Reinforcement Learning

上海科技大学
ShanghaiTech University

LIMA Lab
Living Machines Laboratory

Learning

WHAT IS LEARNING AND THE DIFFERENT KINDS OF LEARNING THAT WE HAVE

CAN GENETIC ALGORITHMS AND SIMULATED ANNEALING BE USED TO MAKE ROBOTS "LEARN"?

https://robotics.shanghaitech.edu.cn/node/205

A FEW WORDS ON EXPLORATION AND EXPLOITATION

Supervised Learning

Data: (x, y)
x is data, y is label

Goal: Learn a *function* to map x -> y

Examples: Classification, regression, object detection, semantic segmentation, image captioning, etc.

Cat

Classification

REMEMBER THE BACKPROPAGATION EXAMPLE?

Unsupervised Learning

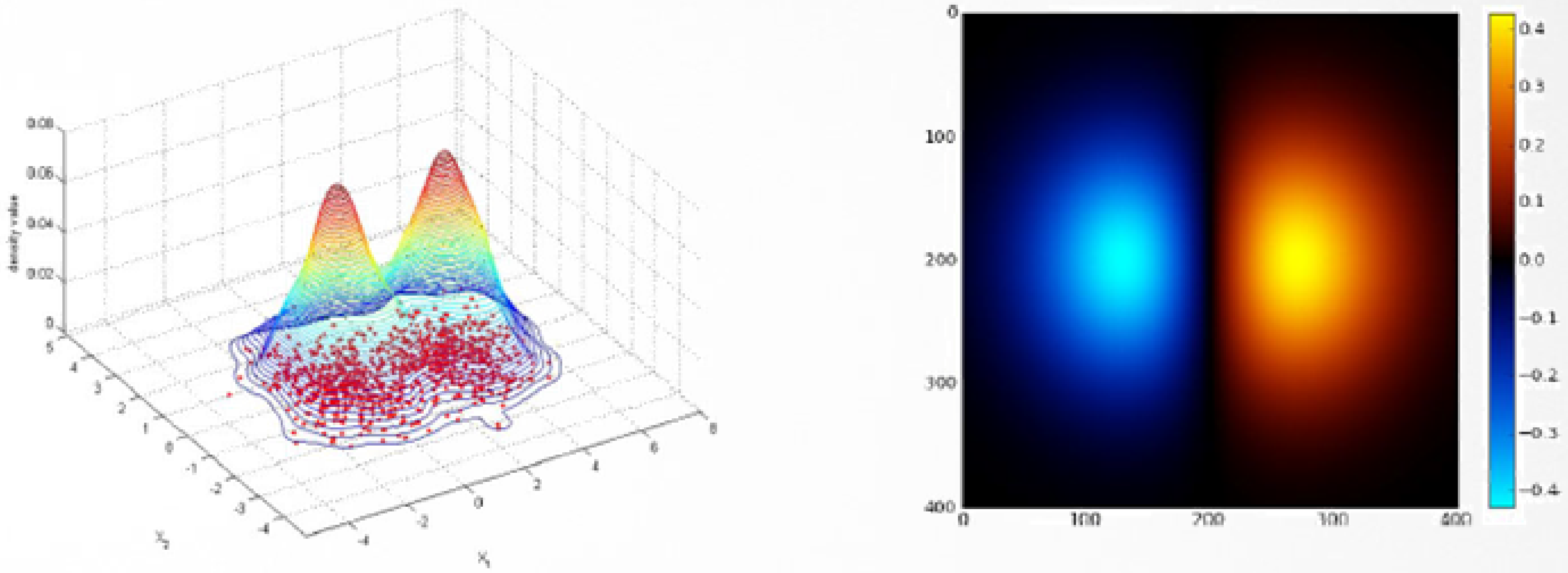
Data: x
Just data, no labels!

Goal: Learn some underlying hidden *structure* of the data

Examples: Clustering, dimensionality reduction, feature learning, density estimation, etc.

Figure copyright Ian Goodfellow, 2016. Reproduced with permission.

1-d density estimation

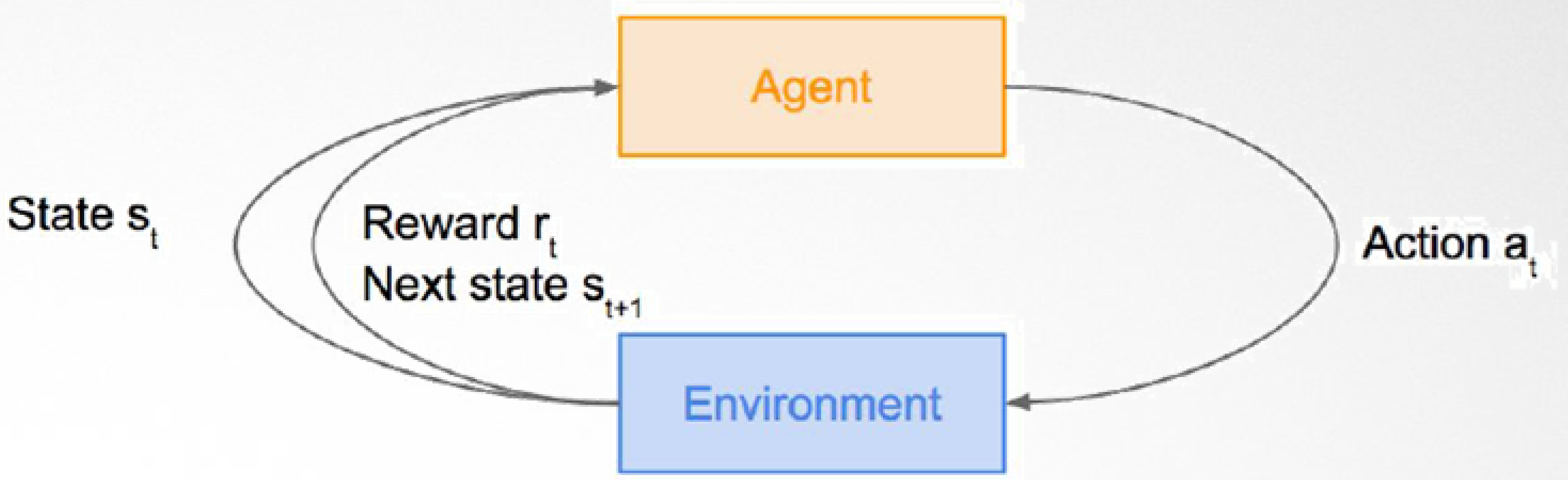


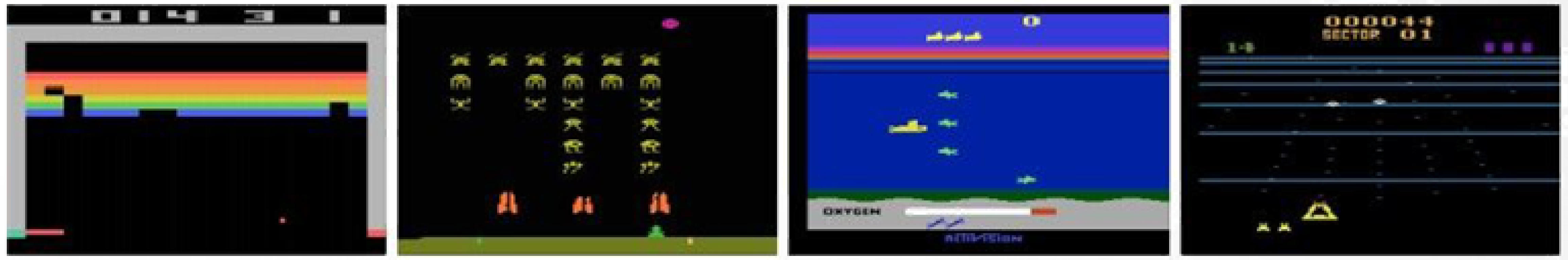
2-d density estimation

Reinforcement Learning

Problems involving an **agent** interacting with an **environment**, which provides numeric **reward** signals

Goal: Learn how to take actions in order to maximize reward





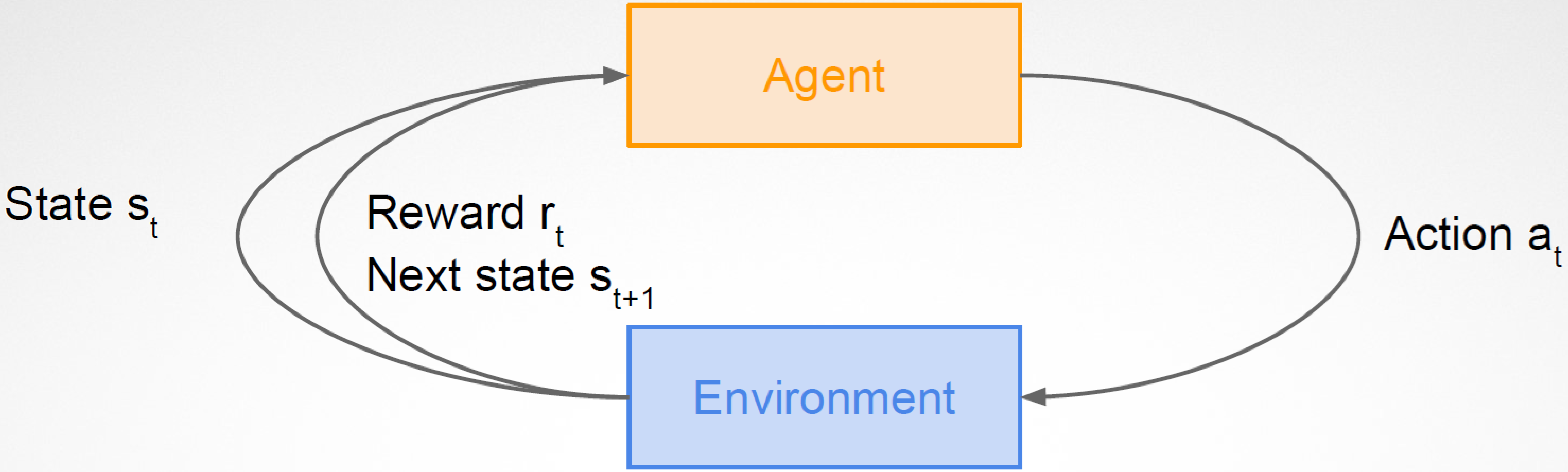
PERFECT FOR ROBOTICS, AS THEIR GOAL IS TO INTERACT WITH THE ENVIRONMENT

Behaviorism and Conditioning

A PSYCHOLOGY STUDY TO UNDERSTAND THE BEHAVIOR OF HUMANS AND ANIMALS.

IT EXPLAINS THAT OUR BEHAVIOR IS EITHER FROM REFLEXES PRODUCED BY A RESPONSE TO THE ENVIRONMENT, OR A CONSEQUENCE OF OUR LIFE'S HISTORY (REWARD OR PUNISHMENT)

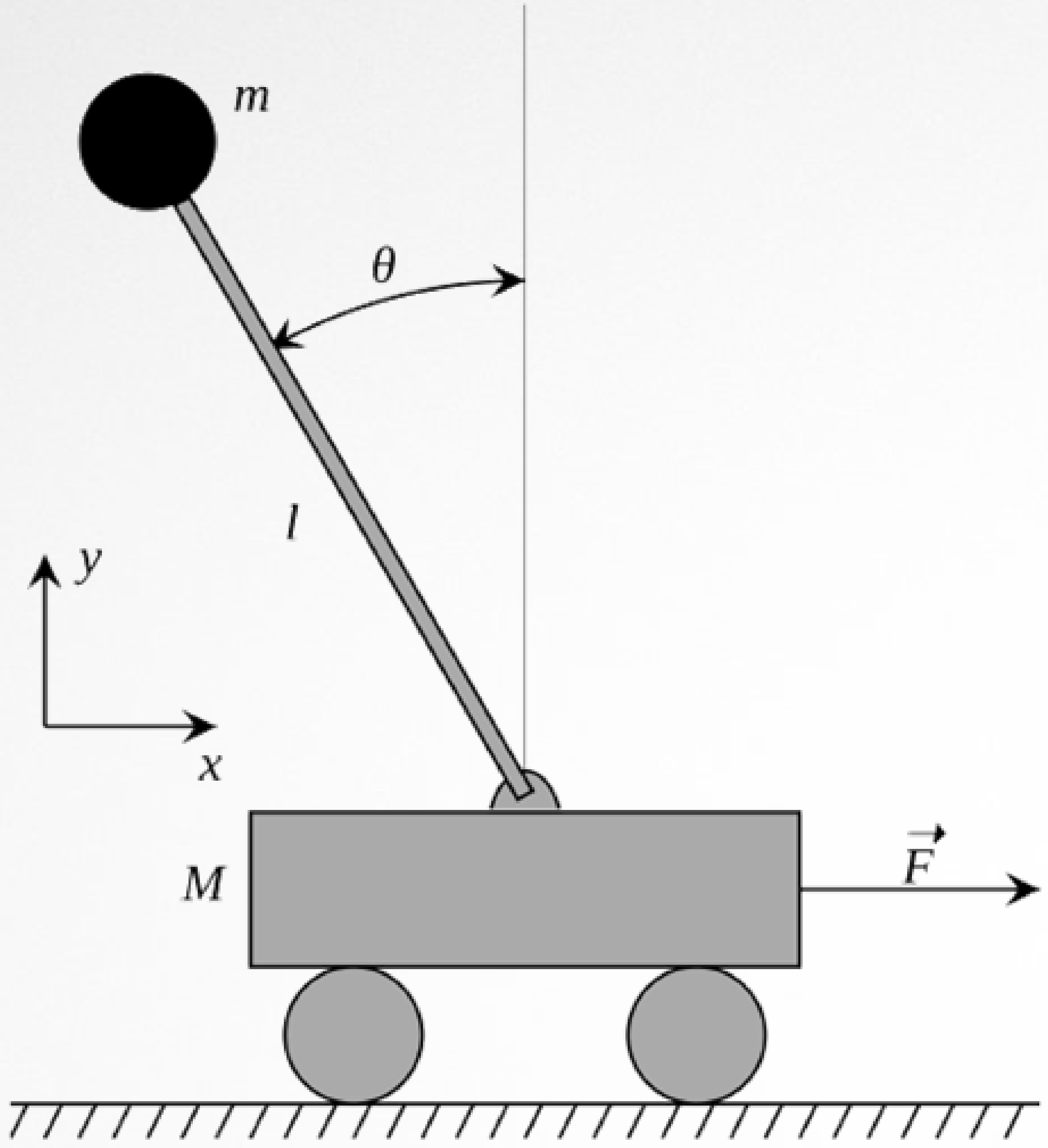
RL - Concept



THE ENVIRONMENT REWARDS THE AGENT WHEN SPECIFIC ACTIONS PLACE IT IN THE "RIGHT STATES"

HOW DOES THIS RELATE WITH THE "EVOLUTIONARY JACKAL" PROBLEM THAT WE JUST SAW?

Inverted Pendulum

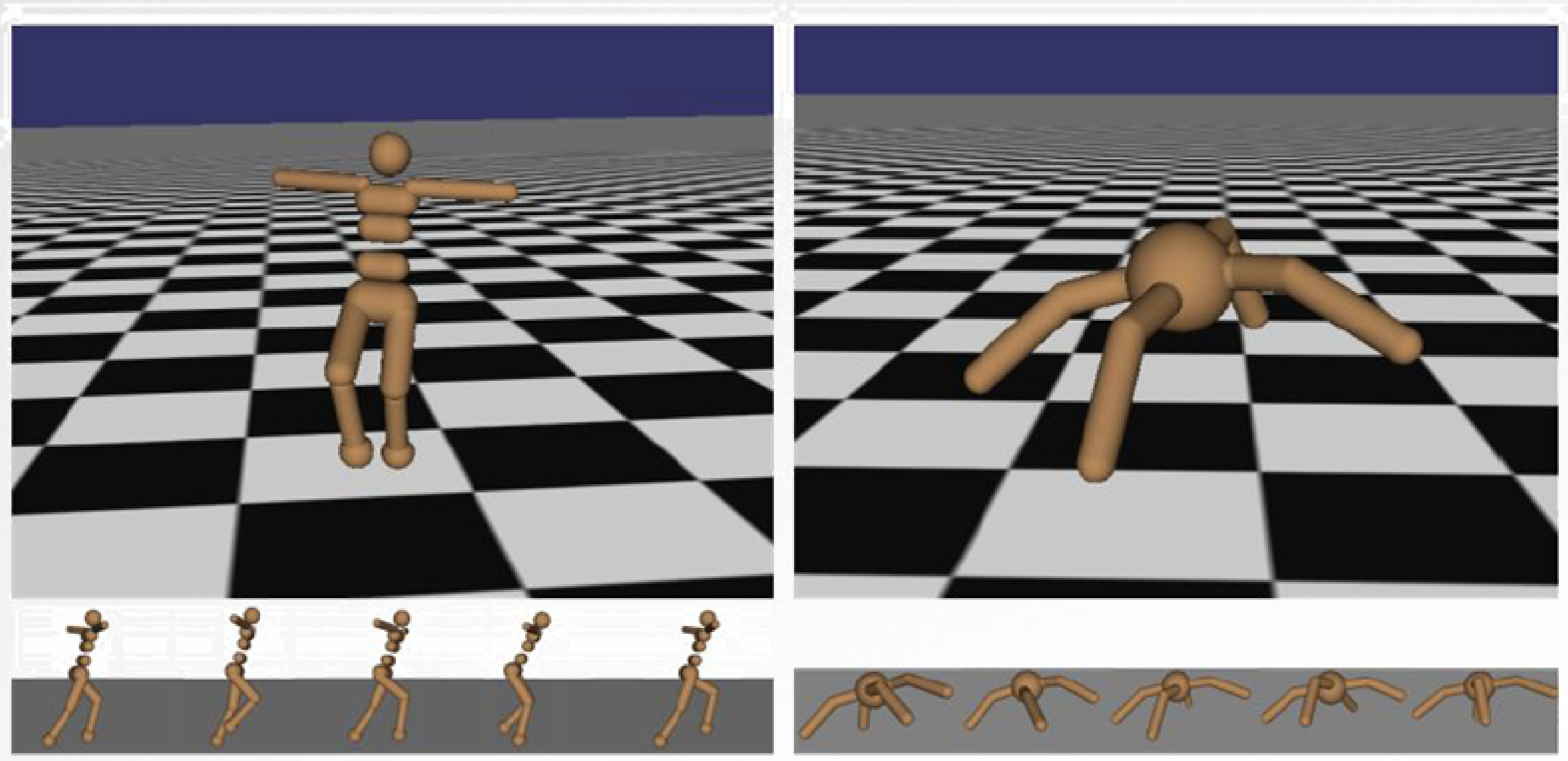


Objective: Balance a pole on top of a movable cart

State: angle, angular speed, position, horizontal velocity
Action: horizontal force applied on the cart
Reward: 1 at each time step if the pole is upright

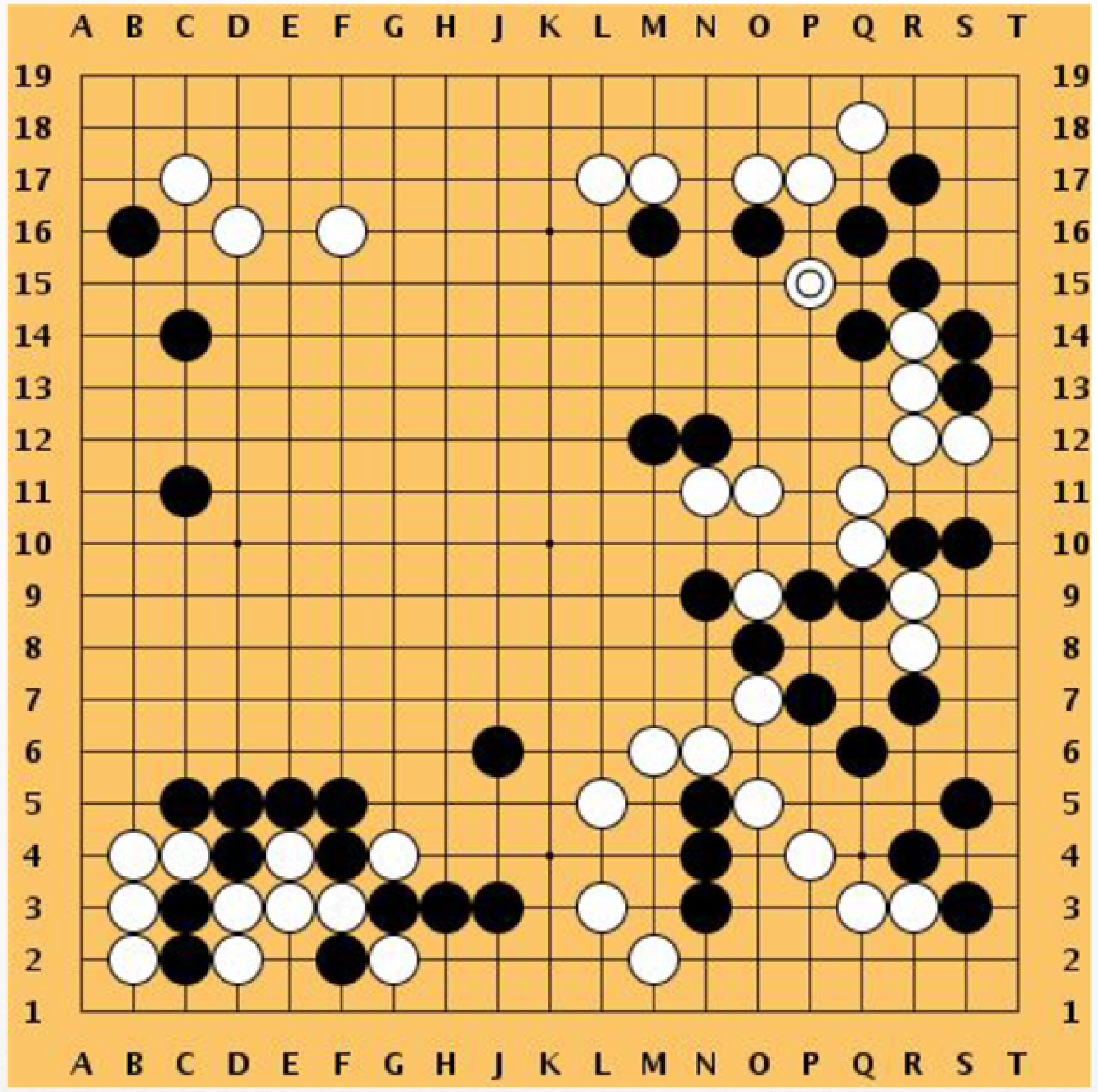
ONE EXAMPLE OF HOW THIS CAN BE DONE WITH AN INVERTED PENDULUM PROBLEM

Other famous problems



Objective: Make the robot move forward

State: Angle and position of the joints
Action: Torques applied on joints
Reward: 1 at each time step upright + forward movement



Objective: Win the game!

State: Position of all pieces
Action: Where to put the next piece down
Reward: 1 if win at the end of the game, 0 otherwise

Markov Decision Process

LIMA Lab
Living Machines Laboratory

- Mathematical formulation of the RL problem
- **Markov property**: Current state completely characterises the state of the world

Defined by: $(\mathcal{S}, \mathcal{A}, \mathcal{R}, \mathbb{P}, \gamma)$

$\mathcal{S}$: set of possible states
$\mathcal{A}$: set of possible actions
$\mathcal{R}$: distribution of reward given (state, action) pair
$\mathbb{P}$: transition probability i.e. distribution over next state given (state, action) pair
γ : discount factor

MDPs - 2

- At time step t=0, environment samples initial state $s_0 \sim p(s_0)$
- Then, for t=0 until done:
  - Agent selects action a_t
  - Environment samples reward $r_t \sim R(. \mid s_t, a_t)$
  - Environment samples next state $s_{t+1} \sim P(. \mid s_t, a_t)$
  - Agent receives reward r_t and next state s_{t+1}

- A policy **π** is a function from S to A that specifies what action to take in each state
- **Objective**: find policy **π*** that maximizes cumulative discounted reward: $\sum_{t \geq 0} \gamma^t r_t$

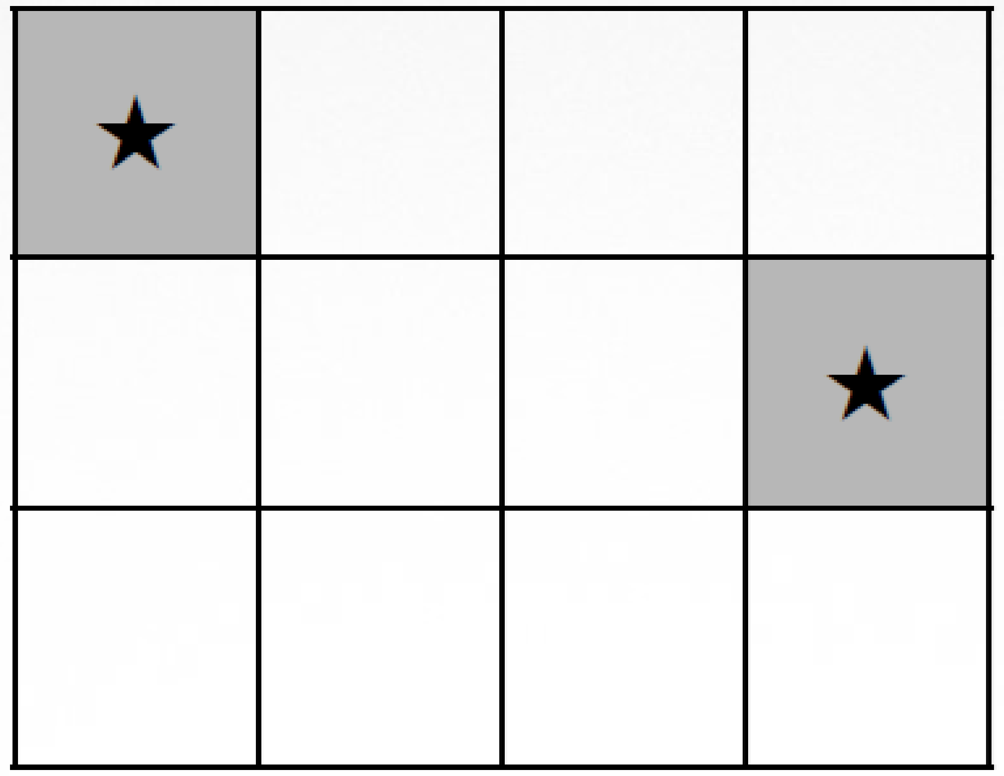
A grid world example

actions = {

1. right
2. left
3. up
4. down

}

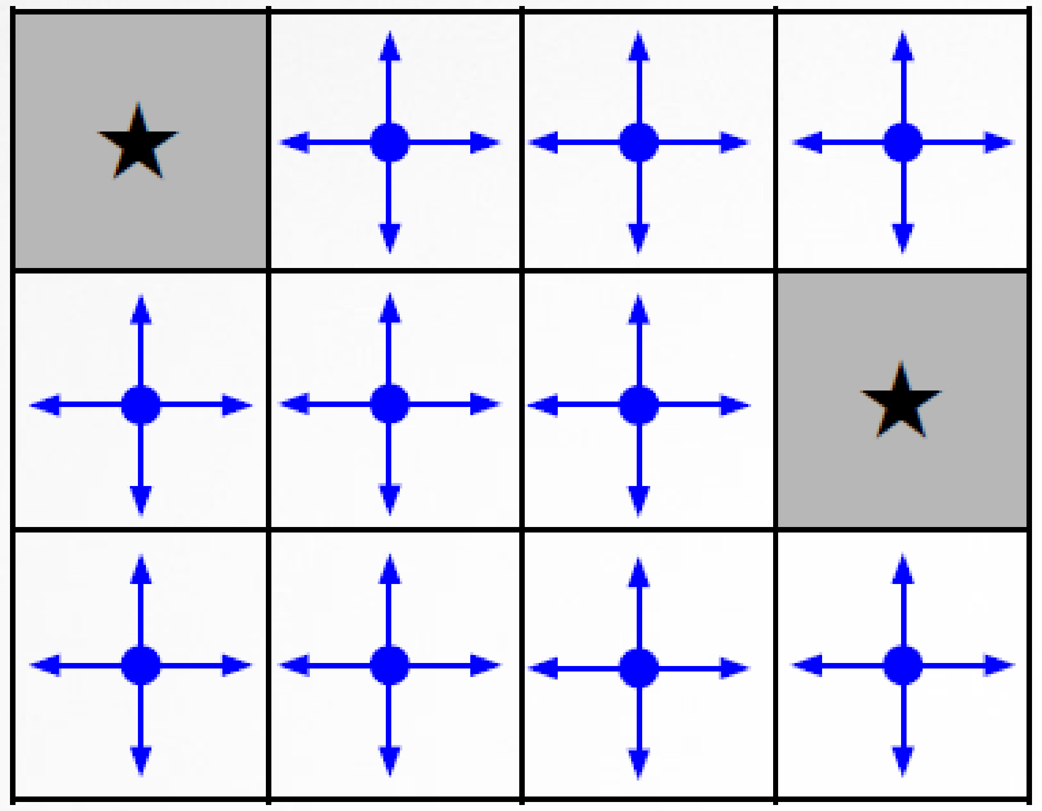
states



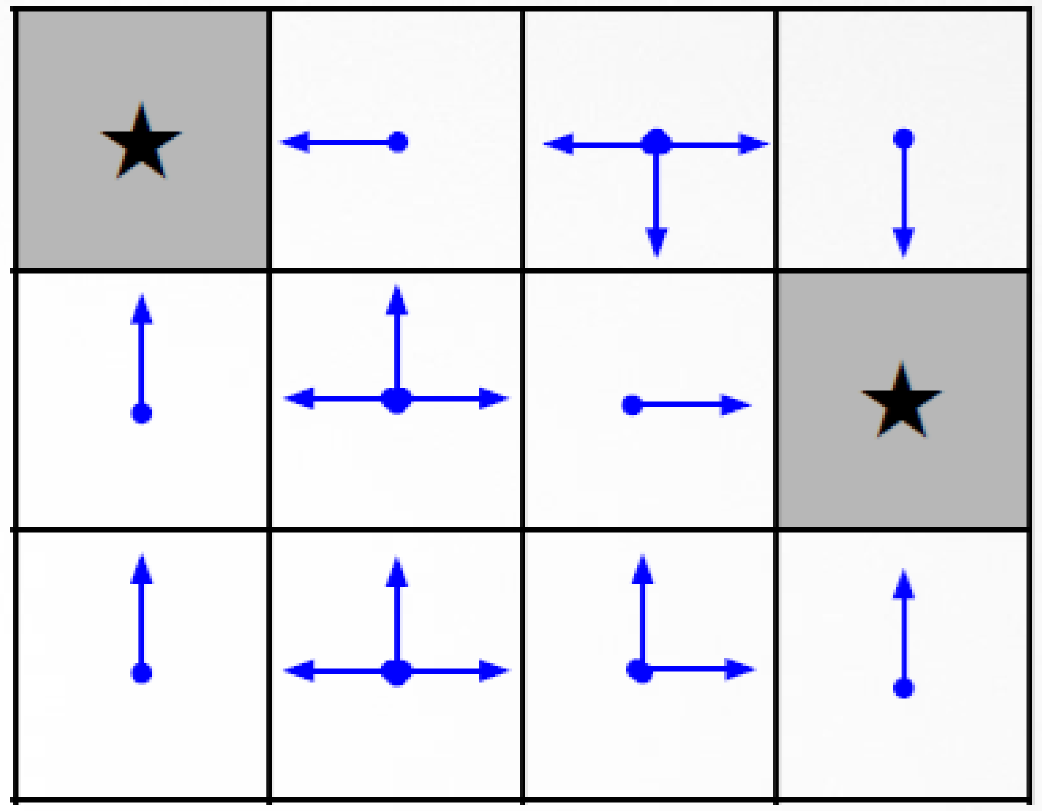
Set a negative "reward" for each transition (e.g. $r = -1$)

Objective: reach one of terminal states (greyed out) in least number of actions

Discovering the Optimal Policy



Random Policy



Optimal Policy

Defining the Optimal Policy

We want to find optimal policy **π*** that maximizes the sum of rewards.

How do we handle the randomness (initial state, transition probability…)?
Maximize the **expected sum of rewards!**

Formally: $\pi^* = \arg\max_{\pi} \mathbb{E}\left[\sum_{t\geq 0} \gamma^t r_t | \pi\right]$ with $s_0 \sim p(s_0), a_t \sim \pi(\cdot|s_t), s_{t+1} \sim p(\cdot|s_t, a_t)$

Value and Q-value functions

LIMA Lab
Living Machines Laboratory

Following a policy produces sample trajectories (or paths) $s_0, a_0, r_0, s_1, a_1, r_1, \ldots$

How good is a state?

The **value function** at state s, is the expected cumulative reward from following the policy from state s:

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

How good is a state-action pair?

The **Q-value function** at state s and action a, is the expected cumulative reward from taking action a in state s and then following the policy:

$$Q^{\pi}(s,a) = \mathbb{E}\left[\sum_{t\geq 0} \gamma^t r_t | s_0 = s, a_0 = a, \pi\right]$$

$$V^*(s) = \max_{a'} Q(s, a')$$

Bellman Equation

The optimal Q-value function Q* is the maximum expected cumulative reward achievable from a given (state, action) pair:

$$Q^*(s,a) = \max_{\pi} \mathbb{E}\left[\sum_{t\geq 0} \gamma^t r_t | s_0 = s, a_0 = a, \pi\right]$$

Q* satisfies the following **Bellman equation**:

$$Q^*(s,a) = \mathbb{E}_{s' \sim \mathcal{E}}\left[r + \gamma \max_{a'} Q^*(s',a') | s, a\right]$$

Intuition: if the optimal state-action values for the next time-step Q*(s',a') are known, then the optimal strategy is to take the action that maximizes the expected value of $r + \gamma Q^*(s',a')$

The optimal policy **π*** corresponds to taking the best action in any state as specified by Q*

A tabular example – Q Learning

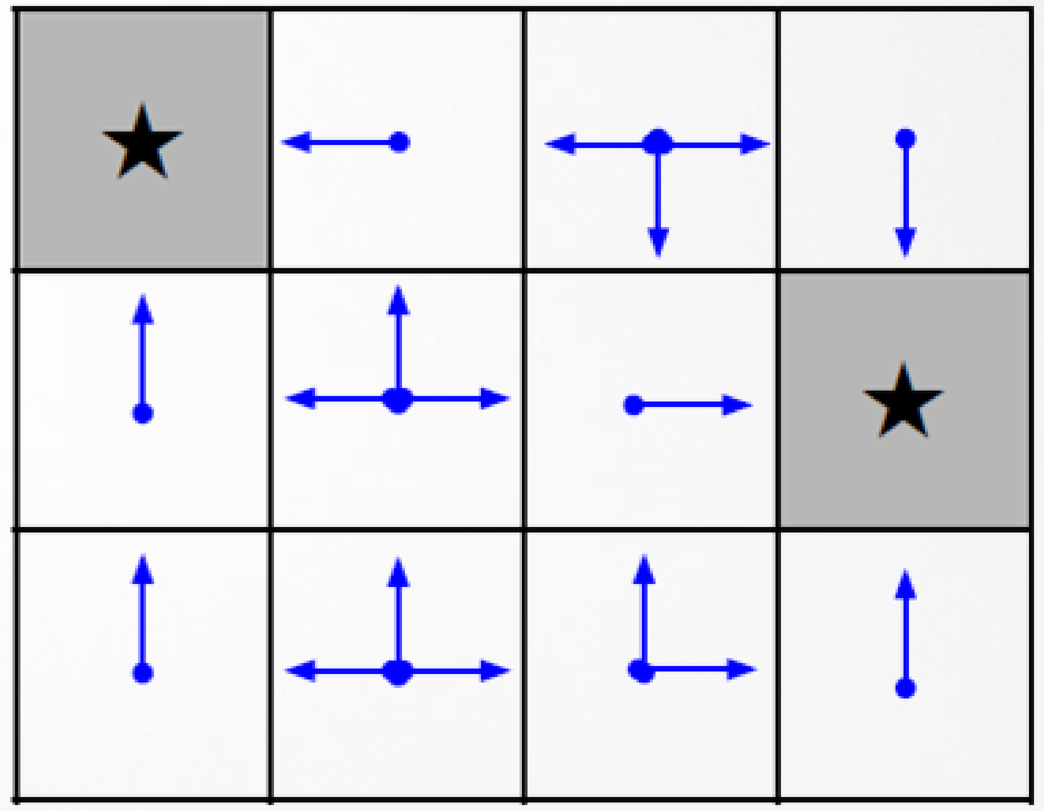
For each s, a initialize table entry $\hat{Q}(s,a) \leftarrow 0$

Observe current state s

Do forever:

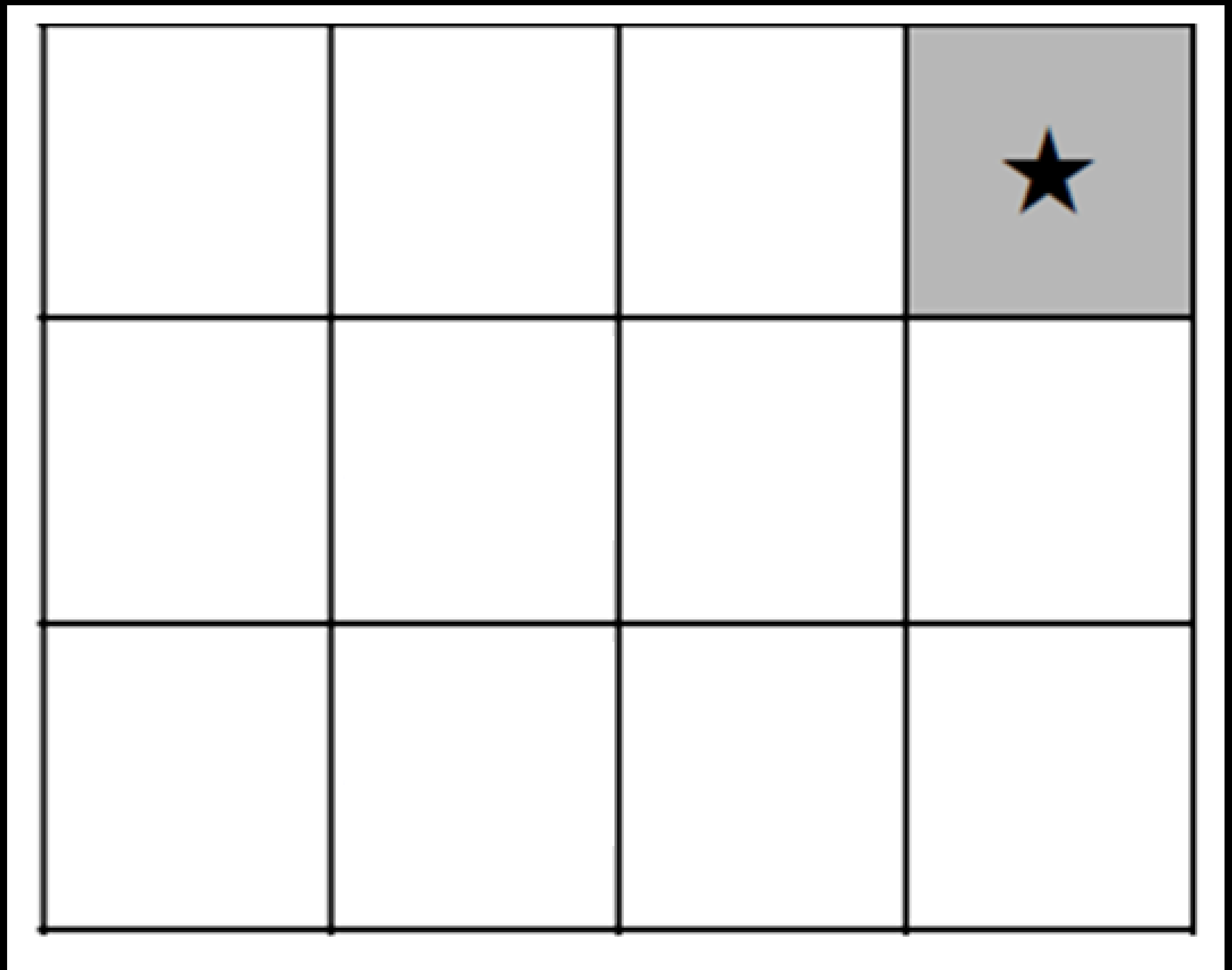
- Select an action a and execute it
- Receive immediate reward r
- Observe the new state s'
- Update the table entry for $\hat{Q}(s,a)$ as follows:

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



Optimal Policy

Example – Q Learning



Mountain Car example

LIMA Lab
Living Machines Laboratory