

# Learning to Act

## Robotic Vision Summer School 2026

Dana Kulić  
Monash University  
[dana.kulic@monash.edu](mailto:dana.kulic@monash.edu)

February 2026

<sup>1</sup>The material covered in this lecture is based on [David Silver's RL lectures at UCL](#), [Mario Martín's RL lectures at UPC](#), and [Sutton and Barton's Introduction to RL book](#). This material was developed in collaboration with Pamela Carreno-Medrano and Tin Tran.

# Overview of the Series

- ▶ Session 1: Introduction to Policy Learning
  - ▶ Motivation
  - ▶ Problem Formulation
  - ▶ Overview of Solution Approaches
- ▶ Session 2: Imitation Learning
  - ▶ Behaviour Cloning
- ▶ Session 3: Hands-On
  - ▶ Imitation and reinforcement learning in grid world
- ▶ Session 4: Multi-task learning
  - ▶ A high-level introduction to recent developments

# Introduction to Policy Learning

# Activity 0: Notebook Setup

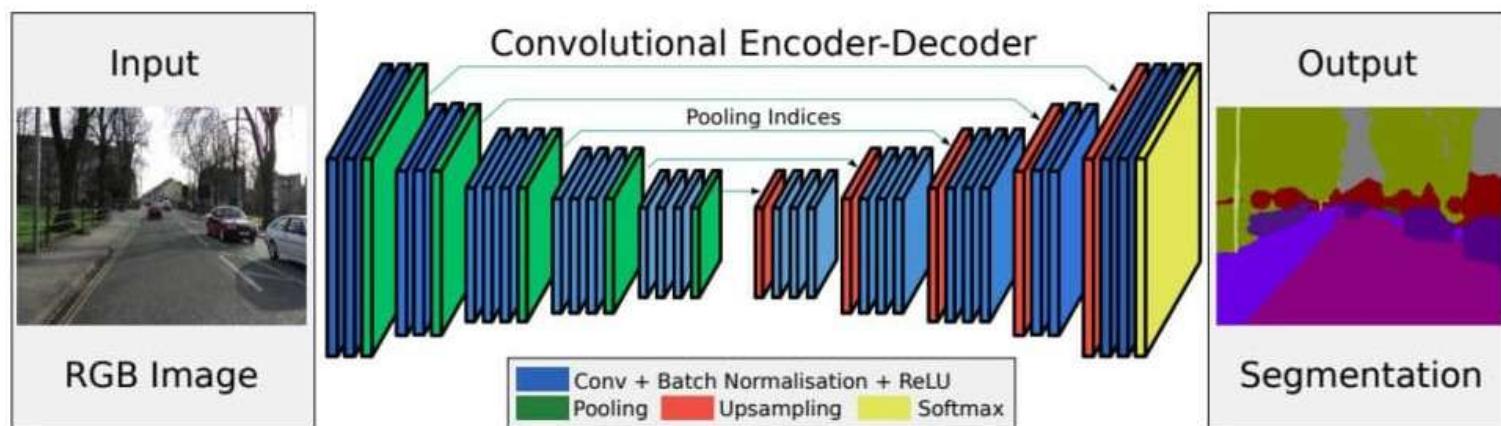
- ▶ Please open your Jupyter notebook environment and open the IntroBC.ipynb notebook in the Reinforcement\_Learning folder
- ▶ We will be making use of the Gymnasium toolkit:  
<https://github.com/Farama-Foundation/Gymnasium>

# Session 1 Overview

- ▶ Introduction to Policy Learning
- ▶ Problem formulation as a Markov Decision Process
- ▶ Overview of Solution Approaches

# From perception to action

- ▶ Robot vision allows the robot to perceive its environment

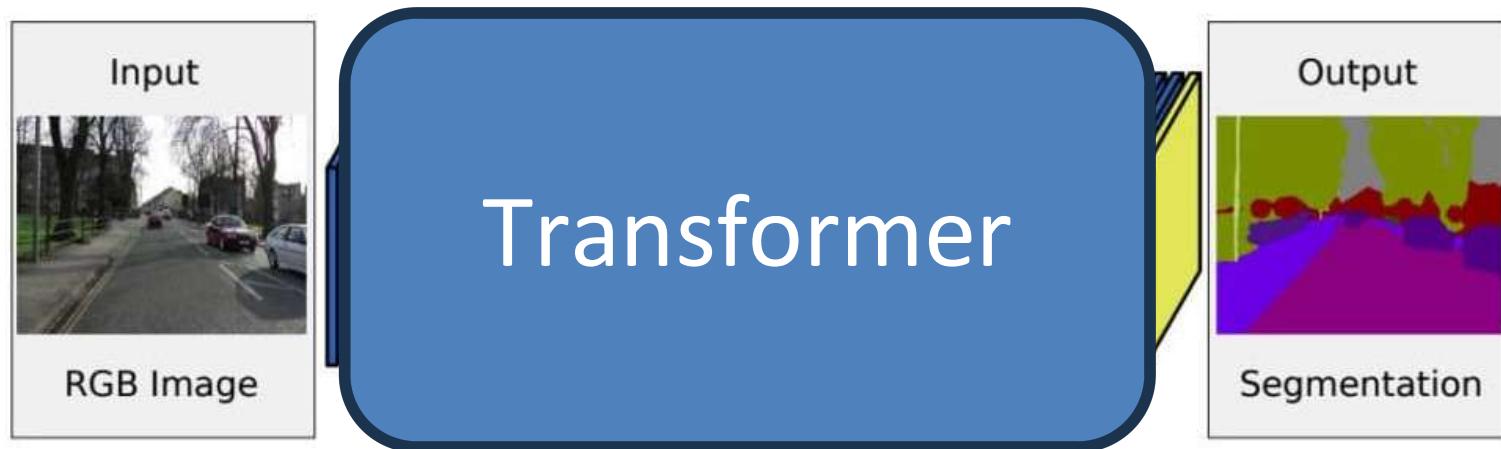


- ▶ Perception is a mapping from sensory data (e.g. pixels) to percepts (labels)

<sup>2</sup>Image courtesy of [Derrick Mwiti](#)

# From perception to action

- ▶ Robot vision allows the robot to perceive its environment



- ▶ Perception is a mapping from sensory data (e.g. pixels) to percepts (labels)
- ▶ This lecture: how do we map from sensory data to action

<sup>2</sup>Image courtesy of [Derrick Mwiti](#)

# Characteristics of robot action

Why take action?



[https://youtu.be/on9CZn\\_S3HA](https://youtu.be/on9CZn_S3HA)

# Characteristics of robot action

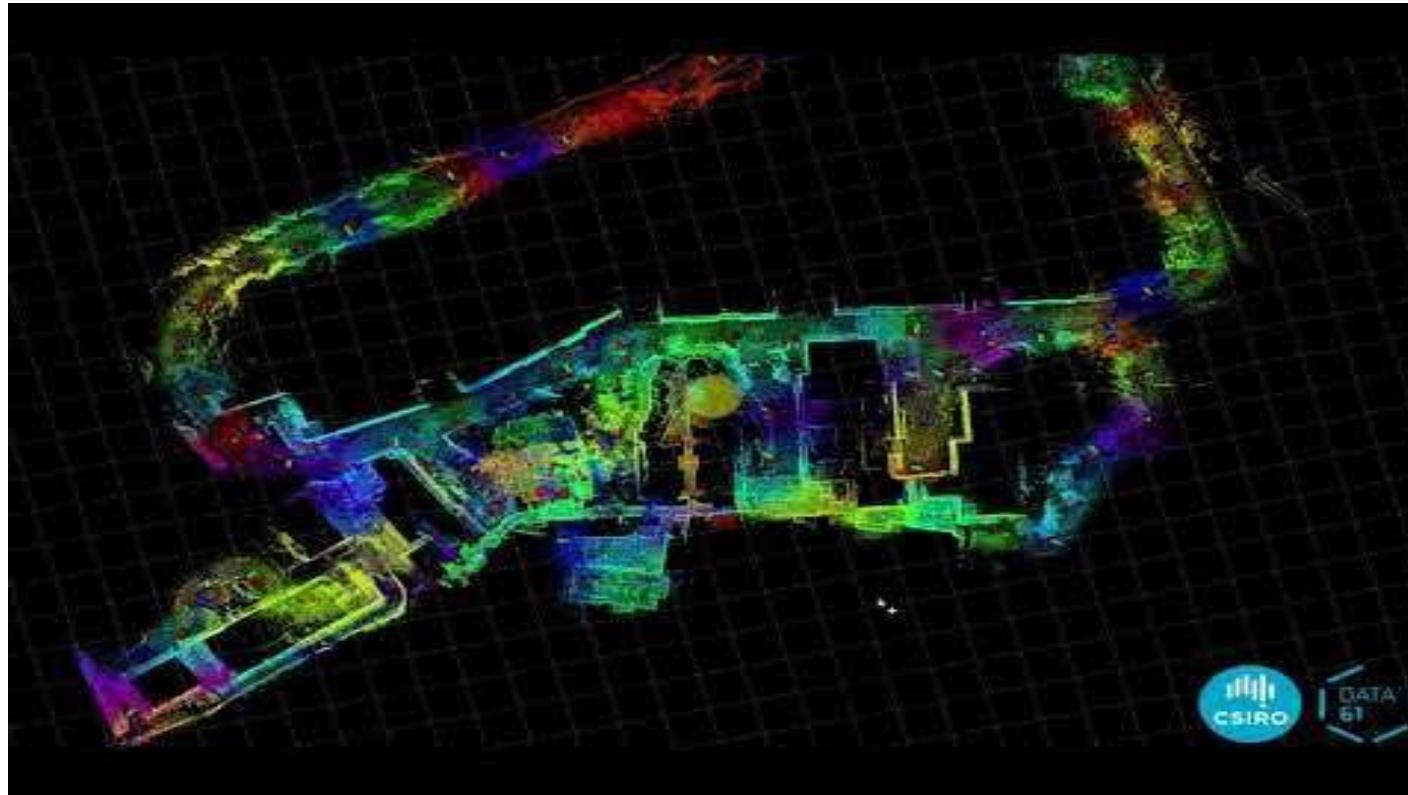
Why take action?



<https://youtu.be/daAvvAICE5o>

# Characteristics of robot action

Why take action?



<https://youtu.be/vaRZTjeldaQ>

# Characteristics of robot action

## Why take action?

- ▶ To accomplish (hopefully useful) tasks
- ▶ To improve perception
- ▶ To improve the robot's knowledge about the environment

# Characteristics of robot action

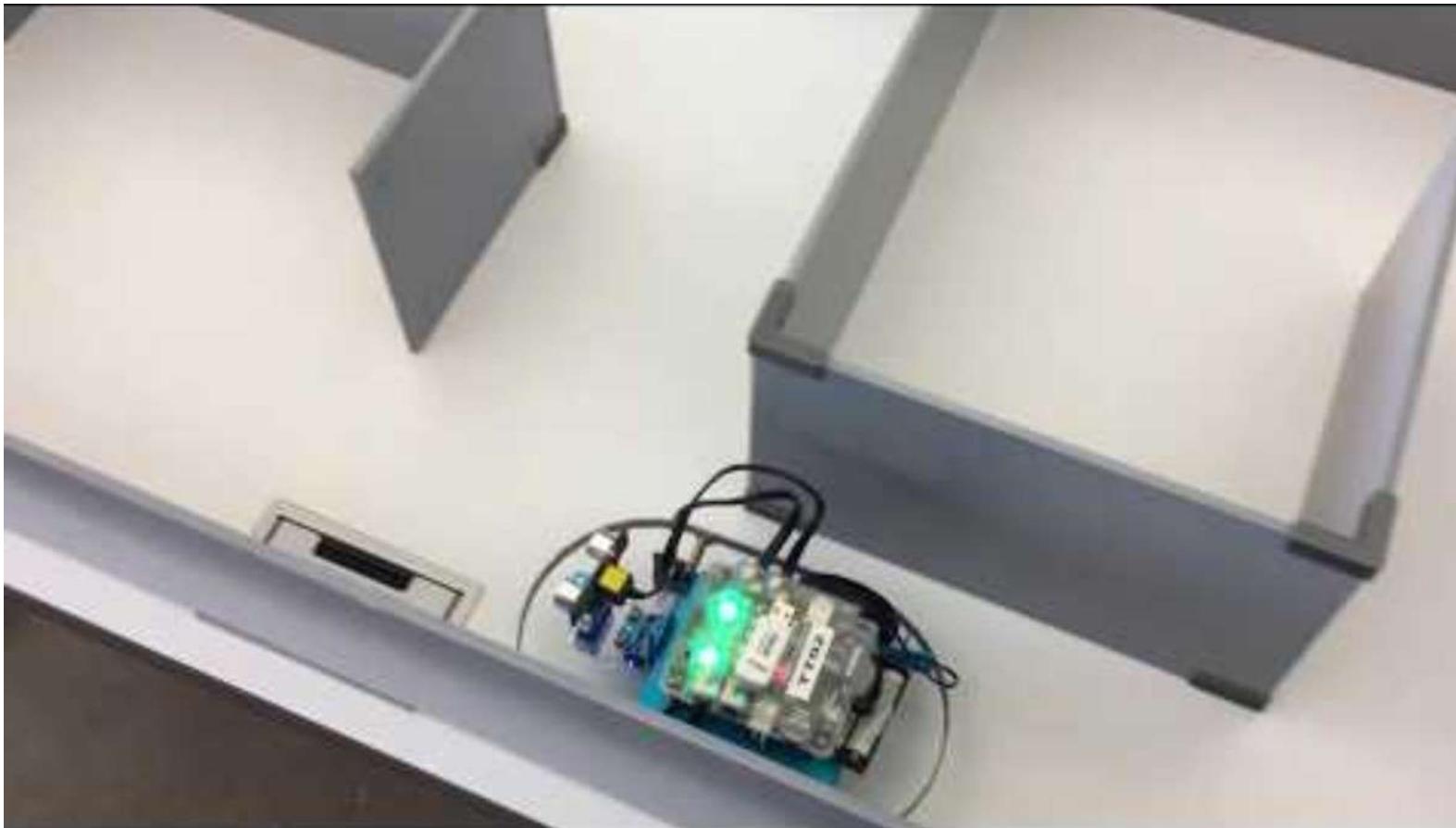
- ▶ Actions modify the world
- ▶ Actions can be costly



<https://www.youtube.com/watch?v=Mt-1smlom M>

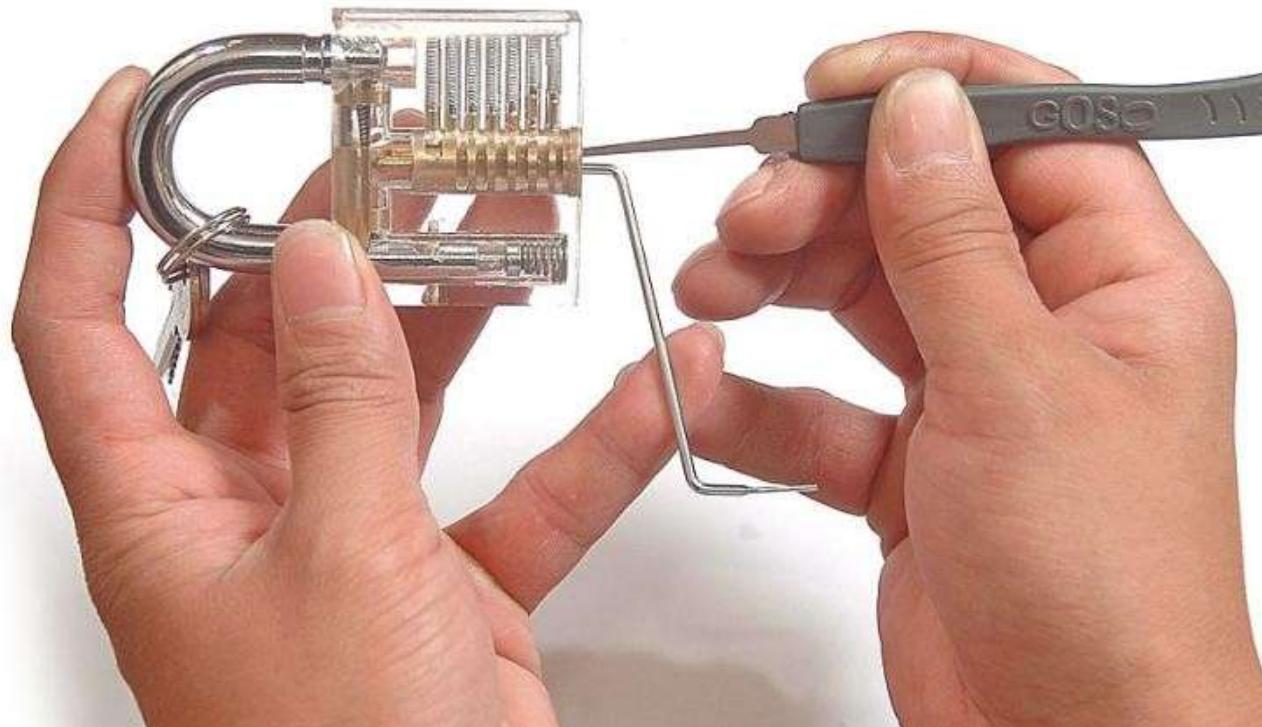
# Characteristics of robot action

- ▶ Consequences may not be immediately apparent



<https://youtu.be/xsSRP68nFK4>

# Action Selection



- ▶ Action Selection is a sequential decision-making problem

# Formalizing Sequential Decision-Making Problems

# Problem Formulation: MDP

- ▶ Markov Decision Processes (MDP) provide a general formalism for modeling and describing sequential decision-making.

# Problem Formulation: MDP

- ▶ Markov Decision Processes (MDP) provide a general formalism for modeling and describing sequential decision-making.
- ▶ MDPs are tuples  $\langle S, A, T, R, \gamma \rangle$  where
  - $S$  is a finite set of states
  - $A$  is a finite set of actions
  - $T$  is a transition function that defines the dynamics of the environment

$$T(s_t, a_t, s_{t+1}) = P(s_{t+1} | s_t, a_t)$$

# Problem Formulation: MDP

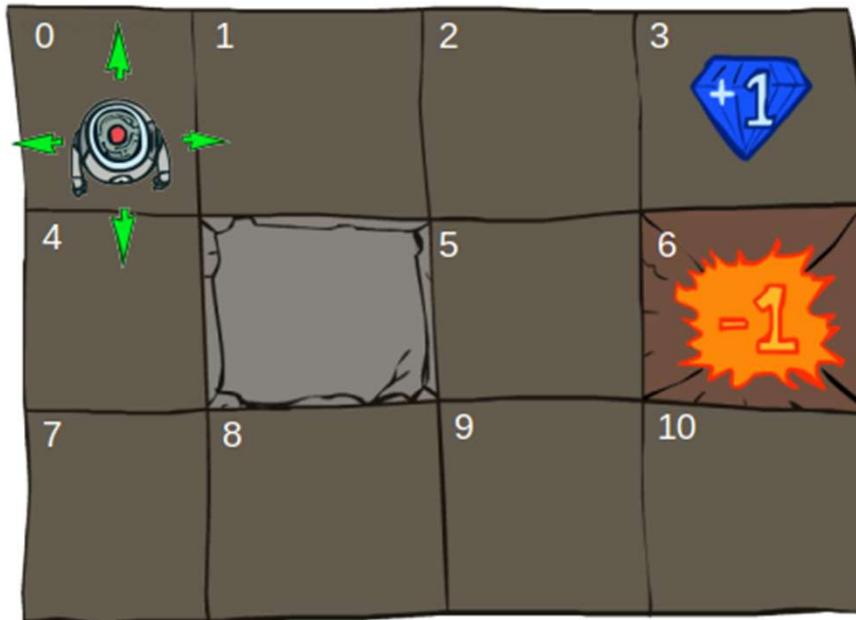
- ▶ Markov Decision Processes (MDP) provide a general formalism for modeling and describing sequential decision-making.
- ▶ MDPs are tuples  $\langle S, A, T, R, \gamma \rangle$  where
  - $S$  is a finite set of states
  - $A$  is a finite set of actions
  - $T$  is a transition function that defines the dynamics of the environment

$$T(s_t, a_t, s_{t+1}) = P(s_{t+1} | s_t, a_t)$$

- ▶ Why do we call this a *Markov* decision process?

# MDP Example (1)

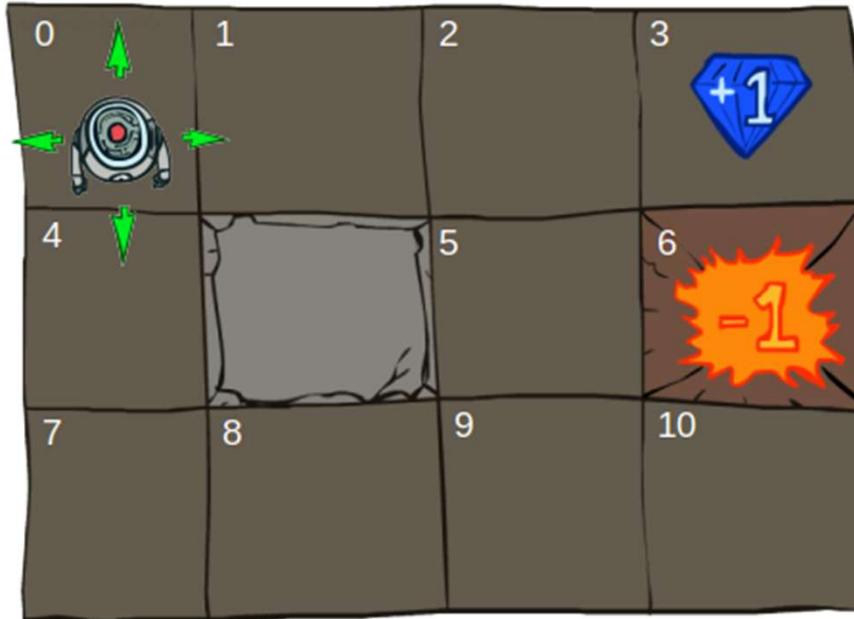
## Grid World



- $S$  = Cells of the grid with indexes  $\{0, 1, \dots, 10\}$
- $A$  = {up, down, left, right}

# MDP Example (1)

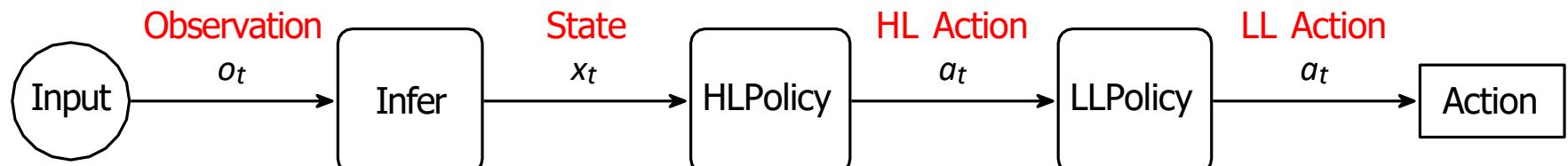
## Grid World



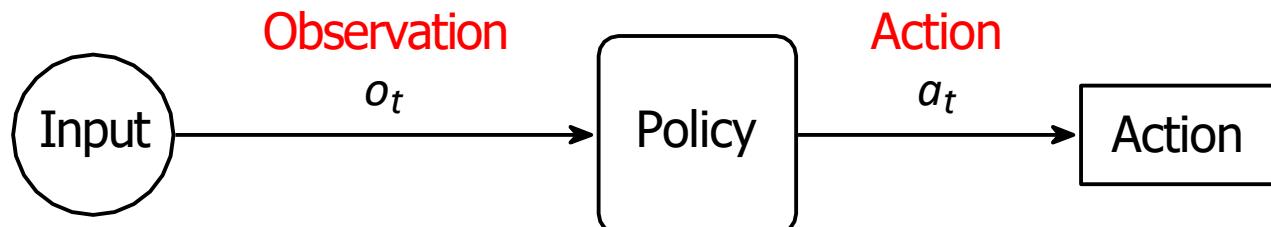
- ▶  $S = \text{Cells of the grid with indexes } \{0, 1, \dots, 10\}$ 
  - ▶ Is this the only choice for states?
- ▶  $A = \{\text{up, down, left, right}\}$ 
  - ▶ Is this the only choice for actions?

# Choices for Problem formulation

Modular



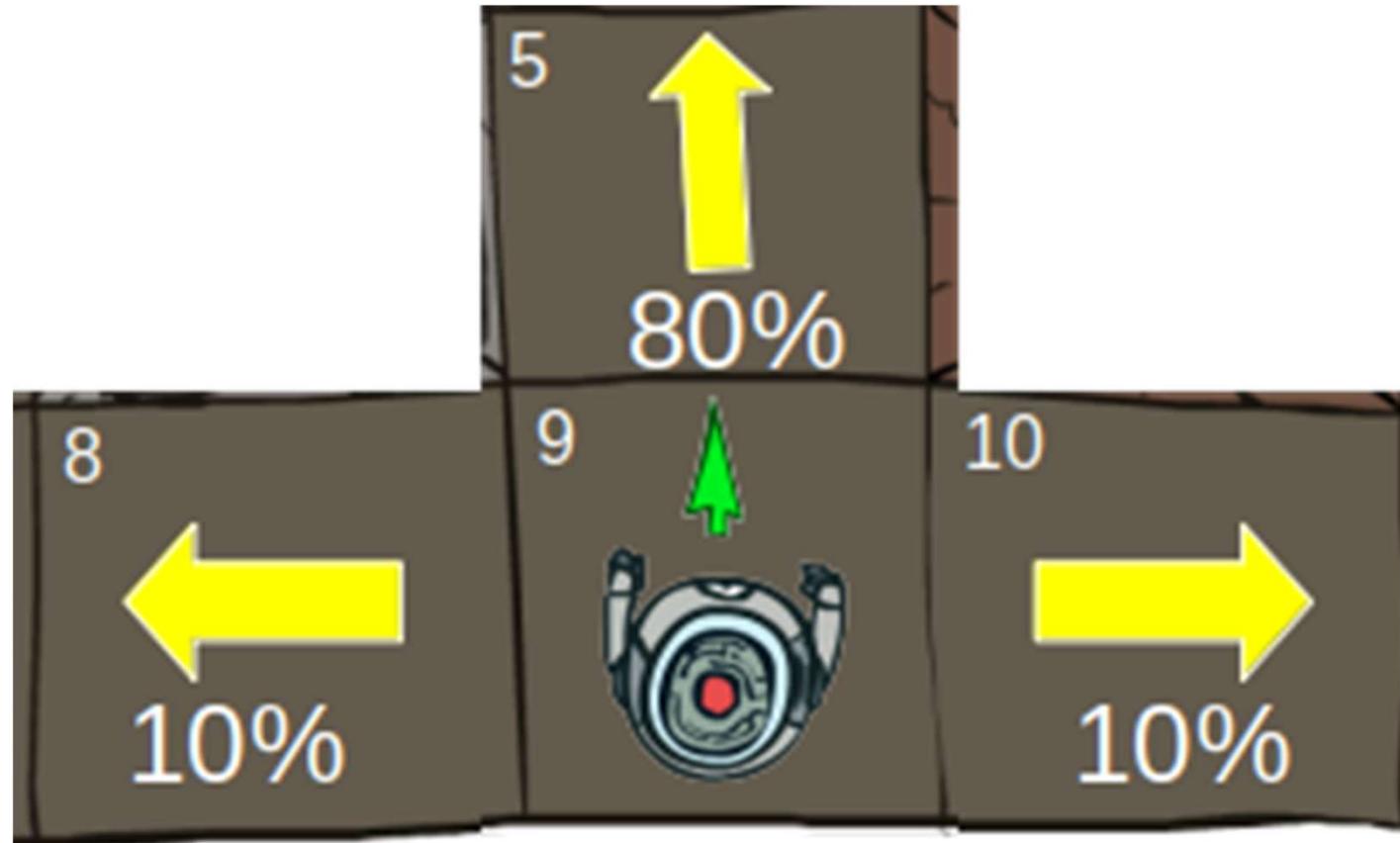
End-to-end



# MDP Example (2)

## Grid World

- ▶ State transitions are noisy



- ▶ Walls block the agent's path

# Problem Formulation: MDP

- ▶ Markov Decision Processes (MDP) provide a general formalism for modeling and describing sequential decision-making under uncertainty.
- ▶ MDPs are tuples  $\langle S, A, T, R, \gamma \rangle$  where
  - $S$  is a finite set of states
  - $A$  is a finite set of actions
  - $T$  is a transition function that defines the dynamics of the environment

$$T(s_t, a_t, s_{t+1}) = P(s_{t+1} | s_t, a_t)$$

- $R$  is a reward function

$$\mathcal{R}(s_t, a_t, s_{t+1}) = r_{t+1},$$

- $\gamma$  is a discount factor  $\gamma \in [0, 1]$  that defines the present value of future rewards

# Problem Formulation: MDP

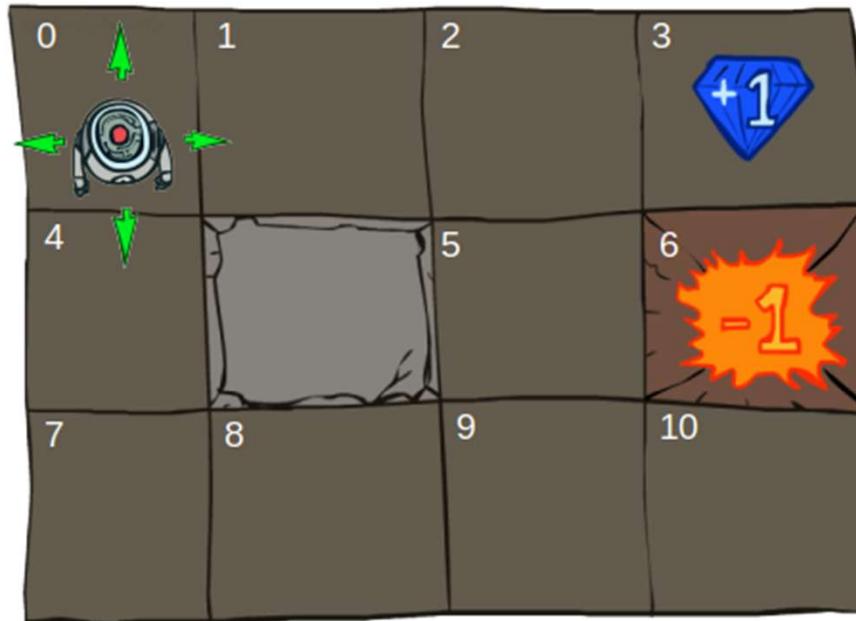
- ▶ Markov Decision Processes (MDP) provide a general formalism for modeling and describing sequential decision-making under uncertainty.
- ▶ MDPs are tuples  $\langle S, A, T, R, \gamma \rangle$  where
  - $S$  is a finite set of states
  - $A$  is a finite set of actions
  - $T$  is a transition function that defines the dynamics of the environment
$$T(s_t, a_t, s_{t+1}) = P(s_{t+1} | s_t, a_t)$$
  - $R$  is a reward function
  - $\gamma$  is a discount factor  $\gamma \in [0, 1]$  that defines the present value of future rewards
- ▶ What is the problem we are trying to solve with this formulation?

# Policies in Finite MDPs

- ▶ A policy is a mapping from states to actions
- ▶ Types of policies:
  - ▶ Deterministic policy: a function that takes state as input and outputs an action
$$a_t = \pi(s_t)$$
  - ▶ Stochastic policy: a distribution over actions given states
$$\pi(a|s) = P(a_t = a|s_t = s)$$
 (stochastic policy)
- ▶ Policies fully define the behaviour of an agent because they specify how to act in any state
- ▶ What are some difficulties we might encounter in trying to find a good policy?

# MDP Example (1)

## Grid World



# Activity 1: Elements of an MDP

- ▶ Let's jump into our Jupyter notebook environment and the IntroBC notebook in the Reinforcement\_Learning folder
- ▶ Take a look at Section 1. Elements of an MDP

# Policy Learning

# How to get the best policy?

# How to get the best policy?

- ▶ Replicate an expert's policy - imitation learning

# How to get the best policy?

- ▶ Replicate an expert's policy - imitation learning
- ▶ Learn from trial and error - reinforcement learning

# How to get the best policy?

- ▶ Replicate an expert's policy - imitation learning
- ▶ Learn from trial and error - reinforcement learning
- ▶ Optimal control

# Optimal Control

Given a **known** model of the transition function  $f(x, u)$ ,

$$x_{t+1} = f(x_t, u_t)$$

and a **known** cost function  $J(x)$ , as an example

$$J(x, u) = w_x x^2 + w_u u^2$$

Solve the optimisation problem:

$$\min_{u_1 \dots T} \sum_{t=0}^T J(x_t, u_t)$$

$$s.t. x_{t+1} = f(x_t, u_t), x_0 = x_s$$

# Imitation Learning

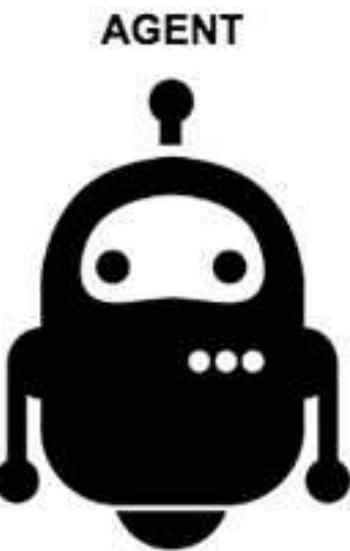
(specifically, Behavioural Cloning)

- ▶ Collect data from demonstration episodes  $D(e_{1:N})$
- ▶ Each episode is a sequence of states and actions  
 $e_i = (s_0, a_1, s_1, a_2, \dots, s_T)$
- ▶ Learn a policy  $\phi(s)$  using supervised learning:

$$L = (a_D(s) - \phi(s))^2$$

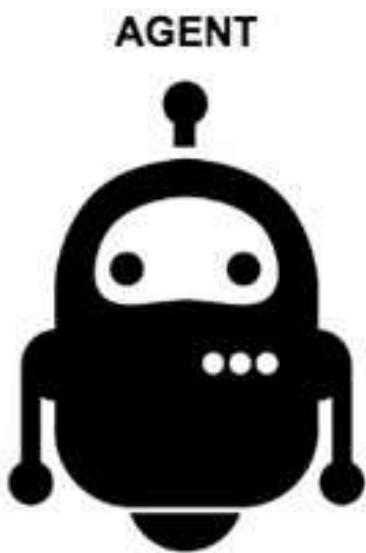
- ▶ The state  $s$  corresponds to the input data
- ▶ The action  $a$  corresponds to the label
- ▶ Behavioural cloning learns the policy function  $\phi(s)$  to minimise the difference between the estimated action and the observed expert action from each state

# Reinforcement Learning



<sup>3</sup>Images courtesy of [Lilian Weng](#)

# Reinforcement Learning

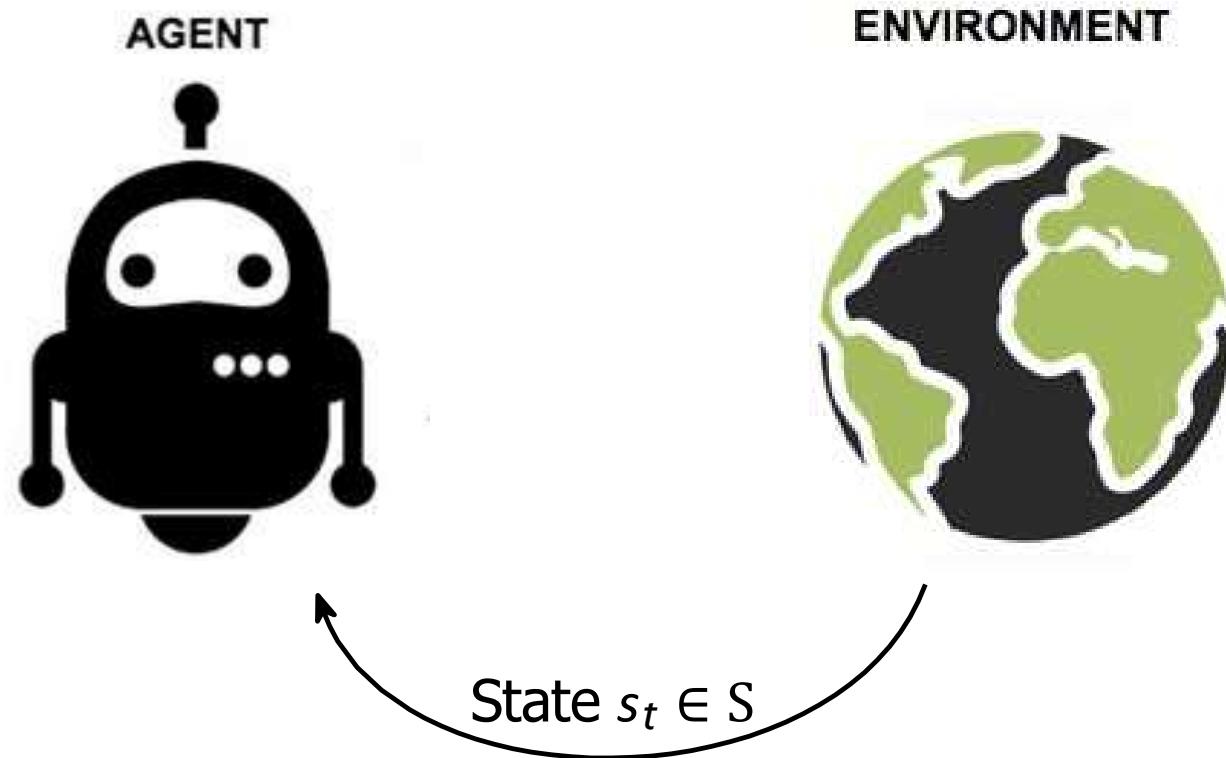


ENVIRONMENT



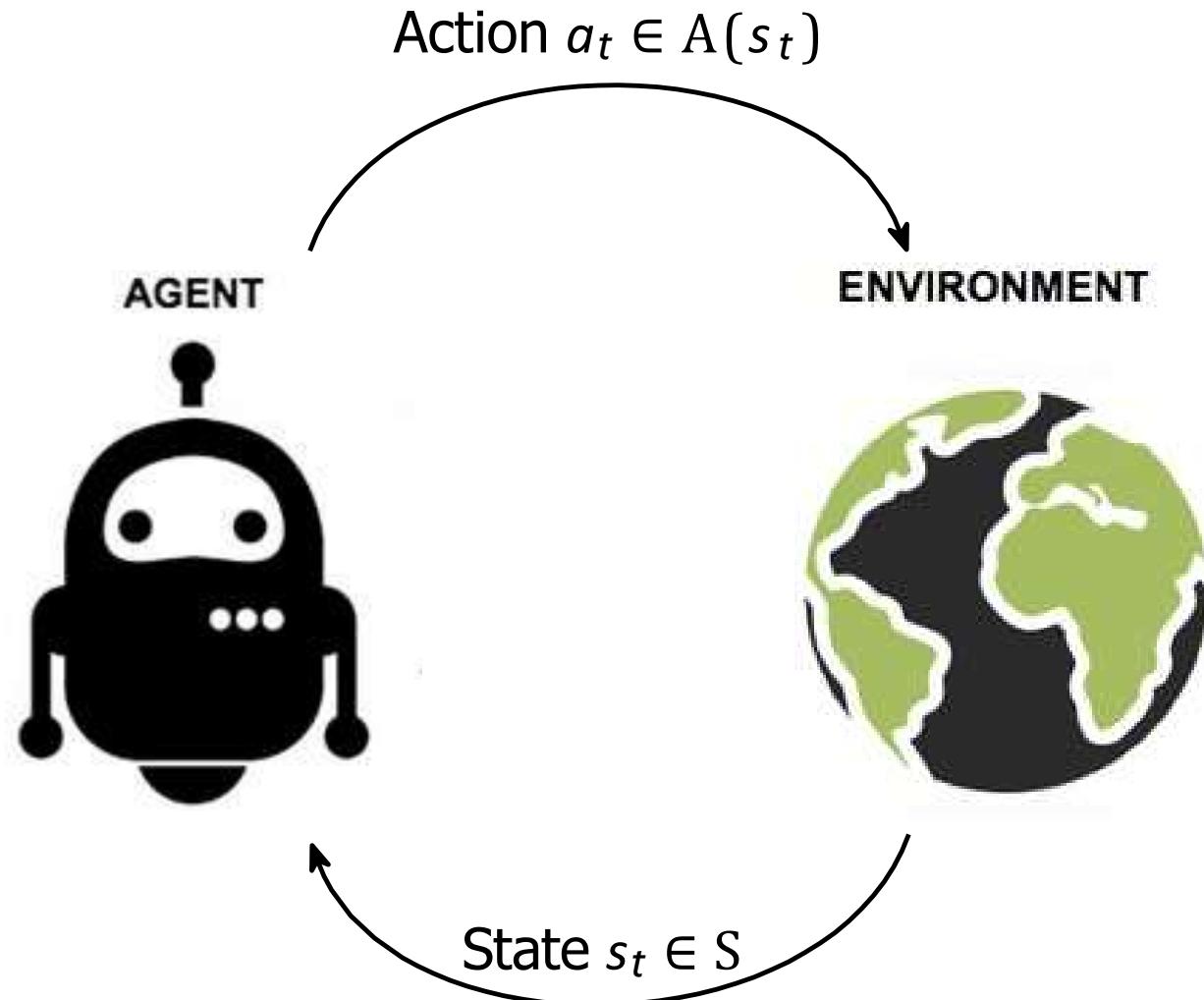
<sup>3</sup>Images courtesy of [Lilian Weng](#)

# Reinforcement Learning



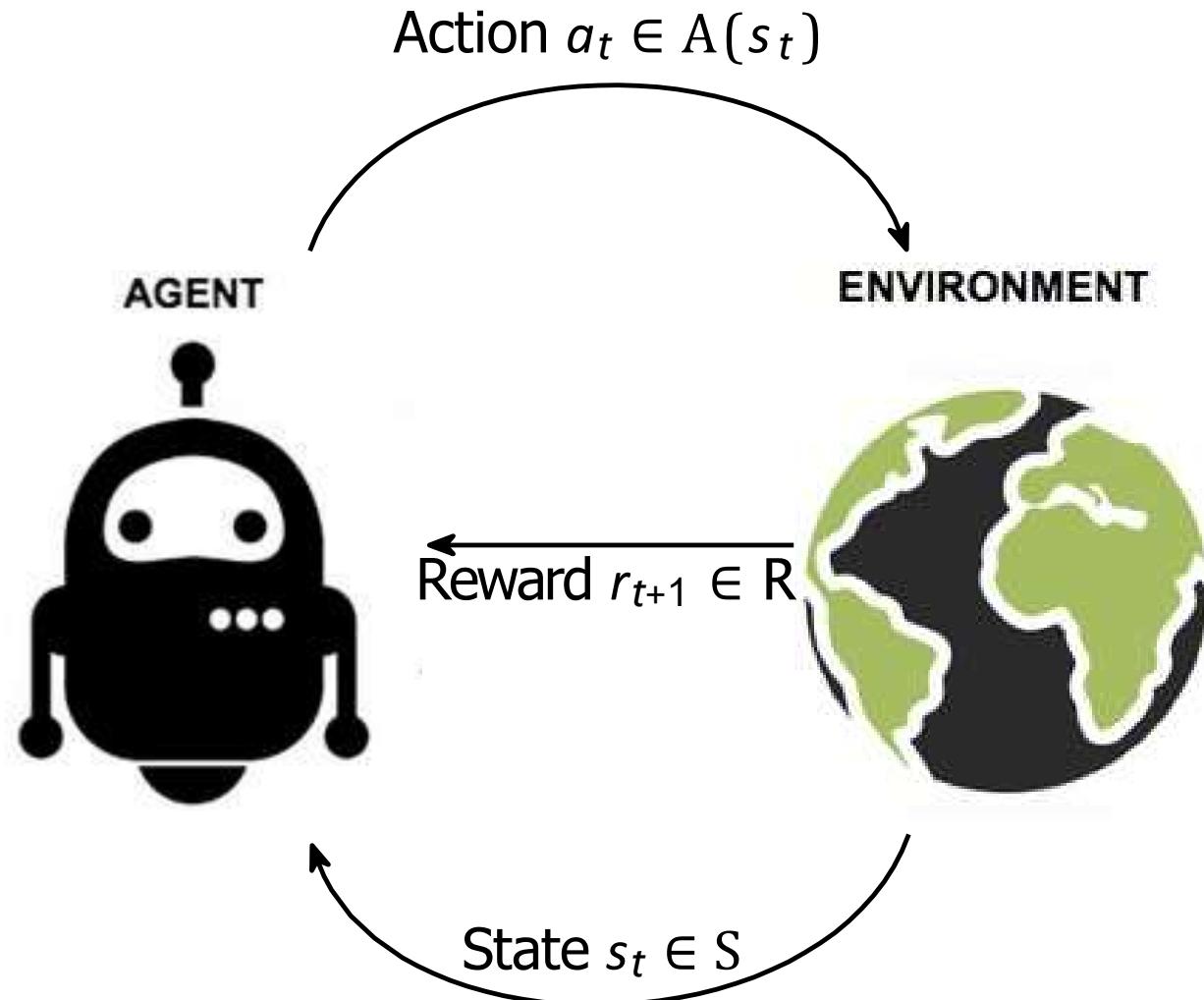
<sup>3</sup>Images courtesy of [Lilian Weng](#)

# Reinforcement Learning



<sup>3</sup>Images courtesy of [Lilian Weng](#)

# Reinforcement Learning



<sup>3</sup>Images courtesy of [Lilian Weng](#)

# Policy Finding Requirements

Method	Transition Function	Reward Function	Expert Demos
Optimal Control			
Reinforcement Learning			
Imitation Learning – IRL			
Imitation Learning – BC			

# Policy Finding Requirements

Method	Transition Function	Reward Function	Expert Demos
Optimal Control	Known	Known	No
Reinforcement Learning	Known or Unknown	Known	No
Imitation Learning – IRL	Known or Unknown	Unknown	Yes
Imitation Learning – BC	Known or Unknown	Implicit	Yes

# Session summary

- ▶ Problems of action selection can be formulated as Markov Decision Processes
  - ▶ Designer needs to formulate the state and action space
- ▶ Our objective is to find the ***policy***: a mapping from state to action
- ▶ Three common approaches:
  - ▶ Optimisation
  - ▶ Reinforcement Learning
  - ▶ Imitation Learning

# Next Lecture

- ▶ A closer look at imitation learning
- ▶ A brief intro to reinforcement learning