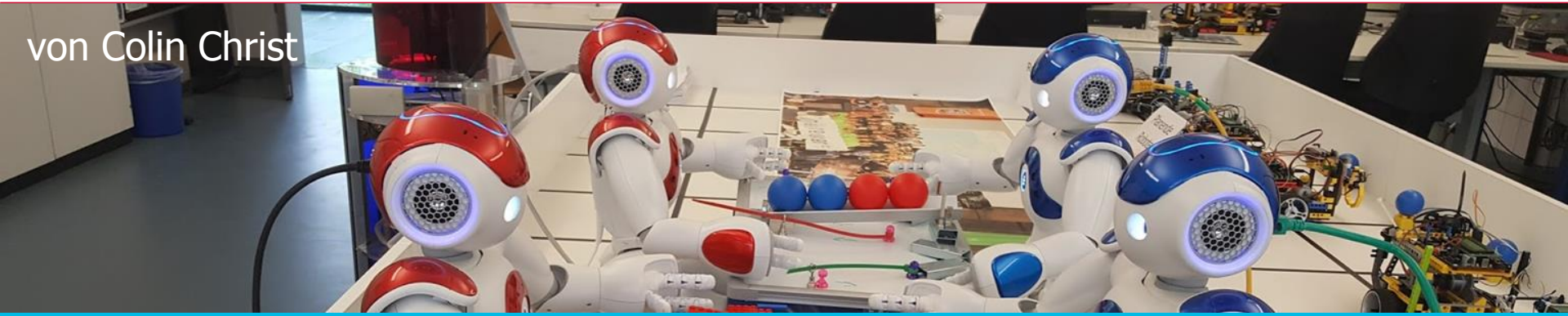
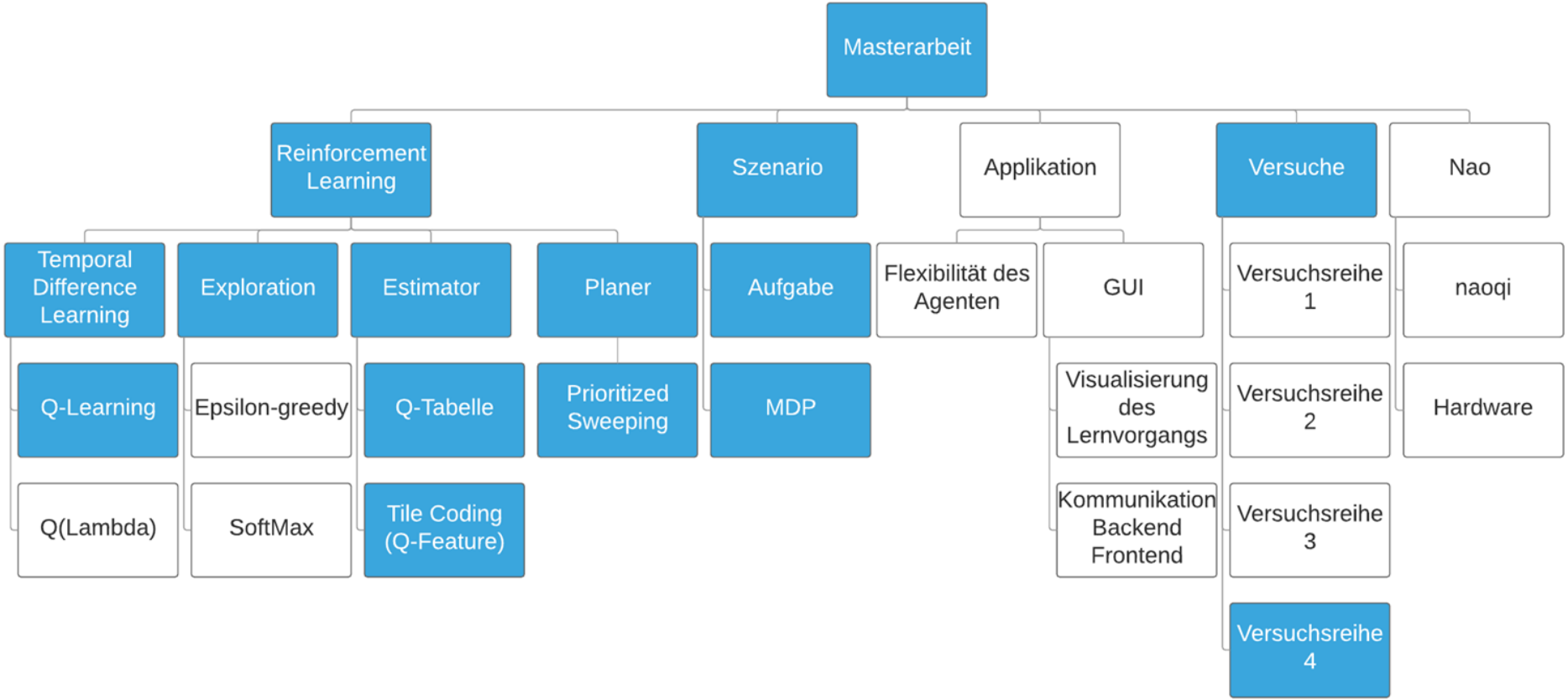
Real-time reinforcement learning von Handlungsstrategien für humanoide Roboter

von Colin Christ



Aufgabenstellung

- Entwicklung einer Applikation zur Demonstration von RL für humanoide Roboter
- Demonstration: Erlernen einer erfolgreichen Handlungsstrategie in einem realen Szenario
  - Szenario
    - soll weitestgehend deterministisch sein,
    - kann aber auch in seltenen Fällen stochastisch reagieren
- Der Lernvorgang soll visualisiert werden
- Das Lernen soll sowohl in der realen Welt als auch in simulierter Form möglich sein
- Eine erfolgreiche Handlungsstrategie soll in einer kurzen Zeit (unter einer Stunde) gefunden werden

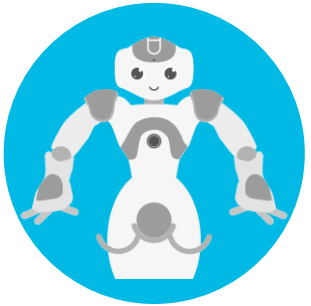

Masterarbeit
Reinforcement Learning
Szenario
Applikation
Versuche
Nao
Temporal Difference Learning
Exploration
Estimator
Planer
Aufgabe
Flexibilität des Agenten
GUI
Versuchsreihe 1
naoqi
Q-Learning
Epsilon-greedy
Q-Tabelle
Prioritized Sweeping
MDP
Visualisierung des Lernvorgangs
Versuchsreihe 2
Hardware
Q(Lambda)
SoftMax
Tile Coding (Q-Feature)
Kommunikation Backend Frontend
Versuchsreihe 3
Versuchsreihe 4

Was sollen Sie aus dieser Präsentation mitnehmen?

- Verständnis über Reinforcement Learnings (RL) und die Vor- und Nachteile gegenüber anderen Machine-Learning-Methoden
- Einblicke in das Projekt
- Übersicht, welche Methoden verwendet werden, um RL zu beschleunigen
- Sie wissen, welche Methoden in dieser Arbeit erfolgreich waren

Motivation - Was ist Reinforcement Learning?

Motivation - Was ist Reinforcement Learning?

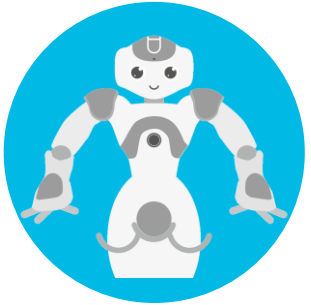


Agent

Motivation - Was ist Reinforcement Learning?

Du bist im Zustand (0,0,0,0,1) und hast 7 ausführbare Aktionen

Umgebung



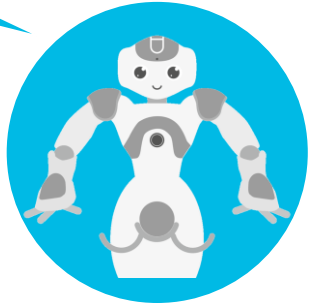
Agent

Motivation - Was ist Reinforcement Learning?

Du bist im Zustand (0,0,0,0,1) und hast 7 ausführbare Aktionen

Ich führe Aktion “Ball Ansehen” aus

Umgebung



Agent

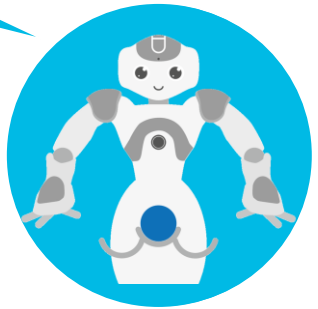
Motivation - Was ist Reinforcement Learning?

Du bist im Zustand (0,0,0,0,1) und hast 7 ausführbare Aktionen

Ich führe Aktion “Ball Ansehen” aus

Du bekommst einen Reward von -15, bist im Zustand (0,0,0,0,3) und hast 7 ausführbare Aktionen

Umgebung



Agent

Motivation - Was ist Reinforcement Learning?

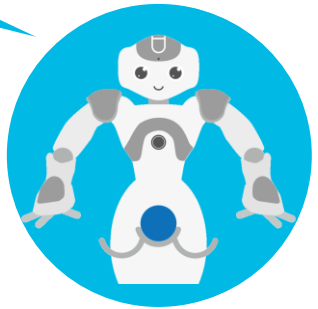
Du bist im Zustand (0,0,0,0,1) und hast 7 ausführbare Aktionen

Ich führe Aktion “Ball Ansehen” aus

Du bekommst einen Reward von -15, bist im Zustand (0,0,0,0,3) und hast 7 ausführbare Aktionen

Ich führe Aktion “Linker Arm Hoch” aus

Umgebung



Agent

Motivation - Was ist Reinforcement Learning?

Du bist im Zustand (0,0,0,0,1) und hast 7 ausführbare Aktionen

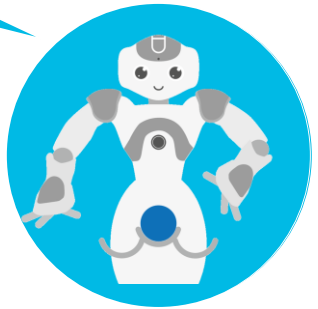
Ich führe Aktion “Ball Ansehen” aus

Du bekommst einen Reward von -15, bist im Zustand (0,0,0,0,3) und hast 7 ausführbare Aktionen

Ich führe Aktion “Linker Arm Hoch” aus

Du bekommst einen Reward von -15, bist im Zustand (0,0,1,0,3) und hast 10 ausführbare Aktionen

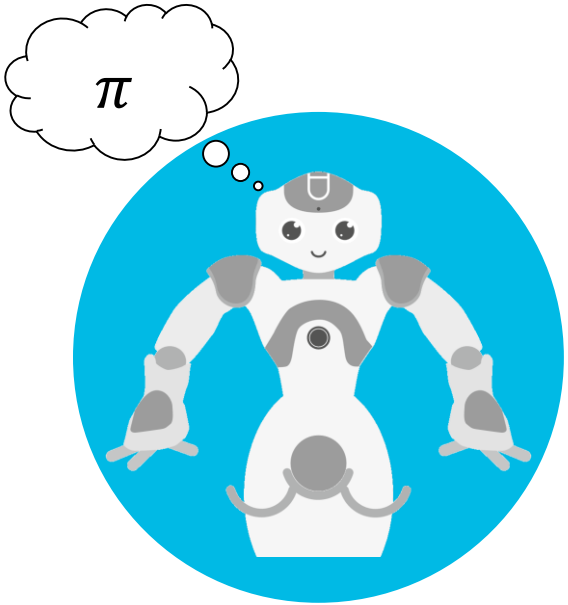
Umgebung



Agent

Motivation - Was ist Reinforcement Learning?

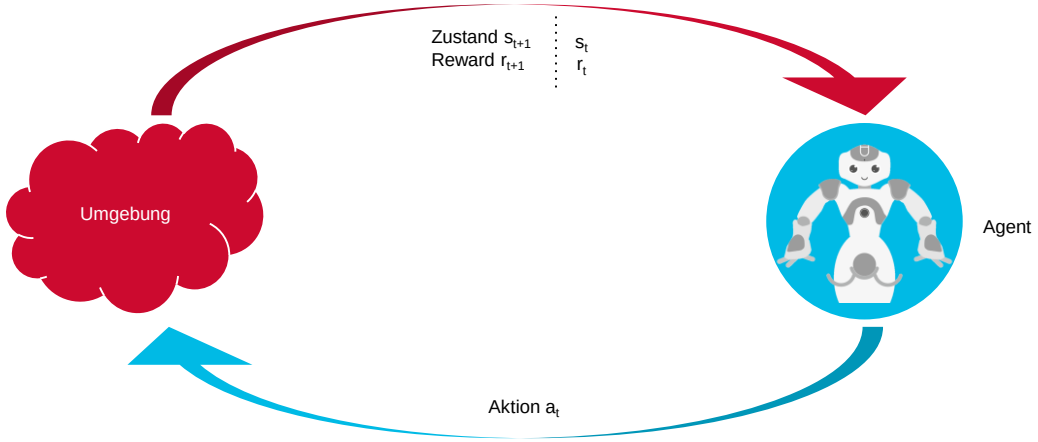
- Return ist der summierte Reward: $R_t = r_{t+1} + r_{t+2} + ... + r_T$
- Der Agent versucht seinen Return zu maximieren
- Dadurch entwickelt er eine Handlungsstrategie (Policy π)
- Die Policy bestimmt die nächste Aktion a in einem Zustand s
  - π: S -> A

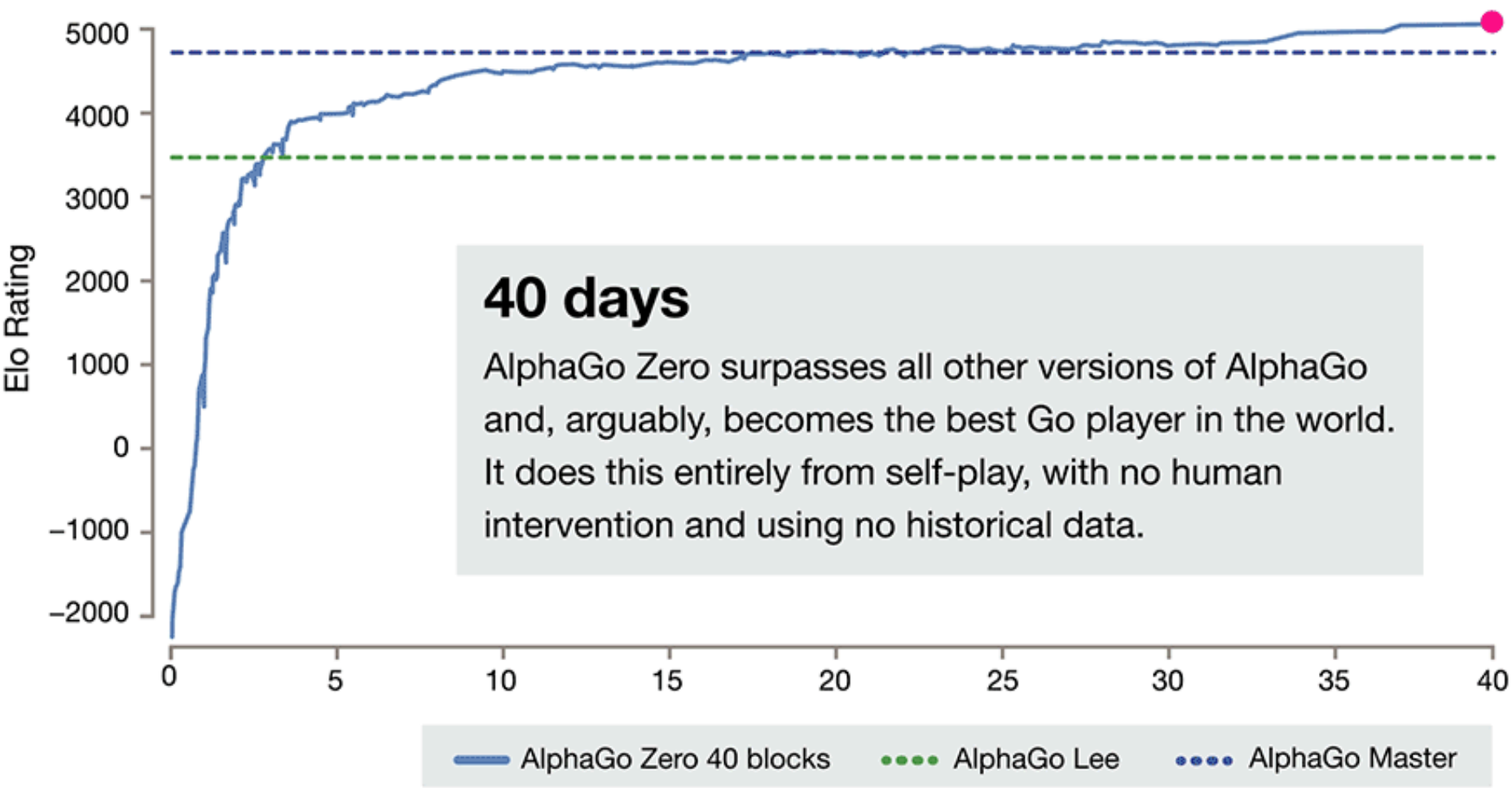


Sutton, Richard S. ; Barto, Andrew G.: Reinforcement Learning: An Introduction. MIT Press, 2012

Motivation - Warum ist Reinforcement Learning wichtig?

- Keine Trainingsmenge notwendig
  - Agent findet Lösungswege durch Trial and Error
- Es können unerwartete Lösungswege gefunden werden
  - Der Lösungsweg wird nicht durch eine Trainingsmenge vorgegeben



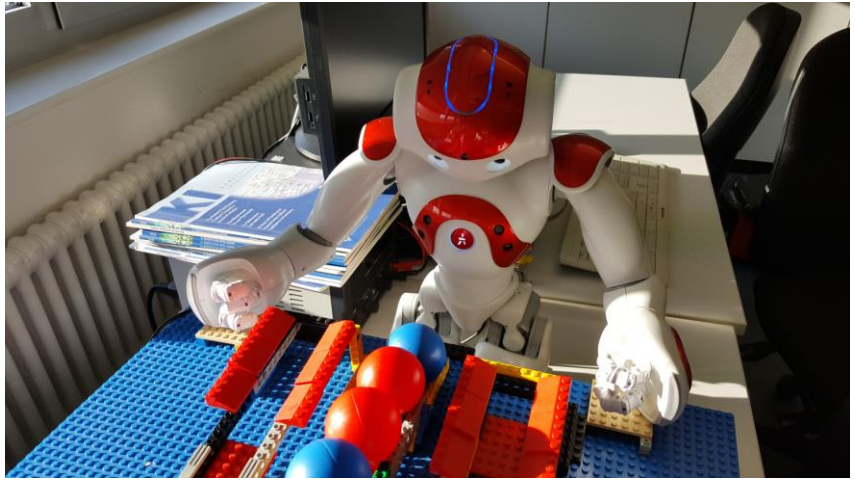


https://deepmind.com/blog/alphago-zero-learning-scratch/

Sergey Levine, Peter Pastor, Alex Krizhevsky und Deirdre Quillen: Learning Hand-Eye Coordination for Robotic Grasping with Deep Learning and Large-Scale Data Collection. (28 Aug 2016)

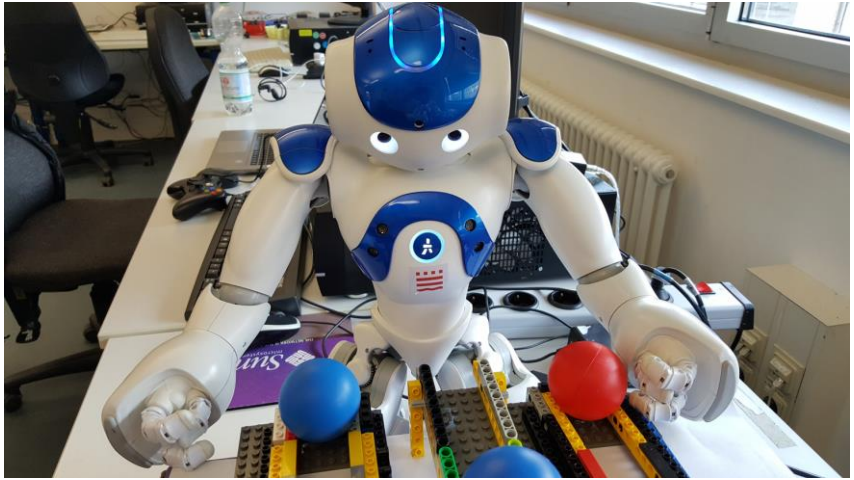
Szenario

Eve



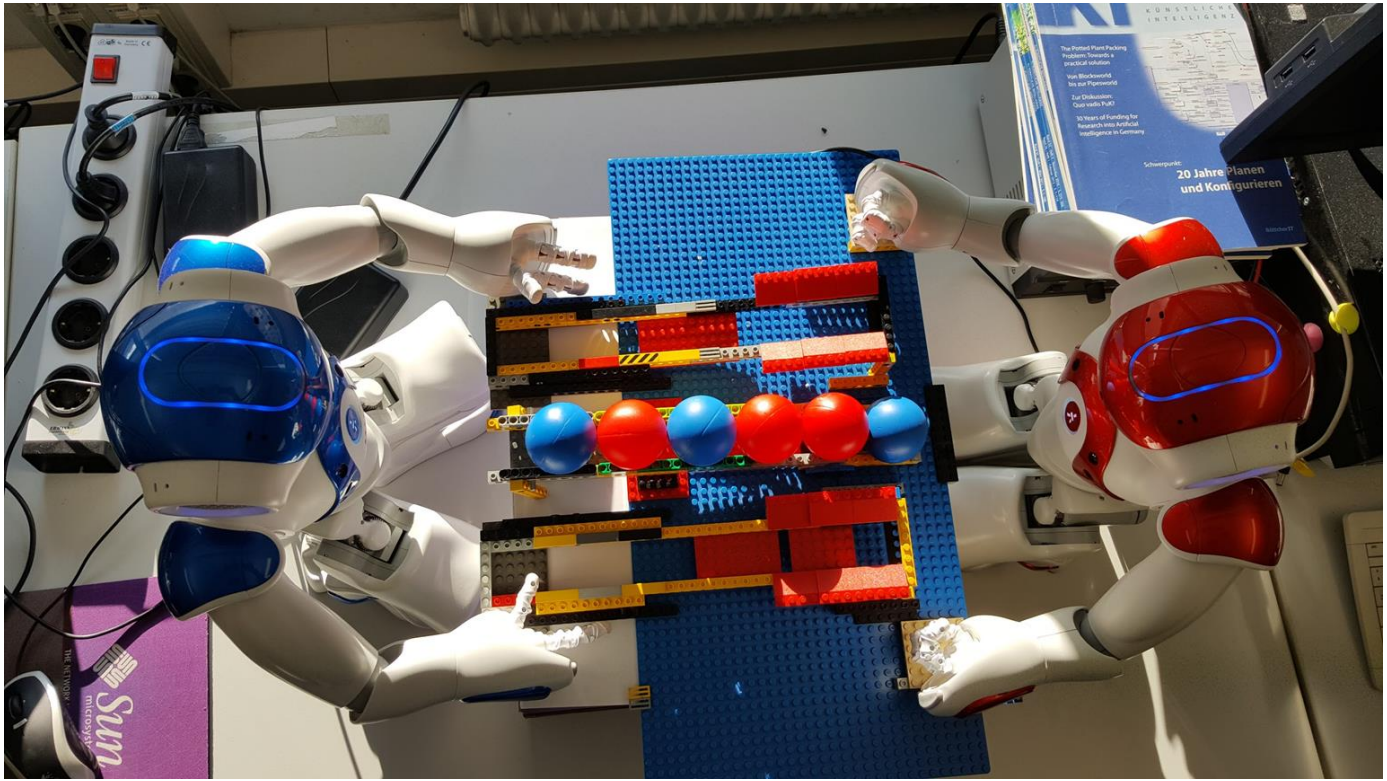
Agent / Lerner

David

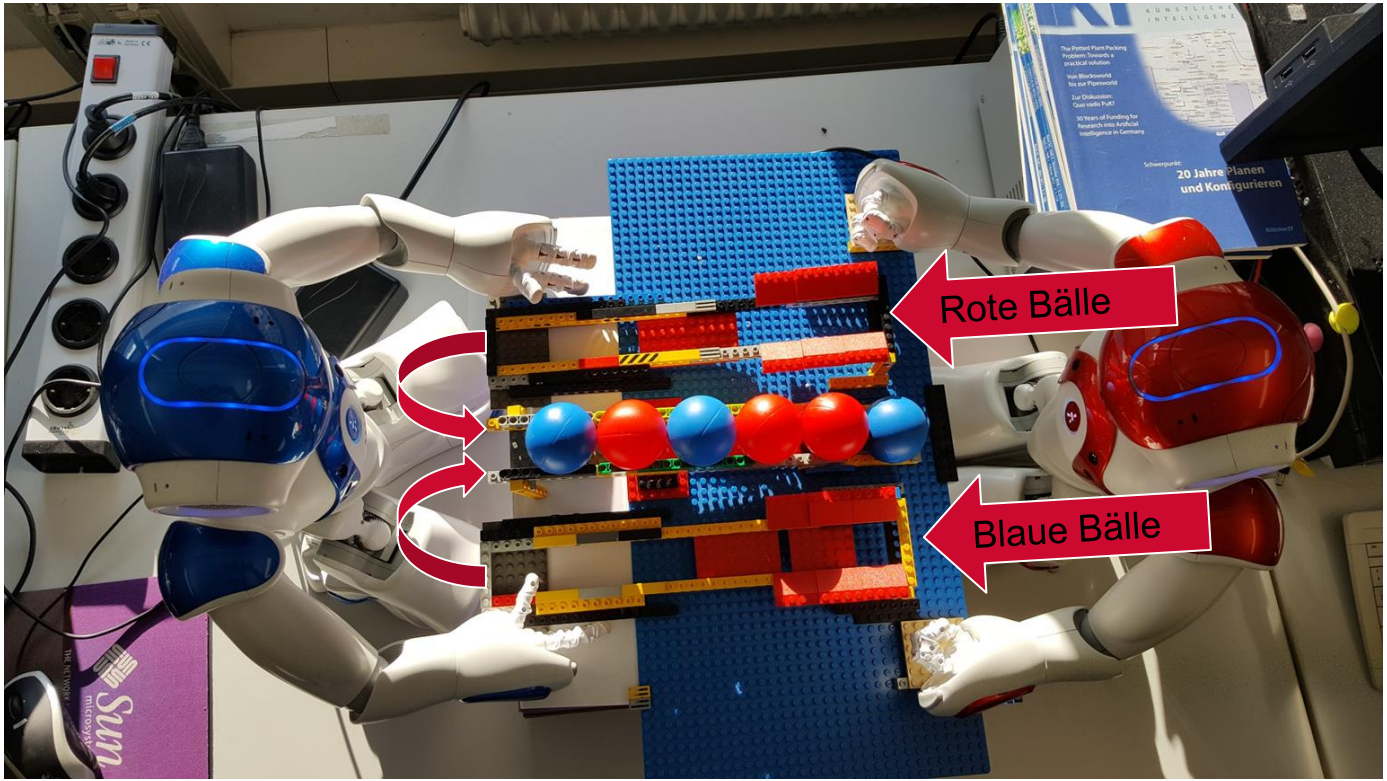


Validator / Ballzurücksortierer

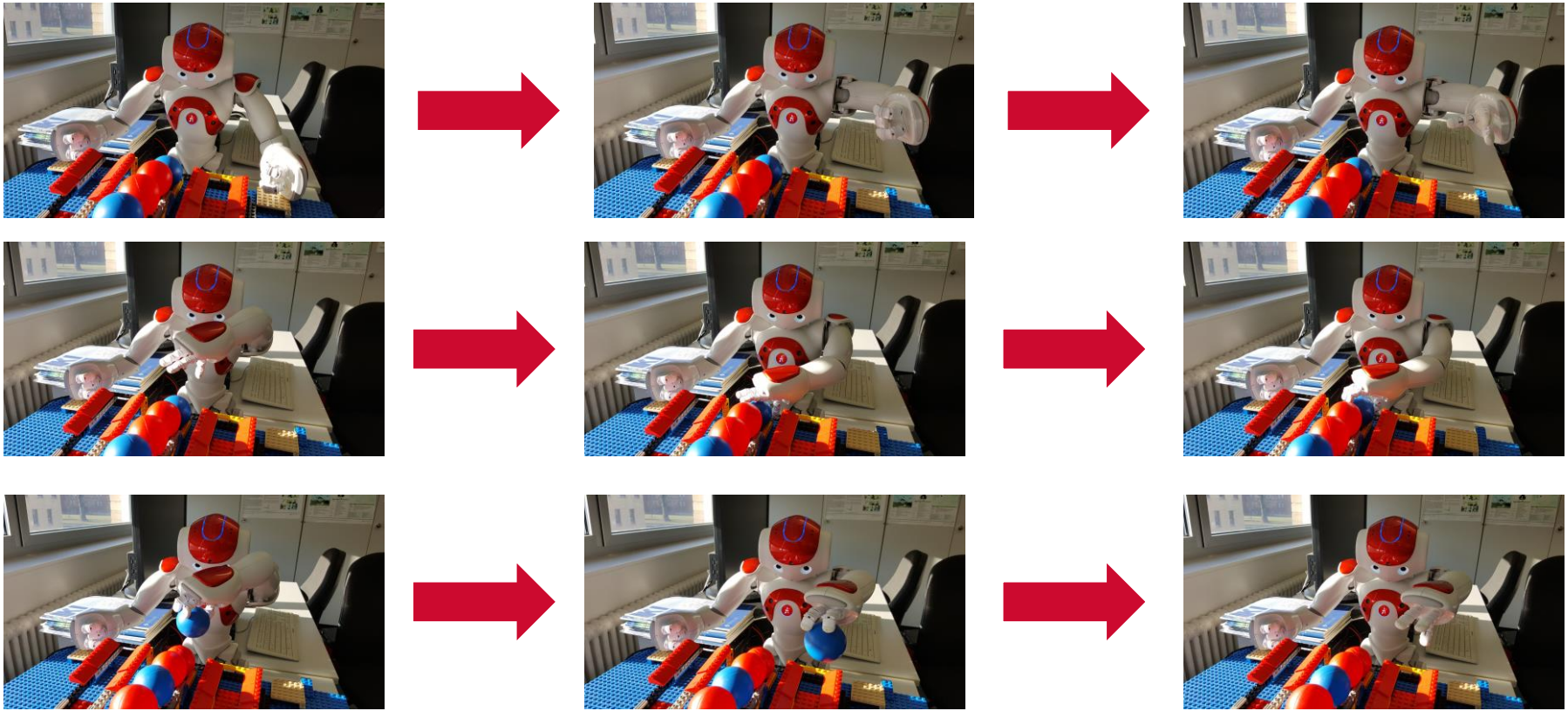
Szenario


The Potted Plant Packing Problem: Towards a practical solution
Von Blocksworld bis zur Pipelineworld
Zur Diskussion: Quo vadis PuK?
30 Years of Funding for Research into Artificial Intelligence in Germany
Schwerpunkt:
20 Jahre Planen und Konfigurieren

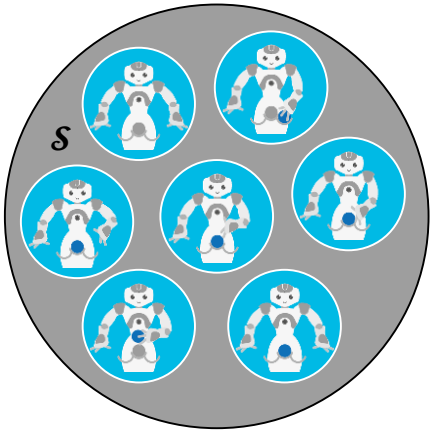
Szenario

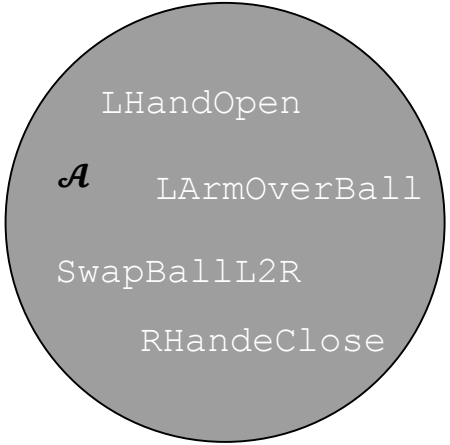


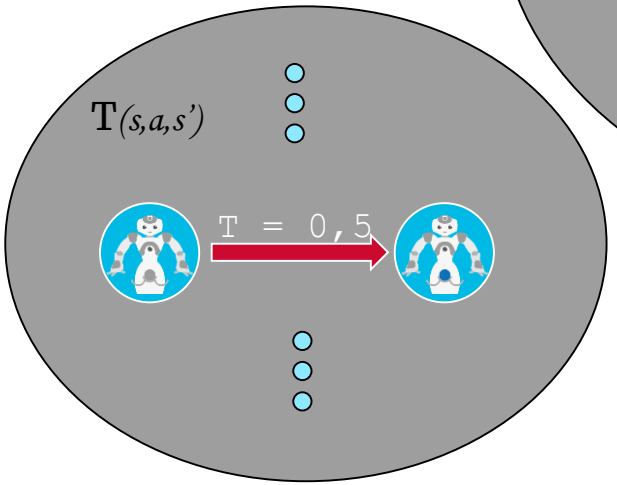
Szenario

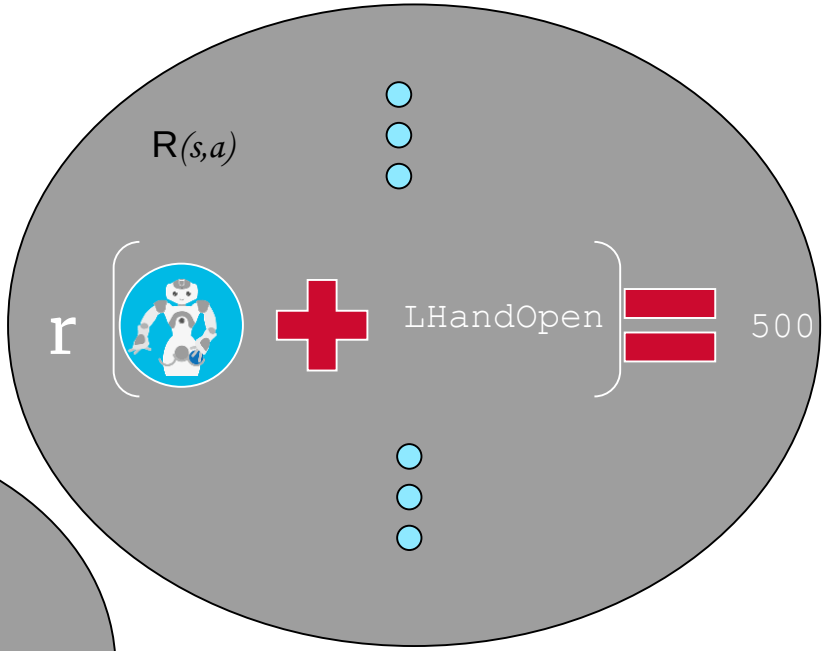


Szenario - MDP



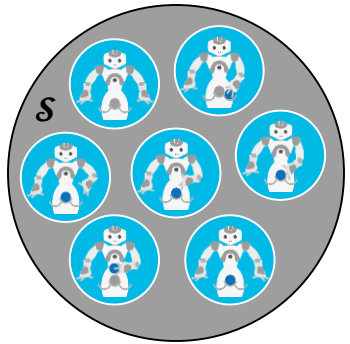




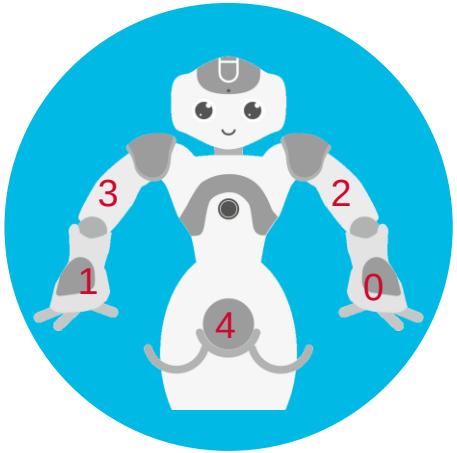


Kaelbling, Leslie P. ; Littman, Michael L. ; Moore, Andrew W.: Reinforcement Learning: A Survey. (1996)

Szenario - MDP - Zustandsraum

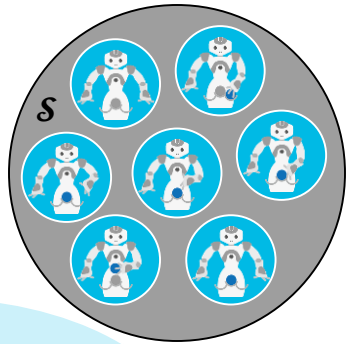


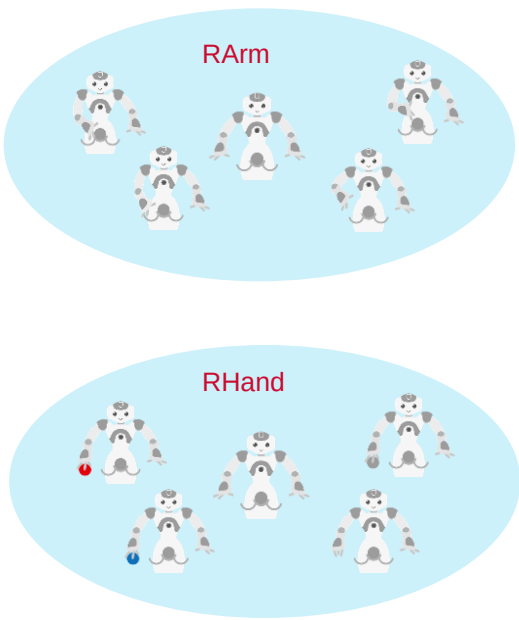
- 5 Dimensionen
- Diskret

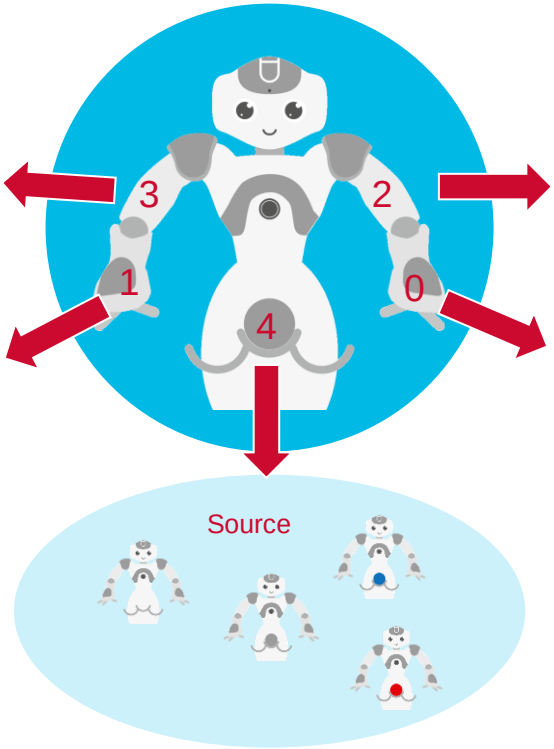


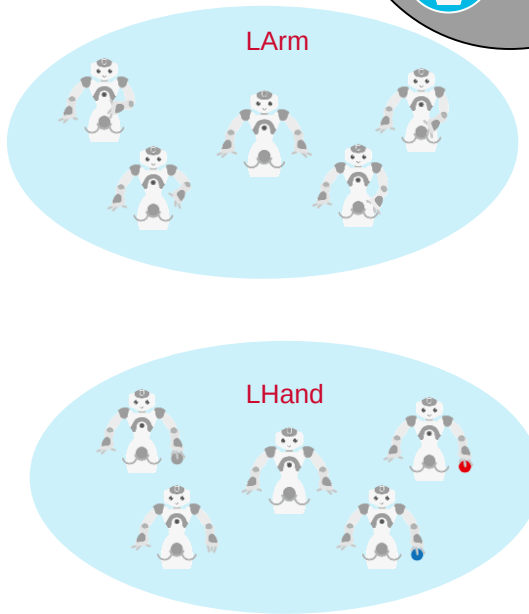
(LHand, RHand, LArm, RArm, Source)

Programm - MDP - Zustandsraum

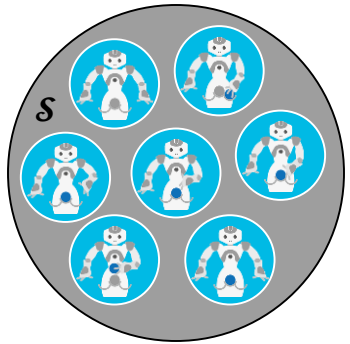


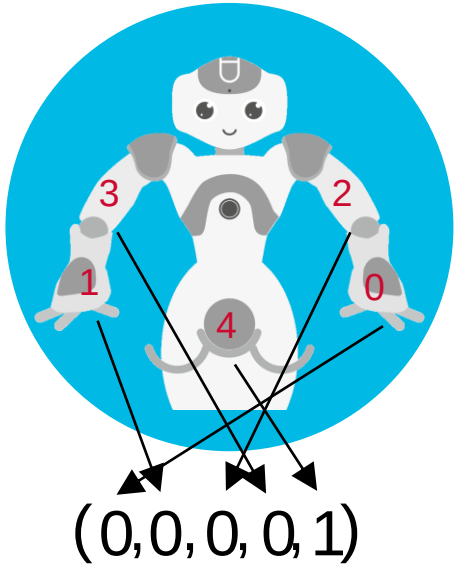






Szenario - MDP - Zustandsraum

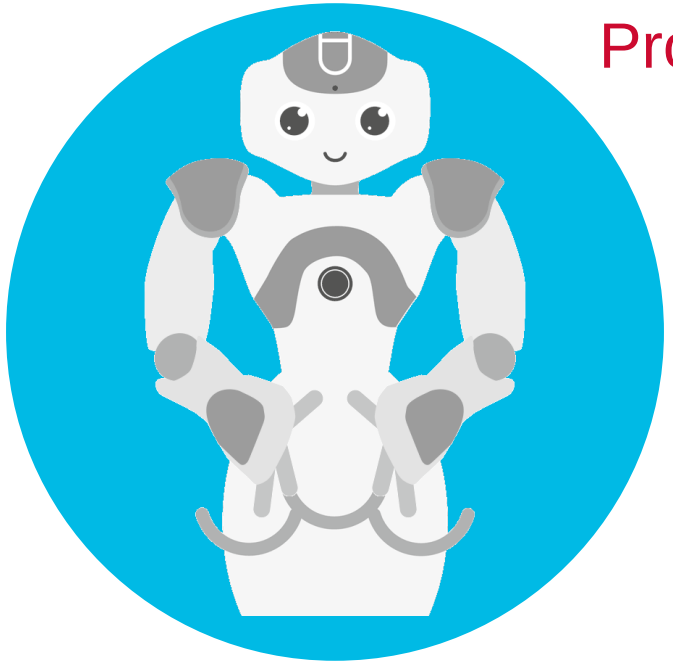




Szenario - MDP - Aktionsraum

	Name	Beschreibung
Hand Aktionen	LHandOpen	Öffnen der linken Hand
	RHandOpen	Öffnen der rechten Hand
	LHandClose	Schließen der linken Hand
	RHandeClose	Schließen der rechten Hand
Koordinaten anfahren	LArm2Start	Linker Arm bewegt sich zur Startposition
	RArm2Start	Rechter Arm bewegt sich zur Startposition
	LArm2Up	Linker Arm bewegt sich zur Position „Arm@Up“
	RArm2Up	Rechter Arm bewegt sich zur Position „Arm@Up“
	LArmOverBall	Linker Arm bewegt sich zur Position „ArmOverBall“
	RArmOverBall	Rechter Arm bewegt sich zur Position „ArmOverBall“
	LArm2Ball	Linker Arm bewegt sich zur Position „Arm@Ball“
	RArm2Ball	Rechter Arm bewegt sich zur Position „Arm@Ball“
	LArmOverDestination	Linker Arm bewegt sich zur Position „ArmOverDestination“
	RArmOverDestination	Rechter Arm bewegt sich zur Position „ArmOverDestination“
Ballfarbe erkennen	BallDetectAtSouce	Erkennt die Farbe des Balls auf der mittleren Schiene (Source)
	LBallDetect	Erkennt die Farbe des Balls in der linken Hand
	RBallDetect	Erkennt die Farbe des Balls in der rechten Hand
Ball in andere Hand tauschen	SwapBallL2R	Wechselt den Ball von der linken in die rechte Hand
	SwapBallR2L	Wechselt den Ball von der rechten in die linke Hand

Szenario - MDP - Aktionsraum



Problem!

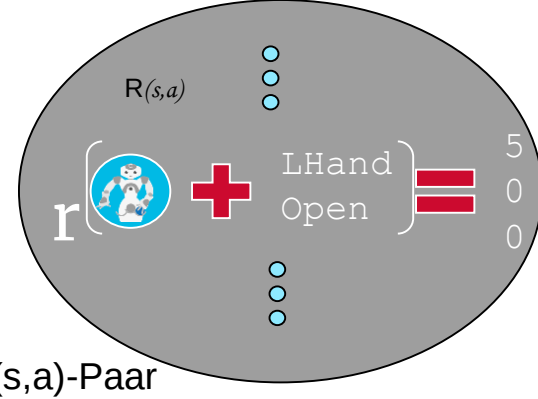
Szenario - MDP - Aktionsraum

Aktionen

	links		rechts	
Name	Hand	Arm	Hand	Arm
LHandOpen	Sonderfall			
RHandOpen	Sonderfall			
LHandClose	0			
RHandeClose			0	
LArm2Start		1, 4		
RArm2Start				1, 4
LArm2Up		0, 2, 4		
RArm2Up				0, 2, 4
LArmOverBall		1, 3, 4		0, 1, 4
RArmOverBall		0, 1, 4		1, 3, 4
LArm2Ball	0	2		0, 1, 4
RArm2Ball	0	0, 1, 4		2
LArmOverDestination		1, 2		
RArmOverDestination				1, 2
BallDetectAtSouce		0, 1, 4		0, 1, 4
LBallDetect	2, 3, 4	1, 4		
RBallDetect			2, 3, 4	1, 4
SwapBallL2R	2, 3, 4	1, 4	0, 1	1, 4
SwapBallR2L	0, 1	1, 4	2, 3, 4	1, 4

Zustands-Dimensionen

Szenario - MDP - Reward



- Weitgehend deterministisch
- Maximaler Reward: 500
- Minimaler Reward: -500
- Schrittreward: -15
- Höhe des Rewards wurde *willkürlich* gewählt
- Wichtig ist hierbei, dass der Schrittreward negativ ist
  - Maximaler Reward > Schrittreward > Minimaler Reward
- In der Realität erkennt der Validator ob der Ball richtig einsortiert wurde und vergibt den Reward

In der Simulation ist nur das (s,a)-Paar für die Vergabe des Rewards entscheidend:

r((4, X, 4, X, X), 'LHandOpen') = 500

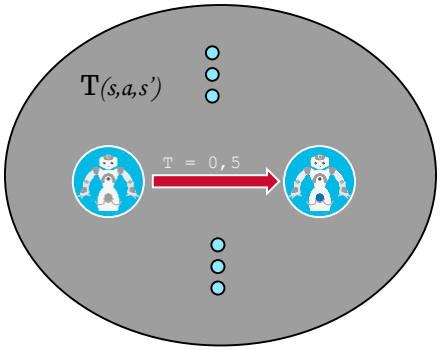
r((X, 3, X, 4, X), 'RHandOpen') = 500

r((X, 4, X, 4, X), 'RHandOpen') = - 500

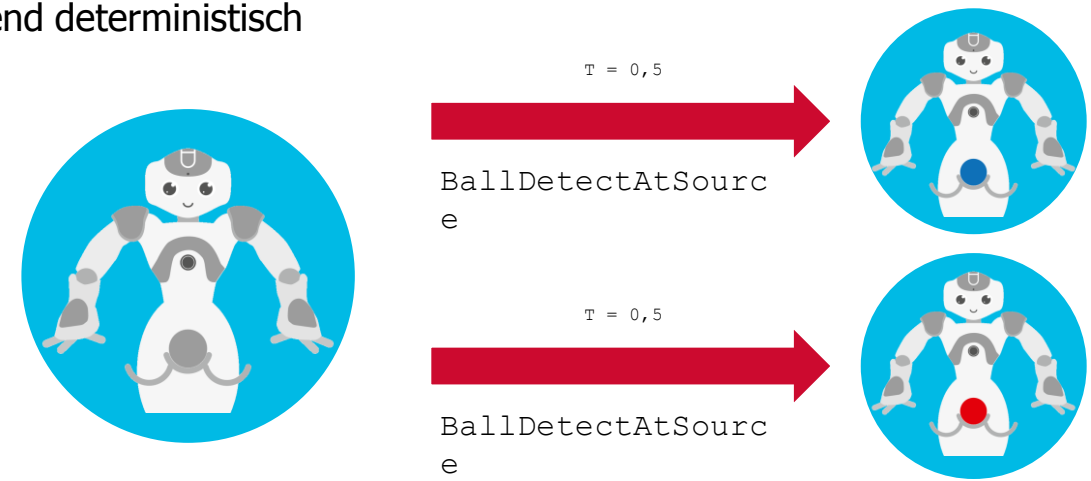
r((3, X, 4, X, X), 'LHandOpen') = - 500

Bei Einsortierung eines unbekannten Balls gibt mit einer Wahrscheinlichkeit von 0,5 einen maximalen -, und mit der Gegenwahrscheinlichkeit einen minimalen Reward

Szenario - MDP - Übergangswahrscheinlichkeit



- Realität
  - Immer stochastisch
- Simulation
  - Weitgehend deterministisch



Programm - Umgesetzte RL-Algorithmen

Temporal Difference Learning Funktionen

Estimator Funktionen

Explorator Funktionen

Planning Funktionen

Programm - Umgesetzte RL-Algorithmen
Temporal Difference Learning Funktionen

- Q-Learning
- Q(λ)

Programm - Umgesetzte RL-Algorithmen
Temporal Difference Learning Funktionen - Q-Learning

- Q gibt den erwarteten Return für die Ausführung einer Aktion in einem Zustand an
  - Q(s,a) = E(R)

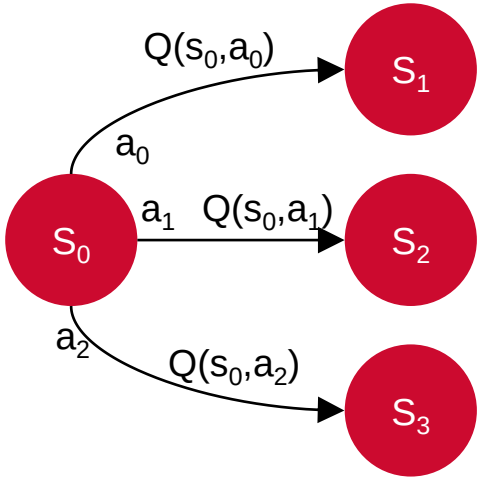
$$\pi^*(s) = \arg\max_a Q^*(s,a)$$

Diskontierungsfaktor

Lernrate

$$Q(s,a) \longleftarrow (1-\alpha)Q(s,a) + \alpha[r + \gamma \max_{a'} Q(s',a')]$$

Höchster Q-Wert in den Folgezuständen



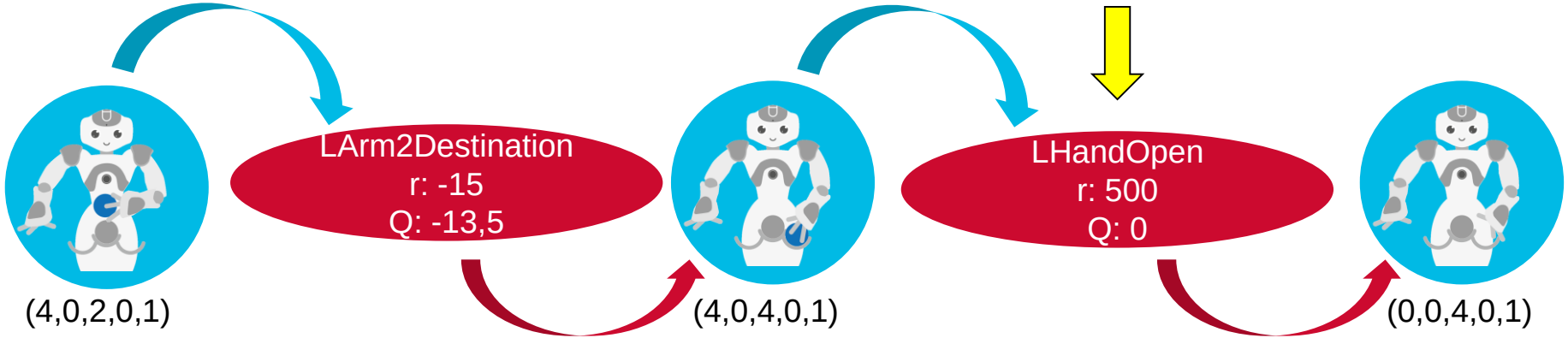
Watkins, Christopher John Cornish Hellaby: Learning from Delayed Rewards. Cambridge, UK, King's College, Diss., May 1989. http://www.cs.rhul.ac.uk/~chrisw/new_thesis.pdf

Programm - Umgesetzte RL-Algorithmen
Estimator Funktionen

- Q-Tabelle
- Tile Coding (Q-Feature)

Programm - Umgesetzte RL-Algorithmen
Estimator Funktionen - Q-Tabelle

Zustände \ Aktionen	LArm2Destination	LHandOpen
(4,0,2,0,1)	-13,5	0
(4,0,4,0,1)	0	0
(0,0,4,0,1)	0	0



Programm - Umgesetzte RL-Algorithmen
Estimator Funktionen - Q-Tabelle

$$Q(s, a) \leftarrow (1 - \alpha)Q(s, a) + \alpha[r + \gamma \max_{a'}(Q(s', a'))]$$

$$Q((4,0,4,0,1), \text{LHandOpen}) \leftarrow (1 - \alpha)Q((4,0,4,0,1), \text{LHandOpen}) + \alpha[r + \gamma \max_{a'}(Q(s', a'))]$$

$\alpha := 0{,}9$; $\gamma := 0{,}9$; $Q((4,0,4,0,1), \text{LHandOpen}) = 0$; $\max_{a'}(Q(s', a')) = 0$; $r = 500$

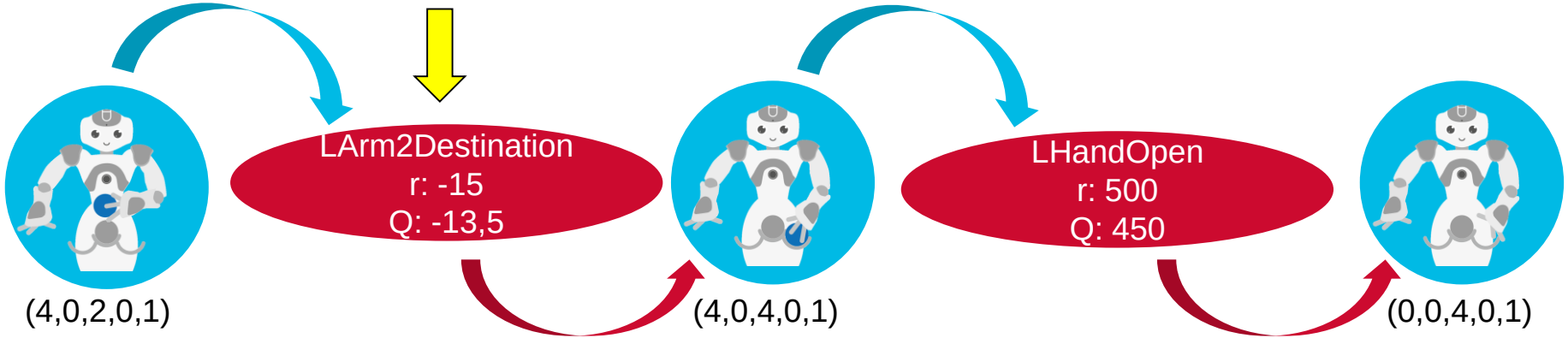
$$Q((4,0,4,0,1), \text{LHandOpen}) \leftarrow (1 - 0{,}9)0 + 0{,}9[500 + 0{,}9\ 0]$$

$$Q((4,0,4,0,1), \text{LHandOpen}) \leftarrow 0{,}9[500]$$

$$Q((4,0,4,0,1), \text{LHandOpen}) \leftarrow 450$$

Programm - Umgesetzte RL-Algorithmen
Estimator Funktionen - Q-Tabelle

Zustände \ Aktionen	LArm2Destination	LHandOpen
(4,0,2,0,1)	-13,5	0
(4,0,4,0,1)	0	450
(0,0,4,0,1)	0	0



Programm - Umgesetzte RL-Algorithmen
Estimator Funktionen - Q-Tabelle

$$Q(s, a) \leftarrow (1 - \alpha)Q(s, a) + \alpha[r + \gamma \max_{a'}(Q(s', a'))]$$

$$Q((4,0,2,0,1), \text{LArm2Destination}) \leftarrow (1 - \alpha)Q((4,0,2,0,1), \text{LArm2Destination}) + \alpha[r + \gamma \max_{a'}(Q(s', a'))]$$

$\alpha := 0,9$; $\gamma := 0,9$; $Q((4,0,2,0,1), \text{LArm2Destination}) = 13,5$; $\max_{a'}(Q(s', a')) = 450$; $r = -15$

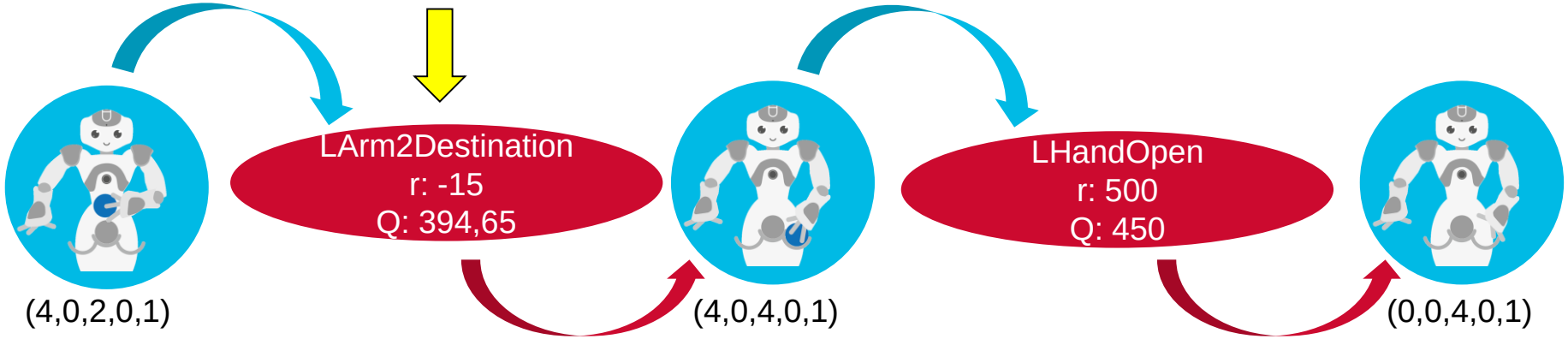
$$Q((4,0,2,0,1), \text{LArm2Destination}) \leftarrow (1 - 0,9)13,5 + 0,9[-15 + 0,9\ 450\ ]$$

$$Q((4,0,2,0,1), \text{LArm2Destination}) \leftarrow -1,35 + 0,9[390]$$

$$Q((4,0,2,0,1), \text{LArm2Destination}) \leftarrow 349,65$$

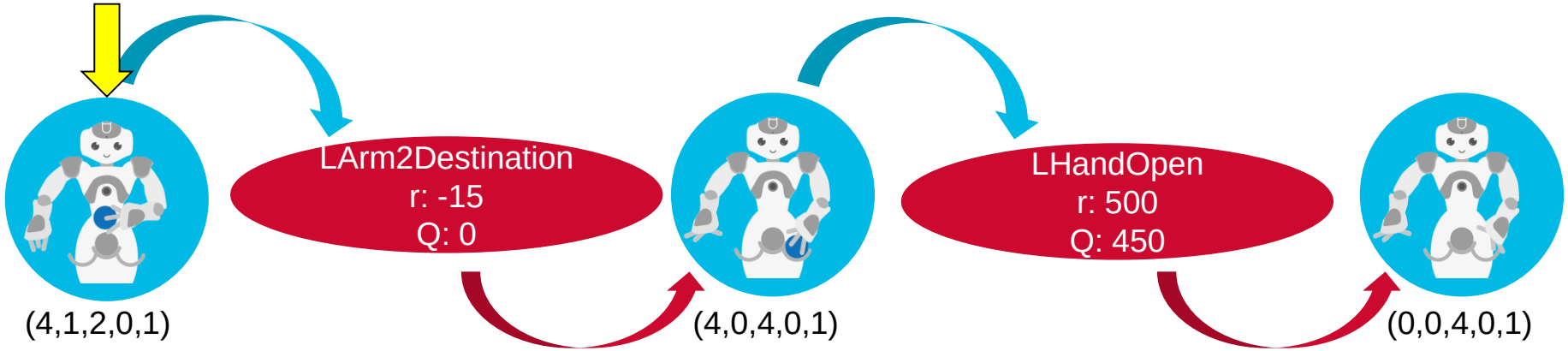
Programm - Umgesetzte RL-Algorithmen
Estimator Funktionen - Q-Tabelle

Zustände \ Aktionen	LArm2Destination	LHandOpen
(4,0,2,0,1)	349,65	0
(4,0,4,0,1)	0	450
(0,0,4,0,1)	0	0

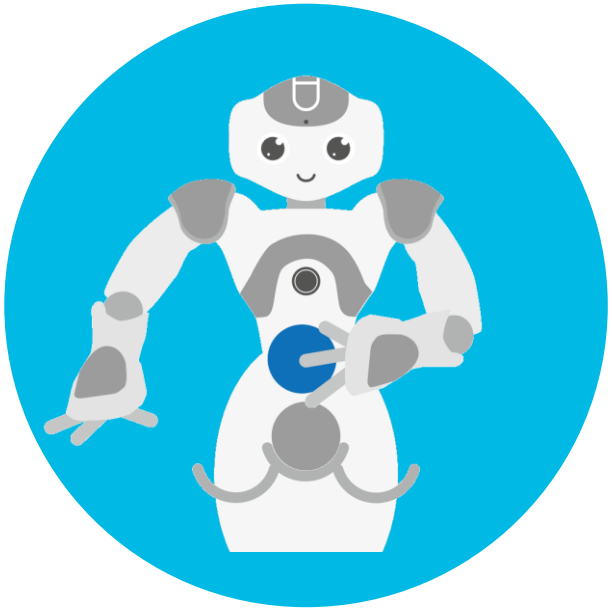


Programm - Umgesetzte RL-Algorithmen
Estimator Funktionen - Q-Tabelle

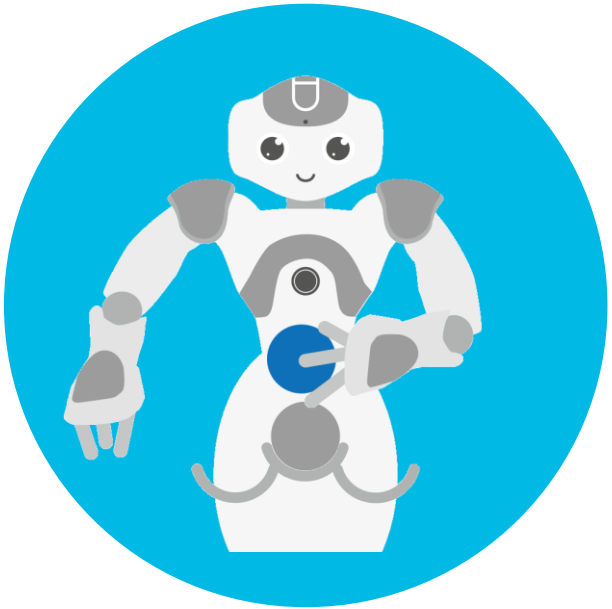
Zustände \ Aktionen	LArm2Destination	LHandOpen
(4,0,2,0,1)	349,65	0
(4,0,4,0,1)	0	450
(0,0,4,0,1)	0	0



Programm - Umgesetzte RL-Algorithmen
Estimator Funktionen - Tile Coding (Q-Feature)



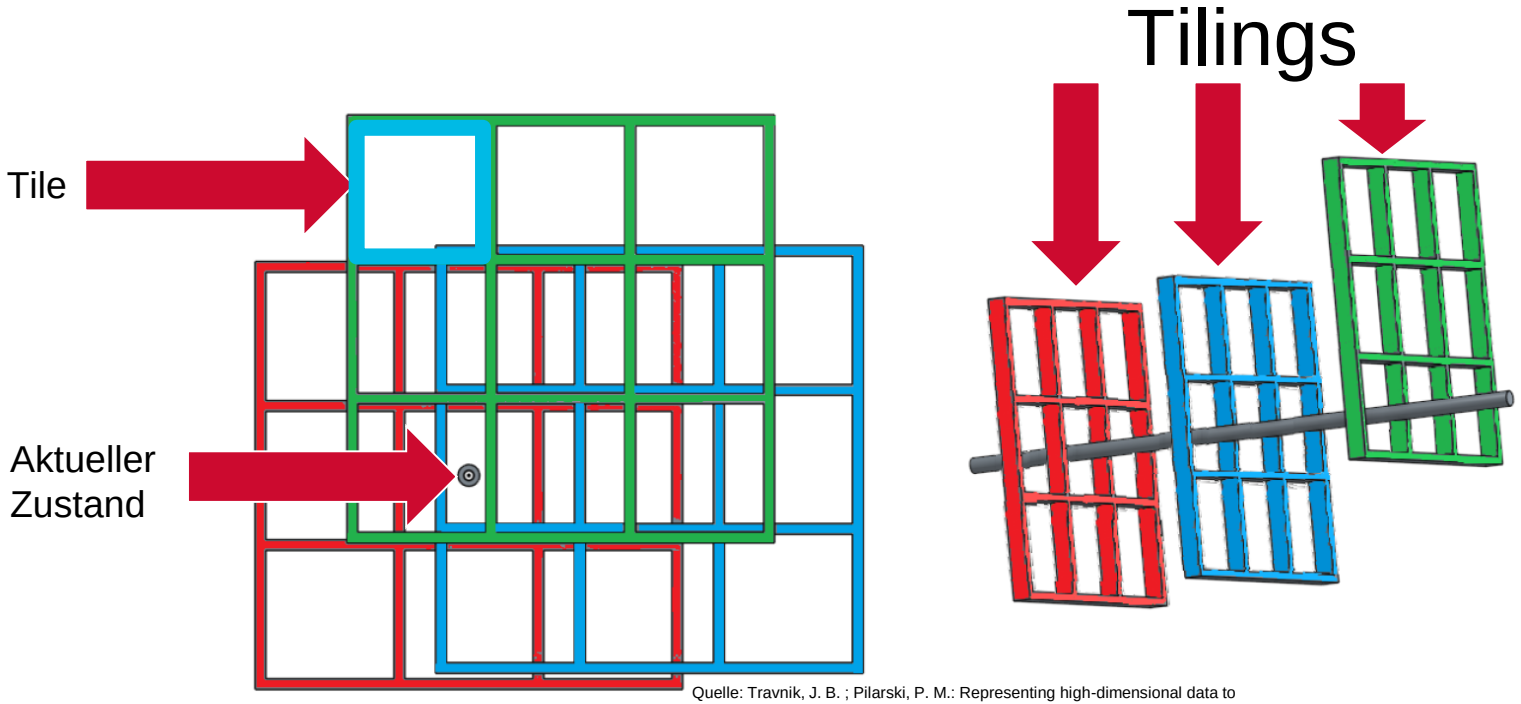
≅



Q((4,0,2,0,1), LArm2Destination) = 349,65

Q((4,1,2,0,1), LArm2Destination) ≅ 349,65

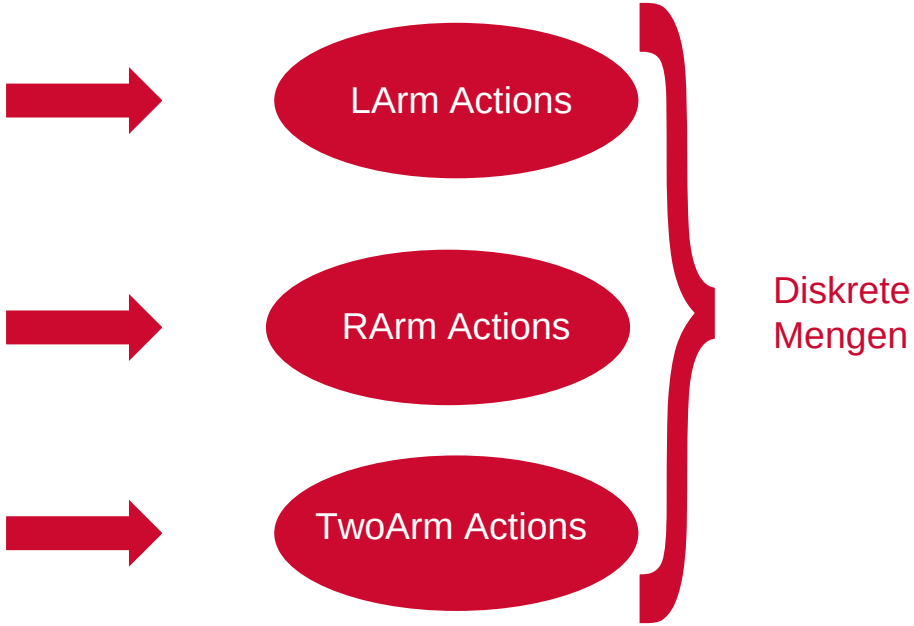
Programm - Umgesetzte RL-Algorithmen
Estimator Funktionen - Tile Coding (Q-Feature)



Quelle: Travnik, J. B. ; Pilarski, P. M.: Representing high-dimensional data to intelligent prostheses and other wearable assistive robots: A first comparison of tile coding and selective Kanerva coding. In: 2017 International Conference on Rehabilitation Robotics (ICORR), 2017, S. 1443–1450

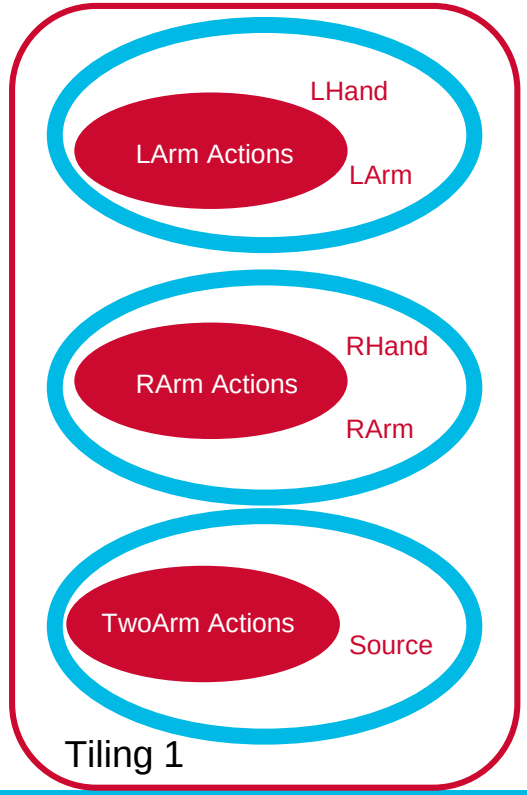
Programm - Umgesetzte RL-Algorithmen
Estimator Funktionen - Tile Coding (Q-Feature)

Name	Beschreibung
LHandOpen	Öffnen der linken Hand
RHandOpen	Öffnen der rechten Hand
LHandClose	Schließen der linken Hand
RHandeClose	Schließen der rechten Hand
LArm2Start	Linker Arm bewegt sich zur Startposition
RArm2Start	Rechter Arm bewegt sich zur Startposition
LArm2Up	Linker Arm bewegt sich zur Position „Arm@Up“
RArm2Up	Rechter Arm bewegt sich zur Position „Arm@Up“
LArmOverBall	Linker Arm bewegt sich zur Position „ArmOverBall“
RArmOverBall	Rechter Arm bewegt sich zur Position „ArmOverBall“
LArm2Ball	Linker Arm bewegt sich zur Position „Arm@Ball“
RArm2Ball	Rechter Arm bewegt sich zur Position „Arm@Ball“
LArmOverDestination	Linker Arm bewegt sich zur Position „ArmOverDestination“
RArmOverDestination	Rechter Arm bewegt sich zur Position „ArmOverDestination“
BallDetectAtSouce	Erkennt die Farbe des Balls auf der mittleren Schiene (Source)
LBallDetect	Erkennt die Farbe des Balls in der linken Hand
RBallDetect	Erkennt die Farbe des Balls in der rechten Hand
SwapBallL2R	Wechselt den Ball von der linken in die rechte Hand
SwapBallR2L	Wechselt den Ball von der rechten in die linke Hand

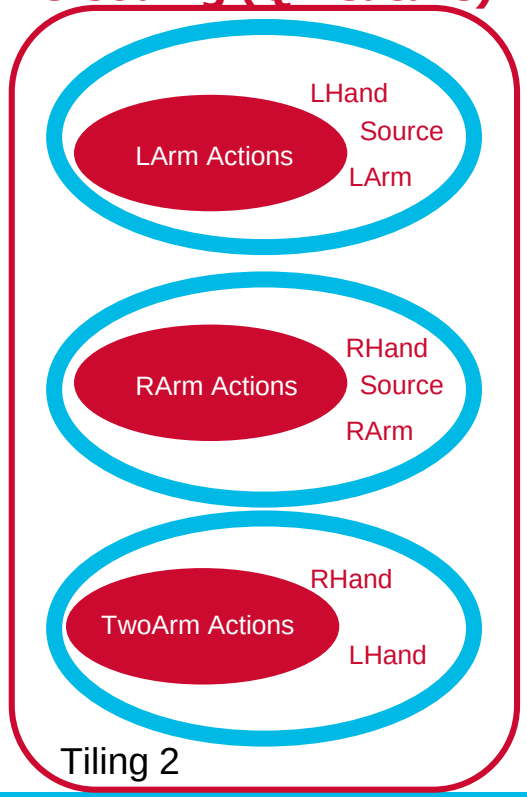


Programm - Umgesetzte RL-Algorithmen

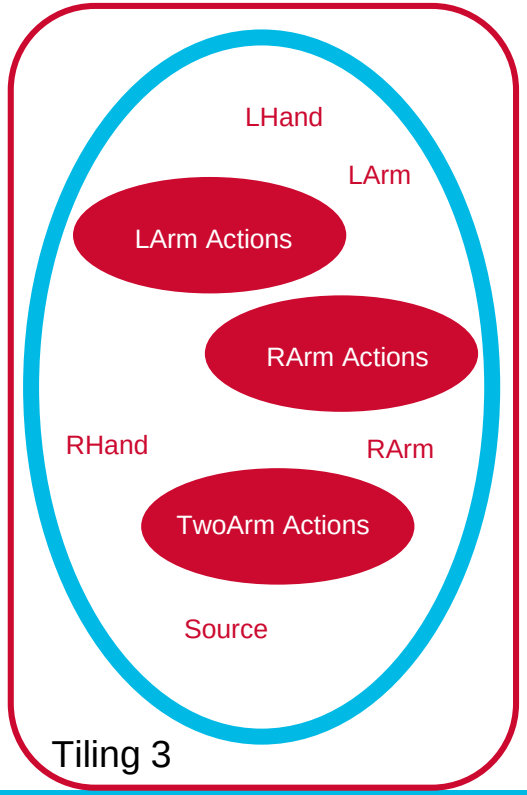
Estimator Funktionen - Tile Coding (Q-Feature)



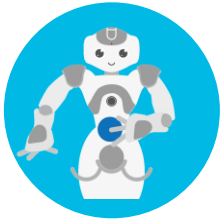
Tiling 1



Tiling 2



Tiling 3



Q((4,0,2,0,1), LArm2Destination) ← 349,65

Zustände \ Aktionen	LArm2Destination	LHandOpen
(4,X,2,X,X)	4,5	0
(4,X,4,X,X)	0	150
(0,X,4,X,X)	0	0

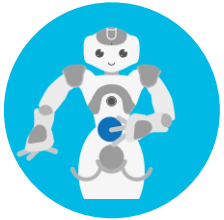
Tiling 1

Zustände \ Aktionen	LArm2Destination	LHandOpen
(4,X,2,X,1)	4,5	0
(4,X,4,X,1)	0	150
(0,X,4,X,1)	0	0

Tiling 2

Zustände \ Aktionen	LArm2Destination	LHand Open
(4,0,2,0,1)	4,5	0
(4,0,4,0,1)	0	150
(0,0,4,0,1)	0	0

Tiling 3



Q((4,0,2,0,1), LArm2Destination) ← 349,65

: 3 : 3 : 3

Zustände \ Aktionen	LArm2Destination	LHandOpen
(4,X,2,X,X)	116,55	0
(4,X,4,X,X)	0	150
(0,X,4,X,X)	0	0

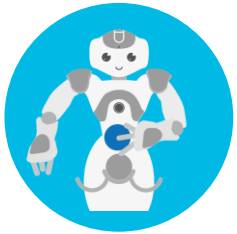
Tiling 1

Zustände \ Aktionen	LArm2Destination	LHandOpen
(4,X,2,X,1)	116,55	0
(4,X,4,X,1)	0	150
(0,X,4,X,1)	0	0

Tiling 2

Zustände \ Aktionen	LArm2Destination	LHand Open
(4,0,2,0,1)	116,55	0
(4,0,4,0,1)	0	150
(0,0,4,0,1)	0	0

Tiling 3



Q((4,1,2,0,1), LArm2Destination) = 233,1

+ + +

Zustände \ Aktionen	LArm2Destination	LHandOpen
(4,X,2,X,X)	116,55	0
(4,X,4,X,X)	0	150
(0,X,4,X,X)	0	0

Tiling 1

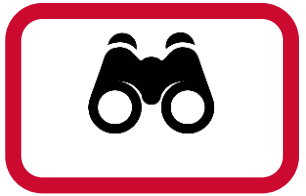
Zustände \ Aktionen	LArm2Destination	LHandOpen
(4,X,2,X,1)	116,55	0
(4,X,4,X,1)	0	150
(0,X,4,X,1)	0	0

Tiling 2

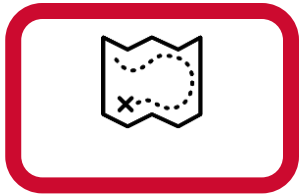
Zustände \ Aktionen	LArm2Destination	LHand Open
(4,0,2,0,1)	116,55	0
(4,0,4,0,1)	0	150
(4,1,2,0,1)	0	0

Tiling 3

Programm - Umgesetzte RL-Algorithmen
Explorator Funktionen



- ε-geedy
- softmax



Programm - Umgesetzte RL-Algorithmen
Planning Funktionen - Prioritized Sweeping

- Benötigt ein Modell der Umgebung
  - M(S,A) ← R, S’
- Annahme: (s,a)-Paare, deren Q-Wert sich stark geändert hat, haben vorangegangene (s’,a’)-Paare, deren Q-Wert sich auch stark ändert
  - Solche (s’,a’)-Paare müssen mit hoher Priorität, entsprechend der neuen Erfahrung, angepasst werden.

Initialize $Q(s,a)$, $Model(s,a)$, for all s, a, and $PQueue$ to empty
Do forever:

(a) $S \leftarrow$ current (nonterminal) state

(b) $A \leftarrow policy(S, Q)$

(c) Execute action A; observe resultant reward, R, and state, S'

(d) $Model(S, A) \leftarrow R, S'$

(e) $P \leftarrow |R + \gamma \max_a Q(S', a) - Q(S, A)|$.

(f) if $P > \theta$, then insert S, A into $PQueue$ with priority P

(g) Repeat n times, while $PQueue$ is not empty:

$S, A \leftarrow first(PQueue)$

$R, S' \leftarrow Model(S, A)$

$Q(S, A) \leftarrow Q(S, A) + \alpha[R + \gamma \max_a Q(S', a) - Q(S, A)]$

Repeat, for all $\bar{S}, \bar{A}$ predicted to lead to S:

$\bar{R} \leftarrow$ predicted reward for $\bar{S}, \bar{A}, S$

$P \leftarrow |\bar{R} + \gamma \max_a Q(S, a) - Q(\bar{S}, \bar{A})|$.

if $P > \theta$ then insert $\bar{S}, \bar{A}$ into $PQueue$ with priority P

Sutton, Richard S. ; Barto, Andrew G.: Reinforcement Learning: An Introduction. MIT Press, 2012

Versuche

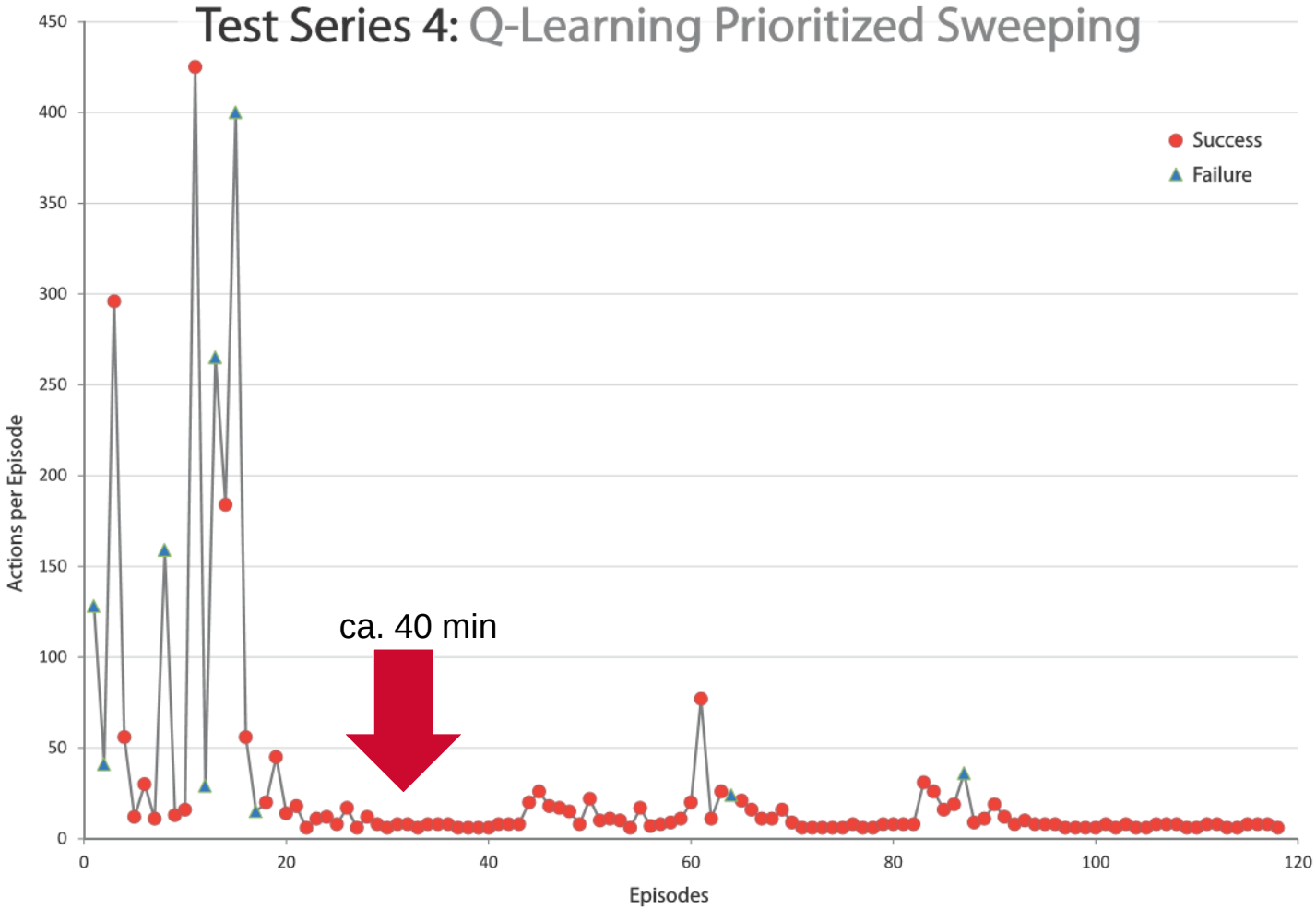
- Alle Versuche nur simuliert
- Laufzeit 1h
- jede Aktion hat 1 sec delay

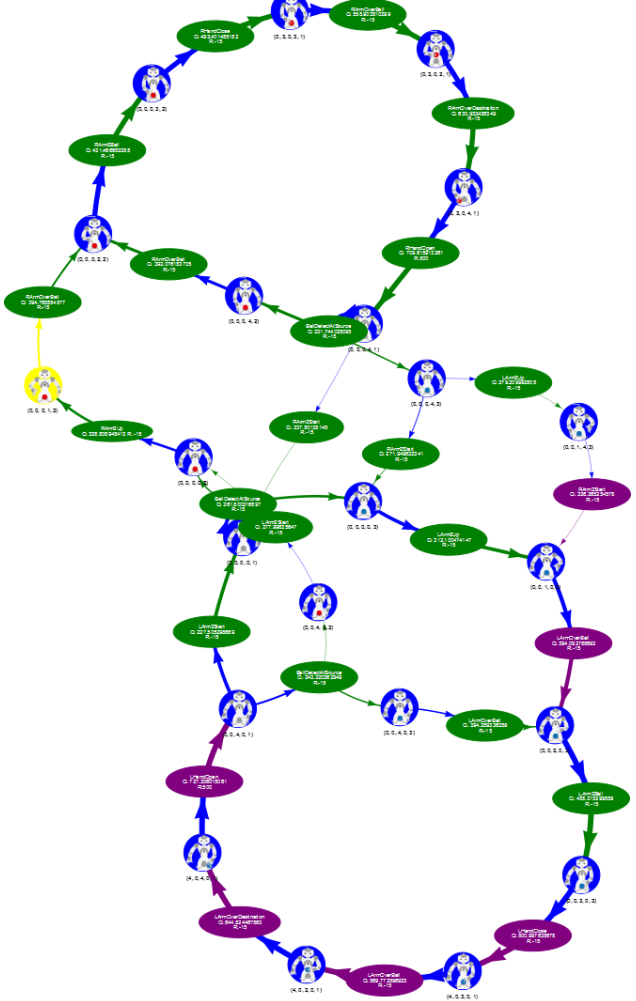
Parameter	Versuchsreihe 1	Versuchsreihe 2	Versuchsreihe 3	Versuchsreihe 4
α	0,7	0,1	0,01	0,9
γ	0,7	0,7	0,7	0,9

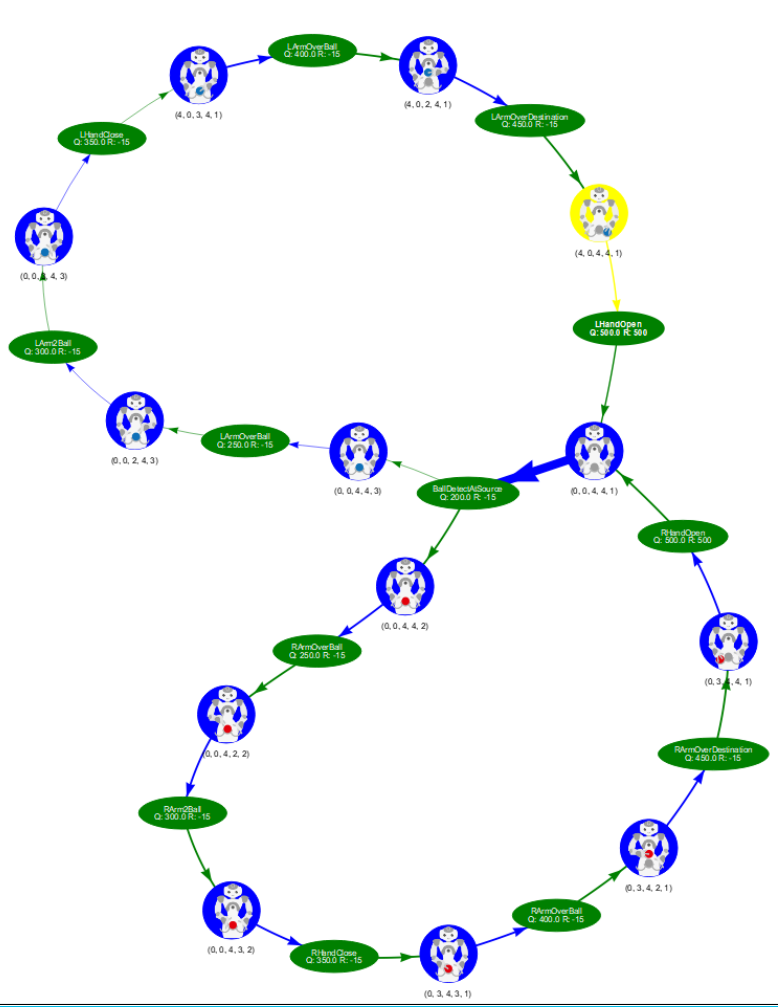
Versuche	1	2	3	4	5
Esimator	Q-Tabelle	Q-Tabelle	Q-Tabelle	Q-Feature	Q-Feature
TD-Learning Funktion	Q-Update	Q-Lambda	Q-Update	Q-Feature-Update	Q-Feature-Update
Planende Funktion	–	–	PS	–	PS

Versuche

- α = 0,9
- γ = 0,9






LArmOverBall
Q: 400.0 R: -15
LArmOverDestination
Q: 450.0 R: -15
LHandClose
Q: 350.0 R: -15
LHandOpen
Q: 500.0 R: 500
LArm2Ball
Q: 300.0 R: -15
LArmOverBall
Q: 250.0 R: -15
BallDetectASource
Q: 200.0 R: -15
RHandOpen
Q: 500.0 R: 500
RArmOverBall
Q: 250.0 R: -15
RArmOverDestination
Q: 450.0 R: -15
RArm2Ball
Q: 300.0 R: -15
RHandClose
Q: 350.0 R: -15
RArmOverBall
Q: 400.0 R: -15

Ergebnis/Fazit

- Programm zur Demonstration von RL mit humanoiden Robotern
- Q-Learning mit PS (α = 0,9 γ = 0,9) hat gute Ergebnisse geliefert
  - Eine gute Policy kann in unter einer Stunde erreicht werden, allerdings ist diese nicht perfekt
  - Verschiedene Methoden des RL lassen sich kombinieren, um das Lernen zu beschleunigen
- Tile Coding (Q-Feature) hat schnell Lösungswege gefunden, aber sich danach nicht weiter verbessert
- Q(λ) lieferte durchweg schlechte Ergebnisse -> evtl. sollte eine andere Form des Algorithmus genutzt werden

Ausblick

- Die Arbeit ist abgeschlossen, jedoch das Projekt ist noch nicht beendet
- Bestehende Algorithmen wie Tile Coding oder Q(λ) können noch verbessert werden
- Neue RL-Methoden können hinzugefügt werden
- Das Überhitzungsproblem kann durch RL gelöst werden

Demo


SONY