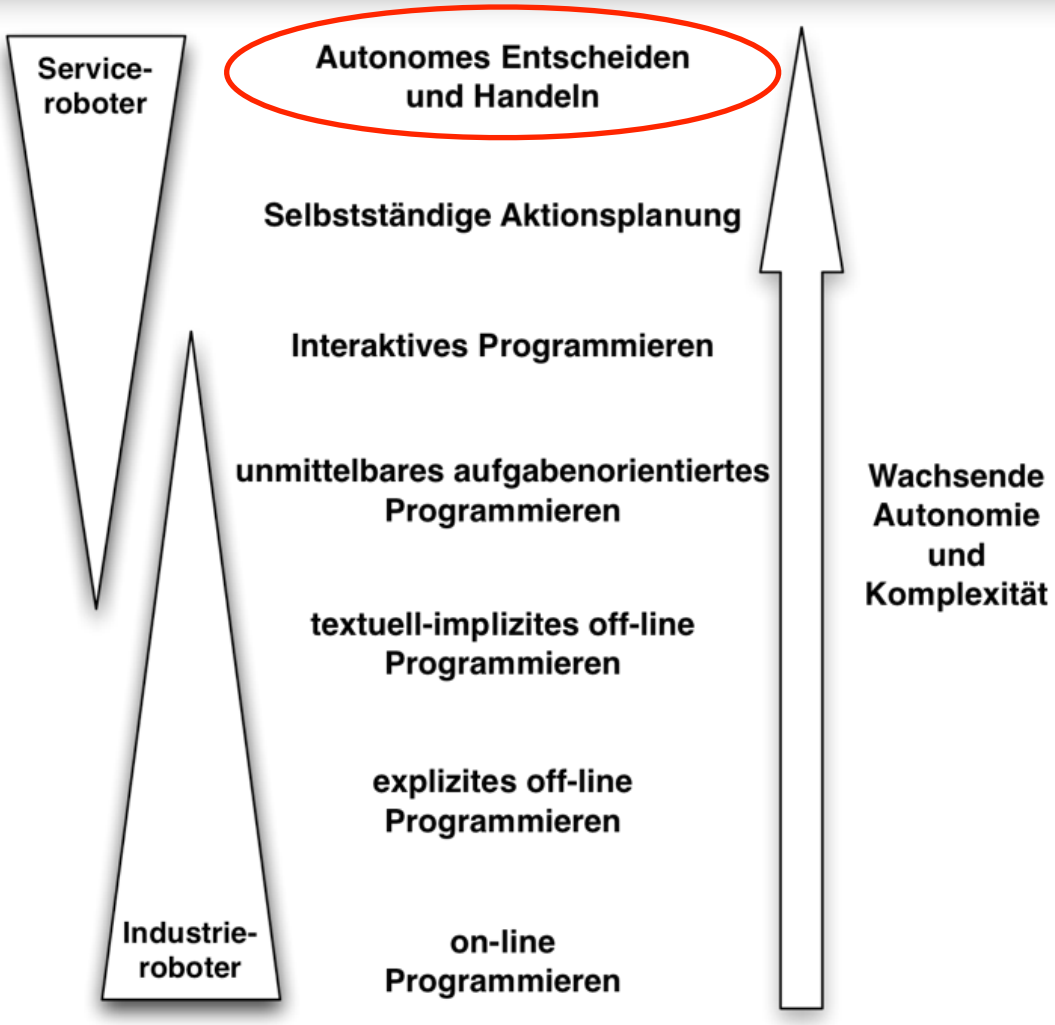
Probabilistische Entscheidungsverfahren I

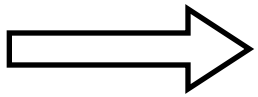
Einordnung: Entscheiden



Entscheiden

Fundamentale Fragen:

- Was ist Entscheiden?
- Welche Arten von Systemen fällen Entscheidungen?
- Welches ist der allgemeinste Rahmen für eine Definition von Entscheidungssystemen?



rationaler Agent

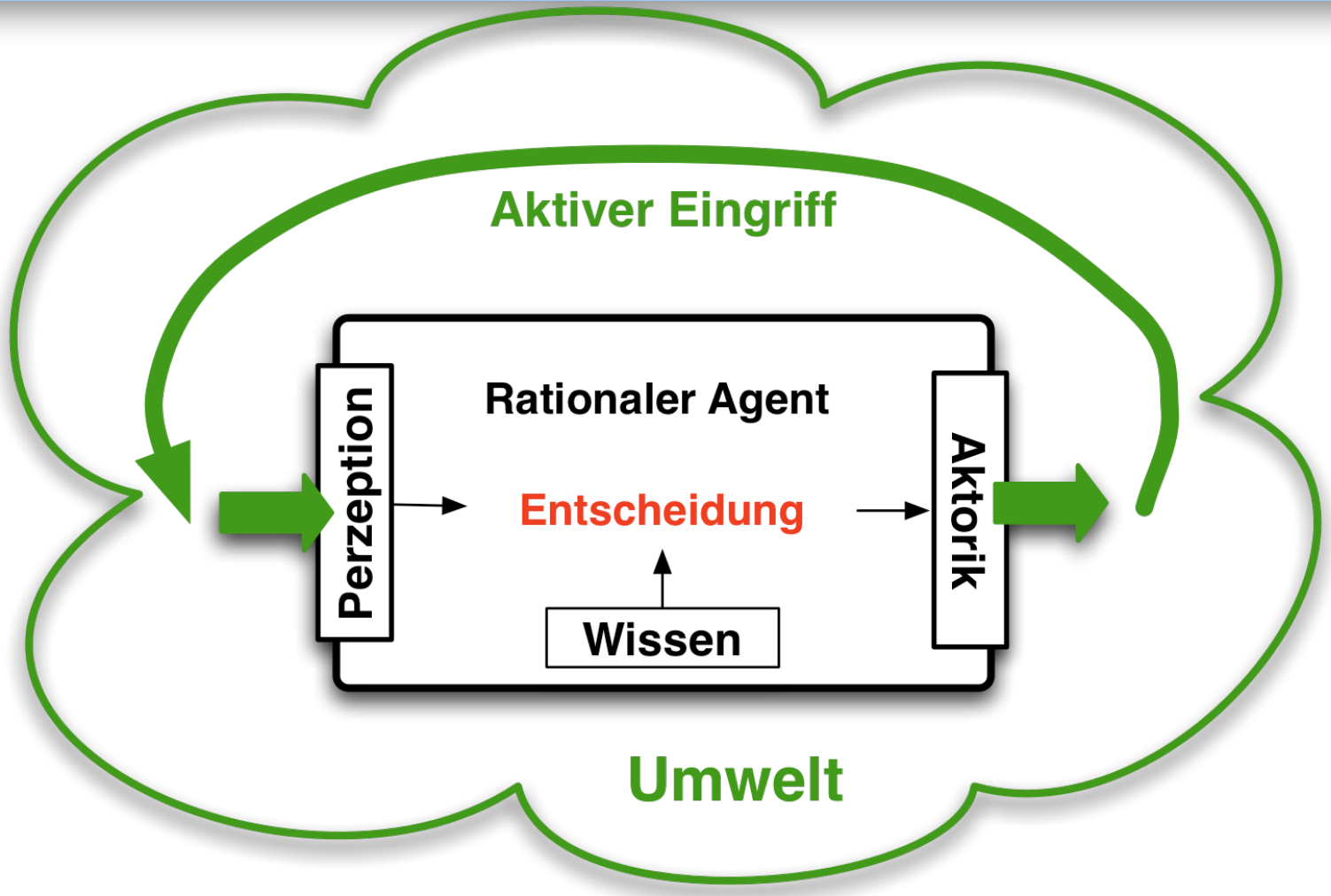
Der rationale Agent

- Allgemeine Definition eines in eine Umwelt eingebetteten, handelnden Systems
- Beliebige Art von Umwelt
- Definiert durch den Agentenzyklus:

  Perzeption - Entscheidung - Handlung

- Der Agentenzyklus ist eine andere Darstellung eines Regelkreises
- Entscheidung ist eine abstrakte Form der Regelung

Der Agentenzyklus



Ein rationaler Agent: Schachcomputer

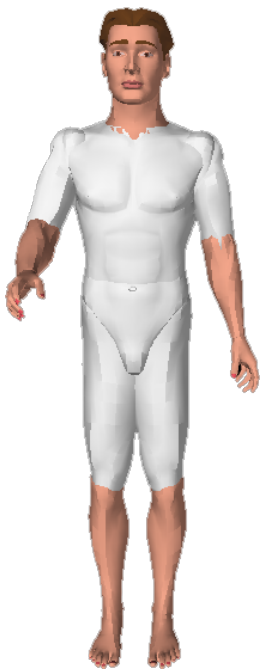
- Keine physische Umwelt
- Sehr beschränkte Umwelt
- Nur ein Ziel: das Spiel zu gewinnen
- Sensorik: Das Spielfeld wird komplett wahrgenommen
- Aktorik: Durchführung eines symbolischen Zugs

Ein rationaler Agent: Industrieroboter

- Oftmals, bei Abwesenheit von Sensorik, kein Agent, sondern nur ein Werkzeug
- Physische Umwelt
- Beschränkte Umwelt
- Wenn überhaupt rationaler Agent, üblicherweise extrem begrenzter Entscheidungsrahmen
- Ziel: Erfolgreiche Ausführung einer Operation

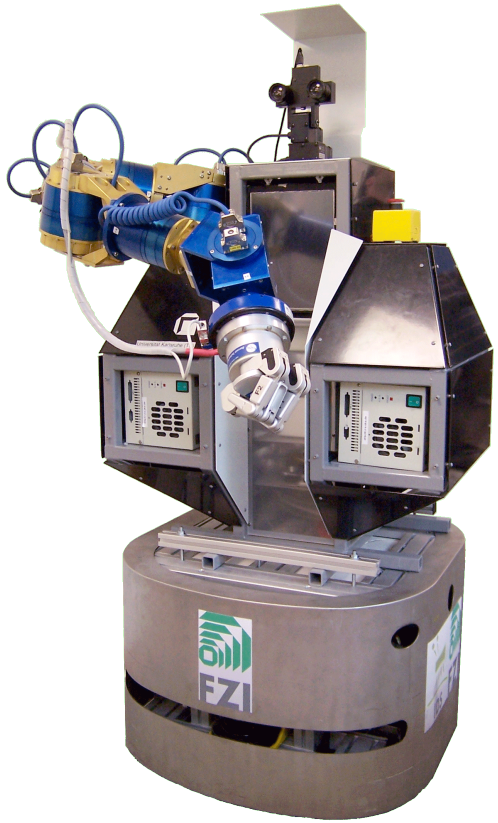
Ein rationaler Agent: Mensch

- Extrem komplexe Umwelt
- Sehr mächtige Perzeption
- Sehr wissensbasiert
- Vielschichtige Motivationen und Zielsetzungen



Ein rationaler Agent: Serviceroboter

- Anforderungen dem Menschen viel ähnlicher als einem Industrieroboter
- Sehr komplexe Umwelt
  - Vielschichtige Zielsetzung
  - Es ist bei Servicerobotern ein ganz anderes Vorgehen, als bei Industrierobotern erforderlich



Charakteristisierung der Umwelt

- Die Beispiele deuten sehr unterschiedliche Umgebungen für rationale Agenten an
- Ein Rahmen für eine nützliche und eindeutige Charakterisierung von Umgebungen ist nötig

Anforderungen:

- Die Charakterisierung muss fundamentale, nicht oberflächliche, Eigenschaften aufzeigen
- Es muss einen Rahmen gebildet werden, in den verschiedene Verfahren und Paradigmen eingeordnet werden können
- Eine Leitlinie für die Entwicklung neuer Verfahren muss gebildet werden

Charakteristika der Umwelt

- Beobachtung: Umwelten verschiedener Agenten können gut durch einige fundamentale, duale Eigenschaften unterschieden werden
- Diese Eigenschaften sind:
  - **Statisch** vs. **dynamisch**
  - **Episodisch** vs. **sequentiell**
  - **Diskret** vs. **kontinuierlich**
  - **Deterministisch** vs. **stochastisch**
  - **Vollständig** vs. **unvollständig beobachtbar**
  - **Einzelagent** vs. **Multiagent**

Statisch vs. Dynamisch

- Statisch:
  - Der Agent ist das einzige Element, welches den Zustand der Umwelt verändert
- Dynamisch:
  - Die Umwelt kann sich auch ohne zutun des Agenten verändern

Episodisch vs. Sequentiell

- Episodisch:
  - Der zeitliche Verlauf des Geschehens ist in abgeschlossene Einheiten unterteilt, zwischen denen keinerlei kausale Zusammenhänge bestehen
- Sequentiell:
  - Die vollständige zeitliche Vergangenheit hat Auswirkungen auf die Gegenwart, die Gegenwart auf die gesamte Zukunft

Diskret vs. Kontinuierlich

- Diskret:
  - Die Zustände der Umwelt sind diskret, ebenso der zeitliche Verlauf des Geschehens
- Kontinuierlich:
  - Der Zustandsraum der Umwelt ist kontinuierlich und die Zeit fließt ebenfalls kontinuierlich

Deterministisch vs. Stochastisch

- Deterministisch:
  - Der Ausgang einer jeden Handlung ist eindeutig bestimmt
- Stochastisch:
  - Eine Handlung kann mit bestimmten Wahrscheinlichkeiten zu verschiedenen Ausgängen führen

Vollständig vs. Unvollständig beobachtbar

- Vollständig beobachtbar:
  - Der Agent kann die komplette Umwelt jederzeit vollständig und exakt wahrnehmen
- Unvollständig beobachtbar:
  - Der Agent kann die Umwelt nur eingeschränkt und fehlerbehaftet wahrnehmen

Einzelagent vs. Multiagent

- Einzelagent:
  - Nur eine als Agent modellierbare Entität agiert in der Umwelt
- Multiagent:
  - Viele als Agenten modellierbare Entitäten agieren in der Umwelt und können kooperieren oder konkurrieren

Einordnung der Charakteristika

- Die Beschreibung der Umwelt durch das jeweils komplexere Merkmal wird nur gewählt, wenn das einfachere Merkmal im Szenario des Agenten keine hinreichende Beschreibung ist

- *Hinreichend* ist hierbei auch in dem Sinne zu verstehen, dass es oftmals noch keine ausreichenden Verfahren gibt, um die komplexeren Merkmale zu berücksichtigen

Die Umwelt des Schachcomputers

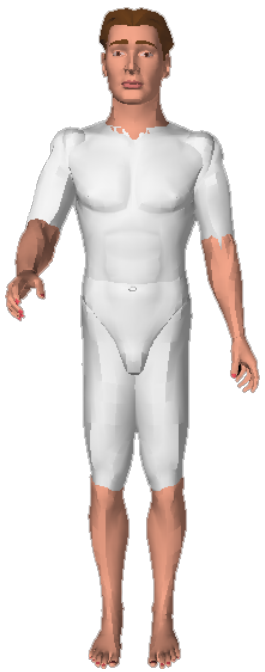
- Semi-Statisch
- Episodisch
- Diskret
- Deterministisch
- Vollständig beobachtbar
- Multiagent

Umwelt eines Industrieroboters

- Dynamisch
- Episodisch
- Diskret oder kontinuierlich
- Deterministisch
- Vollständig beobachtbar
- Einzelagent oder Multiagent

Die Umwelt eines Menschen

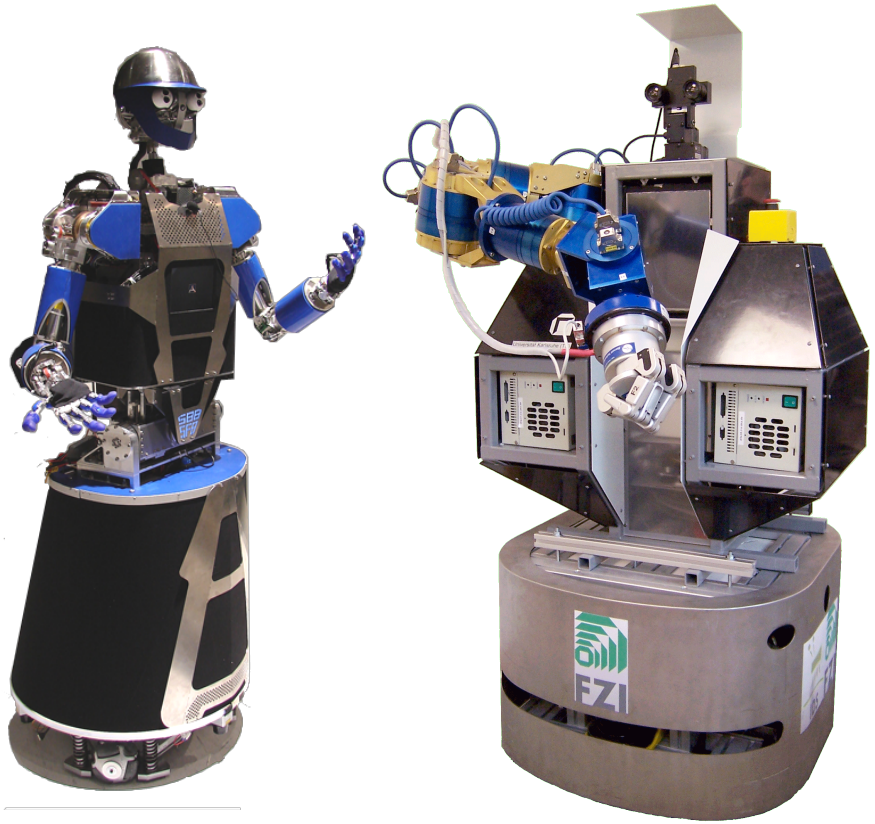
- Dynamisch
- Sequentiell
- Kontinuierlich
- Stochastisch
- Unvollständig beobachtbar
- Multiagent



Die Umwelt eines Serviceroboters

Wie beim Menschen:

- Dynamisch
- Sequentiell
- Kontinuierlich
- Stochastisch
- Unvollständig beobachtbar
- Multiagent



Motivation des Agenten

- Ein Agent benötigt Motivation und Absichten als Fundament für Entscheidungen
- In der Entscheidungstheorie durch das Konzept der ***Utility*** (dt. Nutzwertfunktion) abgebildet
- Dieses Konzept ist deutlich allgemeiner, als spezielle Ziele z.B. in der logikbasierten Planung
- Ursprünglich - wie die meisten Begriffe der Entscheidungstheorie - in der Ökonomie entwickelt

Das Konzept der Utility

- Die utility (auch *utility function*) modelliert in der Ökonomie ein numerisches Maß der Befriedigung eines Agenten (Menschen) durch einen bestimmten Konsum
- In der Robotik/KI wird jedoch ein Motivationsmaß eines Agenten nicht *modelliert*, sondern ***konstruiert***
- Die utility wird hier genutzt, um einem künstlichen Agenten bestimmte Motivationen einzupflanzen
- Die utility ist ein allgemeines Konzept: es können konkurrierende und ergänzende Zielsetzungen fusioniert werden
- Im Rahmen der utility können auch Absichten gegeneinander abgewogen werden

Annahmen der klassischen Aktionsplanung

- Ursprüngliche, klassische Aktionsplanung nimmt an, dass die Umwelt folgende Eigenschaften habe:
  - Statisch
  - Episodisch
  - Diskret
  - Deterministisch
  - Vollständig beobachtbar

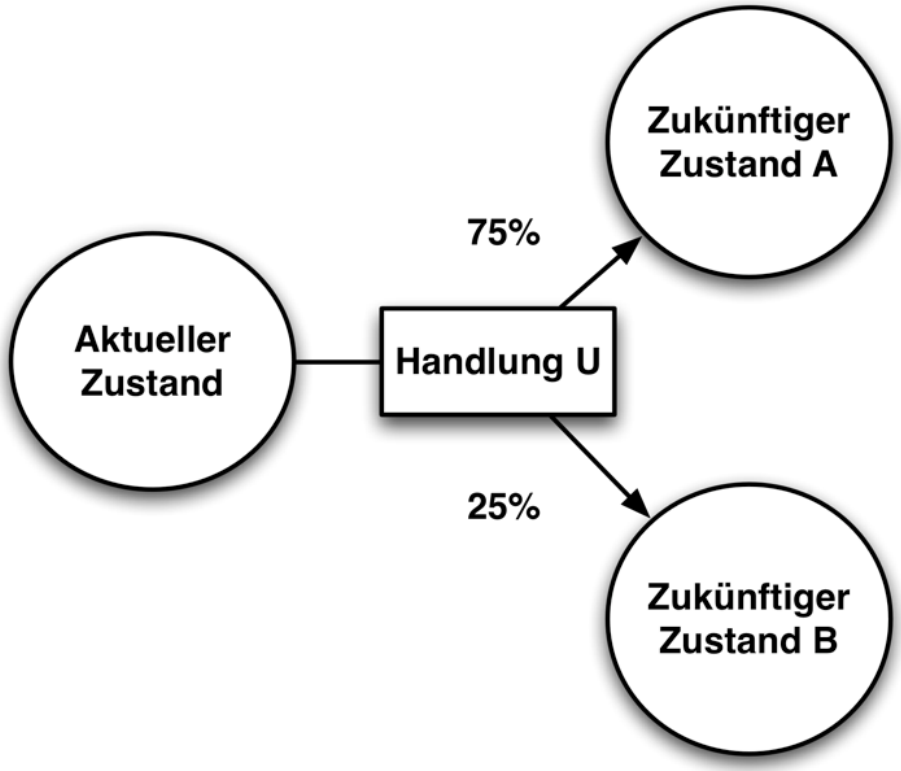
  - Konkrete, nicht überlappende Zielsetzungen

Grenzen der klassischen Aktionsplanung

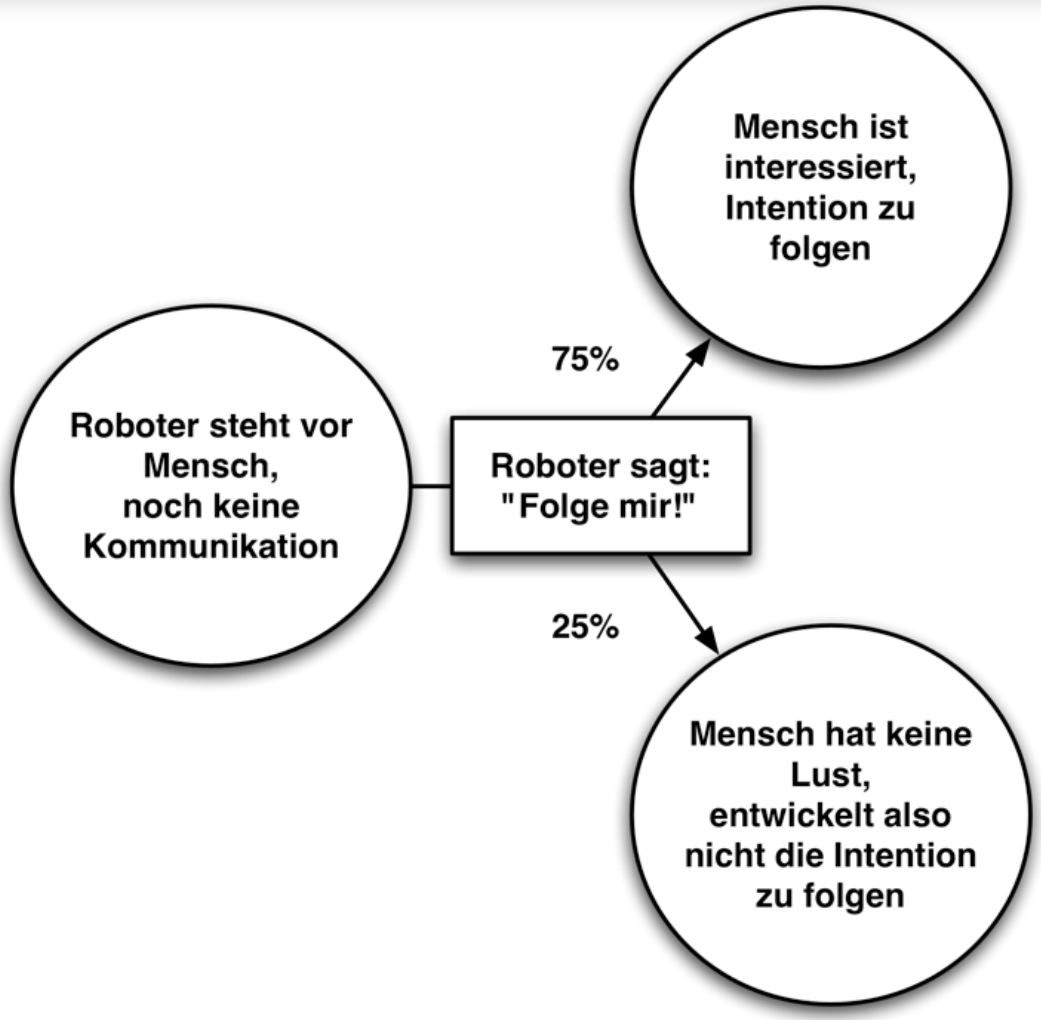
- Für folgende Eigenschaften gibt es Erweiterungen des klassischen Aktionsplanens:
  - Dynamisch
  - Sequentiell
  - Kontinuierlich
- Für die folgenden Eigenschaften nicht:
  - Stochastisch
  - Unvollständig beobachtbar
  - Unterschiedlich stark konkurrierende Ziele

Eine stochastische Umwelt

- Übergangswahrscheinlichkeiten beschreiben den Ausgang von Handlungen

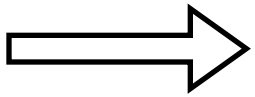


Beispiel: stochastische Umwelt



Methoden für stochastische Umgebungen

- Klassische, logikbasierte Aktionsplanung kann nicht mit numerischen Wahrscheinlichkeiten umgehen
- In stochastischen Umgebungen ist dies essenziell



probabilistische Methoden

Probabilistische Entscheidungsfindung

- Probabilistische Methoden handhaben die stochastischen Eigenschaften der komplexen Umgebungen
- Anderer Ansatz als Planung mit klassischer Logik
- Die Methoden basieren auf einfacher Wahrscheinlichkeitstheorie
- Eine wichtige, allgemeine Regel ist die Bayes Regel:

$$p(x|y) = \frac{p(y$$

Der Satz von Bayes erlaubt in gewissem Sinn das Umkehren von Schlussfolgerungen:

Die Berechnung von $P(\text{Ereignis} \mid \text{Ursache})$ ist häufig einfach, aber oft ist eigentlich $P(\text{Ursache} \mid \text{Ereignis})$ gesucht, also ein Vertauschen der Argumente. Das Verfahren ist auch als

Exkurs: Frequenstistisch vs. Bayesianisch

- Frequentistische Interpretation von Wahrscheinlichkeiten:
  - Klar definiertes Experiment
  - Wahrscheinlichkeiten als Resultat sehr vieler Durchführungen (lim -> unendlich)

- Bayes'sche Interpretation von Wahrscheinlichkeiten:
  - Wahrscheinlichkeiten repräsentieren ein „Maß des Wissenstands"
  - z.B. Übergangswahrscheinlichkeiten als a-piori Modell über das Verhalten des Systems

Markov Prozesse

- Markovprozess: Verbreitetes Werkzeug für die Modellierung stochastischer Prozesse
- Zentrale Eigenschaft: Die zukünftige Entwicklung kann durch Kenntnis einer **begrenzten** Vorgeschichte prognostiziert werden
- Erst durch diese Markoveigenschaft sind Prognosen bezüglich der Zukunft in einem solchen stochastischen Prozess möglich
- Es gibt zwei Arten von Markov Prozessen:
  - Zeitdiskrete Markov Prozesse
  - Zeitstetige Markov Prozesse

Zeitdiskreter Markov Prozess

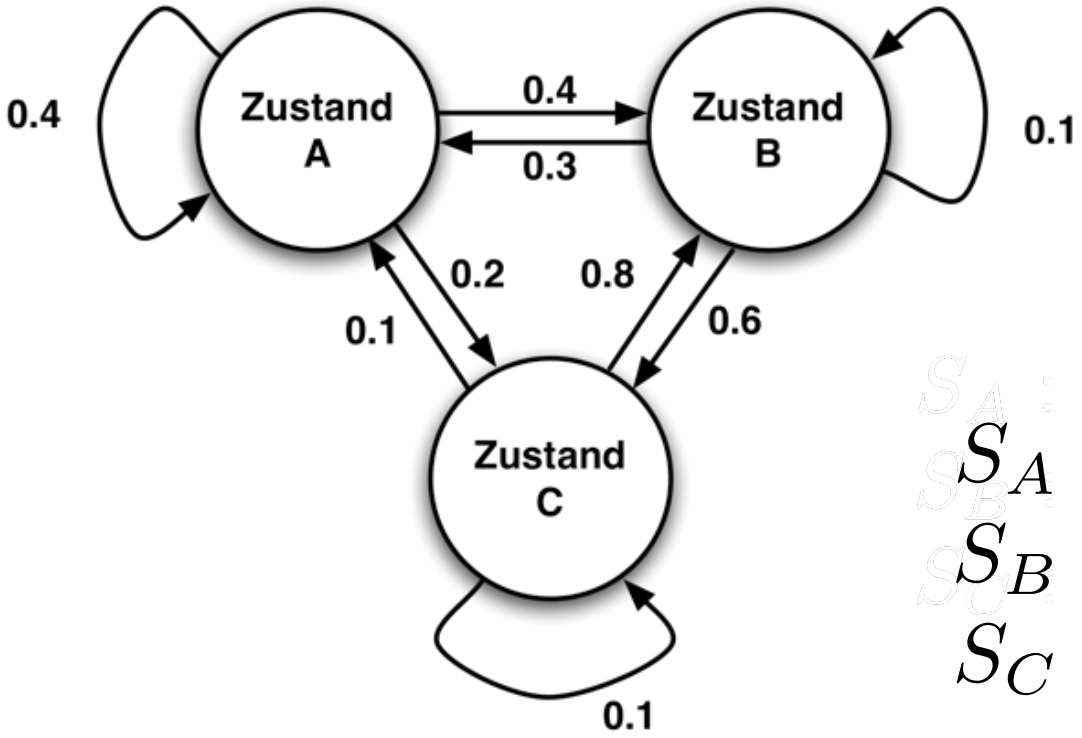
- Markov Prozesse mit diskreter Zeit werden Markov Ketten genannt
- Übergänge in einer Markov Kette sind Sprünge in einer diskreten Zeit
- Mathematisch wird die Wahrscheinlichkeit für einen zukünftigen Zustand als bedingte Wahrscheinlichkeit, abhängig von einer endlichen Zahl an vergangenen Zuständen, modelliert
- Die Ordnung einer Markov Kette gibt an, welcher Ausschnitt der Vergangenheit eine Rolle spielt
- Die Ordnung ist immer endlich, sonst wäre die Markov Eigenschaft nicht erfüllt
- Je geringer die Ordnung, desto einfacher die Modellierung

Markovkette 1. Ordnung

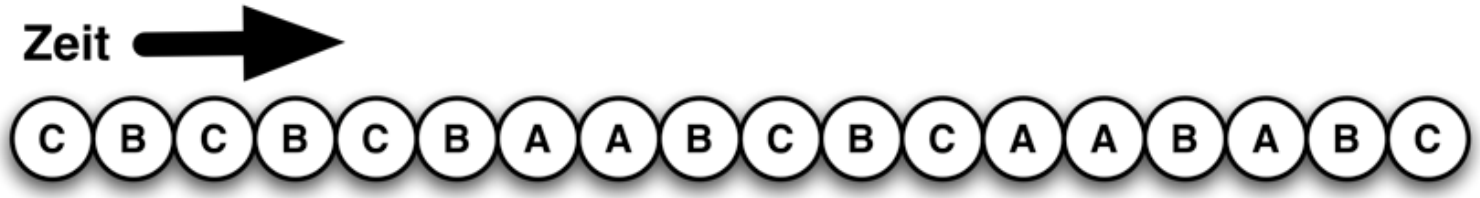
- Nur die Gegenwart spielt eine Rolle für die Betrachtung der Zukunft
- Die Wahrscheinlichkeit des Nachfolgezustands hängt nur vom aktuellen Zustand ab

$$Pr(X_{n+1} = x | X_n = x_n, ..., X_1 = x_1, X_0 = x_0)$$
$$= Pr(X_{n+1} = x | X_n = x_n)$$

Beispiel: Markovkette 1. Ordnung



	S'_A	S'_B	S'_C
S_A :	0.4	0.4	0.2
S_B :	0.3	0.1	0.6
S_C :	0.1	0.8	0.1

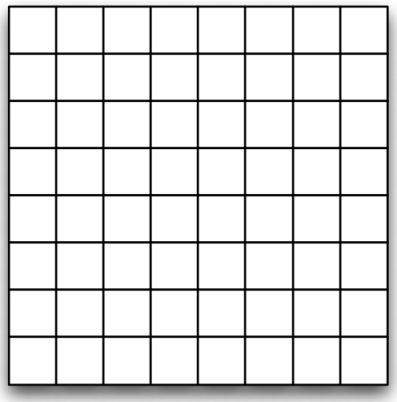


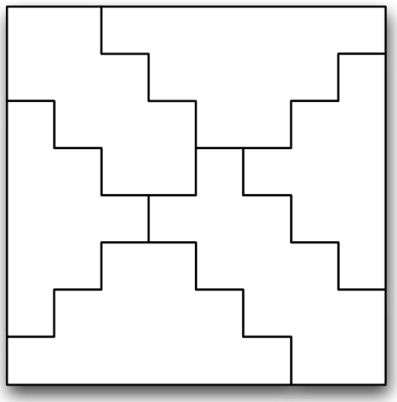
Markovketten: Modellierung der Welt

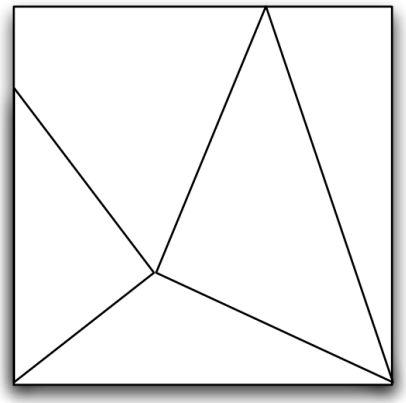
- Markovketten können von Agenten zur Modellierung von stochastischen Umgebungen genutzt werden
- Bei diskreten Markovketten muss die Welt dazu in eine diskrete Zustandsmenge zerlegt werden
- Die Zerlegung kann für verschiedene Umweltaspekte getrennt erfolgen, z.B.:
  - Roboterposition
  - Objektform
  - Intentionen eines Menschen

Beispiel: Welt als diskrete Zustände

- Einfaches Beispiel: Unterteilung von Räumen in diskrete Regionen



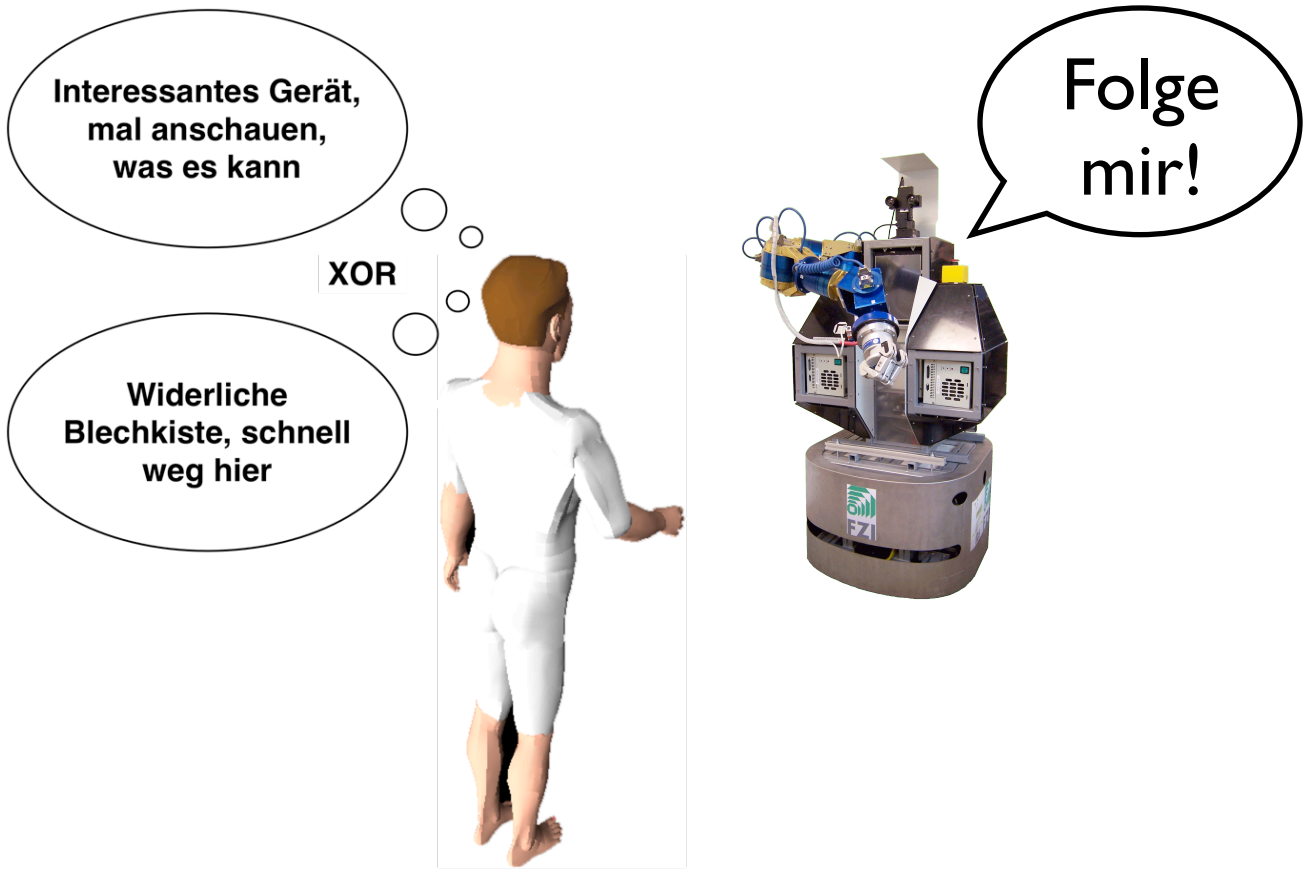




Die Position eines mobilen Roboters wird durch die Region, in der er sich befindet, repräsentiert

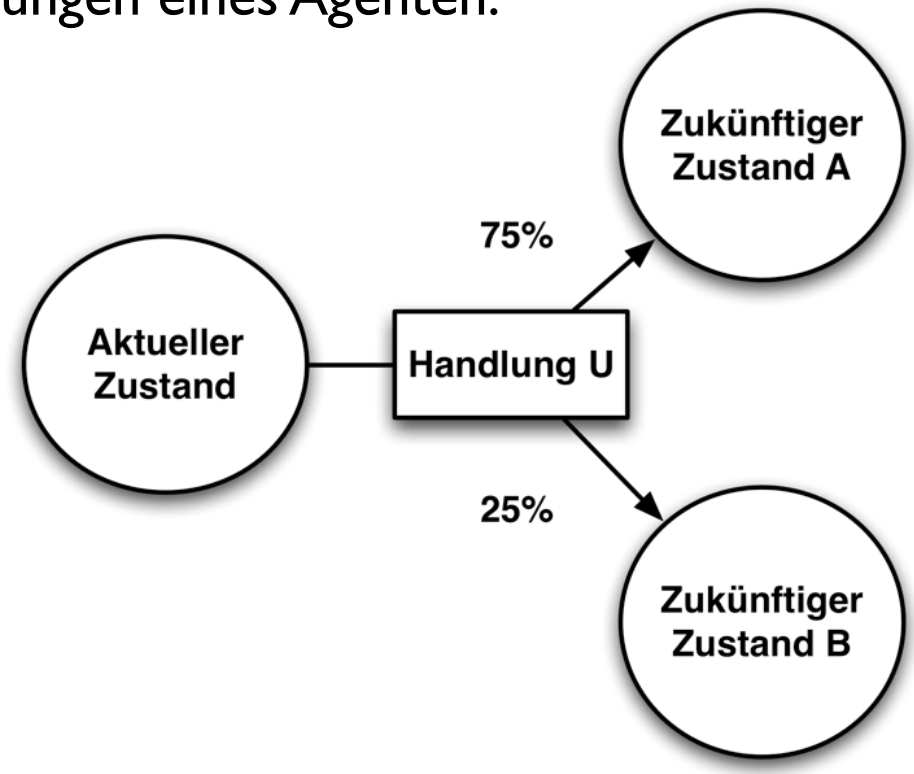
Beispiel: Welt als diskrete Zustände

- Abstraktes Beispiel: Modellierung menschlicher Intentionen über Mensch-Maschine Dialog



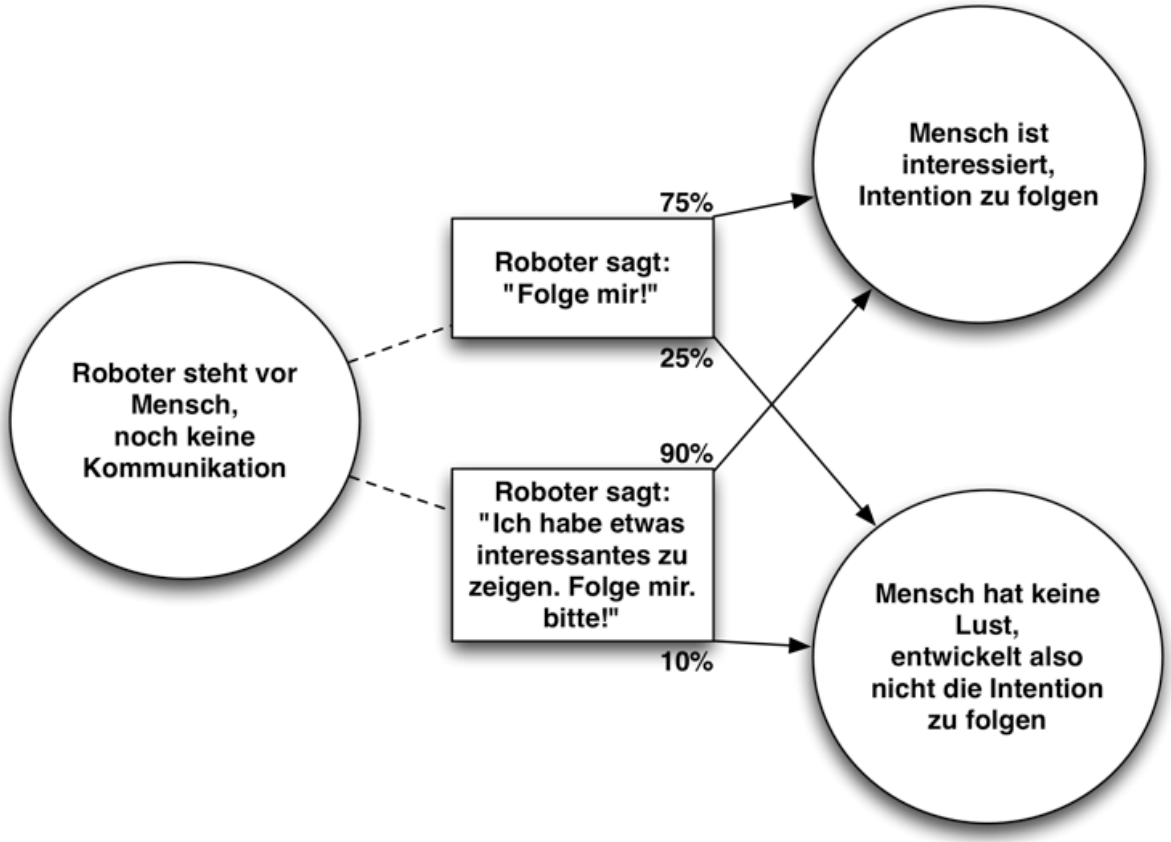
Welt als stochastischer Prozess

- Bestimmte Auslöser führen zu stochastischen Zustandsübergängen (Bayesianische Modellierung), z.B. Handlungen eines Agenten:



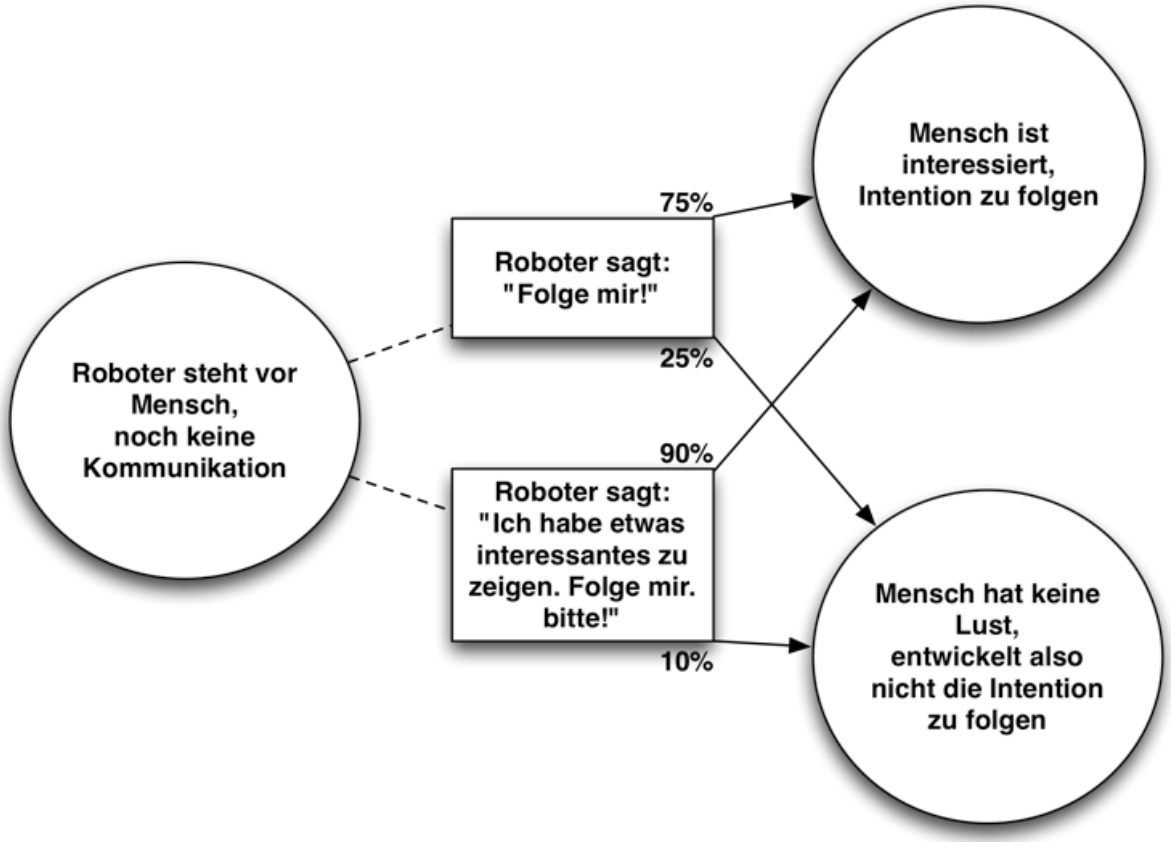
Beispiel: Verschiedene Übergänge

- Verschiedene Auslöser haben unterschiedliche Übergangswahrscheinlichkeiten:



Wahlmöglichkeit von Auslösern

- Durch die Wahl der Handlung kann der Agent die Übergangswahrscheinlichkeiten steuern:



Markov Entscheidungsprozesse

- Ketten von Handlungen bzw. Auslösern führen zu Bayes-Netzwerken
- Für die Entscheidungsfindung in diesen stochastischen Prozessen existiert ein fundiertes Modellierungsrahmenwerk spezieller Bayes-Netze:

Markov Decision Processes (MDPs)

(= dt. Markov Entscheidungsprozesse)

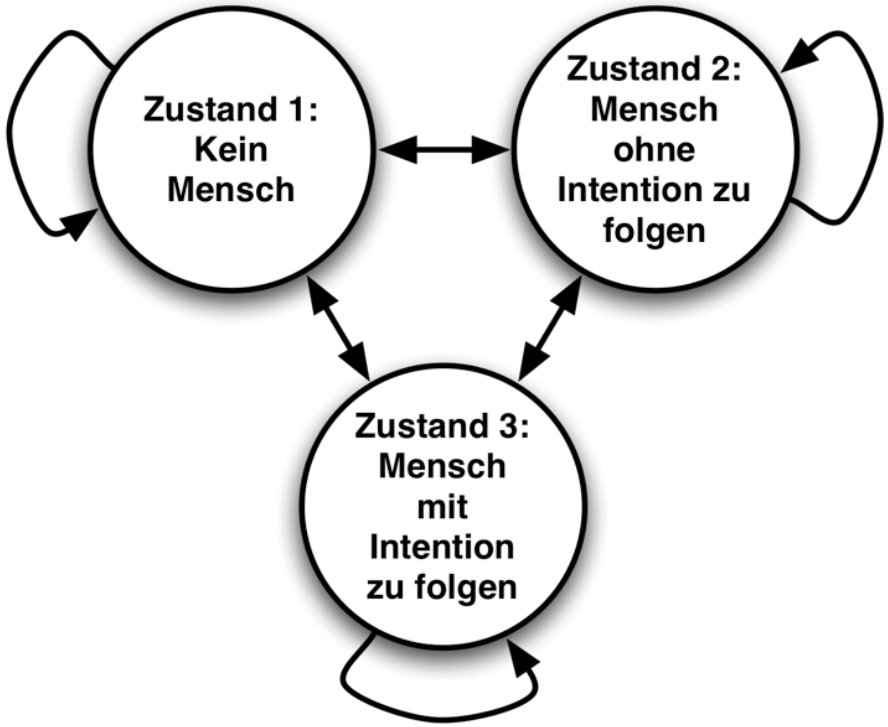
MDP: Struktur

- Die Struktur eines MDPs wird durch folgendes Modell beschrieben:
  - Eine Menge von symbolischen **Zuständen** der Welt *S*
  - Eine Menge von symbolischen **Handlungen** des Agenten *U*
  - Ein **Übergangsmodell** (*transition model*) *T(s', u, s)* welches die stochastischen Übergänge modelliert
  - Ein **Belohnungsmodell** (*reward model*) *R(s, u)* welches die Ziele und Absichten des Agenten modelliert

MDP: Statische Eigenschaften

- In einem MDP ist die Welt zu jedem Zeitpunkt in genau einem Zustand
- In einem diskreten MDP ist dies ein symbolischer Zustand, welcher alle Eigenschaften der Welt repräsentiert
- Der Agent kann in jedem Agentenzyklus genau eine Handlung ausführen
- In einem diskreten MDP ist dies eine symbolische Handlung
- Im reinen MDP nimmt der Agent den aktuellen Zustand immer und eindeutig wahr

MDP Beispiel: statische Eigenschaften



- 3 Zustände
- 3 Handlungen

Handlung 1: Sage "Folge mir"
Handlung 2: Führe an einen Ort
Handlung 3: Tue nichts

MDP: Transitionsmodell

- T(s‘, u, s) = p(transition)
  - Wenn der alte Zustand s war und der Agent die Handlung u ausgeführt hat, dann beschreibt dieser Eintrag im Modell die Wahrscheinlichkeit, dass der neue Zustand s‘ sein wird
- Das Transitionsmodell ist ein *Tensor 3. Stufe*
- In der Praxis oft als Vektor (über U) von Matrizen (S, S‘) dargestellt
- |T| = |S| * |S| * |U|
- Die Grösse wächst also kubisch

MDP Beispiel: Transitionsmodell

Ein Mensch erscheint im Szenario mit 25% Wahrscheinlichkeit, völlig unabhängig vom Roboterverhalten.

Das Verschwinden des Menschen ist abhängig vom Roboterverhalten

|T| = |S| * |S| * |U| ; |S| = 3, |U| = 3 ➛ 3 * 3 * 3 = 27 Einträge

Zustände:
S1: Kein Mensch
S2: Mensch, keine Intention zu folgen
S3: Mensch, Intention zu folgen

Handlungen:
U1: Sage „Folge mir!“
U2: Führe an einen Ort
U3: Tue nichts

U_1 (Sage "Folge mir!")

	S'_1	S'_2	S'_3
S_1 :	0.75	0.25	0.0
S_2 :	0.0	0.25	0.75
S_3 :	0.25	0.25	0.5

U_2 (Führe an einen Ort)

	S'_1	S'_2	S'_3
S_1 :	0.75	0.25	0.0
S_2 :	0.25	0.75	0.0
S_3 :	0.50	0.25	0.25

U_3 (Tue nichts)

	S'_1	S'_2	S'_3
S_1 :	0.75	0.25	0.0
S_2 :	0.25	0.75	0.0
S_3 :	0.25	0.50	0.25

MDP: Belohnungsmodell

- R(s, u) = Belohnung oder Strafe
  - Wenn der Zustand s war und der Agent führt die Handlung u aus, erhält er die sofortige Belohnung (oder negative Belohnung = Strafe) wie vom Modell an R(s, u) gegeben
- Das Belohnungsmodell ist eine Matrix
- Die Einträge können beliebige Realzahlen sein, je nach Modellierung

- Durch das Belohnungsmodell können auch konkurrierende oder ergänzende Zielsetzungen modelliert werden

MDP Beispiel: Belohnungsmodell

- Sprechen und führen kostet immer -1 Strafe, verglichen mit tue nichts, denn der Roboter ist faul und möchte Energie sparen
- Einen folgenden Menschen an einen Ort zu führen gibt +5 Belohnung

Zustände:
S1: Kein Mensch
S2: Mensch, keine Intention zu folgen
S3: Mensch, Intention zu folgen

Handlungen:
U1: Sage „Folge mir!“
U2: Führe an einen Ort
U3: Tue nichts

Belohnungsmodell:

	U_1	U_2	U_3
S_1 :	-1	-1	0
S_2 :	-1	-1	0
S_3 :	-1	4	0

MDP: Ziel des Agenten

- Das Ziel des rationalen Agenten im MDP ist nun, die Summe der Belohnungen bis zu einem bestimmten Zeithorizont (im reinen MDP meistens unendlich) zu maximieren
- Dies führt zu einer **Gewinnmaximierung** bzw. **Risikoabschätzung** in die Zukunft als Grundlage für den Entscheidungsprozess

- Das einzige explizite Ziel des Agenten ist also die Gewinnmaximierung
- Sie bezieht implizit alle gegebenen Zielsetzungen in den Entscheidungsprozess ein

MDP: Gewinnmaximierung

- Da in einer stochastischen Welt keine festen Handlungsketten existieren, wird eine Entscheidungsfunktion (*policy*) berechnet
- Die Entscheidungsfunktion π gibt für jeden Zustand s eine Handlung zurück: $\pi(s)$
- Ein Agent in einem MDP muss nun eine optimale Entscheidungsfunktion π^* berechnen, welche immer die Handlung mit der größten, langfristigen Gewinnerwartung wählt: $\pi^*(s)$
- Es gibt immer mindestens eine $\pi^*(s)$, welche zu einem gegebenen MDP Modell optimale Entscheidungen liefert

MDP: Gewinnmaximierung Horizont ∞

- Bei unendlichem Horizont ist die optimale Entscheidungsfunktion stationär: es gibt für einen Zustand immer dieselbe optimale Handlung
- In einem voll beobachtbaren (reinen) MDP ist die Variante mit unendlichem Horizont somit die einfachste
- Im Folgenden wird bei reinen MDPs nur der Fall des unendlichen Zeithorizonts berücksichtigt

MDP: Utility

- Die Utility einer Zustandskette ist die Summe ihrer Belohnungen bzw. Strafen:

$$U_h([s_0, s_1, s_2, ...]) = R(s_0) + R(s_1) + R(s_2) + ...$$

- Die Einführung eines Abschlags (*discount*) γ auf Zustände in der Zukunft ermöglicht eine endliche utility für alle unendlichen Zustandsketten

$$U_h([s_0, s_1, s_2, ...]) = R(s_0) + \gamma R(s_1) + \gamma^2 R(s_2) + ...$$

MDP: Entscheidungsfunktion

- Es existieren verschiedene Verfahren, um eine optimale Entscheidungsfunktion aus dem gegebenen MDP Modell (S, A, T, B) zu berechnen. Unter anderem:
  - Policy iteration
  - Reinforcement learning einer policy

- Das wichtigste aber ist:

Value Iteration

MDP Value Iteration: Idee

- Für jeden Zustand wird eine utility berechnet, die die *erwartete utility* aller möglichen, folgenden Zustandssequenzen widerspiegelt
- Daraus kann die Entscheidungsfunktion direkt gewonnen werden
- Die Zustandssequenzen wiederum hängen von der Entscheidungsfunktion ab

⟹ verschränktes Problem

MDP Value Iteration: Ansatz

- Das Problem wird durch einen iterativen Ansatz gelöst
- Die wirkliche utility eines Zustands wäre die utility bezogen auf die optimale policy:

$$U_{real}(s) = U^{\pi^*}(s)$$

- Die wirkliche utility ist jedoch zu Beginn nicht bekannt, da die optimale policy noch berechnet werden muss
- Zuerst wird eine beliebige policy π_{start} gewählt
- Dazu wird jedem Zustand eine initiale utility zugewiesen

MDP Value Iteration: Ansatz

- Die tatsächliche utility jedes Zustands, bezogen auf eine policy π, ist die Summe aller zukünftigen Belohnungen:

$$U^{\pi}(s) = E[\sum_{t=0}^{\infty} \gamma^t R(s_t, u_t) | \pi, s_0 = s]$$

- Die Summe der zukünftigen Belohnungen hängt jedoch von der Handlungswahl und damit von π ab

MDP Value Iteration: Handlungswahl

- Es kann nun eine policy gefunden werden, die auf den aktuell zugewiesenen utilities basiert und 1-Schritt weit optimal ist
- Nach der utility Funktion wird nun die Handlung, welche die erwartete zukünftige Belohung maximiert, gewählt:

$$\pi^{1step*}(s) = argmax_u \sum_{s'} T(s', u, s)U(s')$$

- Durch diese Formel wird auch ein Zusammenhang zwischen einem Zustand und seinen Nachbarn gebildet

MDP Value Iteration: Iterationsschritt

- Nun wird die policy Zuweisung wieder in die utility-Berechnung zurückgeführt
- Es entsteht eine Iterationsformel zwischen benachbarten Zuständen:

$$U(s) = \gamma max_u(R(s,u) + \sum_{s'} T(s',u,s)U(s'))$$

= Bellmann Formel

- Somit kann rückwärts von utility(s‘) auf utility(s) geschlossen werden

MDP Value Iteration: Ablauf

- Beginn: irgendwelche niedrigen Werte für die utilities aller Zustände
- Iterationsschritt: Anwendung der Bellmann Formel auf jeden Zustand unter Nutzung aller seiner Nachbarn
- Vielfache Anwendung des Iterationsschritts auf alle Zustände
- Irgendwann wird ein Gleichgewicht erreicht, welches die optimale policy π^* ist

- Die Iteration konvergiert zur optimalen policy

MDP Value Iteration: Algorithmus

- In der Praxis werden die Iterationen bis zu einer akzeptablen Konvergenzschranke durchgeführt

MDPdiscreteValueIteration():
    repeat until convergence bound
        for each state s **in** S **do**
            $U'(s) \leftarrow \gamma max_u(R(s,u) + \sum_{s'} T(s',u,s)U(s'))$
        endfor
    end repeat
    return U

MDP VI Beispiel: Modell

Zustände:
S1: Kein Mensch
S2: Mensch, keine Intention zu folgen
S3: Mensch, Intention zu folgen

Handlungen:
U1: Sage „Folge mir!“
U2: Führe an einen Ort
U3: Tue nichts

U_1 (Sage "Folge mir!")

	S'_1	S'_2	S'_3
S_1 :	0.75	0.25	0.0
S_2 :	0.0	0.25	0.75
S_3 :	0.25	0.25	0.5

U_2 (Führe an einen Ort)

	S'_1	S'_2	S'_3
S_1 :	0.75	0.25	0.0
S_2 :	0.25	0.75	0.0
S_3 :	0.50	0.25	0.25

U_3 (Tue nichts)

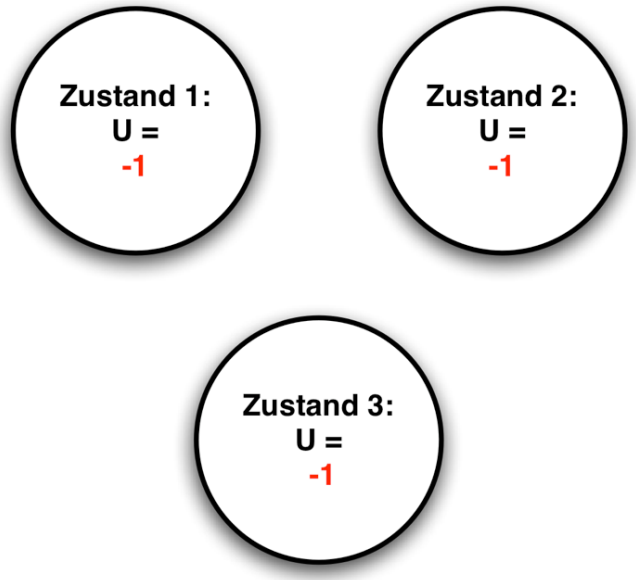
	S'_1	S'_2	S'_3
S_1 :	0.75	0.25	0.0
S_2 :	0.25	0.75	0.0
S_3 :	0.25	0.50	0.25

Belohnungsmodell:

	U_1	U_2	U_3
S_1 :	-1	-1	0
S_2 :	-1	-1	0
S_3 :	-1	4	0

MDP VI Beispiel: Start

- Initialisiere:
- U(s1=kein Mensch) = -1
- U(s2=Mensch, folgt nicht) = -1
- U(s3=Mensch, folgt) = -1
- Wähle $\gamma = 0.9$

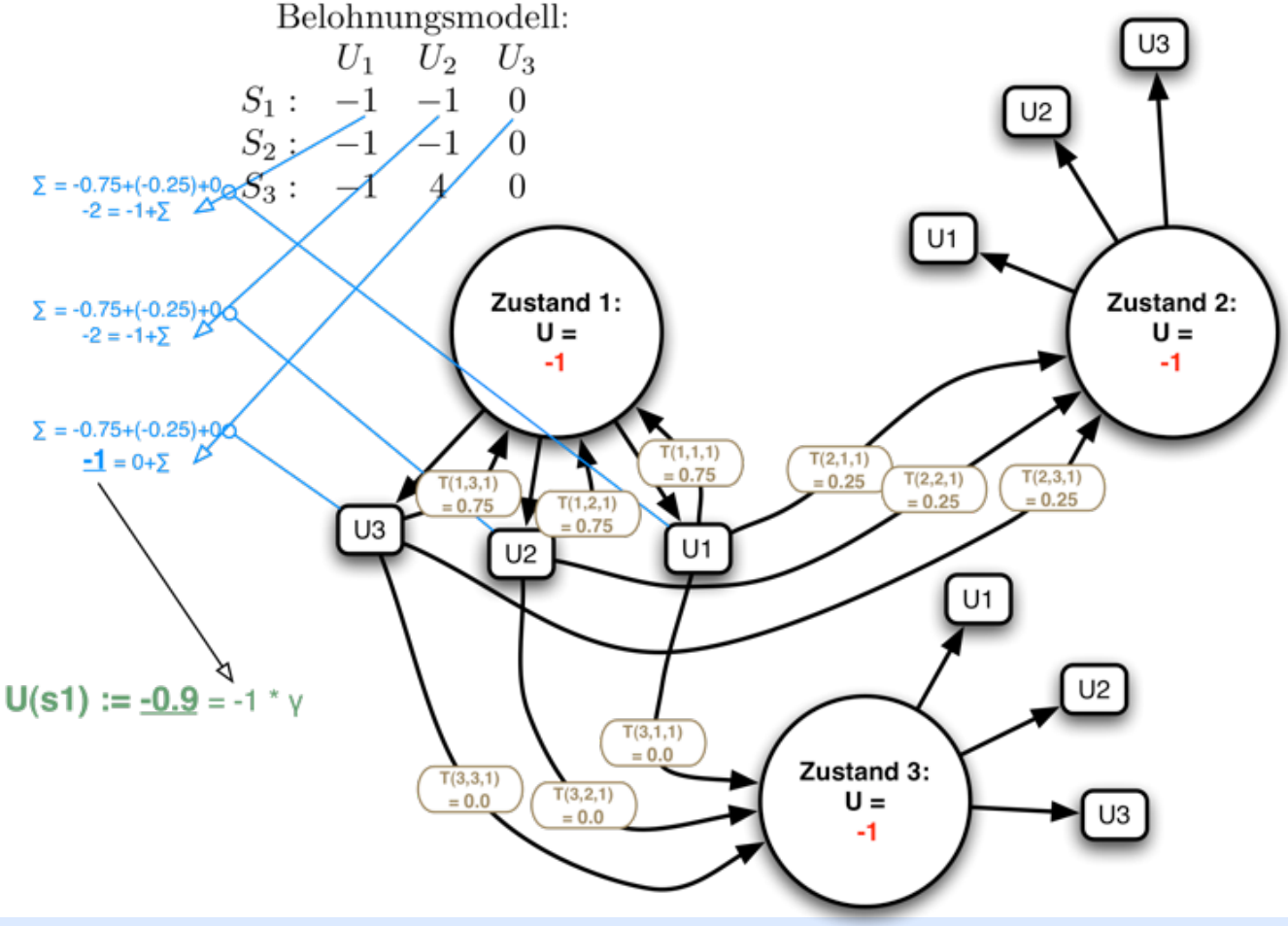


MDP VI Beispiel: Iteration

$$U(s) = \gamma max_u(R(s,u) + \sum_{s'} T(s',u,s)U(s'))$$

Belohnungsmodell:

	U_1	U_2	U_3
S_1 :	−1	−1	0
S_2 :	−1	−1	0
S_3 :	−1	4	0



U_1 (Sage "Folge mir!")

	S'_1	S'_2	S'_3
S_1 :	0.75	0.25	0.0
S_2 :	0.0	0.25	0.75
S_3 :	0.25	0.25	0.5

U_2 (Führe an einen Ort)

	S'_1	S'_2	S'_3
S_1 :	0.75	0.25	0.0
S_2 :	0.25	0.75	0.0
S_3 :	0.50	0.25	0.25

U_3 (Tue nichts)

	S'_1	S'_2	S'_3
S_1 :	0.75	0.25	0.0
S_2 :	0.25	0.75	0.0
S_3 :	0.25	0.50	0.25

MDP VI Beispiel: Iterationsschritt

S1: u1: (∑ = -1) + (r = -1) = -2

u2: (∑ = -1) + (r = -1) = -2

u3: (∑ = -1) + (r = 0) = -1

max(S1): (u3 = -1) * (γ = 0.9) = -0.9

S2: u1: (∑ = -1) + (r = -1) = -2

u2: (∑ = -0.975) + (r = -1) = -1.975

u3: (∑ = -0.975) + (r = 0) = -0.975

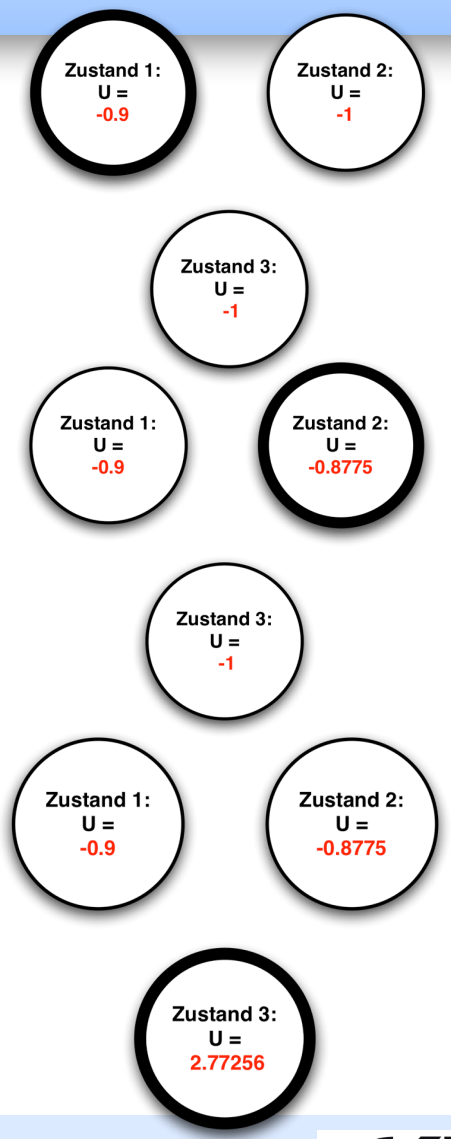
max(S2): (u3 = -0.975) * (γ = 0.9) = -0.8775

S3: u1: (∑ = -0.94437) + (r = -1) = -1.94437

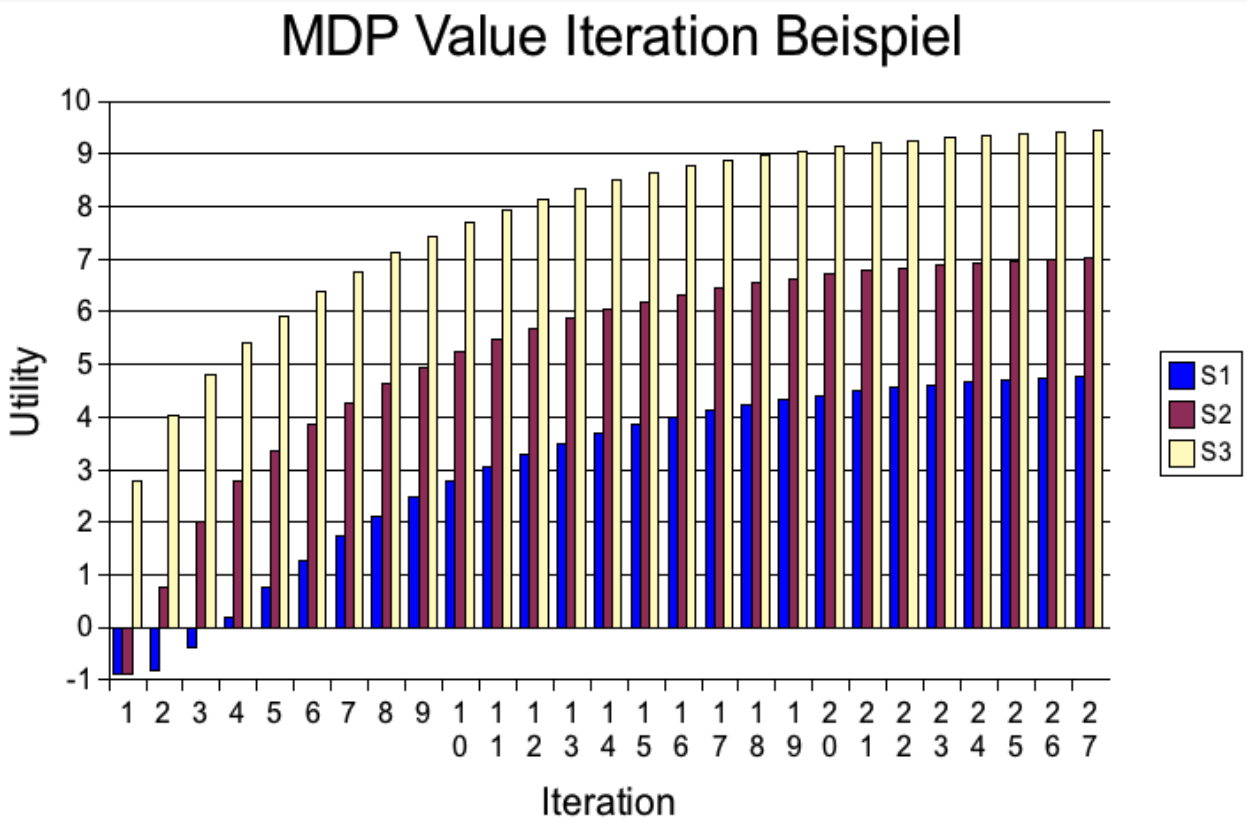
u2: (∑ = -0.919375) + (r = 4) = 3.08062

u3: (∑ = -0.91375) + (r = 0) = -0.91375

max(S3): (u2 = 3.08062) * (γ = 0.9) = 2.77256



MDP VI Beispiel: Ergebnis



S1: u3 am besten ab Iterationsschritt 1

S2: u1 am besten ab Iterationsschritt 2

S3: u2 am besten ab Iterationsschritt 1

MDP Value Iteration: Animation

http://www.cs.ubc.ca/~poole/demos/mdp/vi.html

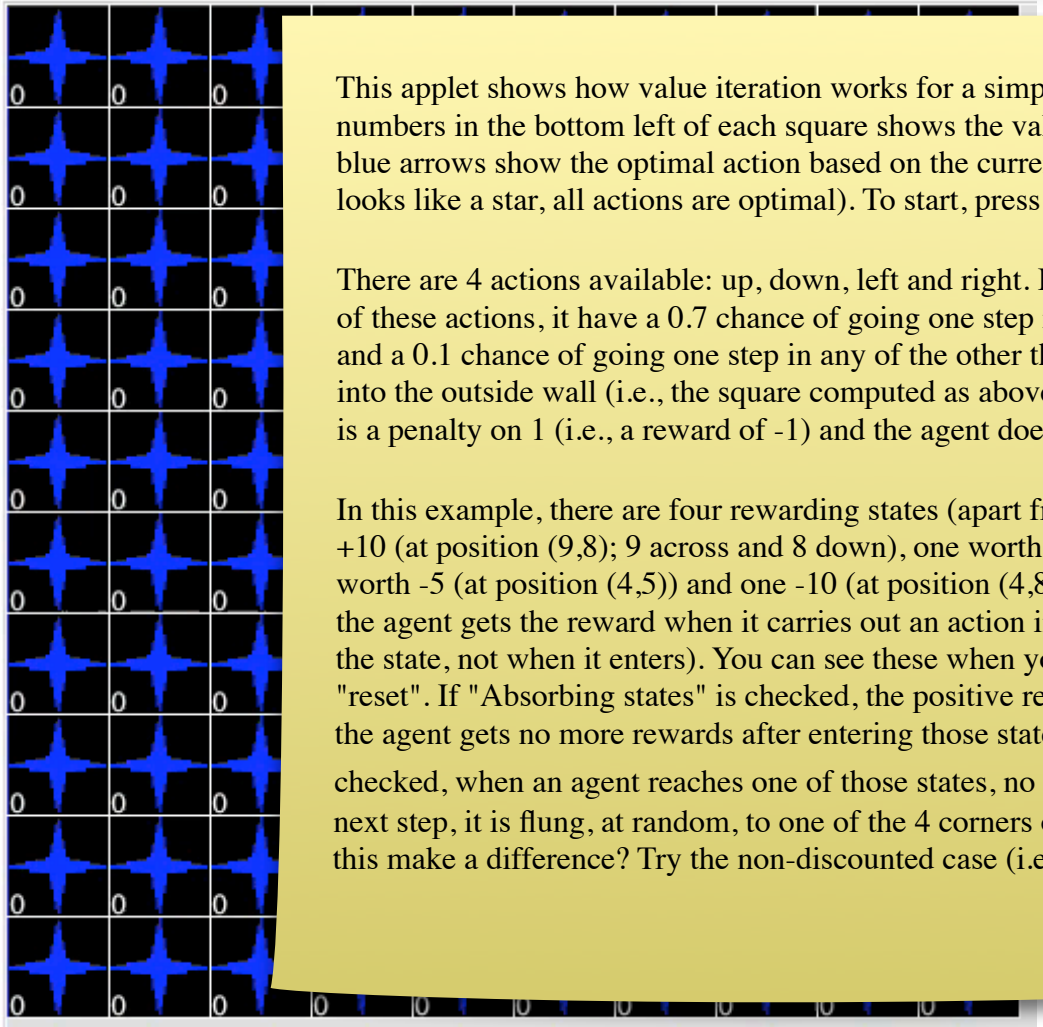
This applet shows how value iteration works for a simple 10x10 grid world. The numbers in the bottom left of each square shows the value of the grid point. The blue arrows show the optimal action based on the current value function (when it looks like a star, all actions are optimal). To start, press "step".

There are 4 actions available: up, down, left and right. If the agent carries out one of these actions, it have a 0.7 chance of going one step in the desired direction and a 0.1 chance of going one step in any of the other three directions. If it bumps into the outside wall (i.e., the square computed as above is outside the grid), there is a penalty on 1 (i.e., a reward of -1) and the agent doesn't actually move.

In this example, there are four rewarding states (apart from the walls), one worth +10 (at position (9,8); 9 across and 8 down), one worth +3 (at position (8,3)), one worth -5 (at position (4,5)) and one -10 (at position (4,8)). In each if these states the agent gets the reward when it carries out an action in that state (when it leaves the state, not when it enters). You can see these when you press "step" once after a "reset". If "Absorbing states" is checked, the positive reward states are absorbing; the agent gets no more rewards after entering those states. If the box is not checked, when an agent reaches one of those states, no matter what it does at the next step, it is flung, at random, to one of the 4 corners of the grid world. [Does this make a difference? Try the non-discounted case (i.e., where discount=1.0).]

MDP Value Iteration: Animation

http://www.cs.ubc.ca/~poole/demos/mdp/vi.html



This applet shows how value iteration works for a simple 10x10 grid world. The numbers in the bottom left of each square shows the value of the grid point. The blue arrows show the optimal action based on the current value function (when it looks like a star, all actions are optimal). To start, press "step".

There are 4 actions available: up, down, left and right. If the agent carries out one of these actions, it have a 0.7 chance of going one step in the desired direction and a 0.1 chance of going one step in any of the other three directions. If it bumps into the outside wall (i.e., the square computed as above is outside the grid), there is a penalty on 1 (i.e., a reward of -1) and the agent doesn't actually move.

In this example, there are four rewarding states (apart from the walls), one worth +10 (at position (9,8); 9 across and 8 down), one worth +3 (at position (8,3)), one worth -5 (at position (4,5)) and one -10 (at position (4,8)). In each if these states the agent gets the reward when it carries out an action in that state (when it leaves the state, not when it enters). You can see these when you press "step" once after a "reset". If "Absorbing states" is checked, the positive reward states are absorbing; the agent gets no more rewards after entering those states. If the box is not checked, when an agent reaches one of those states, no matter what it does at the next step, it is flung, at random, to one of the 4 corners of the grid world. [Does this make a difference? Try the non-discounted case (i.e., where discount=1.0).]

MDP Value Iteration: Eigenschaften

- MDP Value Iteration ist eine *Kontraktion*: der Algorithmus konvergiert immer
- In der Praxis konvergiert er bei kleinerem γ schneller
- MDP Value Iteration ist verhältnismäßig schnell
  - Dies gilt im Vergleich zu anderen Verfahren und anderen Arten von MDPs (siehe PE II)
- Fast optimale Entscheidungsfunktionen sind in der Praxis ausreichend

MDP: Schlußfolgerung

- Es gibt ein algorithmisches Konzept für optimale Entscheidungsfindung in stochastischen Umgebungen
- Ein rationaler Agent kann in Bezug auf Belohnungsmaximierung entscheiden
- Die (fast) optimale Belohnungsmaximierung ist berechenbar
- Die Modellierung eines Szenarios ist allerdings schwierig

- Nur: Welche Umgebungseigenschaften decken reine MDPs ab?

MDP: Umgebungseigenschaften

- Dynamisch
- Sequentiell
- Diskret (hier), aber auch kontinuierlich
- Stochastisch
- ***Vollständig*** beobachtbar
- (Multiagent)

MDP: Beschränkungen

- Es gibt Erweitungen von MDPs für kontinuierliche Umgebungen (Zustände, Handlungen)
- Es gibt eine Erweiterung von MDPs für unvollständig beobachtbare Umgebungen:

Partially Observable Markov Decision Processes (POMDPs)

Ausblick: POMDPs

- POMDPs können mit Sensorunschärfen und unvollständigem Weltwissen umgehen
- Daher vor allem auch für echte, autonome Roboter interessant
- Leider deutlich komplexer als vollständig beobachtbare MDPs
- Sehr spezielle Lösungsmethoden notwendig, um die Rechenkomplexität im Rahmen zu halten

Übungsblatt

- Übungsblatt 2 mit Aufgaben zu den folgenden Vorlesungen:
  - Aktionsplanungsverfahren (Lineares, Nichtlineares Planen)
  - Probabilistische Entscheidungsverfahren I (MDPs)
  - Probabilistische Entscheidungsverfahren II (POMDPs)

MDP Grundlagenliteratur

- Artificial Intelligence: A Modern Approach
  2nd Edition; S. Russel, P. Norvig; 2003
  Kapitel 16 & 17
- Probabilistic Robotics
  S. Thrun, W. Burgard, D. Fox; 2005
  Kapitel 14
- Planning Algorithms
  S. M. LaValle; 2006
  Kapitel III

Youtube: Stanford Lecture
Machine Learning (Andrew Ng)