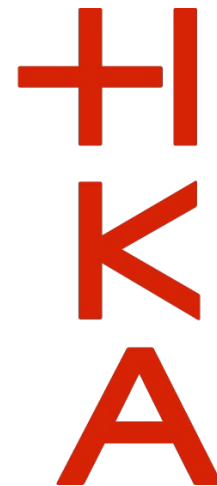




KI Labor - Wintersemester 2022

**R**einforcement **L**earning



Jochen Gietzen, Stefan Käser, Maximilian Blanck, Pascal  
Fecht, **Adrian Westermeier, Tim Bossenmaier**

Karlsruhe, 9. Dezember 2022

# Schedule



Datum	Thema	Inhalt	Präsenz
30. Sept.	Allg.	Organisation, Teamfindung, Vorstellung CV	Ja
7. Okt.	Ausfall (DMA Techday)		
14. Okt.	CV	Q&A Sessions	Nein
21. Okt.	CV	<u>Sprintwechsel</u> , Vorstellung Assignment	Ja
28. Okt.	CV	Q&A Sessions	Nein
4. Nov.	CV / NLP	Abgabe CV, Vorstellung NLP	Ja
11. Nov.	NLP	Q&A Sessions	Nein
18. Nov.	NLP	<u>Sprintwechsel</u> , Vorstellung Assignment	Ja
25. Nov.	NLP	Q&A Sessions	Nein
2. Dez.	Ausfall (Winter Plenum)		
9. Dez.	NLP / RL	Abgabe NLP, Vorstellung RL <input type="checkbox"/>	Ja
16. Dez.	RL	Q&A Sessions	Nein
23. Dez.	RL	<u>Sprintwechsel</u> , Vorstellung Assignment	Ja / Nein
13. Jan.	RL	Q&A Sessions	Nein
20. Jan.	RL	Abgabe RL, Abschluss KI Labor	Ja



Adrian Westermeier  
Machine Learning Engineer  
seit 2022



Tim Bossenmaier  
Softwareentwickler Datenplattformen  
seit 2021

# Agenda

## › **Theorie**

- Problemstellung & Lösungsansatz
- Value Functions
- Monte-Carlo und Temporal-Difference Methoden
- Q-Learning

## › **Übungsaufgaben**

- Menace Gym (Aufgabe 1)
- CartPole Gym mit Q-Learning (Aufgabe 2)

# Reinforcement Learning



(mathematische)  
Psychologie

Kontroll-  
Theorie

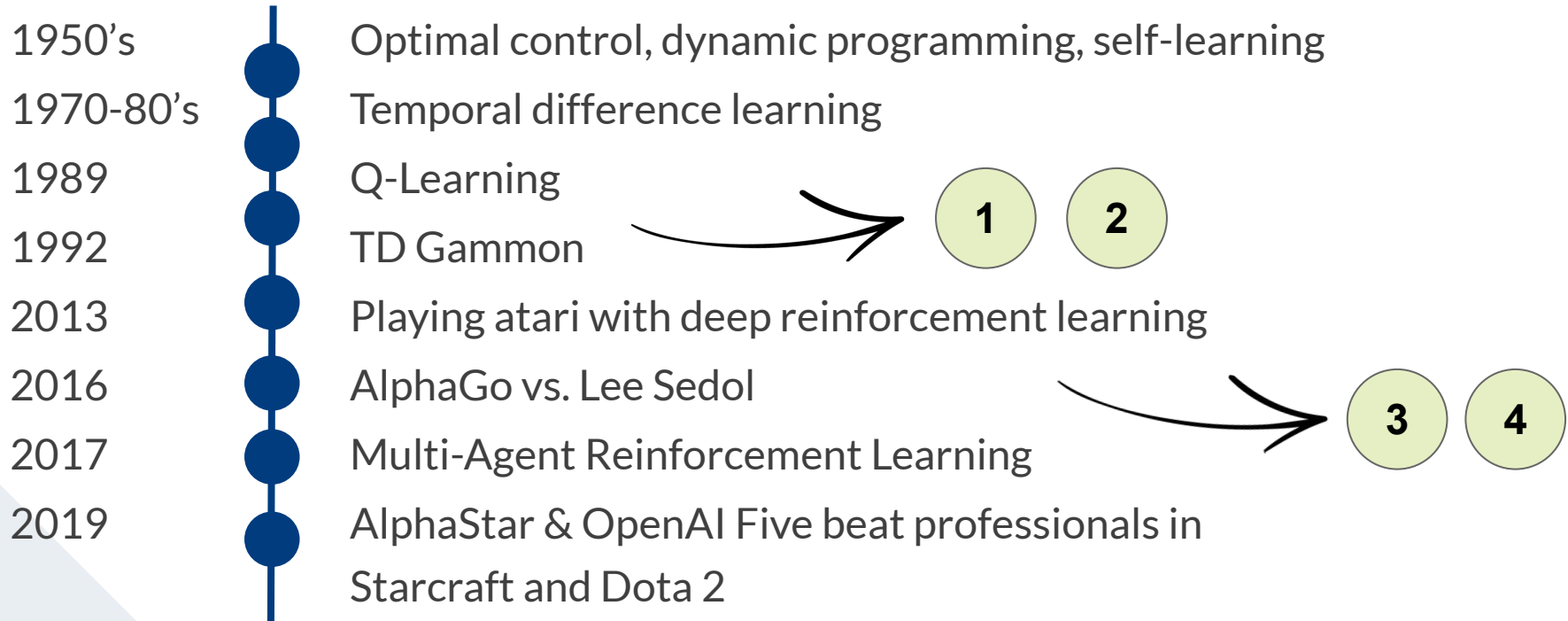
Reinforcement  
Learning

Künstliche  
Intelligenz

Neuro-  
wissenschaften

Operations  
Research

# Meilensteine im Reinforcement Learning





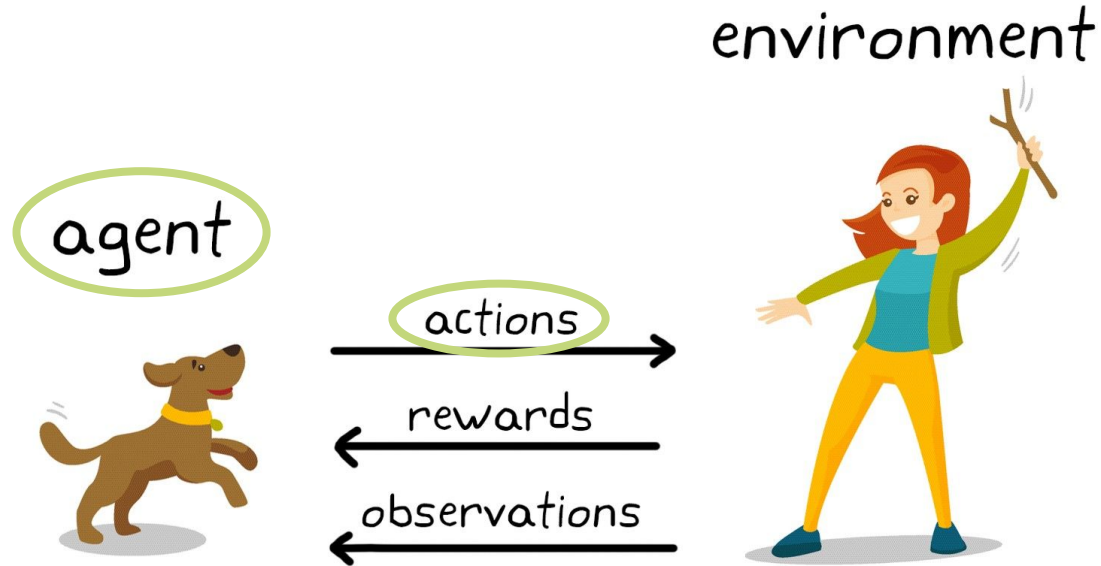
# “Robots that learn a little like humans do: By trial and error.”

## Law of effect (nach Thorndike, 1898):

responses that produce a satisfying effect in a particular situation become more likely to occur again in that situation, and responses that produce a discomforting effect become less likely to occur again in that situation.

**Menschen lernen Verhalten durch  
Belohnung und Strafe**

# Reinforcement Learning



# Vergleich mit (un)überwachtem Lernen

(Un)Supervised Learning

Reinforcement Learning

Lernen mit Datensätzen

Lernen durch Ausprobieren

Ziel: Loss minimieren

Ziel: Reward maximieren

Interaktion mit Umwelt  
nicht Teil des Systems

Interaktion mit Umwelt ist  
zentraler Teil des Systems

Getrennte Trainings- &  
Durchführungsphase

Kontinuierliches Lernen /  
Exploration vs. Exploitation

# Beispiel: Tic-Tac-Toe

- 9 Felder
- je 3 mögliche Belegungen

⇒ # Zustände  $\leq 3^9 = 19.683$

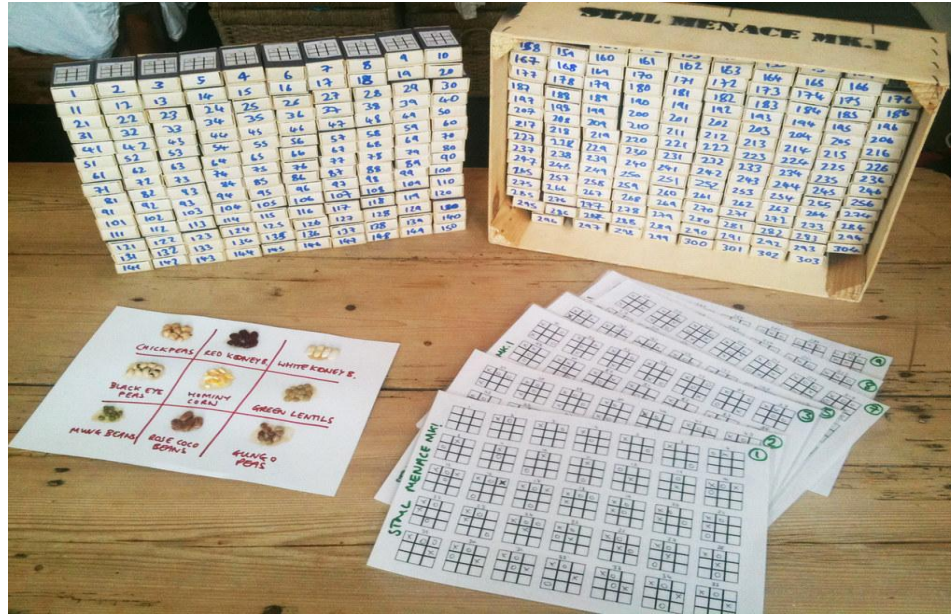
# Spiele  $\leq 9! = 362.880$

Lösbar, aber aufwendige  
Programmierung



# MENACE [Michie1963]

## Matchbox Educable Naughts And Crosses Engine



[Michie 1963]: <https://people.csail.mit.edu/brooks/idocs/matchbox.pdf>

Fotos: James Bridle, <http://jamesbridle.com/works/menace>

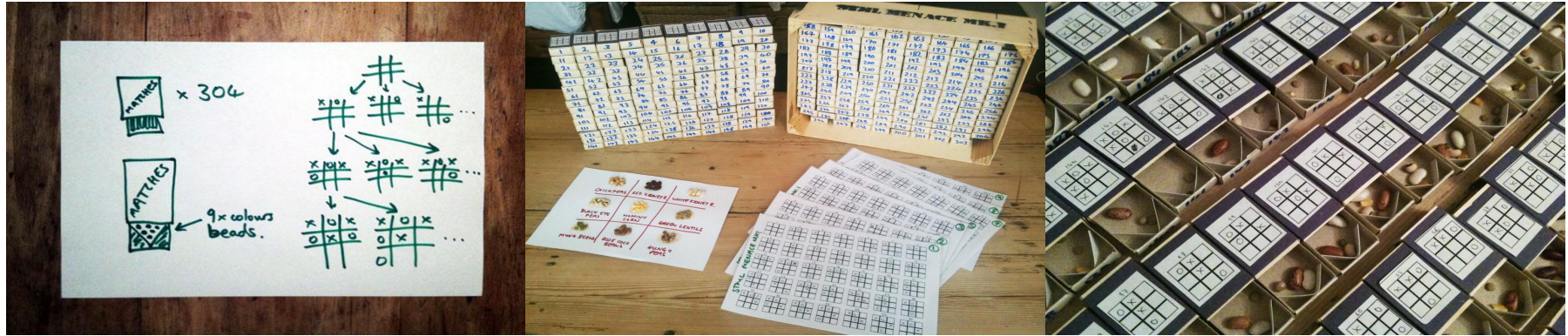
# MENACE [Michie1963]



Moderne Implementierung: <https://www.youtube.com/watch?v=R9c-neaxeU>



# MENACE [Michie1963]



- Eine Schachtel pro Spielzustand
- Perlen in Schachteln für mögliche Spielzüge
- Spielzug bestimmen = Perle aus Schachtel ziehen

[Michie 1963]: <https://people.csail.mit.edu/brooks/idocs/matchbox.pdf>

Fotos: James Bridle, <http://jamesbridle.com/works/menace>

# MENACE [Michie1963]

Nach dem Spiel: Lernen

## Gewonnen

je 2 Perlen gleicher Farbe  
in Schachtel zurücklegen

## Unentschieden

Perlen zurücklegen

## Verloren

Perlen entfernen



[Michie1963]: <https://people.csail.mit.edu/brooks/idocs/matchbox.pdf>

Fotos: James Bridle, <http://jamesbridle.com/works/menace>



# MENACE ist Reinforcement Learning

## MENACE

Regeln, Gegner, Spielfeld

Schachtel

Perlen/Spielzüge

Perlen zurück/weg legen

Zug/Spielrunde

## Reinforcement Learning

Umwelt implizit (z.T. Zustand)

Zustand  $s_t \in \mathcal{S}$

Aktionen  $a_t \in \mathcal{A}$

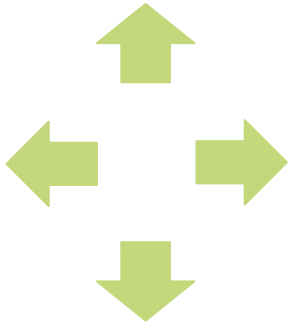
Reward  $r : \mathcal{S} \times \mathcal{A} \mapsto \mathbb{R}$

Zeit/Episode  $t = 0, 1, 2, \dots$

# Gridworld

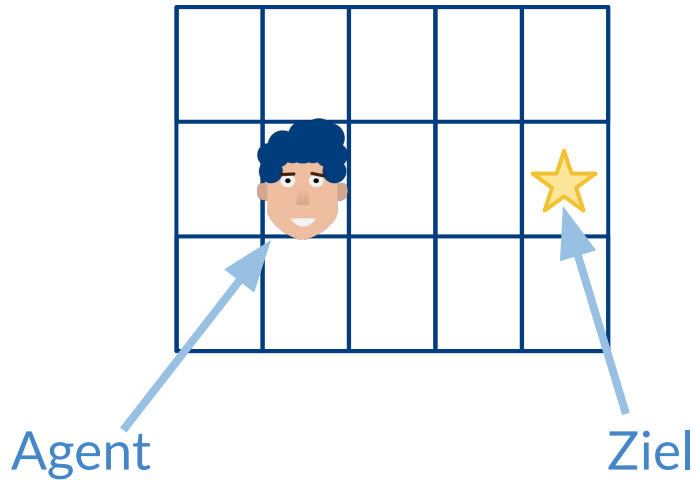
Actions

$$a_t \in \mathcal{A}$$



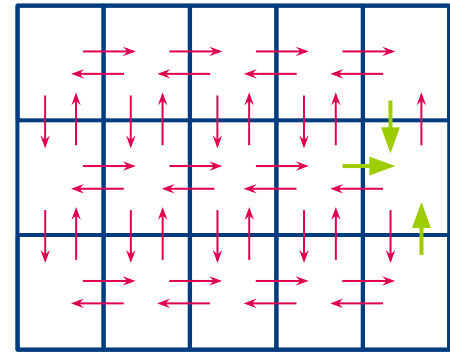
States

$$s_t \in \mathcal{S}$$

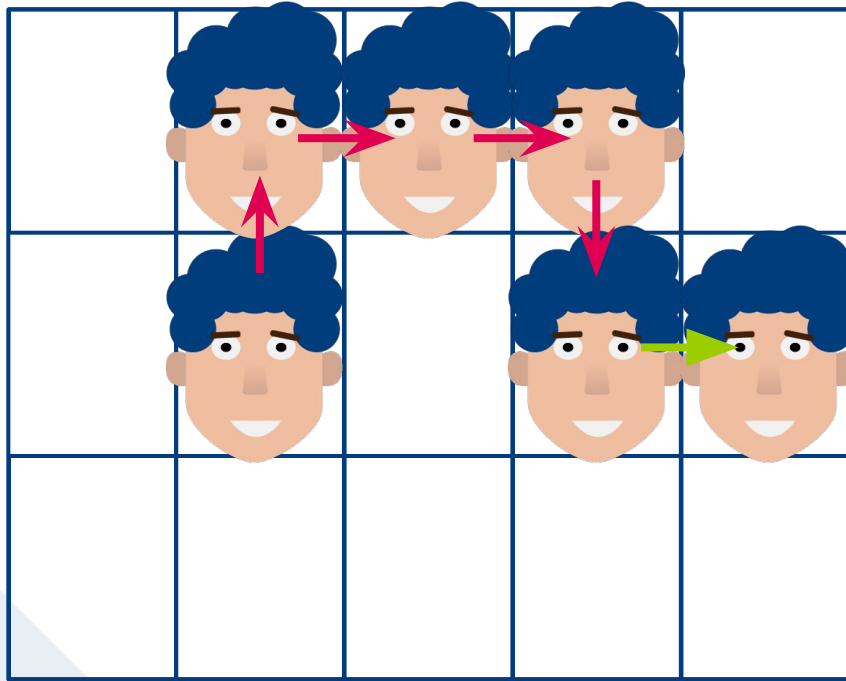


Rewards

$$R_t = r(s_t, a_t)$$



# Buchhaltung



Aktion



Reward gesamt

$$R_0 = -1$$

$$R_0 + R_1 = -2$$

$$R_0 + R_1 + R_2 = -3$$

$$R_0 + \dots + R_3 = -4$$

$$R_0 + \dots + R_4 = -3$$

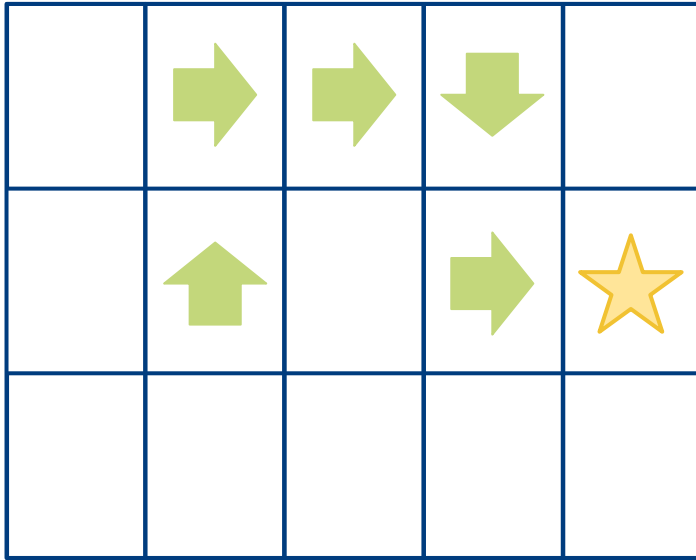
Ziel: maximiere gesammelte Rewards

# Zukünftiger Reward

$$\begin{aligned} \underline{G_t} &= R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \dots \\ \text{Return ab Zeitpunkt } t &= \sum_{k=0}^{\infty} \gamma^k R_{t+k+1} \quad 0 \leq \underline{\gamma} \leq 1 \\ &\quad \underbrace{\hspace{10em}}_{= r(s_{t+k+1}, a_{t+k+1})} \end{aligned}$$

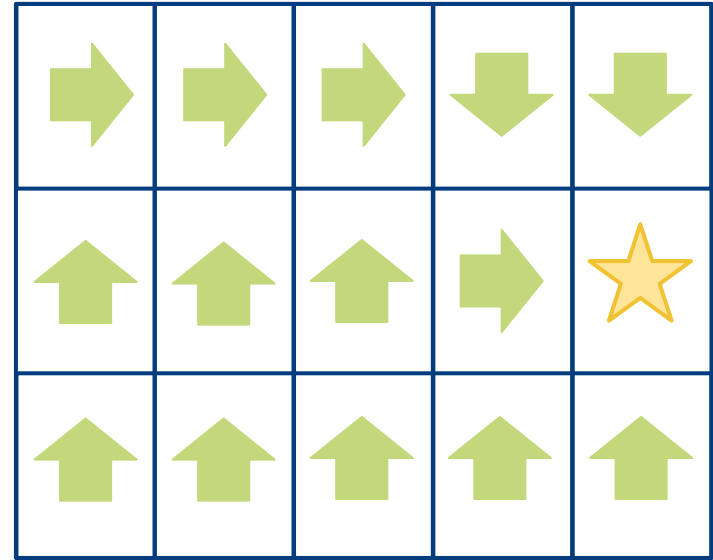
Discount Factor

# Wie die nächste Aktion auswählen?



Plan

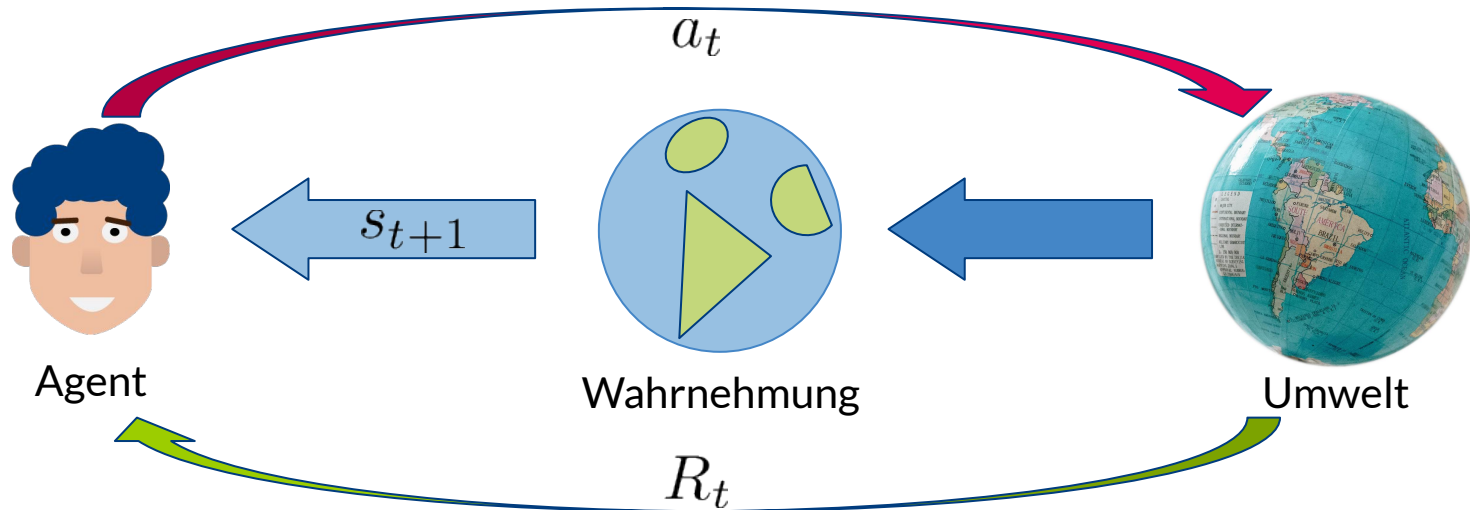
$a_1, a_2, \dots$



Policy

$a = \pi(s)$

# Modell für Zustandsübergänge



Modell für Statusübergang:  $P(s_{t+1} | a_t, s_t, \dots, a_0, s_0)$

... und Reward:  $P(s_{t+1}, R_t | a_t, s_t, \dots, R_0, a_0, s_0)$

# Aktionen ändern die Umwelt

## Transition probabilities $P(s_{t+1}|a_t, s_t)$

Beispiel: Geradeaus gehen



$P(s_{t+1}|a_t, s_t)$  eher groß

<https://unsplash.com/photos/UUvAux9zCFA>



$P(s_{t+1}|a_t, s_t)$  eher klein

[https://unsplash.com/photos/2HHUOHU\\_fBE](https://unsplash.com/photos/2HHUOHU_fBE)

# Transition probabilities $P(s_{t+1}|a_t,s_t)$

		tatsächlicher Zustandsübergang				
		←	→	↑	↓	
Aktion des Agenten	←	1	80%	0%	10%	10%
	→	2	0%	80%	10%	10%
	↑	3	10%	10%	80%	0%
	↓	4	10%	10%	0%	80%





# Markov-Annahme

Ein stochastischer Prozess hat die **Markov** Eigenschaft, wenn der aktuelle Zustand nur vom vorherigen Zustand abhängt:

$$P(x_t | x_{t-1}, \dots, x_0) = P(x_t | x_{t-1})$$

Zustandsübergangs-und-Reward-Modell:

$$P(s_{t+1}, R_t | a_t, s_t, \dots, R_0, a_0, s_0) = P(s_{t+1}, R_t | a_t, s_t)$$

# Markov Decision Process (MDP)

Formale Beschreibung der Interaktion im RL

States  $s \in \mathcal{S}$

Actions  $a \in \mathcal{A}$

Time  $t$

Model  $P(s_{t+1}, R_t | a_t, s_t)$

Reward  $r(s, a)$

## Markov-Annahme:

Zustandsübergang und Reward  
hängen nur von vorherigen  
Zustand und Aktion ab



# Policy $\pi$

Eine Policy definiert das Agenten-Verhalten für **alle** Zustände  $s$

Deterministisch:  $a = \pi(s)$

↓	↓	↓	↓	↓
↓	↓	↓	↓	x
→	→	→	→	↑

Stochastisch:  $a \sim P_{\pi}(a|s)$

→	→	→	→	↓
↑ →	↑ →	↑ →	↑ →	x
↑ →	↑ →	↑ →	↑ →	↑

# Wie findet ein Agent eine gute Policy?

Reminder: Agent will Return  $G_t$  maximieren

$$G_t = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1}$$


Kurzsichtiger Agent:

$$\begin{aligned} \gamma &:= 0 \rightsquigarrow G_t = R_{t+1} = r(s_{t+1}, a_{t+1}) \\ &\Rightarrow \pi(s_t) = \arg \max_a r(s_{t+1}, a) \end{aligned}$$


# Kurzsichtiger Agent

... wählt eine Aktion, die den nächsten Reward maximiert

State  $s$

				X

Rewards

-1	-1	-1	-1	-1
-1		-1	-1	+1
-1	-1	-1	-1	-1

# Bessere Strategie

Wähle Policy  $\pi$ , die den **erwarteten Return** maximiert:

$$\pi^* = \arg \max_{\pi} \mathbb{E}_{\pi} [G_t]$$


mit

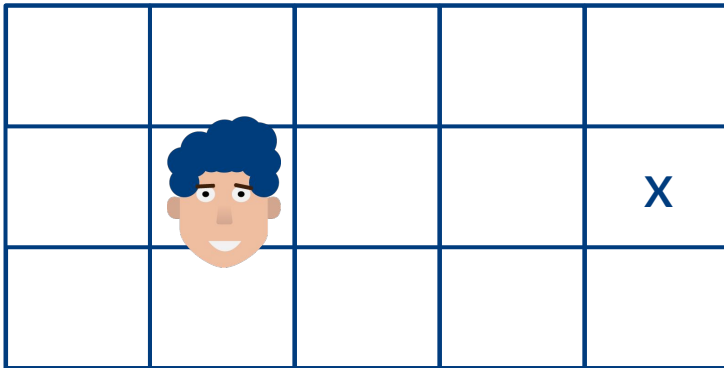
$$G_t = \sum_{k=0}^{\infty} \gamma^k r(s_{t+k+1}, \pi(s_{t+k}))$$

$$\mathbb{E}[x] = \sum_{x \in \mathcal{X}} P(x) \cdot x$$

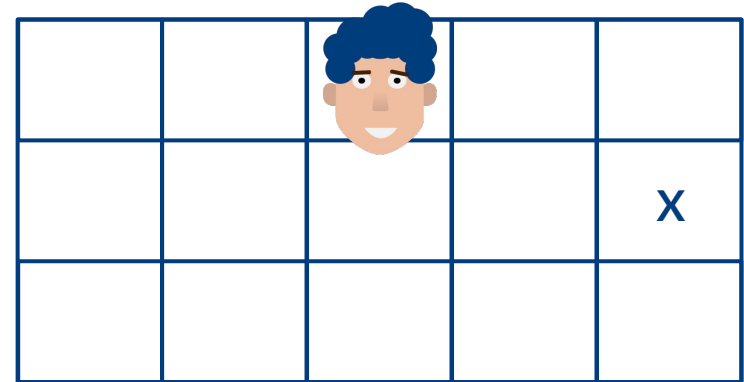
# Value Functions

Welcher Zustand ist besser?

State 1



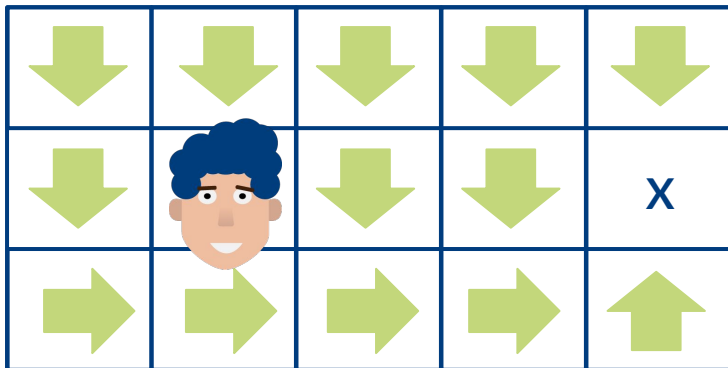
State 2



# State-value function

Welcher State verspricht größten Return?

State  $s$  und Policy  $\pi$



State-value function  $v_{\pi}(s)$

-7	-6	-5	-4	-1
-6	-5	-4	-3	+1
-5	-4	-3	-2	-1

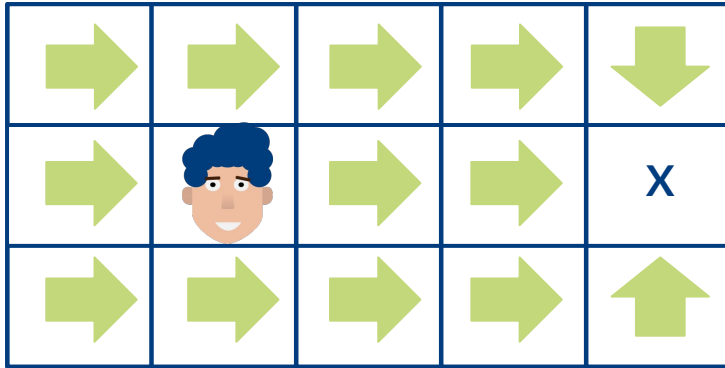
$$v_{\pi}(s) = \mathbb{E}_{\pi}[G_t | S_t = s] = \mathbb{E}[R_{t+1} + G_{t+1}(S_{t+1}) | S_t = s]$$



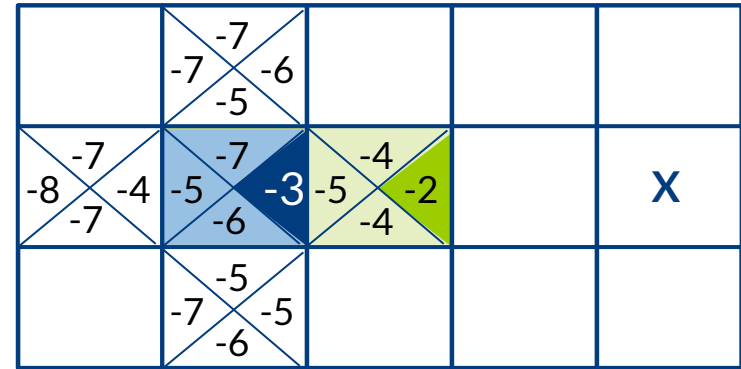
# State-Action-value function $q_{\pi}(s,a)$

Welches State-Action Paar verspricht größten Return?

State  $s$  und Policy  $\pi$



Action-value function  $q_{\pi}(s,a)$







$$\begin{aligned} q_{\pi}(s, a) &= \mathbb{E}_{\pi}[G_t | S_t = s, A_t = a] \\ &= \mathbb{E}[R_{t+1} + G_{t+1}(S_{t+1}, A_{t+1}) | S_t = s, A_t = a] \end{aligned}$$

# Action-value function $q_{\pi}(s,a)$

Action-value function  $q_{\pi}(s,a)$

	<div>-7 -7 -6 -5</div>			
<div>-8 -7 -4 -7</div>	<div>-7 -5 -6 -3</div>	<div>-4 -5 -4 -2</div>		X
	<div>-5 -7 -5 -6</div>			

Q-Table

				
$s_{12}$	-6	-7	-7	-5
$s_{21}$	-4	-8	-7	-7
$s_{22}$	-3	-5	-7	-6
$s_{23}$	-2	-5	-4	-4
$s_{32}$	-5	-7	-5	-6

# Bellman Equations

Wie hängen Zustände, Rewards und Folgezustände zusammen?

$$\underline{v_{\pi}(s)} = \sum_a \pi(a | s) \sum_{s', r} p(s', r | s, a) [\underline{r + \gamma v_{\pi}(s')}]$$

Aktueller Zustand

Erfahrung

Folgezustand

$$\underline{q_{\pi}(s, a)} = \sum_{s', r} p(s', r | s, a) \left( r(s, a) + \gamma \cdot \sum_{a'} \pi(a' | s') \cdot \underline{q_{\pi}(s', a')} \right)$$

# Optimal Policies $\pi^*$



-5 →	-4 →	-3 →	-2 →	↓ -1
-4 →	-3 →	-2 →	-1 →	X
-5 →	-4 →	-3 →	-2 →	↑ -1

-5 →	-4 →	-3 →	-2 →	↓ -1
↓ -6	↓ -5	↓ -4	-1 →	X
-5 →	-4 →	-3 →	-2 →	↑ -1

Optimale Policy ist besser  
alle andere Policies:

$$\pi^* \geq \pi, \forall \pi$$

Was bedeutet besser?

$$\pi \geq \pi', \text{ if } v_\pi(s) \geq v_{\pi'}(s), \forall s$$

# Bellman Equations für optimale Policies

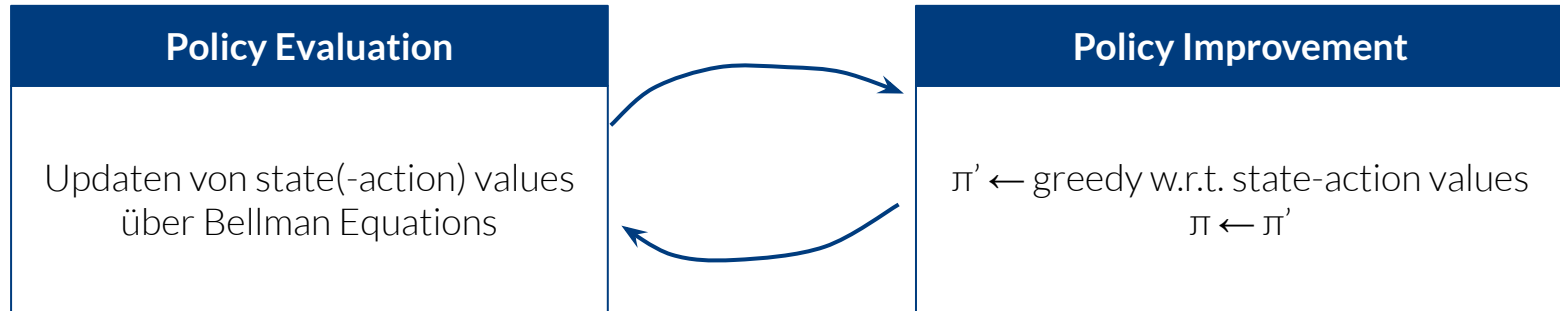
The diagram illustrates the Bellman equations for optimal policies, with arrows indicating the components of each equation:

- Aktueller Zustand** (Current State) points to  $s$  in both equations.
- Erfahrung** (Experience) points to the summation over  $s', r$  in both equations.
- Folgezustand** (Next State) points to  $s'$  in both equations.

The equations are:

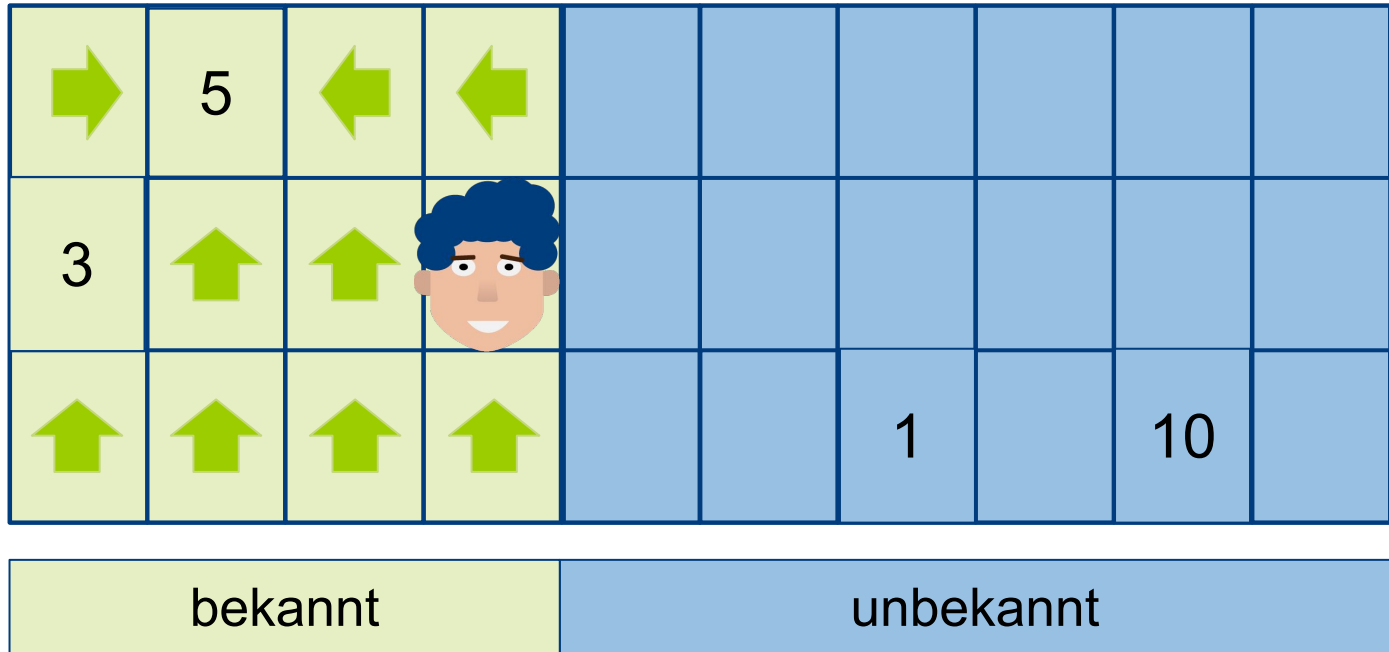
$$\underline{v_*(s)} = \max_a \sum_{s', r} p(s', r | s, a) [r + \gamma \underline{v_*(s')}]$$
$$\underline{q_*(s, a)} = \sum_{s', r} p(s', r | s, a) \left[ r + \gamma \max_{a'} \underline{q_*(s', a')} \right]$$

# Iterative Policy Improvement



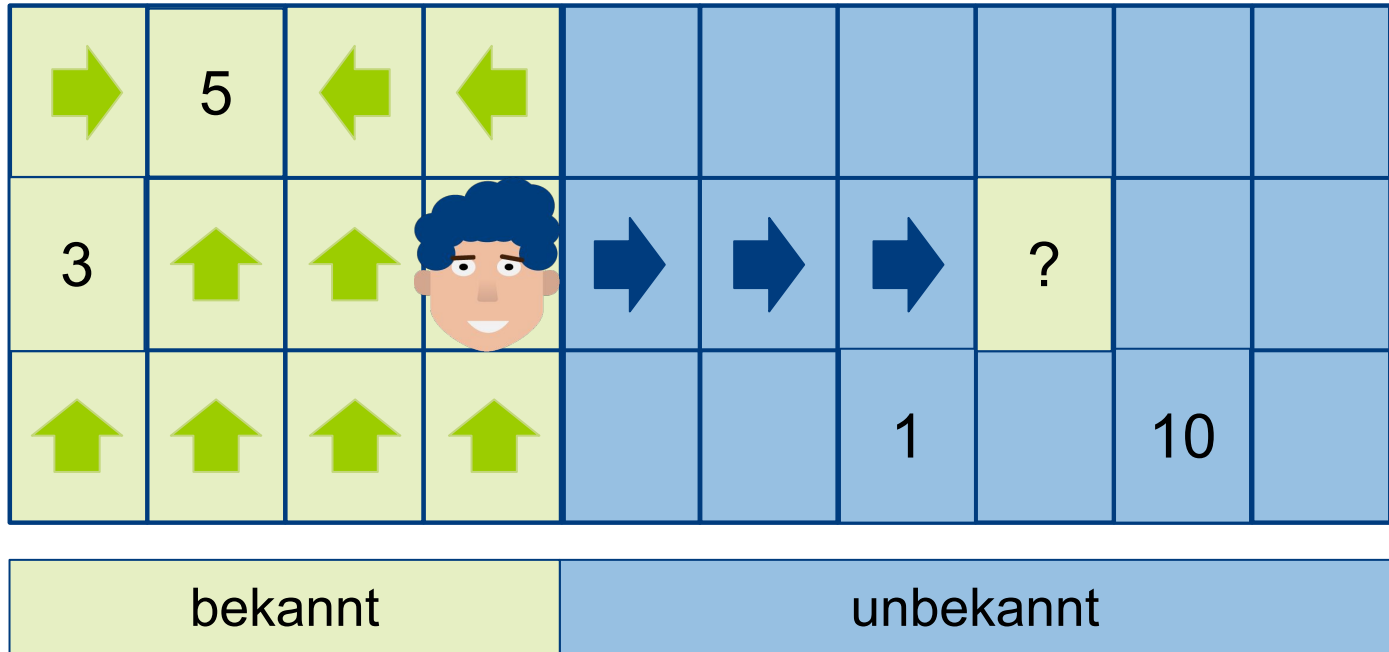
# Exploitation

Maximierung des Rewards gg. bekannter Information (“epsilon-greedy”)



# Exploration

Erschließung neuer, unbekannter Bereiche





# Iterative Policy Improvement

## 1. Initialization

$V(s) \in \mathbb{R}$  and  $\pi(s) \in \mathcal{A}(s)$  arbitrarily for all  $s \in \mathcal{S}$

## 2. Policy Evaluation

Repeat

$\Delta \leftarrow 0$

For each  $s \in \mathcal{S}$ :

$v \leftarrow V(s)$

$V(s) \leftarrow \sum_{s',r} p(s', r | s, \pi(s)) [r + \gamma V(s')]$

$\Delta \leftarrow \max(\Delta, |v - V(s)|)$

until  $\Delta < \theta$  (a small positive number)

## 3. Policy Improvement

*policy-stable*  $\leftarrow true$

For each  $s \in \mathcal{S}$ :

$a \leftarrow \pi(s)$

$\pi(s) \leftarrow \arg \max_a \sum_{s',r} p(s', r | s, a) [r + \gamma V(s')]$

If  $a \neq \pi(s)$ , then *policy-stable*  $\leftarrow false$

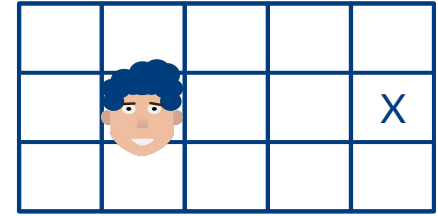
If *policy-stable*, then stop and return  $V$  and  $\pi$ ; else go to 2

Bellman Equations

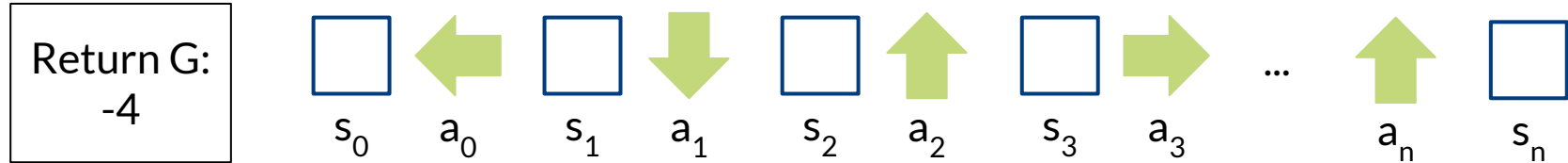
# Monte Carlo Methods

Betrachte Return von kompletten Episoden

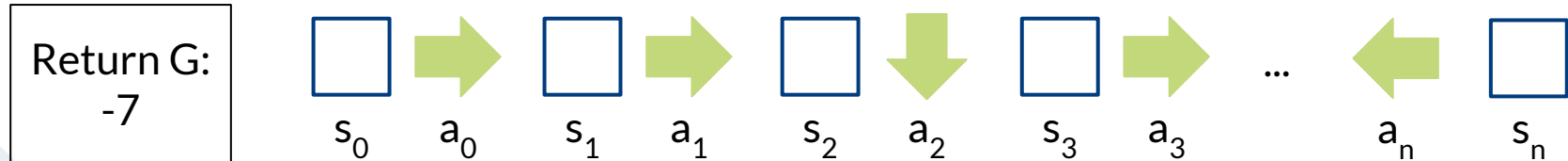
Policy  $\pi$



Episode 1



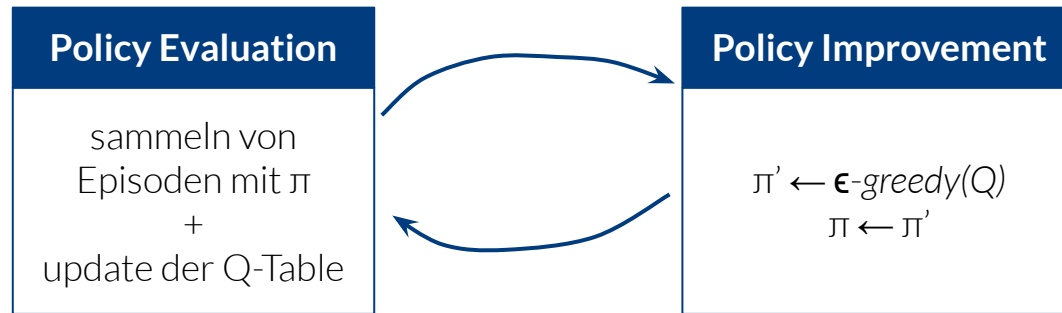
Episode 2



⋮

# Monte Carlo Control

Control Problem: Approximiere die beste Policy  
(ohne Wissen über Dynamik der Umgebung!)



$$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha(G_t - Q(S_t, A_t))$$

alternative  
Schätzung

aktuelle  
Schätzung

# Temporal-Difference Methods

## Monte Carlo Control

$$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha \left( \underbrace{G_t}_{\text{alternative Schätzung}} - \underbrace{Q(S_t, A_t)}_{\text{aktuelle Schätzung}} \right)$$

alternative  
Schätzung

aktuelle  
Schätzung

## Temporal-Difference Control (SARSA)

$$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha \left( \underbrace{R_{t+1} + \gamma Q(S_{t+1}, A_{t+1})}_{\text{alternative Schätzung}} - \underbrace{Q(S_t, A_t)}_{\text{aktuelle Schätzung}} \right)$$

alternative  
Schätzung

aktuelle  
Schätzung

# Q-Learning

## Off-Policy TD-Control

TD Update

$$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha(R_{t+1} + \gamma Q(S_{t+1}, A_{t+1}) - Q(S_t, A_t))$$



alternative  
Schätzung

Q-Learning  
Update

$$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha(R_{t+1} + \gamma \max_a Q(S_{t+1}, a) - Q(S_t, A_t))$$



alternative  
Schätzung

# Q-Learning

## Off-Policy TD-Control

Initialize  $Q(s, a), \forall s \in \mathcal{S}, a \in \mathcal{A}(s)$ , arbitrarily, and  $Q(\text{terminal-state}, \cdot) = 0$   
Repeat (for each episode):  
    Initialize  $S$   
    Repeat (for each step of episode):  
        Choose  $A$  from  $S$  using policy derived from  $Q$  (e.g.,  $\epsilon$ -greedy)  
        Take action  $A$ , observe  $R, S'$   
         $Q(S, A) \leftarrow Q(S, A) + \alpha [R + \gamma \max_a Q(S', a) - Q(S, A)]$   
         $S \leftarrow S'$ ;  
    until  $S$  is terminal

# Aufgaben

# OpenAI Gym

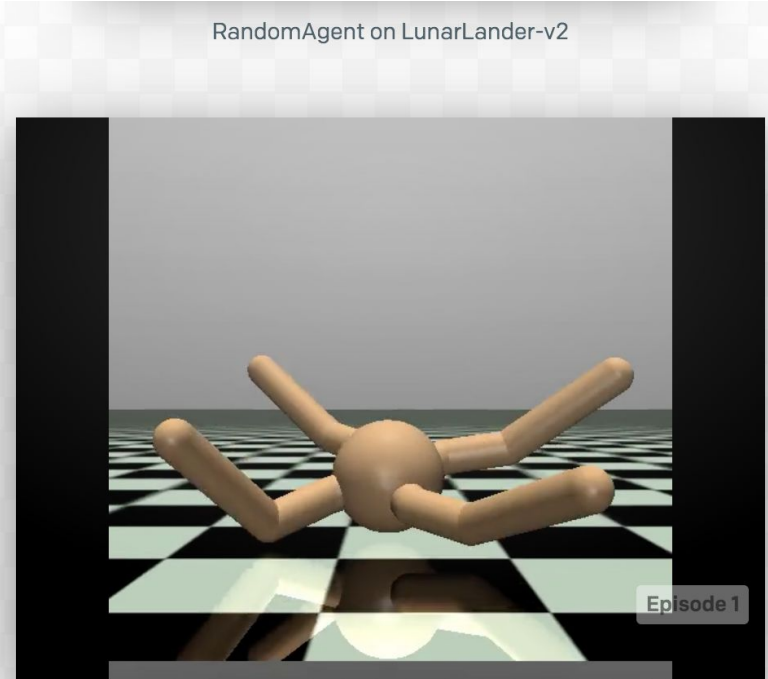


## Gym

Gym is a toolkit for developing and comparing reinforcement learning algorithms. It supports teaching agents everything from walking to playing games like Pong or Pinball.

[View documentation >](#)

[View on GitHub >](#)



RandomAgent on Ant-v2



# OpenAI Gym

```
import gym

env = gym.make('CartPole-v0')
for i_episode in range(20):
    observation = env.reset()
    for t in range(100):
        env.render()
        action = env.action_space.sample()
        observation, reward, done, info = env.step(action)
        if done:
            print("Episode finished after {} timesteps".format(t+1))
            break
    env.close()
```

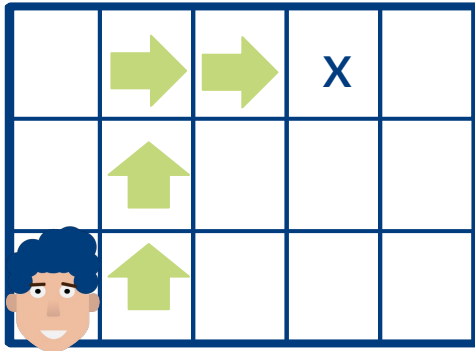


# Aufgabe 1: Einstieg in RL mit MENACE

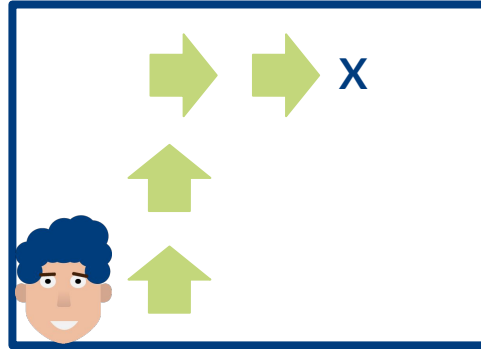
- › **Jupyter Lab Notebook**

# Zustands- und Aktionsräume

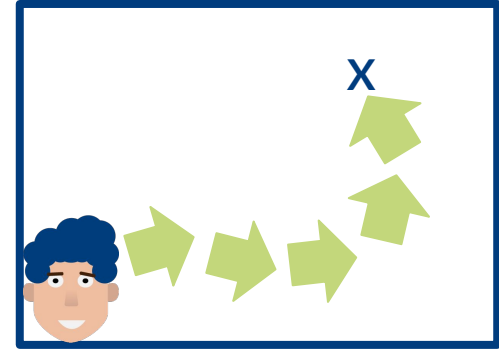
Wie unterscheiden sich diese Environments?



Zustände & Aktionen  
diskret



Zustände kontinuierlich &  
Aktionen diskret



Zustände & Aktionen  
kontinuierlich

# Aufgabe 2: CartPole Gym mit Q-Learning

- › **Jupyter Lab Notebook**

# Literatur

- › Kostenlose "Standard"-Lektüre für den Einstieg in RL:  
*Reinforcement Learning: An Introduction (Sutton and Barto)*, siehe <http://incompleteideas.net/book/RLbook2018.pdf>
- › Ausführlich und gut erklärter Einstieg in RL (Video-Lektionen):  
*UCL Course on RL (David Silver, Google DeepMind)*, siehe <https://www.davidsilver.uk/teaching/>
- › *Algorithms in Reinforcement Learning* von Csaba Szepesvári, siehe <https://sites.ualberta.ca/~szepesva/papers/RLAlgsInMDPs.pdf>
- › Blog mit Videos zum Einstieg in RL und Q-Learning, DQN und vieles mehr:  
*Reinforcement Learning – Introducing Goal Oriented Intelligence*, siehe <https://deeplizard.com/learn/video/nyjbcRQ-uQ8>

# Vielen Dank

Adrian Westermeier

[adrian.westermeier@inovex.de](mailto:adrian.westermeier@inovex.de)

Tim Bossenmaier

[tim.bossenmaier@inovex.de](mailto:tim.bossenmaier@inovex.de)

