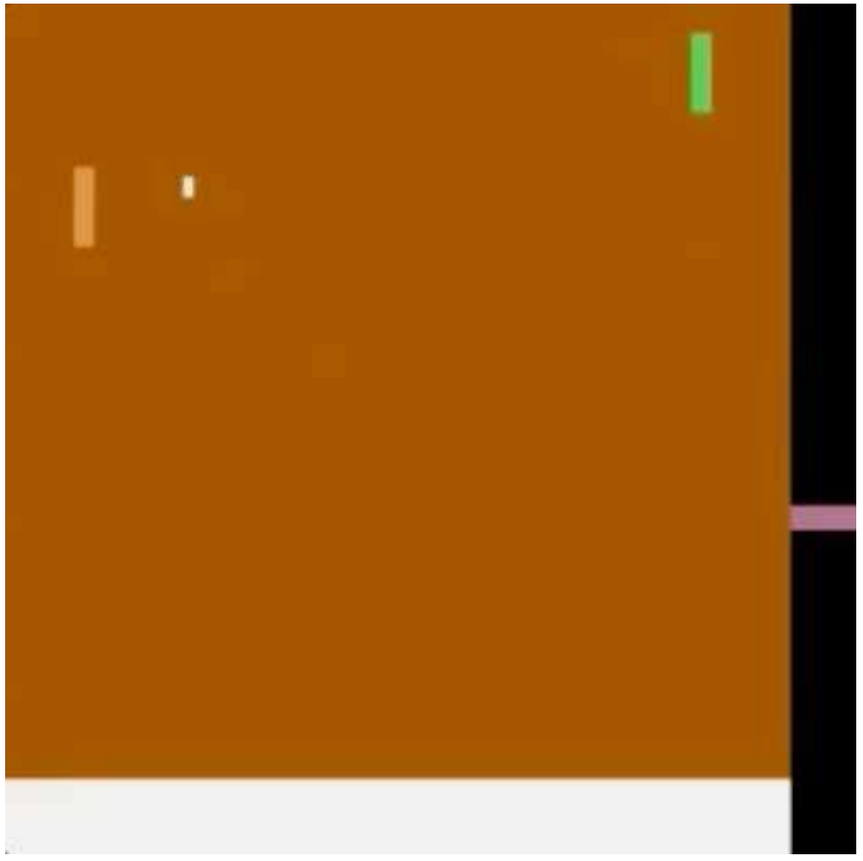
CartPole-v1

OpenAI Gym

Atari Game Environments



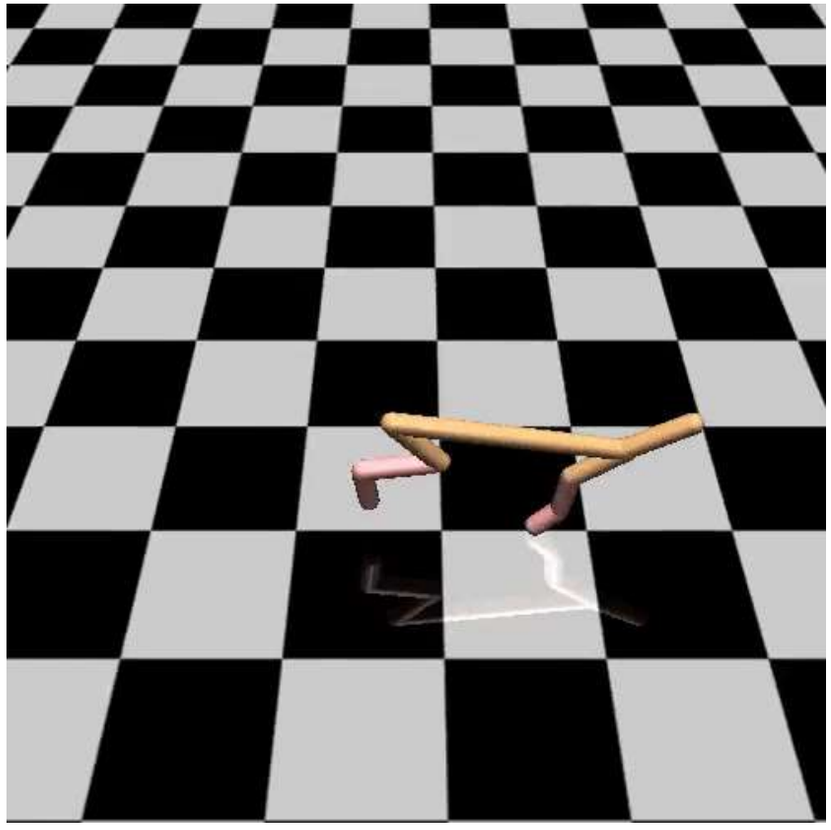
Breakout-v0



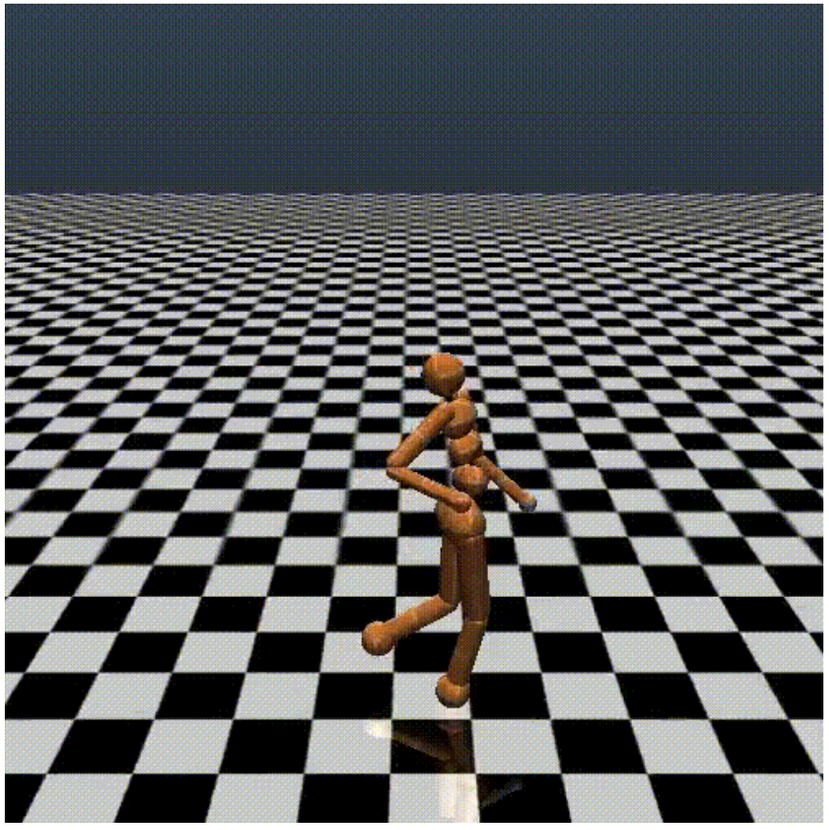
Pong-v0

OpenAI Gym

MuJoCo - Multi-Joint dynamics with Contact



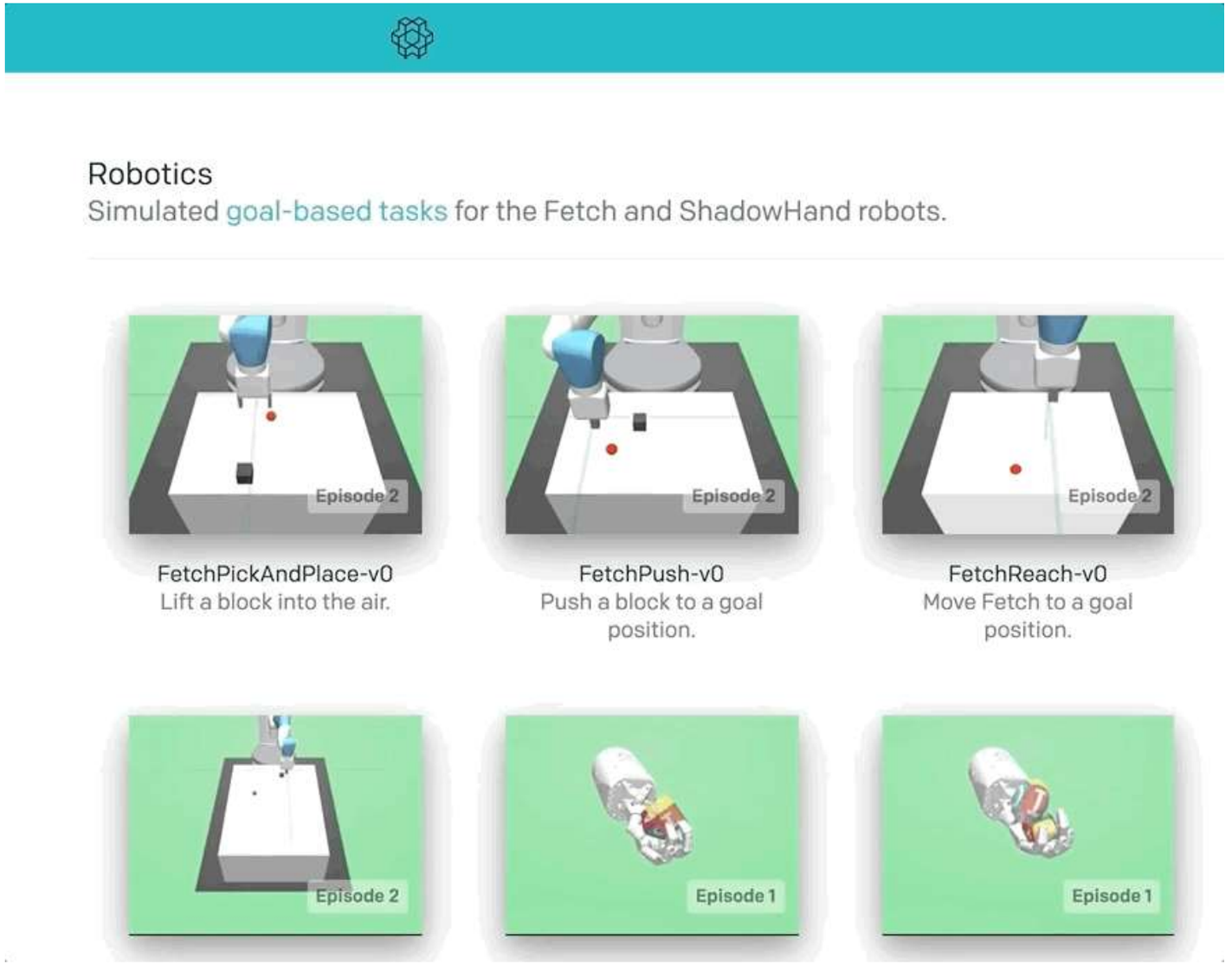
HalfCheetah-v2



Humanoid-v2

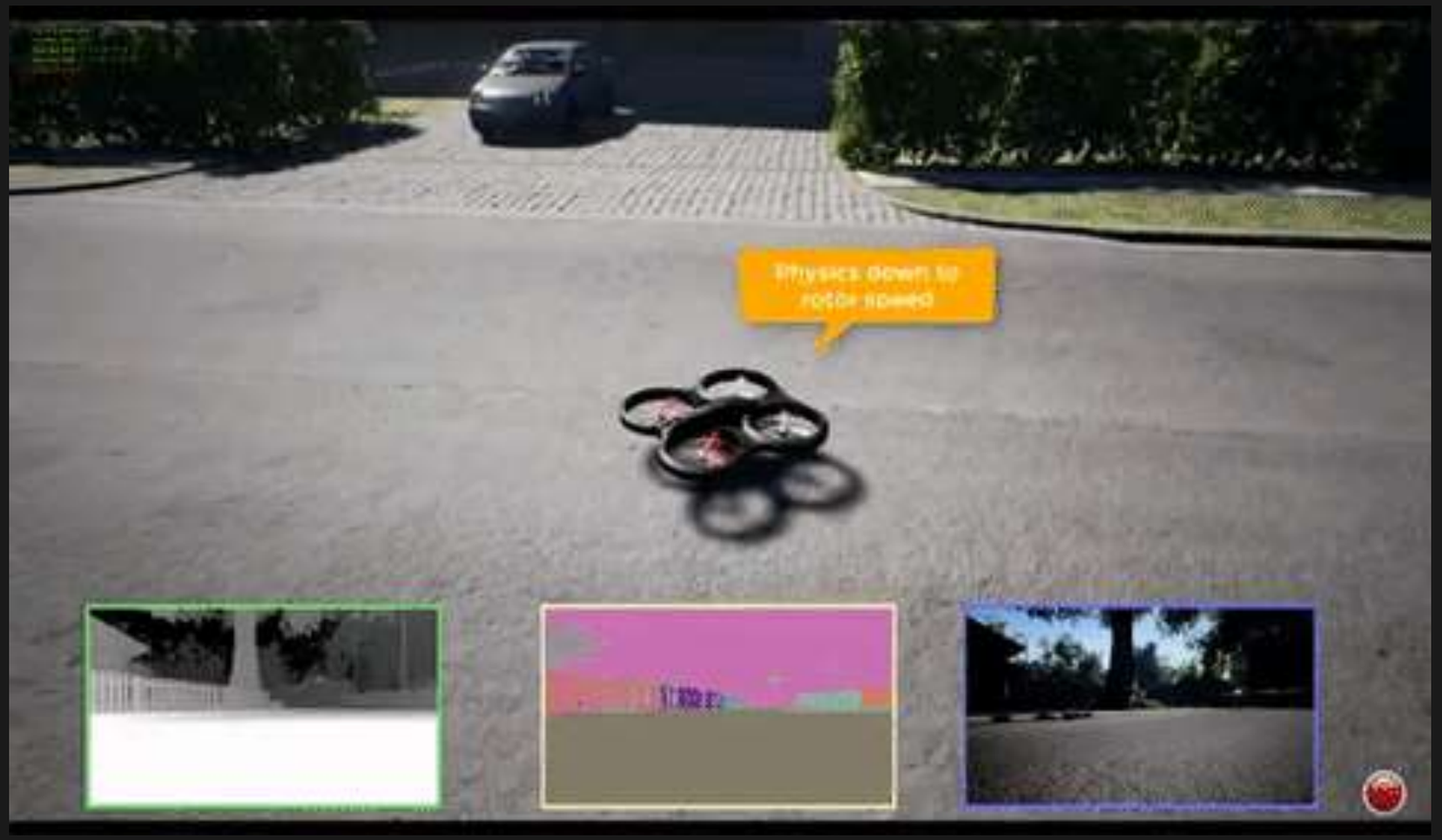
OpenAI Gym

Robotics



Microsoft AirSim

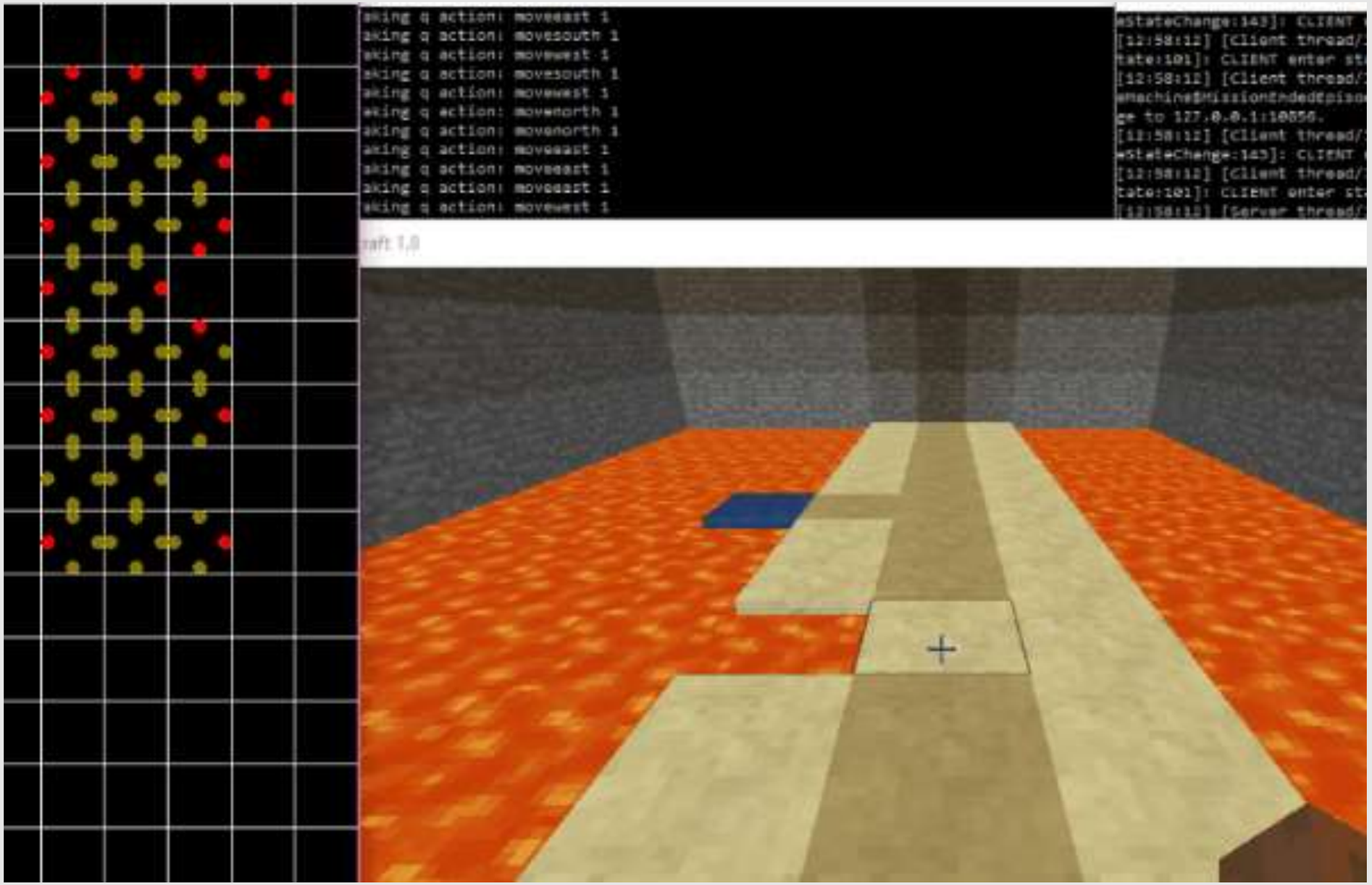
https://github.com/Microsoft/AirSim


Physics down to
reduce speed


98
New plugins are available

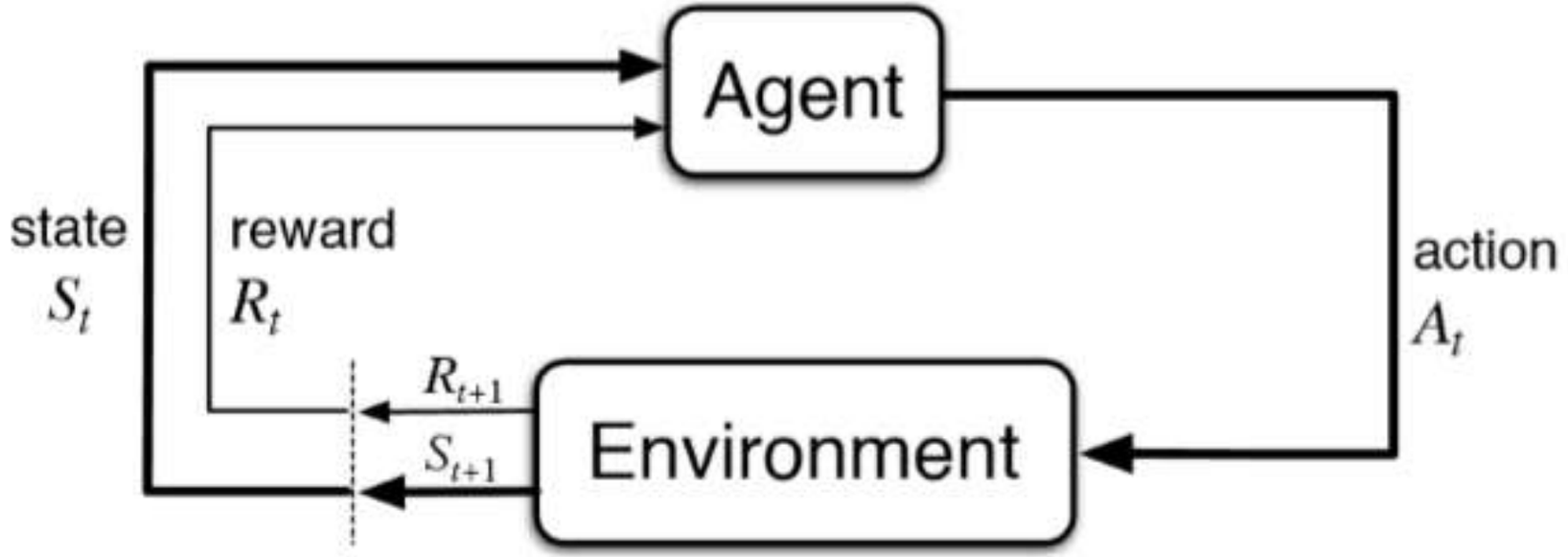
Project Malmo

https://github.com/Microsoft/malmo

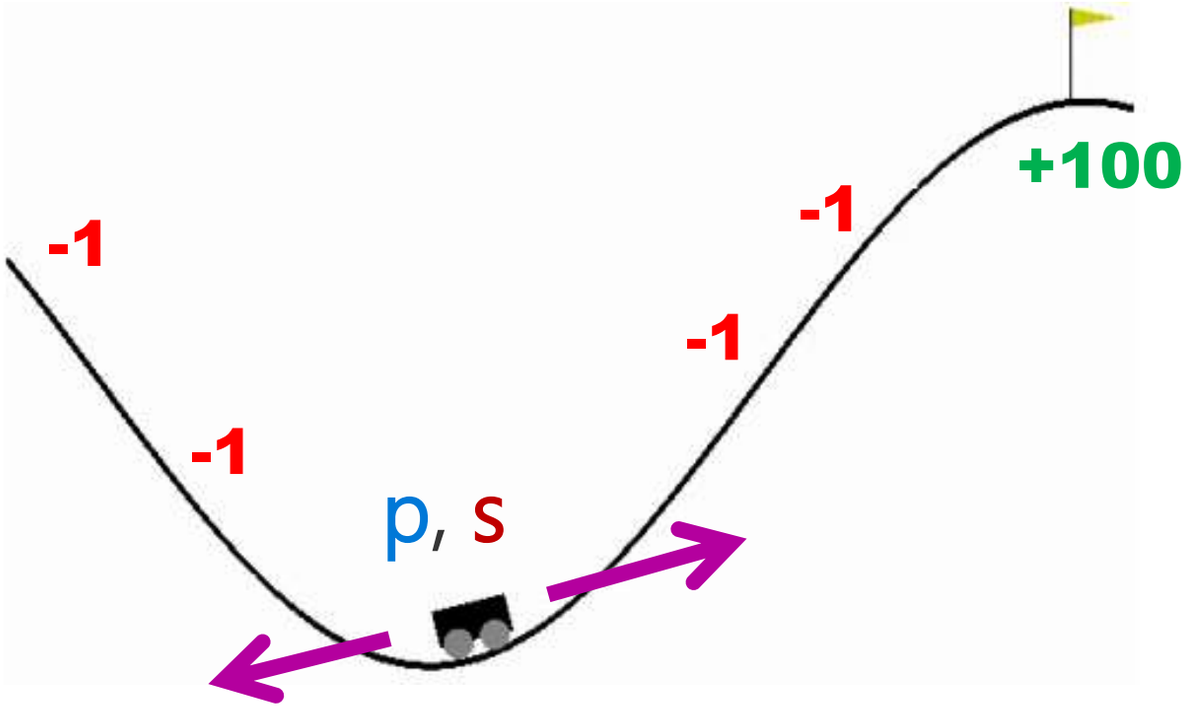

aking q action: moveeast 1
aking q action: movesouth 1
aking q action: movewest 1
aking q action: movesouth 1
aking q action: movewest 1
aking q action: movenorth 1
aking q action: movenorth 1
aking q action: moveeast 1
aking q action: moveeast 1
aking q action: moveeast 1
aking q action: movewest 1

Agent

- Learns to achieve goals by interacting with environment
- Basic loop:
  - Senses: State, Reward
  - Effect: selects an Action

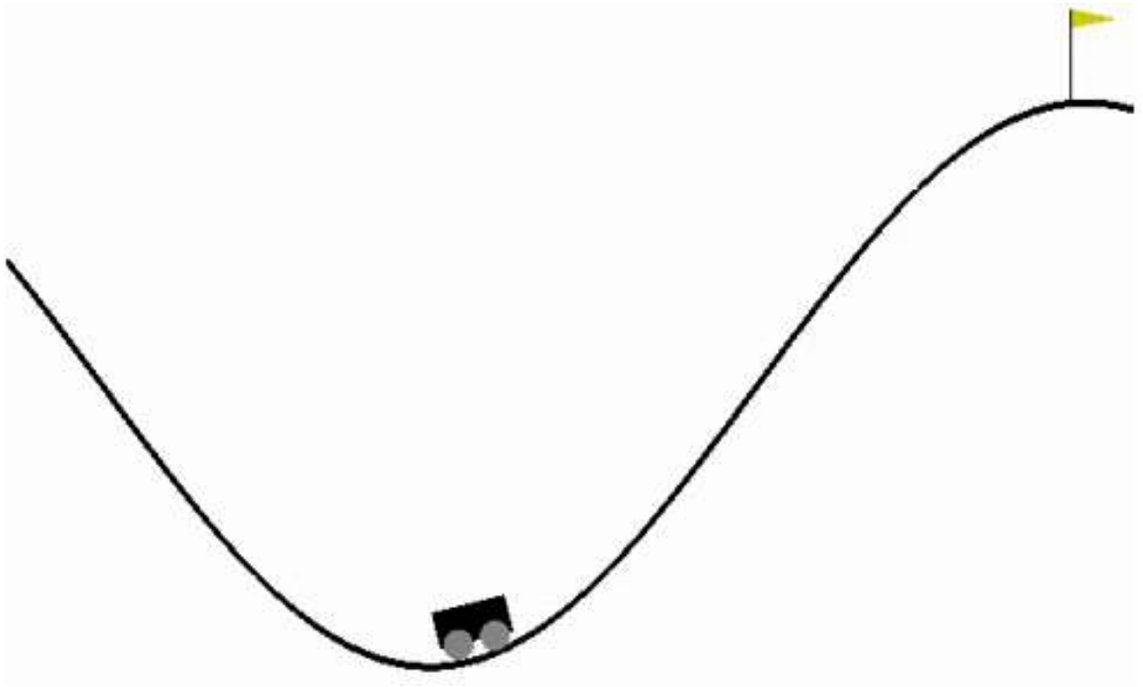


Notation



- Time Step t
- State Space S = (position, speed)
- Action Space A = (push left , push right)
- Reward r_t = 100 if goal reached; -1 otherwise
- Policy π = select Action with maximum return (Greedy)

Notation



- Return

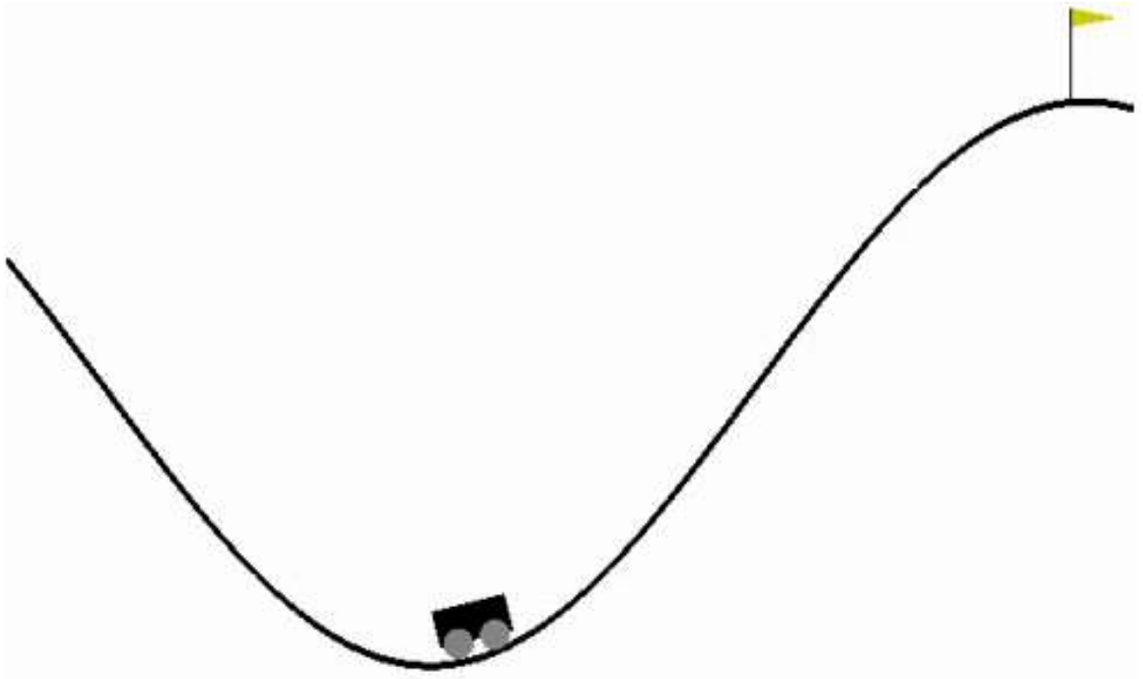
$$G_t = r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+3} + \gamma^3 r_{t+4} + \ldots.$$

$$= r_{t+1} + \gamma G_{t+1}$$

$\gamma < 1$

discount factor

Notation



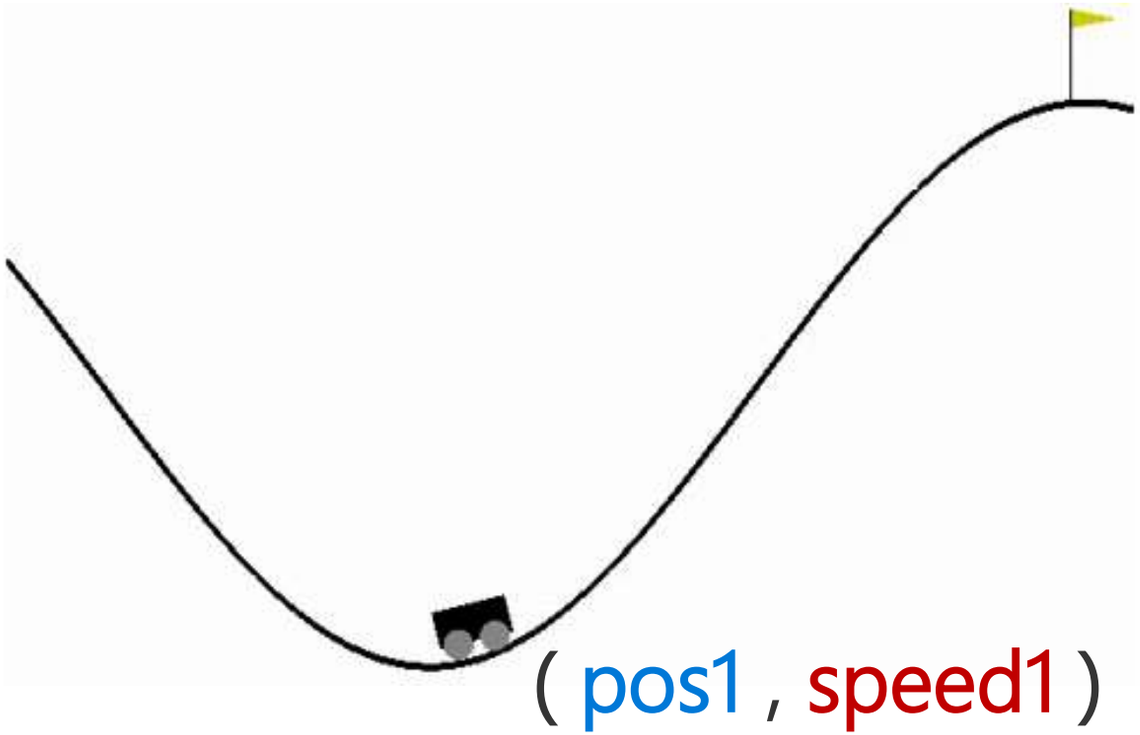
- Action-Value Function

$Q^{\pi}(s, a)$ = expected return starting from state s,

taking action a,

and thereafter following policy π

Example



$$Q^{\pi}(\,(\,\text{pos1}, \text{speed1}\,), \text{push left}) = 21$$

$$Q^{\pi}(\,(\,\text{pos1}, \text{speed1}\,), \text{push right}) = 26$$

Q-learning

How can we determine the function $\mathbf{Q^{\pi}(s, a)}$?

(Q-learning)

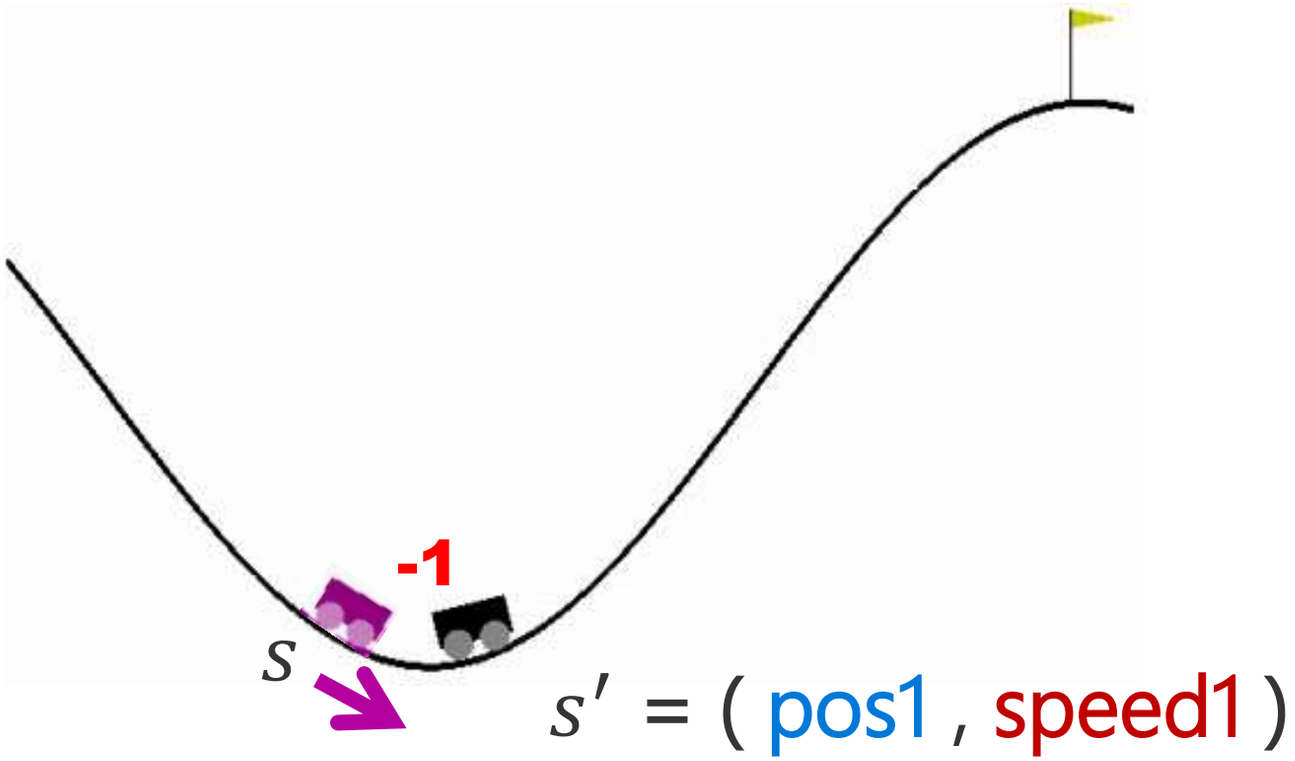
Q-learning

- Given a new experience, (s, a, r, s')

$$Q(s, a) = r + \gamma \, max_{a'} Q(s', a')$$

(Bellman Equation)

Example



$$Q^{\pi}(s', \text{push left}) = 21$$

$$Q^{\pi}(s', \text{push right}) = 26$$

Q-learning

- Given a new experience, $(s, push\ right, -1, s')$

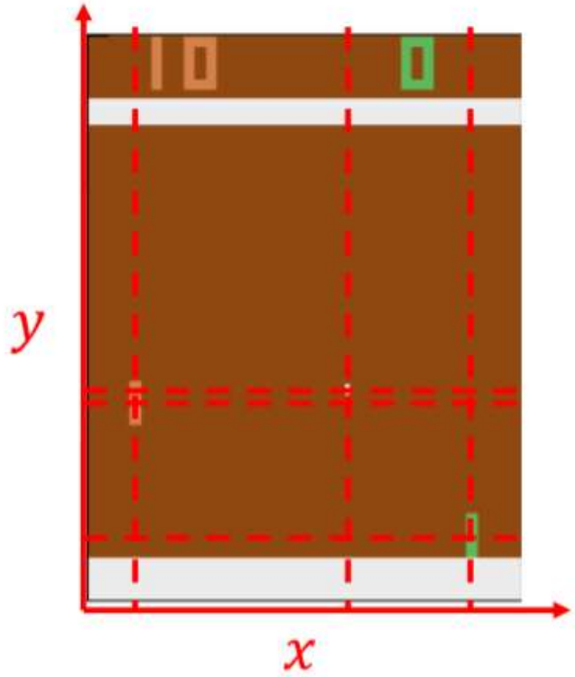
$$Q(s, push\ right) = -1 + \gamma * 26$$

(Bellman Equation)

Tabular Q-learning

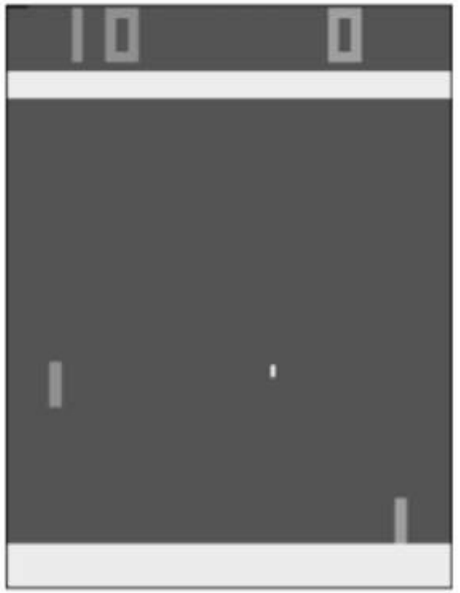
State	Action	Q (s,a)
S1	A1	0
S1	A2	-2
S2	A1	1
S2	A2	-1
S3	A1	3
…	…	…
S99	A1	100

RL Challenge #1: Representation



Example: $k = 3\ x\ 2$

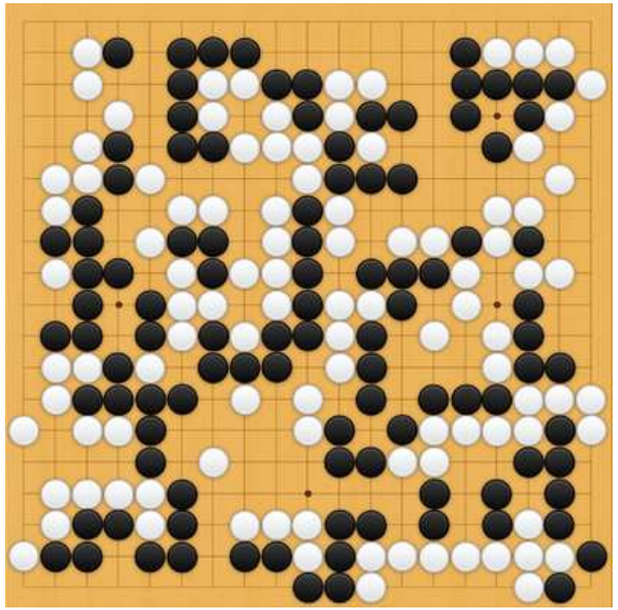
indicate x, y coordinate of ball and paddles



Example: $k = (84\ x\ 84)256$

84x84 8-bit gray-scale pixel values of scaled image

RL Challenge #2: Generalization



- The agent should be able to deal with previously unseen states
- States might be only partially observable

Q-learning

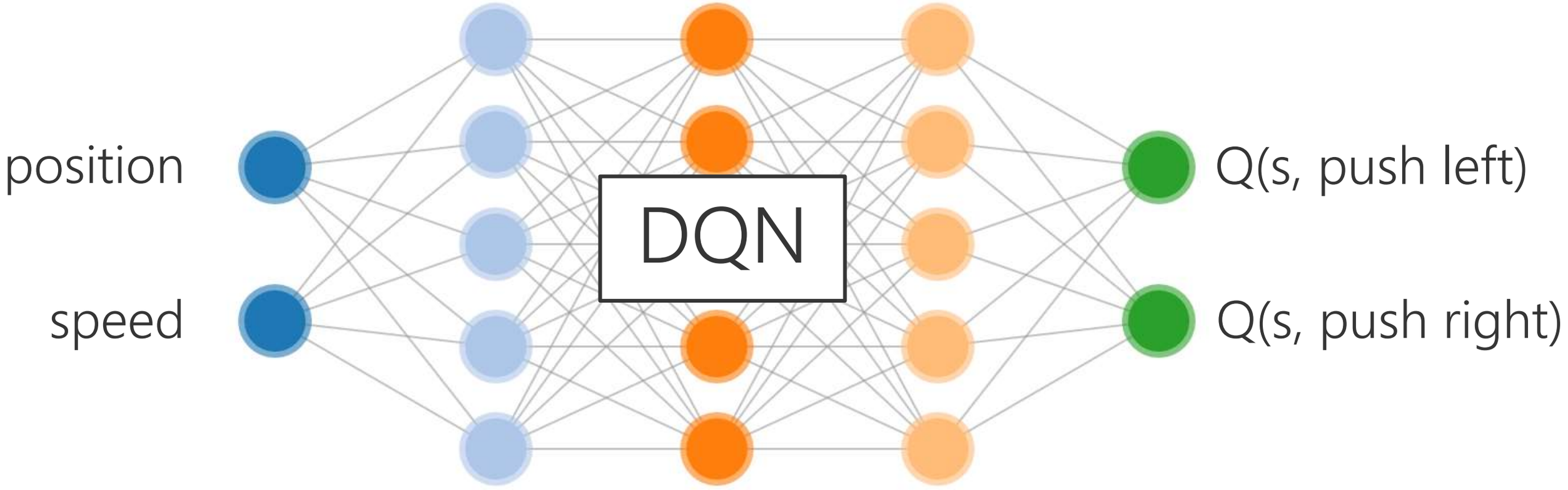
How can we approximate the function $\mathbf{Q^{\pi}(s, a)}$?

(given representation and generalization constraints)

Deep Q-learning

$$(s, a, r, s') \quad => \quad \underbrace{L} = \| \underbrace{r + \gamma \, max_{a'} Q(s', a')} - \underbrace{Q(s, a)} \|^2$$

loss *"target"* *"current"*



- *Playing Atari with Deep Reinforcement Learning* (Mnih et al, 2013)

A (very naive) Q-learning Algorithm

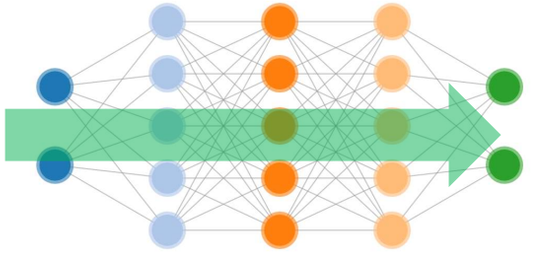
```
qmodel.init('random')
policy.init('greedy')
for N episodes do:
      state = environment.init_episode()
      do:
         a = policy.select_action(state, qmodel)
         (reward, next_state, done) = environment.step(state, action)
         qmodel.learn((state, action, reward, next_state))
         state = next_state
      while (!done)
end for
```

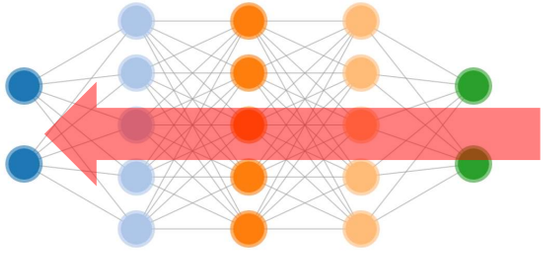
A (pretty naive) Q-learning Algorithm

```
do:
    a = policy.select_action(state, qmodel)

    (reward, next_state, done) =
        environment.step(state, action)

    qmodel.learn((state, action, reward, next_state))
    state = next_state
while (!done)
```





Actual Implementation of Q-learning with CNTK (http://cntk.ai/)

README.md

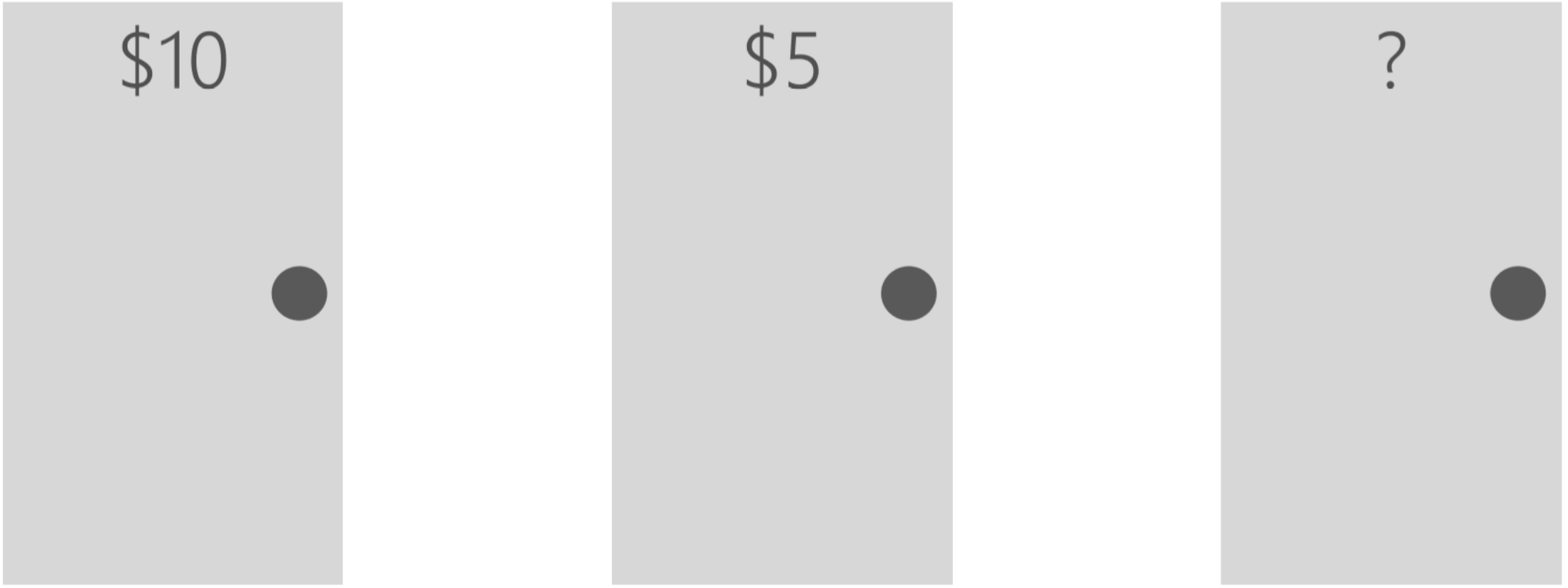
CNTK Examples: Reinforcement Learning

Overview

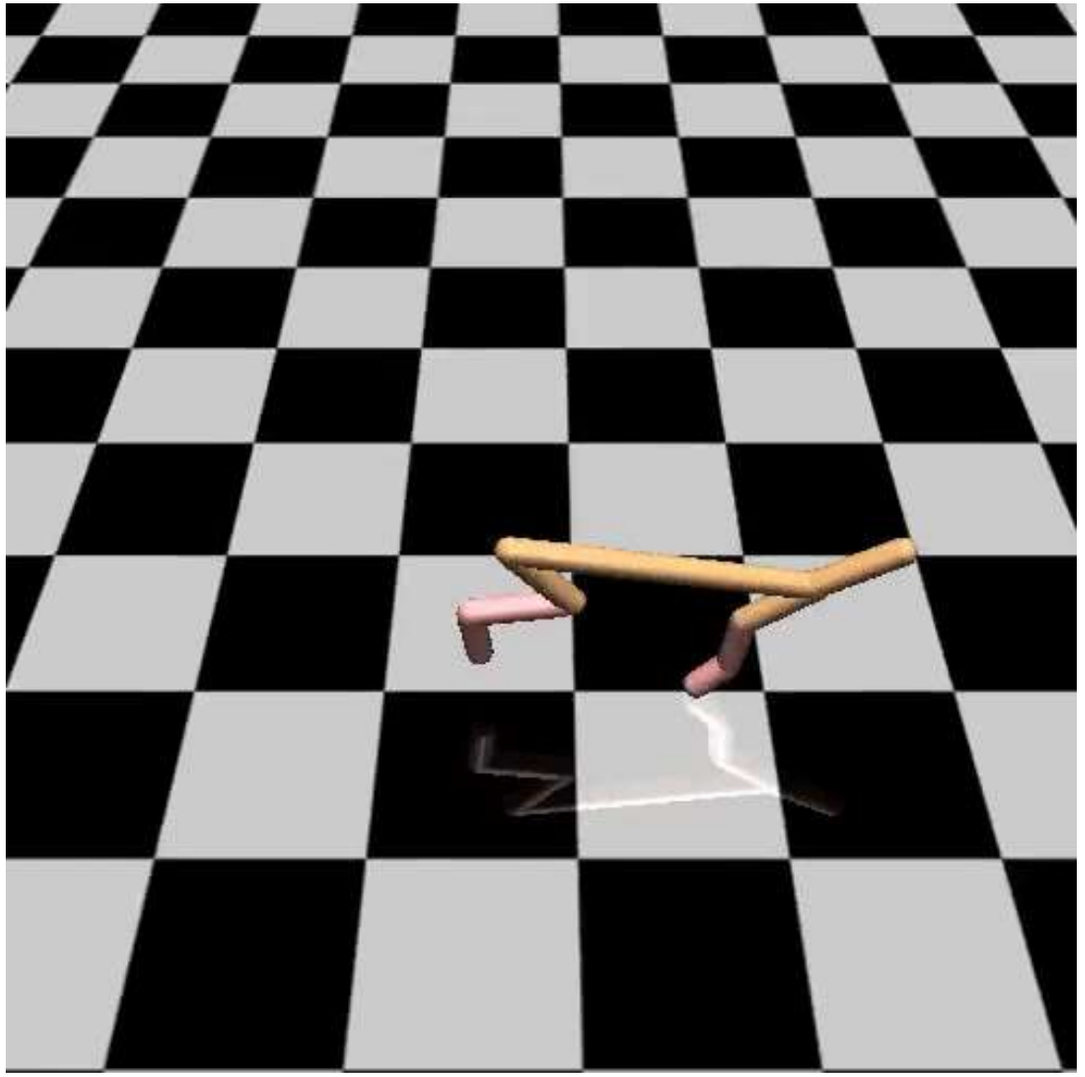
Data:	**Dynamic Atari Learning Environment (ALE)**
Purpose	This folder contains examples that demonstrate how to use CNTK to define Deep Q Neural Network (Mnih & al, 2013), a value-based Deep Reinforcement Learning method inspired by Q-Learning method
Network	DeepQNeuralNetwork (DQN).
Training	Adam.
Comments	See below.

https://github.com/Microsoft/CNTK/tree/master/Examples/ReinforcementLearning

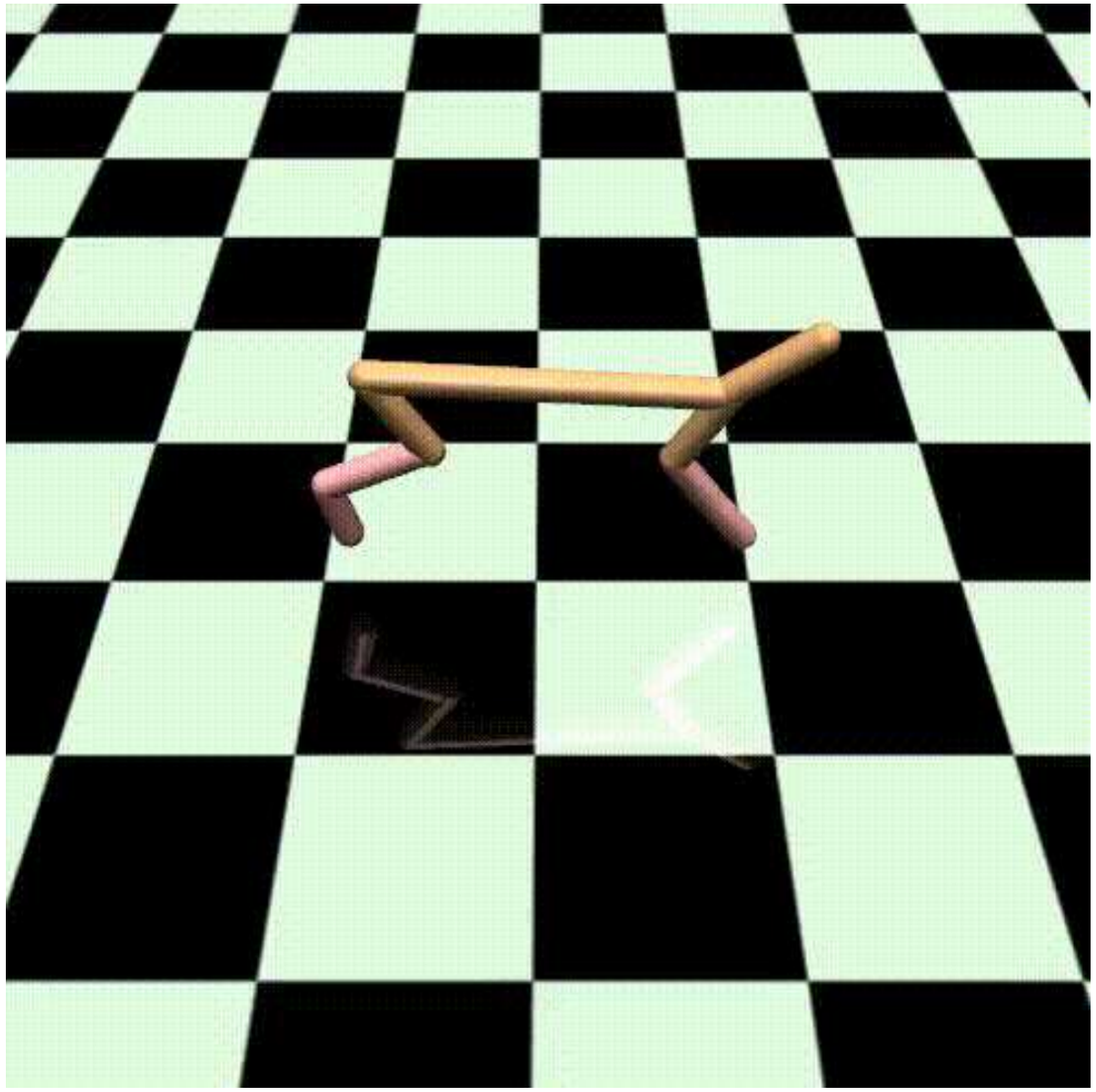
RL Challenge #3: Exploration vs Exploitation



Exploration / Exploitation Dilemma

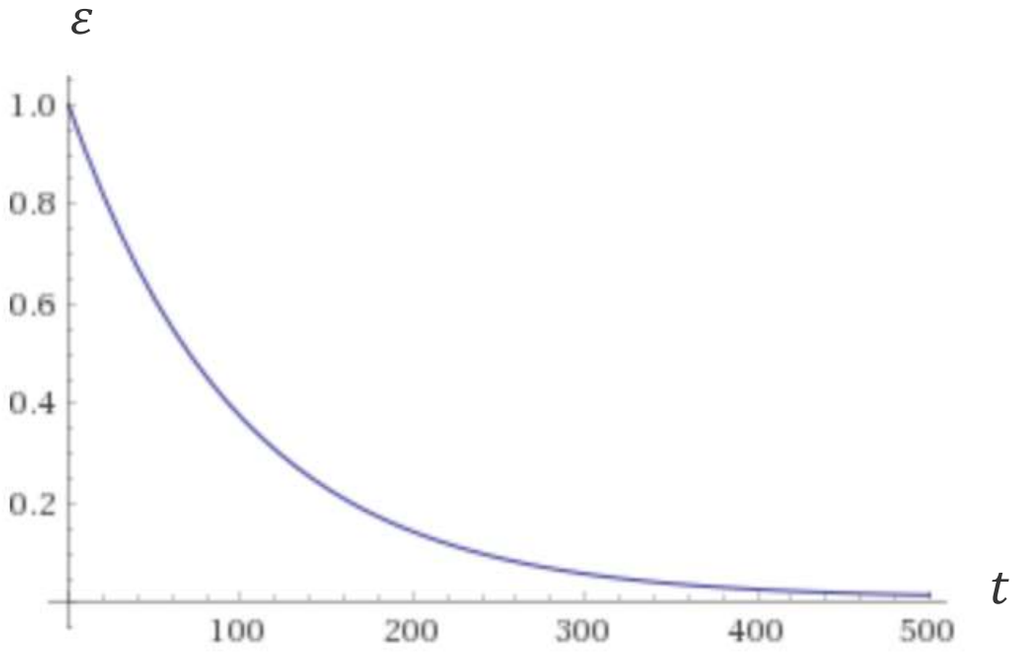


Exploration / Exploitation Dilemma



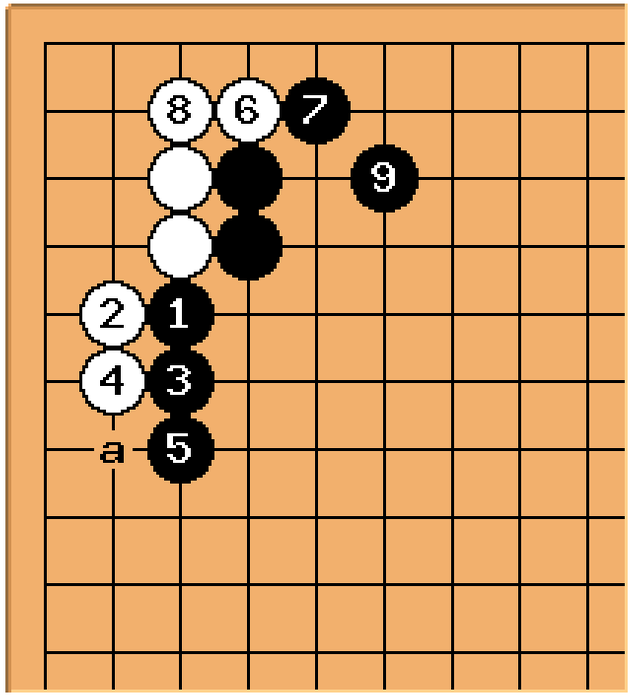
Exploration / Exploitation Dilemma

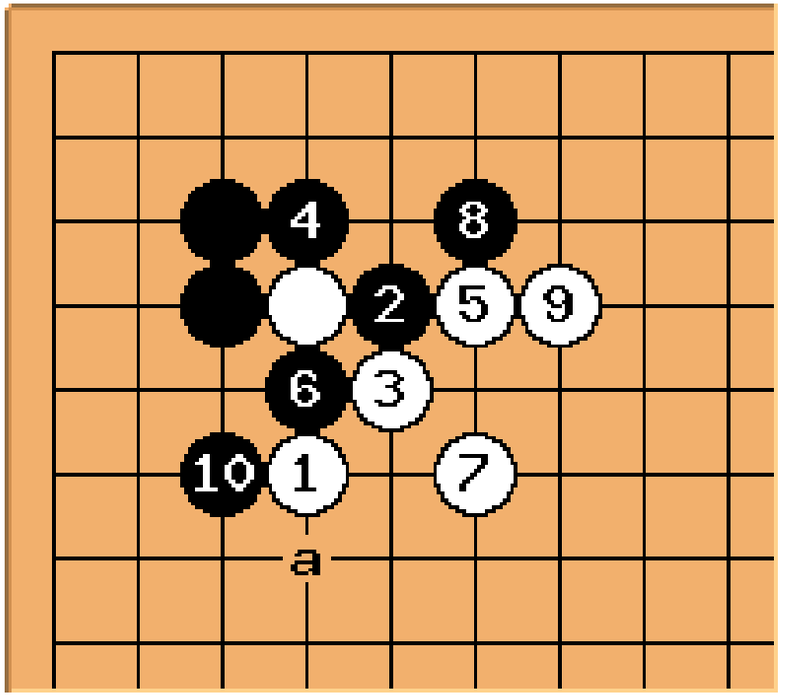
- Use an ε-Greedy policy
  - If (*random value*) < ε, choose a random action
  - Else, choose Greedy action



$$\varepsilon = \varepsilon_{min} + (\varepsilon_{max} - \varepsilon_{min})e^{-\lambda t}$$

Exploration / Exploitation Dilemma





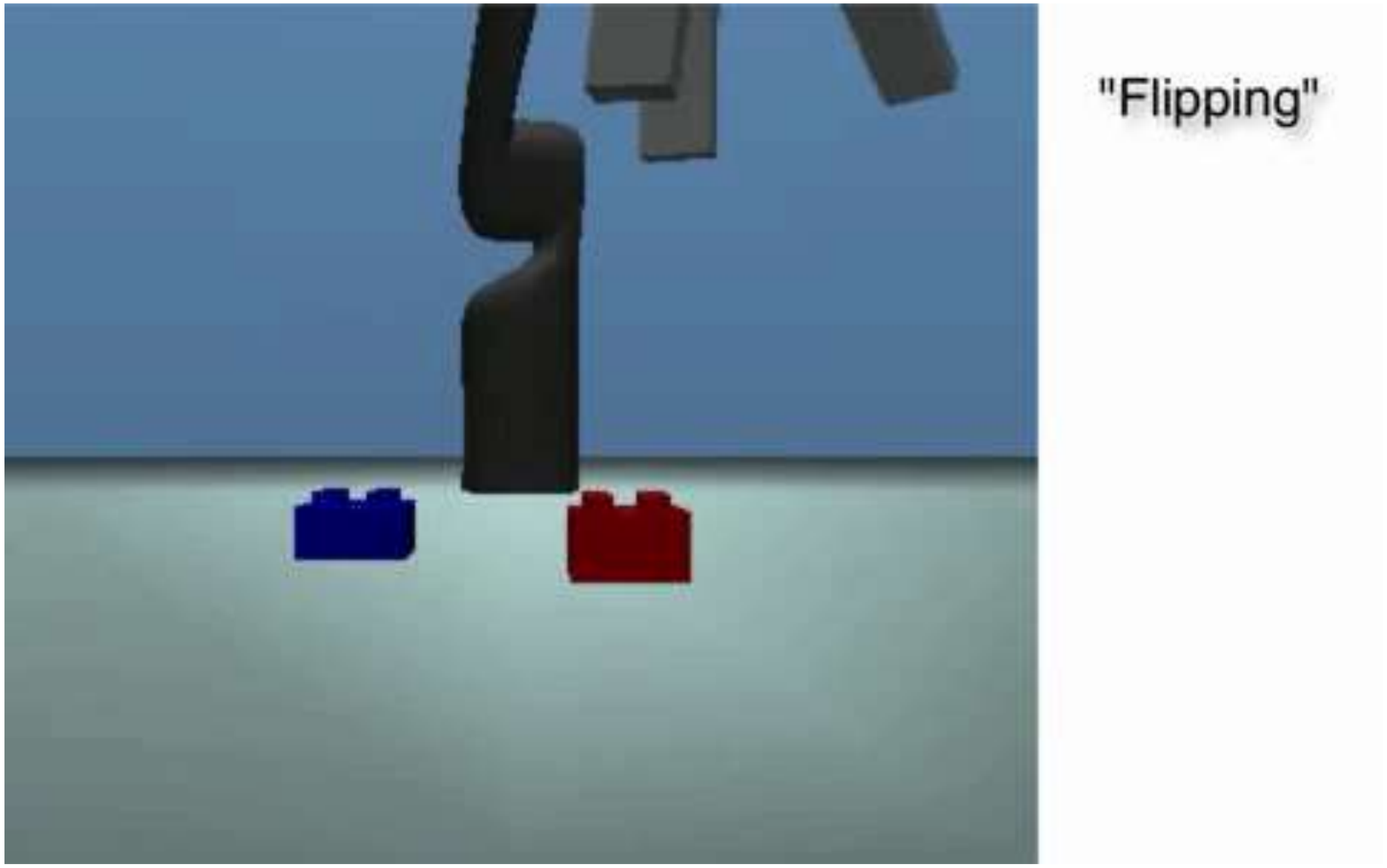
https://senseis.xmp.net/?Joseki


Malibu

RL Challenge #4: Reward Function Design

- You get what you incentivize! (a.k.a. “The Cobra Effect”)

RL Challenge #4: Reward Function Design



Reward Function Shaping

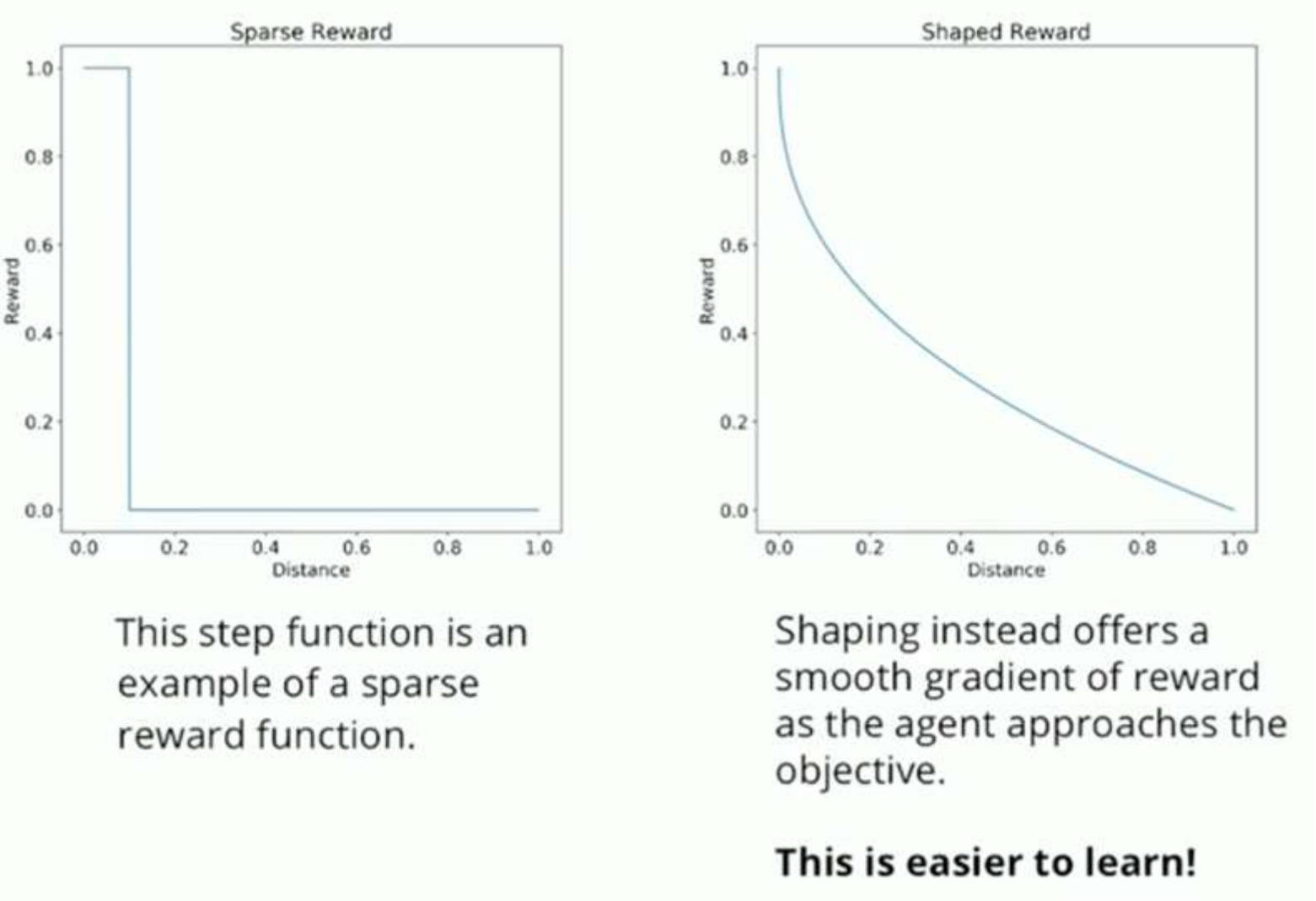


Image source: https://bons.ai/blog/deep-reinforcement-learning-enterprise

Recap: Main RL Challenges

- Representation
- Generalization
- Exploration / Exploitation
- Reward Function Design

Main RL Challenges… In the real world?

- Representation
- Generalization
- Exploration / Exploitation
- Reward Function Design
- Cost of acquiring experience
- Cost of failing

- Robotic Control
- Autonomous Vehicles
- IT Network Security
- Financial Trading
- Fleet Logistics
- Process Planning…

Microsoft

Thank You