Reinforcement Learning

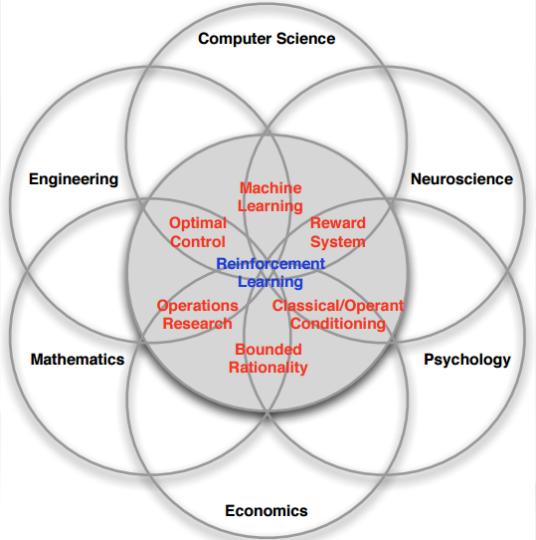
an introduction

Sindre B. Remman & Thomas N. Larsen

NTNU

TABLE OF CONTENTS

What is RL?


Computer Science
Engineering
Neuroscience
Machine Learning
Optimal Control
Reward System
Reinforcement Learning
Operations Research
Classical/Operant Conditioning
Bounded Rationality
Mathematics
Psychology
Economics

What is RL?

Use cases:

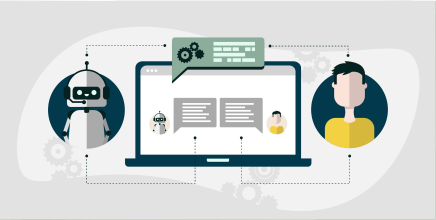
- AlphaGo

What is RL?

Use cases:

- AlphaGo
- Autonomous Cars

What is RL?

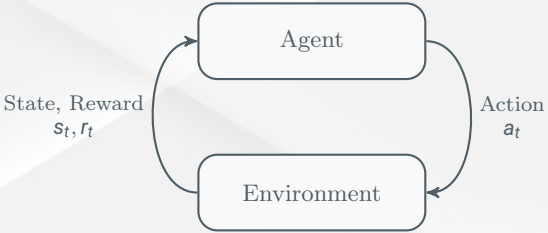


Use cases:

- AlphaGo
- Autonomous Cars
- Natural Language Processing

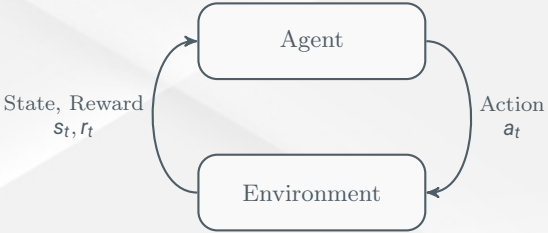
Agents and environments

- Agent
- Environment

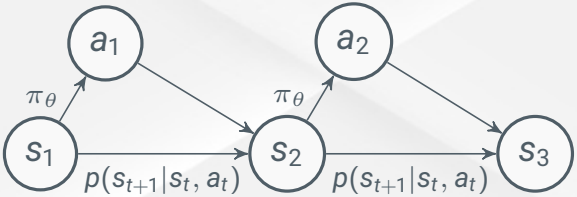


Agents and environments

- Agent
  - Learner and decision-maker.
- Environment
  - Everything outside the agent that the agent interacts with



Markov Decision Processes (MDPs)



Markov Decision Process

The RL objective

$$\theta^* = \arg\max_\theta \mathbb{E}_{\tau \sim p_\theta(\tau)} \left[\sum_t r(s_t, a_t) \right]$$

- Policy Gradient methods

The RL objective

$$\theta^* = \arg\max_{\theta} \mathbb{E}_{\tau \sim p_\theta(\tau)} \left[\sum_t r(s_t, a_t) \right]$$

- Policy Gradient methods
- Actor-Critic methods

The RL objective

$$\theta^* = \arg\max_\theta \mathbb{E}_{\tau \sim p_\theta(\tau)} \left[\sum_t r(s_t, a_t) \right]$$

- Policy Gradient methods
- Actor-Critic methods
- Value-based methods

The RL objective

$$\theta^* = \arg\max_\theta \mathbb{E}_{\tau \sim p_\theta(\tau)} \left[\sum_t r(s_t, a_t) \right]$$

- Policy Gradient methods
- Actor-Critic methods
- Value-based methods
- (Model-based methods)

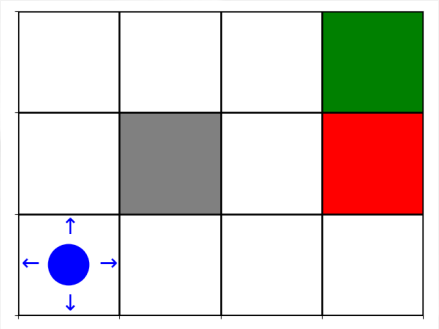
The RL objective

$$\theta^* = \arg\max_\theta \mathbb{E}_{\tau \sim p_\theta(\tau)} \left[\sum_t r(s_t, a_t) \right]$$

- Policy Gradient methods
- Actor-Critic methods
- Value-based methods
- (Model-based methods)
- (On-policy and off-policy)

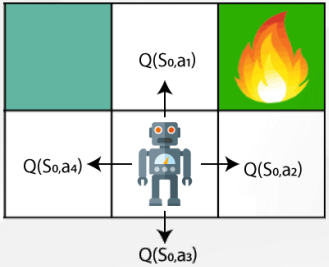
Gridworld environment

- Discrete state and action space.
  - States: 2D coordinates (x,y)
  - Actions: move (left, right, up, down)
- Markov property.
- Stochastic transition probability.
- An episode is finished when a terminal state is reached.
- Rewards:
  - Green: “Good" terminal state (reward = +1)
  - Red: “Bad" terminal state (reward = -1)
  - White: reward = 0
  - Gray: Inaccessible



The Gridworld environment

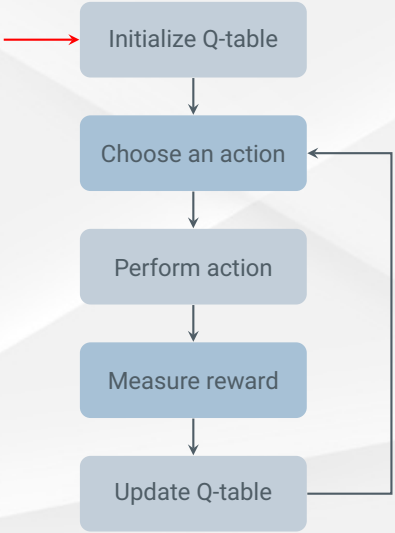
Q-learning



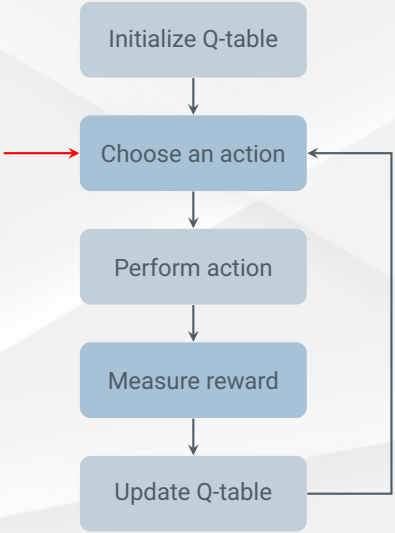
Q_table

		Actions			
		A_1	A_2	...	A_M
States	S_1	$Q(S_1, A_1)$	$Q(S_1, A_2)$	...	$Q(S_1, A_M)$
	S_2	$Q(S_2, A_M)$	$Q(S_2, A_M)$	...	$Q(S_2, A_M)$
	...	...	...	...	...
	S_N	$Q(S_N, A_1)$	$Q(S_N, A_2)$	...	$Q(S_N, A_M)$

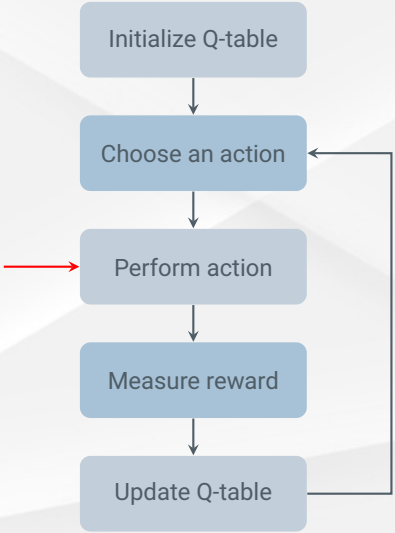
Q-learning overview



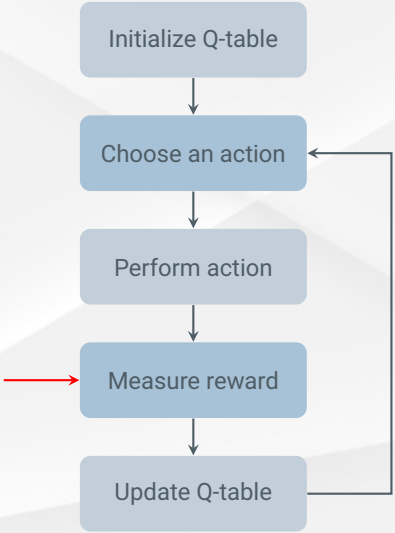
Q-learning overview



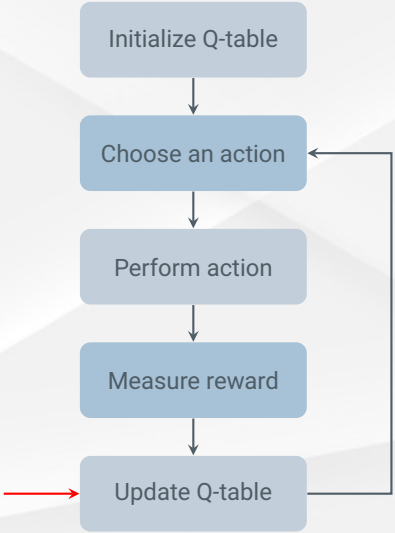
Q-learning overview



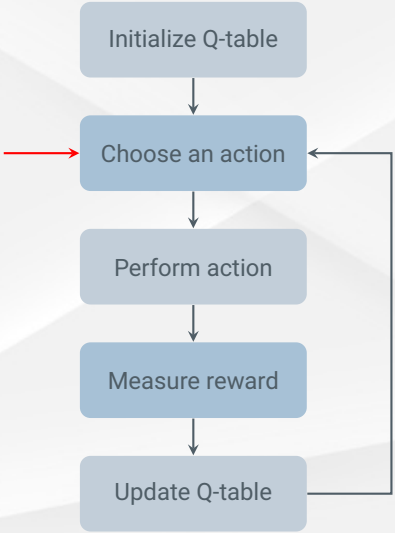
Q-learning overview



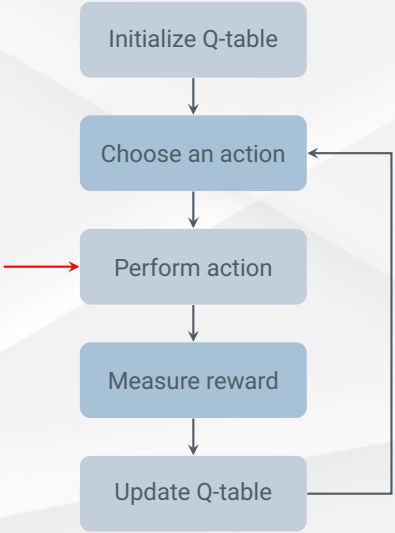
Q-learning overview



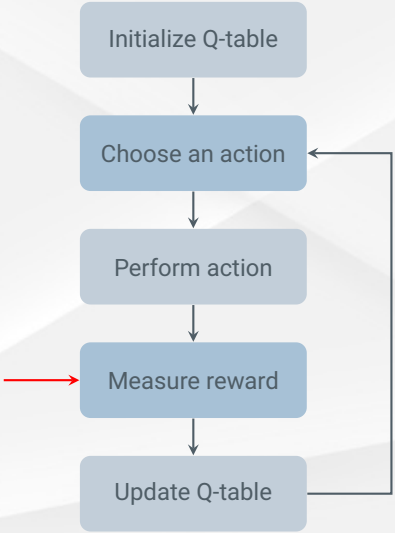
Q-learning overview



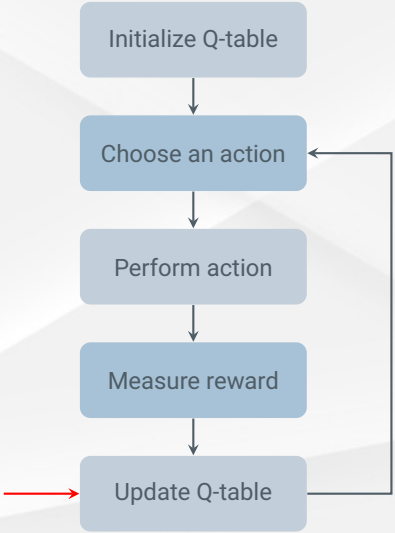
Q-learning overview



Q-learning overview



Q-learning overview

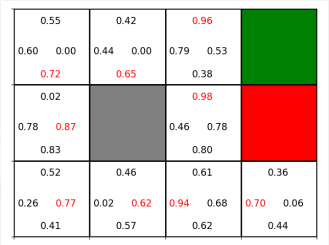


Q-learning algorithm

Algorithm parameters: step size $\alpha \in (0, 1]$, small $\epsilon > 0$, $\gamma \in [0, 1]$

Q-learning algorithm

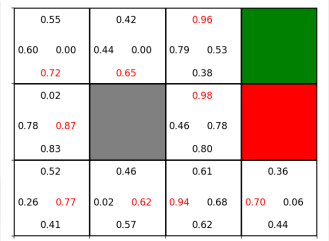
Algorithm parameters: step size $\alpha \in (0, 1]$, small $\epsilon > 0$, $\gamma \in [0, 1]$
Initialize $Q(s, a)$, for all $s \in \mathcal{S}^+, a \in \mathcal{A}(s)$, arbitrarily except that $Q(terminal, \cdot) = 0$



Q-learning algorithm

Algorithm parameters: step size $\alpha \in (0, 1]$, small $\epsilon > 0$, $\gamma \in [0, 1]$
Initialize $Q(s, a)$, for all $s \in \mathcal{S}^+, a \in \mathcal{A}(s)$, arbitrarily except that $Q(terminal, \cdot) = 0$
for episode = 1, M **do**

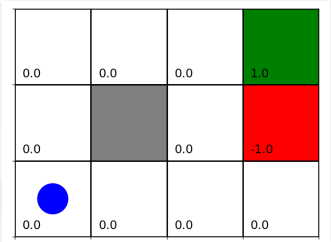
end



Q-learning algorithm

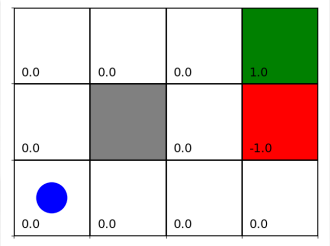
Algorithm parameters: step size $\alpha \in (0, 1]$, small $\epsilon > 0$, $\gamma \in [0, 1]$
Initialize $Q(s, a)$, for all $s \in \mathcal{S}^+, a \in \mathcal{A}(s)$, arbitrarily except that $Q(terminal, \cdot) = 0$
for episode = 1, M **do**
    Initialize $\mathcal{S}$

end



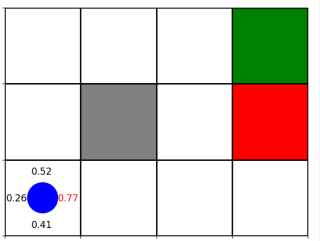
Q-learning algorithm

Algorithm parameters: step size $\alpha \in (0, 1]$, small $\epsilon > 0$, $\gamma \in [0, 1]$
Initialize $Q(s, a)$, for all $s \in \mathcal{S}^+, a \in \mathcal{A}(s)$, arbitrarily except that $Q(terminal, \cdot) = 0$
for episode = 1, M **do**
  Initialize $\mathcal{S}$
  while $S \neq terminal$ **do**
  end
end



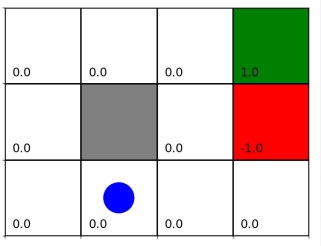
Q-learning algorithm

Algorithm parameters: step size $\alpha \in (0, 1]$, small $\epsilon > 0$, $\gamma \in [0, 1]$
Initialize $Q(s, a)$, for all $s \in \mathcal{S}^+, a \in \mathcal{A}(s)$, arbitrarily except that $Q(terminal, \cdot) = 0$
for episode = 1, M **do**
  Initialize $\mathcal{S}$
  while $S \neq terminal$ **do**
    Choose A from S using policy derived from Q (e.g., ϵ-greedy)
  end
end



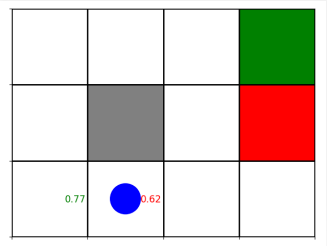
Q-learning algorithm

Algorithm parameters: step size $\alpha \in (0, 1]$, small $\epsilon > 0$, $\gamma \in [0, 1]$
Initialize $Q(s, a)$, for all $s \in \mathcal{S}^+, a \in \mathcal{A}(s)$, arbitrarily except that $Q(terminal, \cdot) = 0$
for episode = 1, M **do**
  Initialize $\mathcal{S}$
  while $S \neq terminal$ **do**
    Choose A from S using policy derived from Q (e.g., ϵ-greedy)
    Take action A, observe R, S'
  end
end



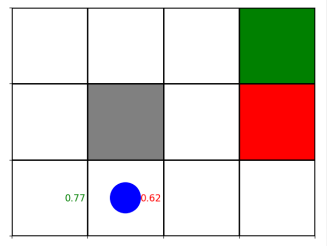
Q-learning algorithm

Algorithm parameters: step size $\alpha \in (0, 1]$, small $\epsilon > 0$, $\gamma \in [0, 1]$
Initialize $Q(s, a)$, for all $s \in \mathcal{S}^+, a \in \mathcal{A}(s)$, arbitrarily except that $Q(terminal, \cdot) = 0$
for episode = 1, M **do**
  Initialize $\mathcal{S}$
  while $S \neq terminal$ **do**
    Choose A from S using policy derived from Q (e.g., ϵ-greedy)
    Take action A, observe R, S'
    $Q(S, A) \leftarrow Q(S, A) + \alpha[R + \gamma \max_a Q(S', a) - Q(S, A)]$
  end
end



Q-learning algorithm

Algorithm parameters: step size $\alpha \in (0, 1]$, small $\epsilon > 0$, $\gamma \in [0, 1]$
Initialize $Q(s, a)$, for all $s \in \mathcal{S}^+, a \in \mathcal{A}(s)$, arbitrarily except that $Q(terminal, \cdot) = 0$
for episode = 1, M **do**
  Initialize $\mathcal{S}$
  while $S \neq terminal$ **do**
    Choose A from S using policy derived from Q (e.g., ϵ-greedy)
    Take action A, observe R, S'
    $Q(S, A) \leftarrow Q(S, A) + \alpha[R + \gamma \max_a Q(S', a) - Q(S, A)]$
  end
end



$$Q(bottom\_left, RIGHT) \leftarrow 0.77 + 0.005 \times [0 + 0.99 \times 0.62 - 0.77]$$

Q-learning algorithm

Algorithm parameters: step size $\alpha \in (0, 1]$, small $\epsilon > 0$, $\gamma \in [0, 1]$
Initialize $Q(s, a)$, for all $s \in \mathcal{S}^+, a \in \mathcal{A}(s)$, arbitrarily except that $Q(terminal, \cdot) = 0$
for episode = 1, M **do**
  Initialize $\mathcal{S}$
  while $S \neq terminal$ **do**
    Choose A from S using policy derived from Q (e.g., ϵ-greedy)
    Take action A, observe R, S'
    $Q(S, A) \leftarrow Q(S, A) + \alpha[R + \gamma \max_a Q(S', a) - Q(S, A)]$
    $S \leftarrow S'$
  end
end

1.41 1.45 1.45 1.68	1.75 1.63 1.89 1.80	1.89 1.86 1.91 1.78	
1.61 1.64 1.63 1.69		1.75 1.72 0.16 1.67	
1.67 1.68 1.70 1.68	1.72 1.70 1.72 1.71	1.74 1.72 1.63 1.72	0.17 1.64 1.33 1.46

After 100k episodes

Q-learning algorithm

Algorithm parameters: step size $\alpha \in (0, 1]$, small $\epsilon > 0$, $\gamma \in [0, 1]$
Initialize $Q(s, a)$, for all $s \in \mathcal{S}^+, a \in \mathcal{A}(s)$, arbitrarily except that $Q(terminal, \cdot) = 0$
for episode = 1, M **do**
  Initialize $\mathcal{S}$
  while $S \neq terminal$ **do**
    Choose A from S using policy derived from Q (e.g., ϵ-greedy)
    Take action A, observe R, S'
    $Q(S, A) \leftarrow Q(S, A) + \alpha[R + \gamma \max_a Q(S', a) - Q(S, A)]$
    $S \leftarrow S'$
  end
end

1.69 1.70 1.86 1.73	1.86 1.78 1.88 1.86	1.89 1.86 1.90 1.77	
1.84 1.70 1.71 1.70		1.77 1.74 0.15 1.62	
1.71 1.69 1.72 1.69	1.70 1.70 1.73 1.70	1.74 1.71 1.64 1.72	0.14 1.63 1.50 1.61

After 200k episodes

Q-learning algorithm

Algorithm parameters: step size $\alpha \in (0, 1]$, small $\epsilon > 0$, $\gamma \in [0, 1]$
Initialize $Q(s, a)$, for all $s \in \mathcal{S}^+, a \in \mathcal{A}(s)$, arbitrarily except that $Q(terminal, \cdot) = 0$
for episode = 1, M **do**
  Initialize $\mathcal{S}$
  while $S \neq terminal$ **do**
    Choose A from S using policy derived from Q (e.g., ϵ-greedy)
    Take action A, observe R, S'
    $Q(S, A) \leftarrow Q(S, A) + \alpha[R + \gamma \max_a Q(S', a) - Q(S, A)]$
    $S \leftarrow S'$
  end
end

1.85 1.84 1.86 1.83	1.87 1.85 1.89 1.87	1.89 1.86 1.91 1.77	
1.84 1.82 1.82 1.81		1.78 1.71 0.09 1.69	
1.82 1.80 1.78 1.80	1.78 1.80 1.76 1.78	1.75 1.78 1.65 1.75	0.11 1.66 1.55 1.60

After 300k episodes

Q-learning algorithm

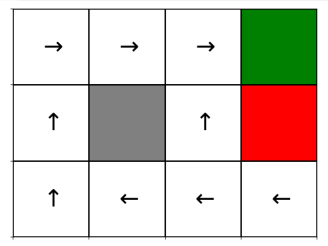
Algorithm parameters: step size $\alpha \in (0, 1]$, small $\epsilon > 0$, $\gamma \in [0, 1]$
Initialize $Q(s, a)$, for all $s \in \mathcal{S}^+, a \in \mathcal{A}(s)$, arbitrarily except that $Q(terminal, \cdot) = 0$
for episode = 1, M **do**
  Initialize $\mathcal{S}$
  while $S \neq terminal$ **do**
    Choose A from S using policy derived from Q (e.g., ϵ-greedy)
    Take action A, observe R, S'
    $Q(S, A) \leftarrow Q(S, A) + \alpha[R + \gamma \max_a Q(S', a) - Q(S, A)]$
    $S \leftarrow S'$
  end
end

1.85 1.84 1.86 1.83	1.87 1.85 1.89 1.87	1.89 1.87 1.91 1.78	
1.84 1.83 1.83 1.81		1.77 1.75 0.13 1.68	
1.82 1.81 1.79 1.80	1.78 1.80 1.77 1.78	1.76 1.78 1.66 1.76	0.10 1.66 1.51 1.61

After 400k episodes

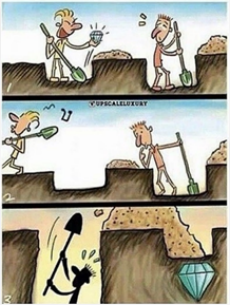
Q-learning algorithm

Algorithm parameters: step size $\alpha \in (0, 1]$, small $\epsilon > 0$, $\gamma \in [0, 1]$
Initialize $Q(s, a)$, for all $s \in \mathcal{S}^+, a \in \mathcal{A}(s)$, arbitrarily except that $Q(terminal, \cdot) = 0$
for episode = 1, M **do**
  Initialize $\mathcal{S}$
  while $S \neq terminal$ **do**
    Choose A from S using policy derived from Q (e.g., ϵ-greedy)
    Take action A, observe R, S'
    $Q(S, A) \leftarrow Q(S, A) + \alpha[R + \gamma \max_a Q(S', a) - Q(S, A)]$
    $S \leftarrow S'$
  end
end

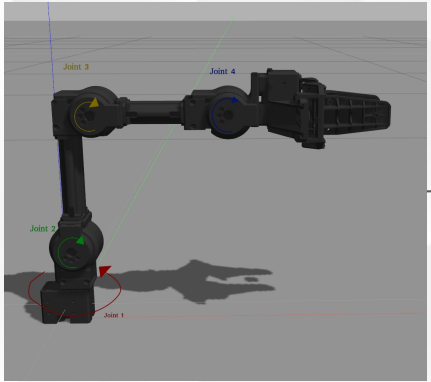


Final policy

Open Problems - Exploration vs. Exploitation



Open Problems - Transfer Learning



Open problems - Explainability

Final frame

Thank you for your attention!