INF FAKULTÄT FÜR INFORMATIK

Intelligente Systeme

Einleitung, reaktive Agenten, S-R-Agent

Prof. Dr. R. Kruse C. Braune C. Doell

{kruse,cbraune,doell}@iws.cs.uni-magdeburg.de

Institut für Wissens- und Sprachverarbeitung
Fakultät für Informatik
Otto-von-Guericke Universität Magdeburg

Übersicht

Zu meiner Person: Rudolf Kruse

1979 Dipl. Mathematik (Nebenfach Informatik) von TU Braunschweig

dort 1980 promoviert, 1984 habilitiert

2 Jahre hauptamtlicher Mitarbeiter bei Fraunhofer

1986 Ruf als Professor für Informatik der TU Braunschweig

seit 1996 Professor an der Universität Magdeburg

Forschung: Data Mining, Explorative Datenanalyse, Fuzzy-Systeme, Neuronale Netze, EA, Bayes'sche Netze

`mailto:kruse@iws.cs.uni-magdeburg.de`

Büro: G29-008, Telefon: 0391 67-58706

Sprechstunde: Mi., 11:00–12:00 Uhr

Zur Arbeitsgruppe: Computational Intelligence

Lehre:

Intelligente Systeme	Bachelor (2 V + 2 Ü, 5 CP)
Evolutionäre Algorithmen	Bachelor (2 V + 2 Ü, 5 CP)
Neuronale Netze	Bachelor (2 V + 2 Ü, 5 CP)
Fuzzy-Systeme	Master (2 V + 2 Ü, 6 CP)
Bayes-Netze	Master (2 V + 2 Ü, 6 CP)
Intelligente Datenanalyse	Master (2 V + 2 Ü, 6 CP)

Seminare: Clustering Algorithms, Classification Algorithms

Forschungsbeispiele:

Analyse und Simulation natürlicher neuronaler Netze (C. Braune)

Entscheidungstheorie / Heuristiken / Neuronale Netze (C. Doell)

Analyse von sozialen Netzen (P. Held)

Zur Vorlesung

Vorlesungstermine: Mo., 13:15–14:45 Uhr, G29-307

Vorlesungsende: 02.02.2015 (Datum der letzten Vorlesung)

Informationen zur Vorlesung:
`http://fuzzy.cs.ovgu.de/wiki/pmwiki.php?n=Lehre.IS1415`

- wöchentliche Vorlesungsfolien als PDF
- Übungsblätter ebenfalls
- wichtige Ankündigungen und Termine!

Inhalte der Vorlesung

Inhalte:

Eigenschaften intelligenter Systeme

Modellierungstechniken für wissensintensive Anwendungen

subsymbolische Lösungsverfahren, heuristische Suchverfahren, lernende Systeme

Modellansätze für kognitive Systeme, Wissensrevision und Ontologien

entscheidungsunterstützende Systeme

weitere aktuelle Methoden für die Entwicklung intelligenter Systeme wie kausale Netze, unscharfes Schließen

Zur Übung

Lernziele:

Befähigung zur Modellierung und Erstellung wissensintensiver Anwendungen durch Auswahl problementsprechender Modellierungstechniken

Anwendung heuristischer Suchverfahren und lernender Systeme zur Bewältigung großer Datenmengen

Befähigung zur Entwicklung und Bewertung intelligenter und entscheidungsunterstützender Systeme

Bewertung und Anwendung von Modellansätzen zur Entwicklung kognitiver Systeme

Ihre Aufgabe:

Nacharbeiten des Vorlesungsstoffs

Bearbeitung der Übungsaufgaben

aktive Teilnahme an den Übungen

Durchführung der Übungen

Sie werden aktiv und erklären Ihre Lösungen!

Sie votieren die Aufgaben, für die Sie bereit sind, einen Lösungsvorschlag zu präsentieren.

Tutor macht auf Fehler aufmerksam und beantwortet Fragen

das „Vorrechnen“ der Aufgaben ist nicht Sinn der Übung

ganz bewusst: keine ausgearbeiteten Musterlösungen

Tutoren: Christian Braune, Christoph Doell und Alexander Dockhorn

`mailto:cbraune@ovgu.de` bzw. `mailto:doell@ovgu.de`

Sprechstunde: stets ansprechbar (wenn Bürotür offen)

Übung: 4 Termine zur Auswahl

Mo., 15:15–16:45 Uhr in Raum G29-E037

Di., 15:15–16:45 Uhr in Raum G29-E037

Mi., 17:15–18:45 Uhr in Raum G29-E037

Do., 15:15–16:45 Uhr in Raum G29-K059

Anmeldung erforderlich!!! (noch bis 19.10.2014, 23:59 möglich)

`https://iws.cs.uni-magdeburg.de:8443/frs/subscribe/IS`

Übungsbeginn: 20., 21., 22. bzw. 23.10.2014 mit 1. Übungsblatt

Zur Prüfung

schriftliche Klausur: 120 Minuten

voraussichtlich im Februar/März 2015

Termine, Räume etc. werden in Vorlesung u. WWW angekündigt

keine Hilfsmittel zugelassen

Bekanntgabe der Ergebnisse: HISQIS

Einsichtnahme in die Klausur ist möglich (Termin im WWW)

Schein: Klausur bestehen

Zur Prüfungszulassung

mindestens 66% der Übungsaufgaben müssen votiert werden

mindestens zweimal im Semester eine Lösung vortragen

Votierung bedeutet: Mit der Aufgabe auseinandergesetzt und einen diskussionsfähigen Lösungsansatz präsentieren können

Votierung heißt nicht: Die perfekte Lösung vortragen können

Inhalt der Vorlesung

Einleitung, reaktive Agenten, S-R-Agent

Neuronale Netze

Fuzzy-Systeme

Evolutionäre Algorithmen

Computational Intelligence

Schwarmbasierte Verfahren

Zustandsagenten, Problemlösen und Suchen

Suchen und Planen

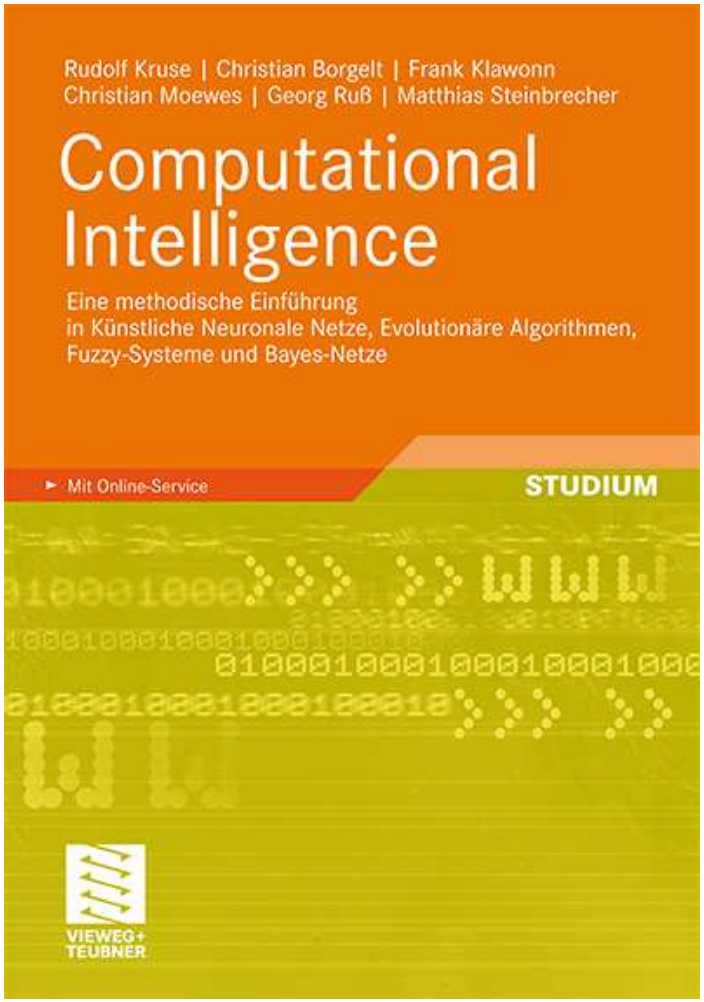
Wissensbasierte Systeme, Logik und Prolog

Unsicherheit

Maschinelles Lernen

Fallbasiertes Schließen

Buch zur Vorlesung



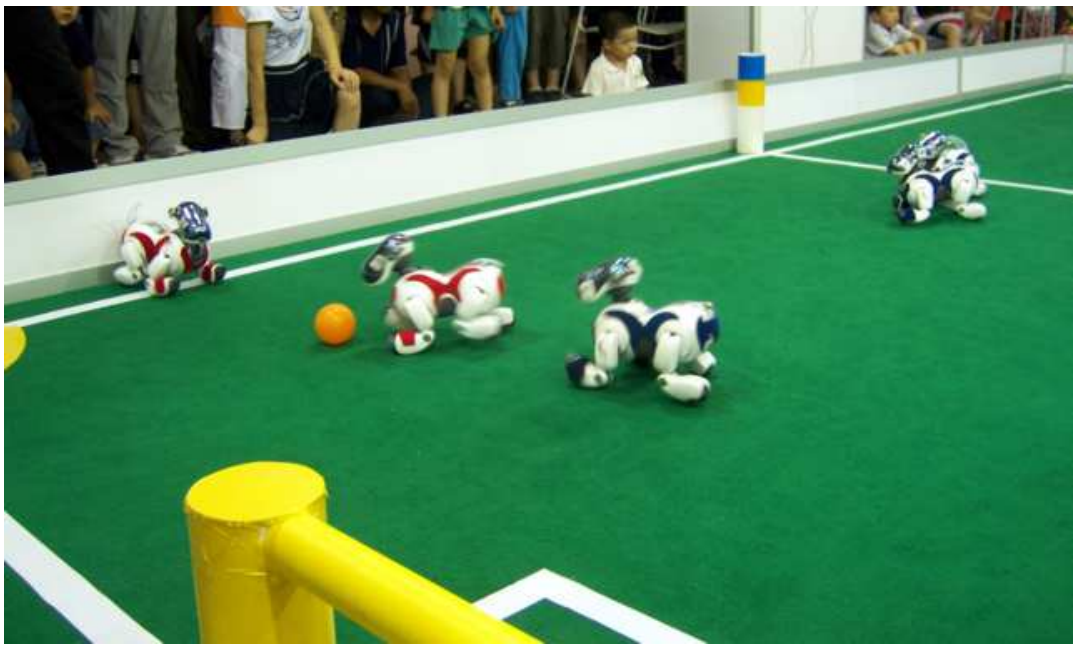
http://www.computational-intelligence.eu/

Übersicht

Intelligente Systeme beim „Robocup“

Ziel:

> *„Bis zum Jahr 2050 soll ein Team von vollständig autonomen humanoiden Robotern entwickelt werden, die gegen das menschliche Fußballweltmeisterschaftsteam gewinnen kann“*



Humanoider Roboter

weitere Informationen zum Robocup: `http://www.robocup.org`

„DARPA Grand Challenge“

Zwei (von elf) Teams: *Stanford Racing* und *Victor Tango*

Bild von `http://www.darpa.mil/GRANDCHALLENGE/gallery.asp`

Was ist ein Intelligentes System

Wissensverarbeitung wird benötigt, um künstliche intelligente Systeme zu entwickeln Begriffsbestimmungen:

Daten: Zeichen, die maschinell verarbeitet werden können

Information: durch entsprechende Interpretation erhalten Daten einen Sinn; Information ist der abstrakte Inhalt von Daten

Wissen: verknüpft Informationen sinngebend miteinander

Weisheit: verknüpft u.a. Erfahrungen, Humor, Wissen miteinander

Merkmale intelligenter Systeme:

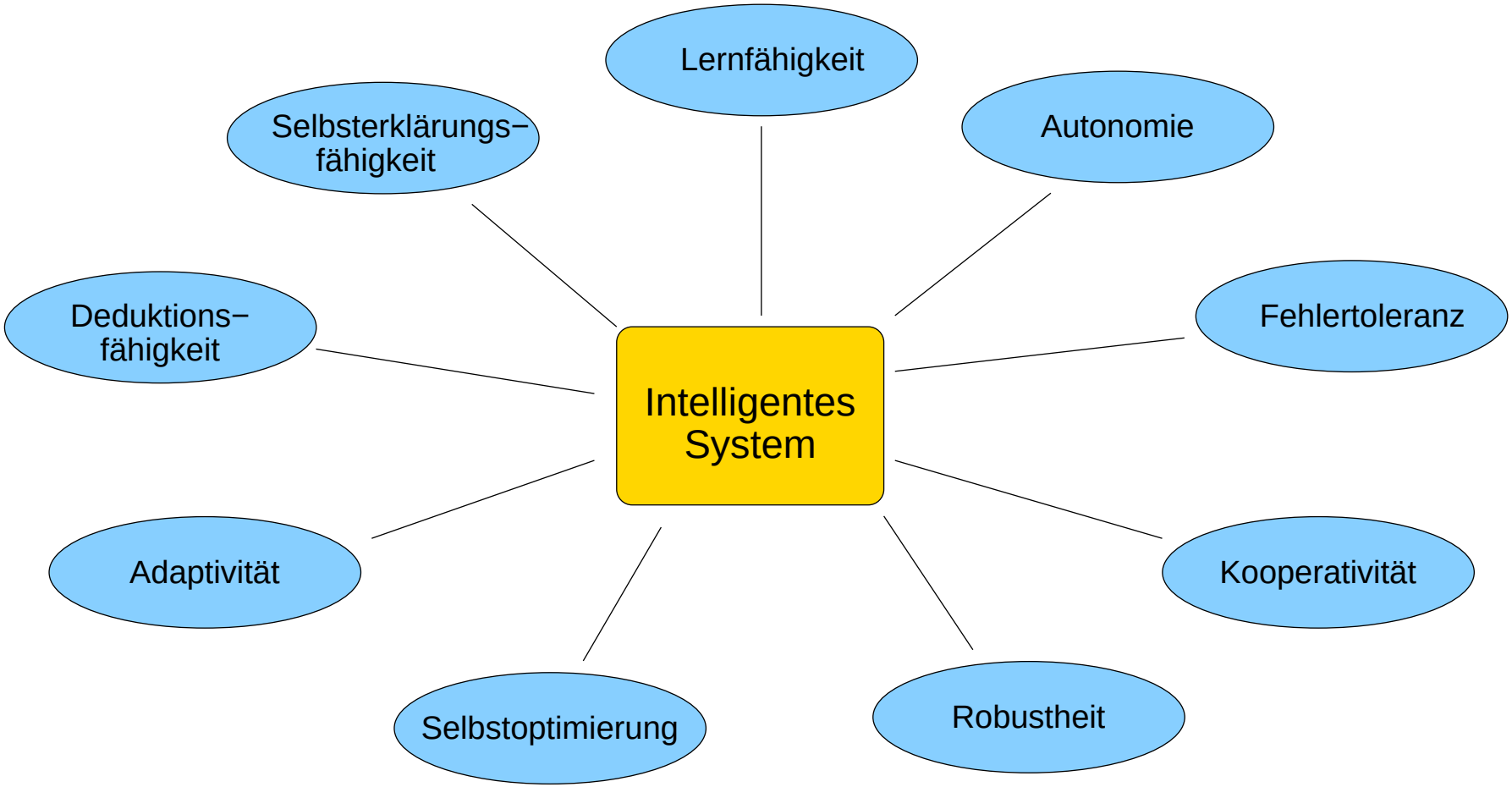
Lernfähigkeit

Autonomie

Fehlertoleranz

...

Merkmale Intelligenter Systeme



Was ist ein Intelligentes System?

Begriff *Intelligentes System* wandelt sich mit wissenschaftlichem Fortschritt:

1967 Taschenrechner

Programme zur symbolischen Integration

1997 Schachcomputer DEEP BLUE gewann gegen Kasparov 3.5 zu 2.5

1998 Automatikgetriebe AG4 für VW New Beetle

2002 Schachcomputer DEEP FRITZ gegen V. Kramnik 4 zu 4

Expertensysteme (wissensbasierte Systeme, intelligente Systeme)

VW Touareg Stanley bei DARPA Grand Challenge 2006

...

Was ist ein Intelligentes System?

Problem: oft anmaßende Voraussagen über Fortschritte der Künstlichen Intelligenz (KI) 1957: H. A. Simon (Nobelpreis 1978) und A. Newell (Turing-Award 1975) behaupten, dass 1967

Rechner Schachweltmeister sein wird,

Computer wichtigen neuen mathematischen Satz entdecken und beweisen wird,

digitaler Rechner Musikstück schreiben wird, welchem von Kritikern beachtlicher ästhetischer Wert bescheinigt wird.

Realistischere Einschätzungen

1993: deutscher Delphi-Report behauptet, dass 2005

Rechner entwickelt werden, die ungenaue Informationen in einer Art gesunden Menschenverstandes verarbeiten können,

Informationsdatenbanken eingesetzt werden, die durch automatisches Lernen ihr Wissen vermehren,

tragbare automatische Übersetzungsgeräte (einfache, alltägliche Konversation in beiden Richtungen) mit Spracheingaben kommerzialisiert werden,

in Büros Geräte weitverbreitet sind, die Texte in handschriftlicher Fließschrift lesen können.

Was ist ein Intelligentes System?

viele Herangehensweisen Gebiet ist hochgradig interdisziplinär

kognitiver Ansatz: Simulation kognitiver Prozesse, Analyse menschlicher Denkweise, *general problem solver*

ingenieurwissenschaftlicher Ansatz: Konstruktion von Systemen, die gewisse menschliche Wahrnehmungs- und Verstandsleistungen maschinell verfügbar machen (Produkte wie Gesichtserkenner, Roboter, Kooperation mit Gehirnforschern, . . .)

Was ist ein Intelligentes System?

spannende philosophische Fragestellungen, wie z.B.

Can machines think?

“can”

mehrere Bedeutungen:

„Kann denken“: heute – irgendwann – im Prinzip

derzeit: Frage „unentscheidbar“, ob Systeme mit menschenähnlichen Fähigkeiten gebaut werden können

wir sind mit Fortschritten auf dem Weg dahin zufrieden und verdienen Geld sowie Ruhm mit innovativen Produkten

Was ist ein Intelligentes System?

"machine"

muss kein Stahlroboter sein

kann auch biologischer Mechanismus sein (Bakterium Haemophilus influenzae Rd hat 10^7 Basenpaare, 1743 Gene, . . .)

Was, wenn menschliches Genom entziffert und verstanden?

auch: Bewusstseinsdiskussion, Leib-Seele-Problem, das „Ich“ im Gehirn

Was ist ein Intelligentes System?

"think"

Menschen sind Maschinen, also können Maschinen denken

Searle (1992): Denken funktioniert nur in speziellen (tatsächlich lebenden) Maschinen

Newell & Simon (1976): Physical symbol system (PSS) hypothesis: PSS hat notwendige und hinreichende Bedingungen für intelligentes Verhalten und PSS ist eine Maschine, die symbolische Daten manipulieren kann (z.B. Computer).

Kohonen u.a. (1980): Entwicklung intelligenter Maschinen nur durch subsymbolische Prozesse (z.B. Signale)

Zadeh (1964): wirklich intelligente Systeme müssen Art *fuzzy logic* benutzen (keine binäre Logik)

Turing-Test (1950) (*"bestanden"* 2014)

Turing-Test (1950)

Der Turing-Test wird von drei Personen gespielt: einem Mann (A), einer Frau (B) und einem Fragesteller (C). Der Fragesteller befindet sich in einem Raum, abgeschottet von A/B, und kommuniziert mit diesen über ein Terminal (teletype). Das Ziel des Spiels für den Fragesteller ist zu bestimmen, welche der beiden Personen der Mann und welche die Frau ist. Er adressiert die beiden mit Variablen X/Y und am Ende des Spiels sagt er "X ist A und Y ist B" oder "X ist B und Y ist A". Der Fragesteller kann Fragen beispielsweise der folgenden Form stellen:

> *C: X, würden Sie mir bitte Ihre Haarlänge verraten?*

Wenn mit "X" A adressiert wird, dann muß A jetzt antworten. Für A geht es darum, C in die Irre zu leiten und ihn zu einer falschen Identifikation zu verleiten.

...

Turing-Test (1950)

Das Ziel für den dritten Spieler B ist, dem Fragesteller zu helfen.

...

Jetzt stellen wir uns die Frage: "Was passiert, wenn eine Maschine den Anteil von A an diesem Spiel übernimmt?" Wird sich der Fragesteller genauso oft falsch entscheiden wenn das Spiel mit einer Maschine gespielt wird, wie wenn es mit Mann/Frau gespielt wird? Diese Fragestellung ersetzt das ursprüngliche "Kann eine Maschine denken?" Der Turing-Test wird oft vereinfacht zu einem Test, in dem eine Maschine versucht, einen menschlichen Fragesteller dazu zu verleiten, sie als Mensch zu identifizieren.

Was ist ein Intelligentes System?

Gebiet der Wissensverarbeitung ist extrem innovativ:
objektorientierte Programmiersprachen

graphische Oberflächen

Expertensysteme

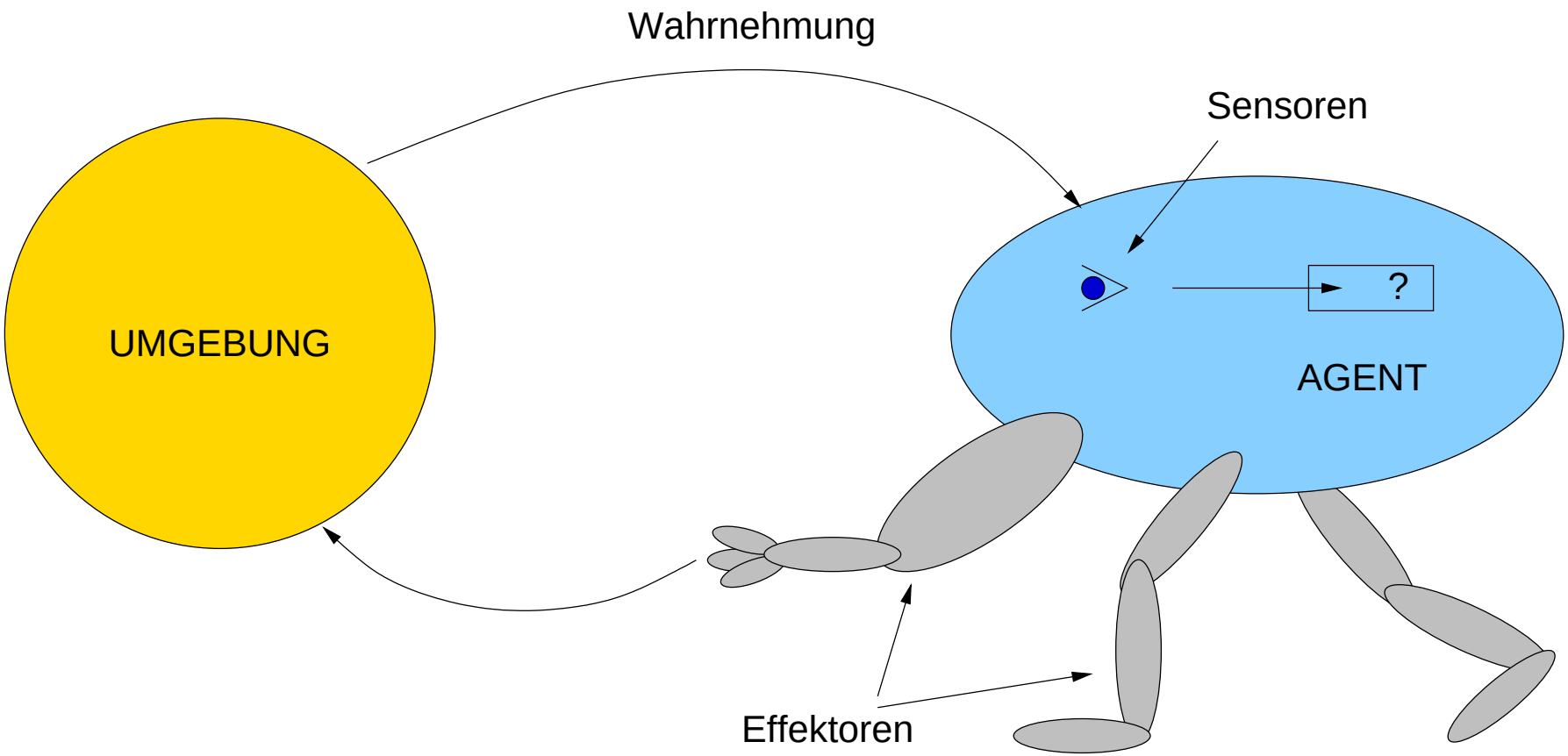
Software-Agenten (Internet)

Autonome Roboter
sind hier erfunden worden.

Übersicht

Agenten

Ein intelligenter Agent interagiert mit seiner Umgebung mittels Sensoren und Effektoren und verfolgt gewisse Ziele:

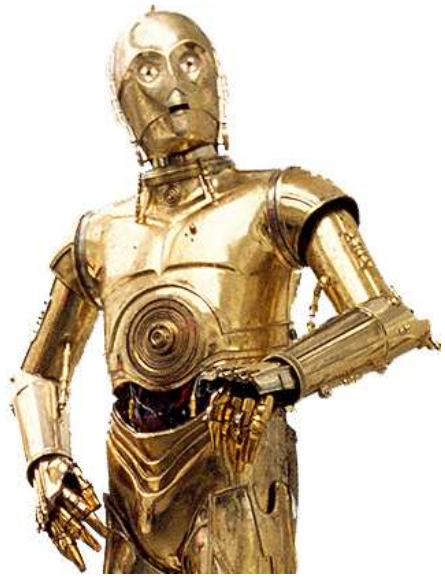


Beispiele für Agenten

Menschen und Tiere

Roboter und Software-Agenten (Softbots)

aber auch: Heizungen, ABS, ...



Agenten

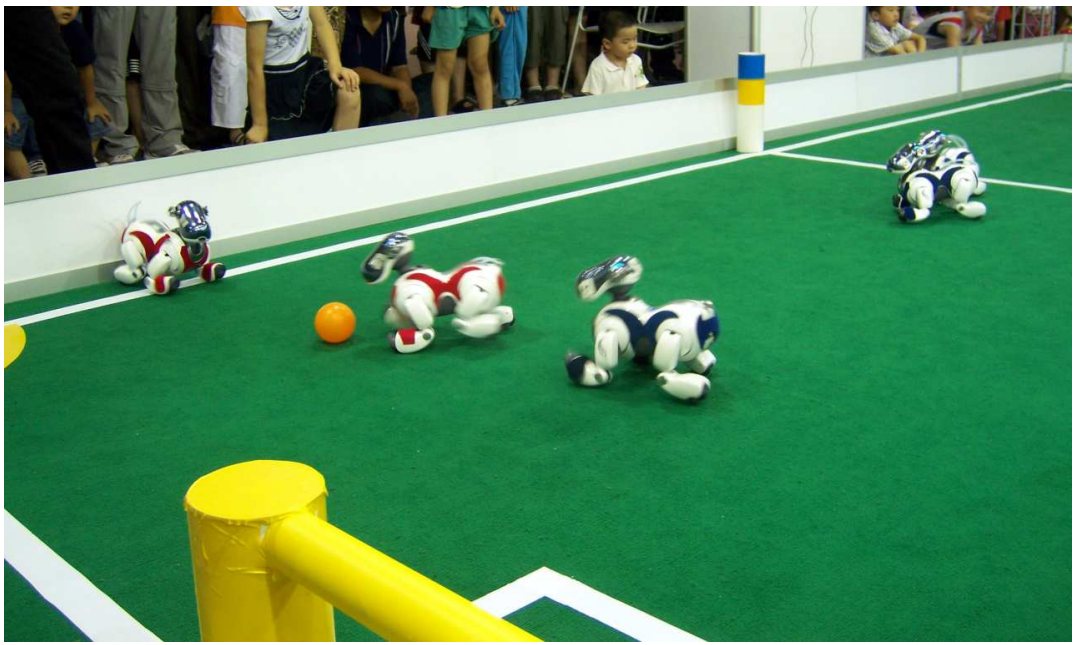
auch andere Definitionen aus unterschiedlichen Fachgebieten, z.B.:

Ein Programm ist ein Softwareagent, wenn es korrekt in einer (Agenten-)Sprache wie ACL, KQML oder KIF kommuniziert.
BDI-Agenten werden durch Überzeugungen (*beliefs*), Wünsche (*desires*) und Absichten (*intentions*) beschrieben; praktisch werden sie mit einer Modallogik und speziellen Datenstrukturen implementiert.

Agenten

Beispiel: Simulation Soccer

RoboCup: Roboterfußball



Beispiel 2: Taxifahrer

Typ	**Taxifahrer**
Wahrnehmung	Kameras, Tachometer, GPS, Mikrofon
Aktionen	Steuern, Schalten, Bremsen, mit Fahrgästen sprechen
Ziele	Sichere, schnelle, legale, komfortable Fahrt; Profit maximieren
Umgebung	Straßen, andere Verkehrsteilnehmer: Fußgänger, Radfahrer; Fahrgäste

Übersicht

Charakterisierung von Agenten

Agenten können charakterisiert werden durch:

Wahrnehmungen (***p****erceptions*)

Aktionen (***a****ctions*)

Ziele (***g****oals*)

Umgebung (***e****nvironment*)

$\Leftrightarrow$*PAGE*

Beispiele von Agenten nach *PAGE*

Art	Wahrnehmung	Aktionen	Ziele	Umgebung
Medizinisches Diagnosesystem	Symptome, Diagnose, Antworten des Patienten	Fragen, Tests, Behandlungen	Gesundheit, geringe Kosten	Patient, Krankenhaus
Satellitenbildanalyse	Punkte verschiedener Intensität	Klassifikation	Korrekte Klassifikation	Satellitenbilder
Roboter	Punkte verschiedener Intensität	Teile aufheben und einsortieren	Teile richtig einsortieren	Förderband mit Teilen

Beispiele von Agenten nach *PAGE*

Art	**Wahrnehmung**	**Aktionen**	**Ziele**	**Umgebung**
Raffinerie-Regler	Temperatur, Druck	Öffnen, Schließen von Ventilen, Temperatur einstellen	Reinheit, Ertrag, Sicherheit maximieren	Raffinerie
Interaktiver Englisch-Tutor	Eingegebene Wörter und Übungen, Vorschläge	Korrekturen ausgeben	Testergebnisse des Studenten maximieren	Menge von Studenten

Typen von Agenten (I)

Unterscheidung von Agenten nach Art und Weise ihrer Umwelt-Interaktionen:

reaktive Agenten:

steuern über ein Reiz-Antwort-Schema ihr Verhalten

reflektive Agenten:

agieren planbasiert, verarbeiten also explizit Pläne, Ziele und Intentionen

situierte Agenten:

verbinden einfaches Reagieren und überlegtes Handeln in dynamischer Umwelt

Typen von Agenten (II)

autonome Agenten:

sind zwischen reflektiven und situierten Agenten einzuordnen (werden meist in Robotik verwendet)

rationale Agenten:

entsprechen reflektiven Agenten, allerdings mit ausgeprägter Bewertungsfunktionalität

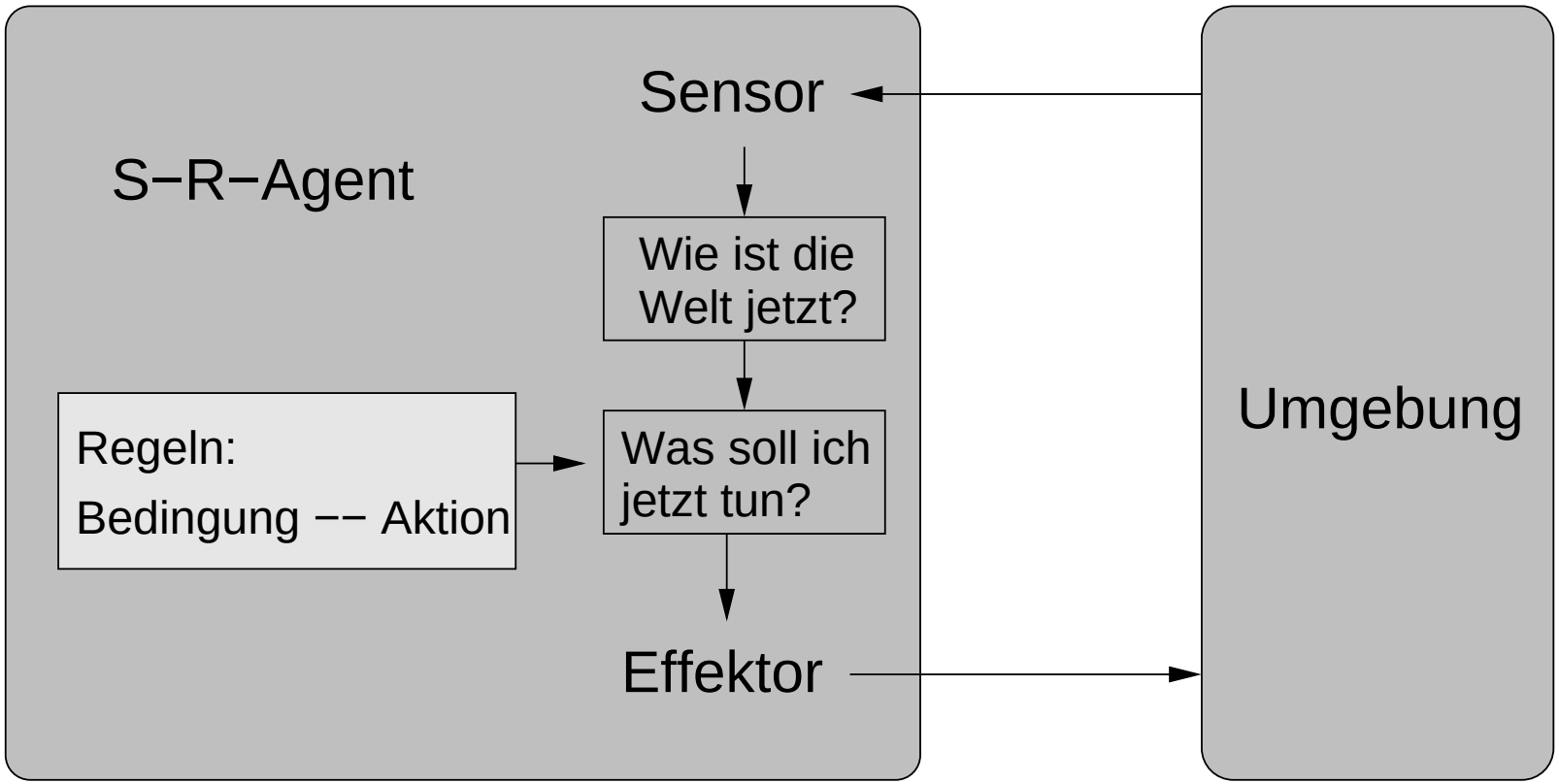
soziale Agenten:

sind in der Lage, ihr Handeln an Gemeinziel auszurichten

Übersicht

Stimulus-Response-Agent

einfacher reaktiver Agent: antwortet unmittelbar auf Wahrnehmungen



„Skelett“ eines Stimulus-Response-Agenten

```
function S-R-Agent(percept) returns action
    static: rules
    state INTERPRET-INPUT(percept)
    rule RULE-MATCH(state, rule)
    action RULE-ACTION(rule)
    return action
```

Agent sucht Regel, deren Bedingung der gegebenen Situation entspricht

er führt zugehörige Aktion (Regel-Konklusion) aus

Gitterwelt (I)

Umwelt = fiktive zweidimensionale Gitterzelleneinheit

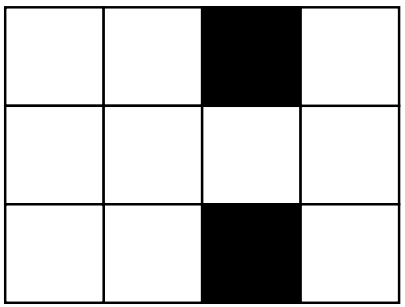
verschiedene (Spielzeug-)Agenten tummeln sich dort

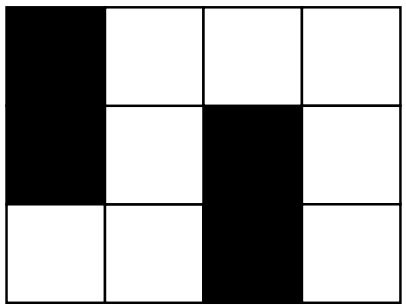
in Zellen können Objekte mit verschiedenen Eigenschaften sein

es gibt Barrieren

Agenten können von Zelle zu Zelle laufen

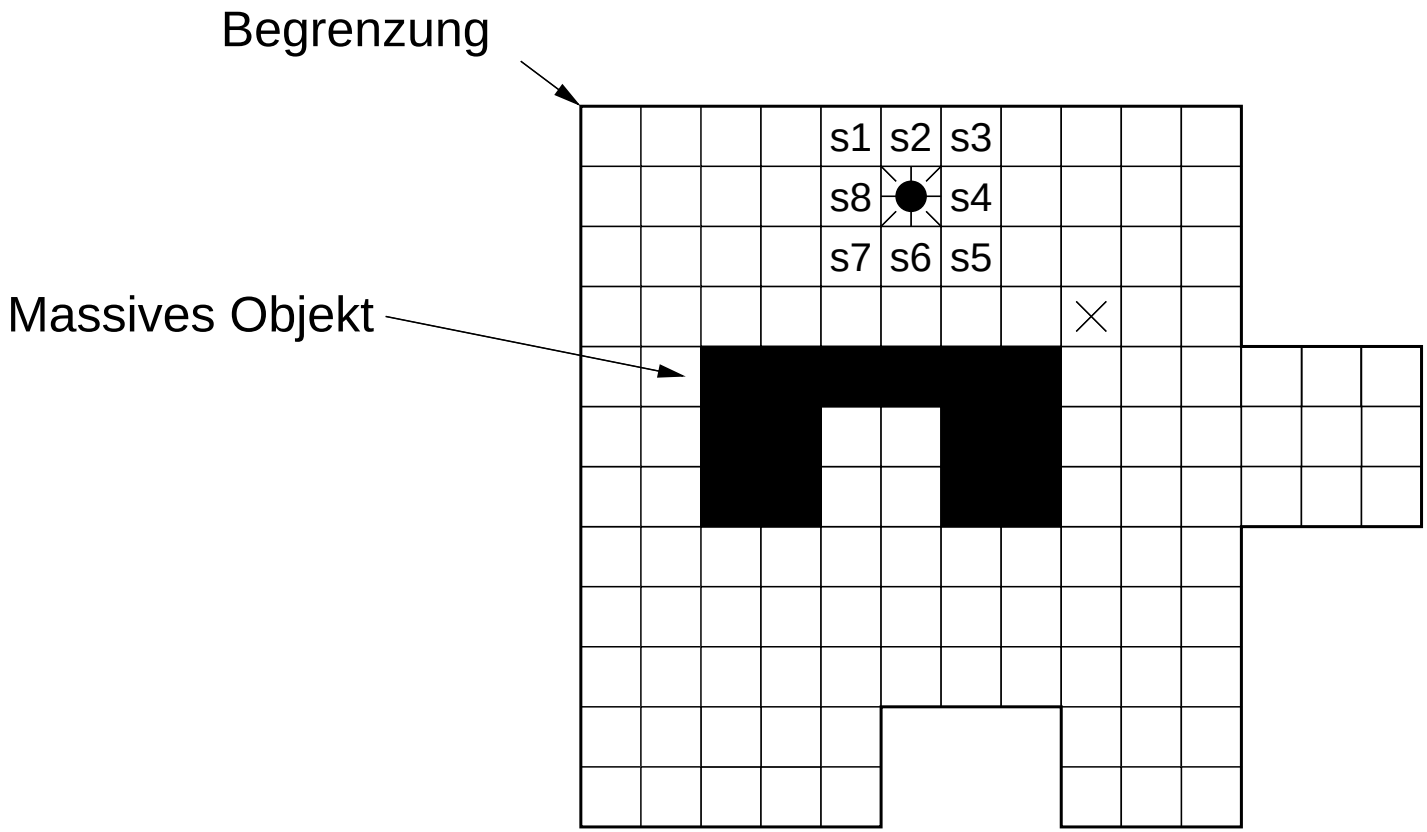
keine engen Zwischenräume, d.h. keine Lücken zwischen Objekten und Begrenzungen, die nur 1 Zelle breit sind (*tight spaces*)





Solche Umgebungen sind *nicht* erlaubt!

Gitterwelt (II)



Roboter kann mithilfe der Sensoren $s_1, \ldots, s_8$ feststellen, welche Zellen in seiner Nachbarschaft belegt sind

Gitterwelt (III)

$s_i \in \{0, 1\}$, Sensoreingabe:

$s_j = 0 \Leftrightarrow$ Zelle s_j ist frei für Roboter

an der mit $\times$ markierten Stelle: $(0, 0, 0, 0, 0, 0, 1, 0)$

vier mögliche Aktionen:

north, east, south, west

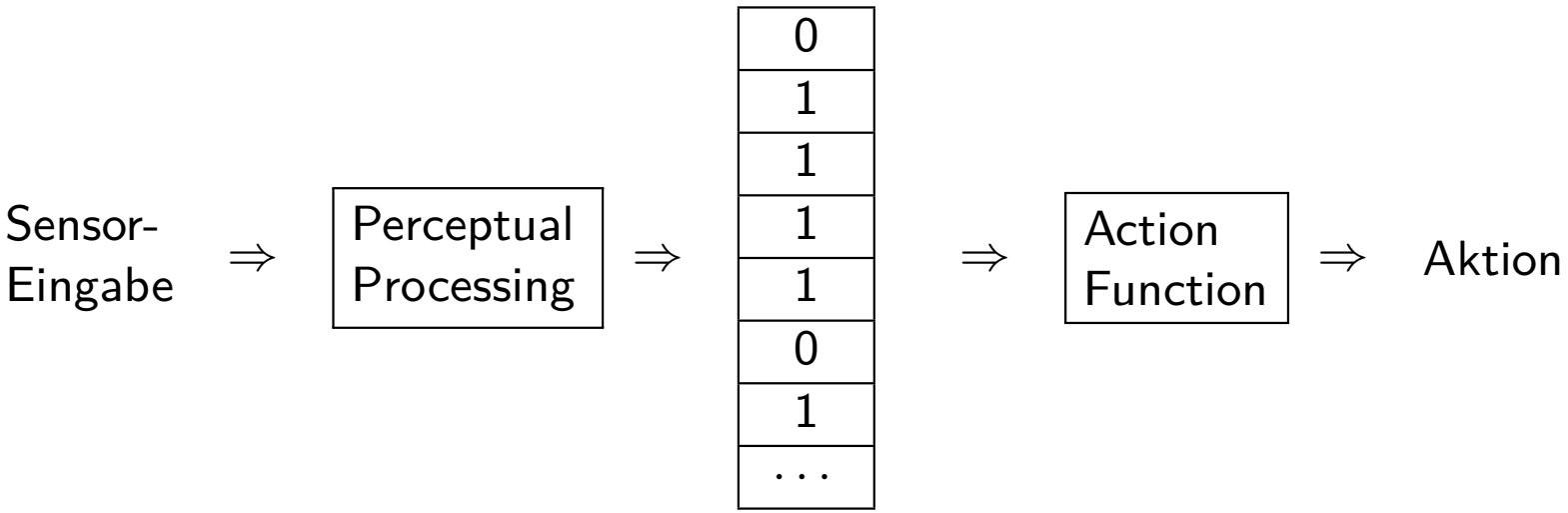
z.B. north bewegt Roboter 1 Zelle nach oben, falls Zelle frei ist, ansonsten wird nicht bewegt

Aufgabe häufig in 2 Schritten gelöst:

Phase: perception processing

Phase: action computation

Komponenten: *Perception* und *Action*



Eigenschaftsvektor $X = (0, 1, 1, 1, 1, 0, 1, \cdots)^T$

vom Entwickler zugewiesene Bedeutungen:

- $(0, 1, \boxed{\mathbf{1}}, 1, 1, 1, \cdots)$: „an einer Wand“
- $(0, 1, 1, 1, \boxed{\mathbf{1}}, 1, \cdots)$: „in einer Ecke“

Beispiel: Wandverfolgung (I)

Aufgabe: gehe zu einer Zelle an Begrenzung eines Objekts und folge dieser Grenze

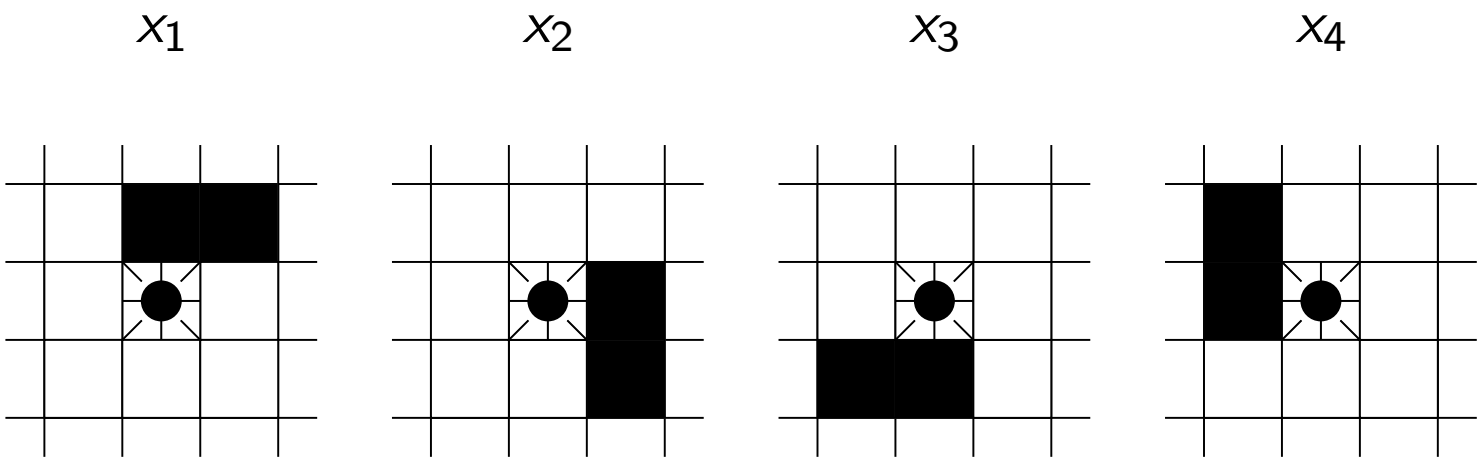
Perception:

- 2^8 verschiedene Sensoreingaben, von denen einige wegen Einschränkung (*keine engen Zwischenräume*) wegfallen
- vier Merkmale $x_1, \cdots, x_4$:

$$\begin{aligned} x_1 = 1 &\Leftrightarrow (s_2 = 1 \vee s_3 = 1) \\ x_2 = 1 &\Leftrightarrow (s_4 = 1 \vee s_5 = 1) \\ x_3 = 1 &\Leftrightarrow (s_6 = 1 \vee s_7 = 1) \\ x_4 = 1 &\Leftrightarrow (s_8 = 1 \vee s_1 = 1) \end{aligned}$$

Beispiel: Wandverfolgung (II)

Merkmal in jedem Diagramm hat genau dann Wert 1, wenn mindestens 1 der markierten Zellen belegt



in komplexen Welten: Informationen sind typischerweise unsicher, vage, oder sogar falsch

Beispiel: Wandverfolgung (III)

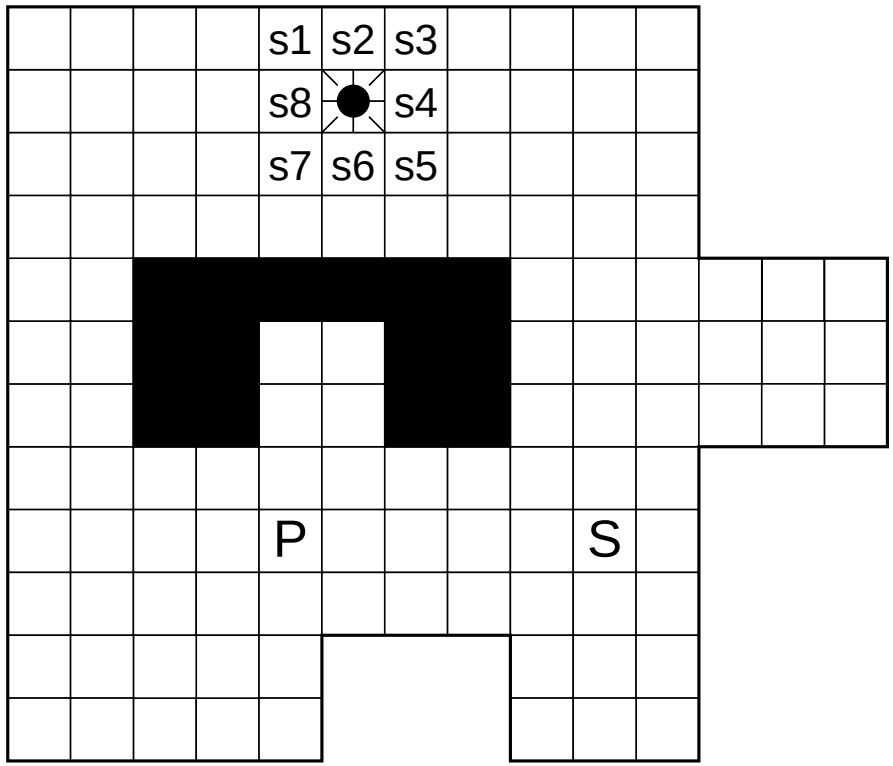
Aktionen:

falls keines der 4 Merkmale Wert 1 hat, führe `north` durch

sonst:

$$\text{If } x_1 = 1 \text{ and } x_2 = 0 \text{ then } \texttt{east}$$
$$\text{If } x_2 = 1 \text{ and } x_3 = 0 \text{ then } \texttt{south}$$
$$\text{If } x_3 = 1 \text{ and } x_4 = 0 \text{ then } \texttt{west}$$
$$\text{If } x_4 = 1 \text{ and } x_1 = 0 \text{ then } \texttt{north}$$

Beispiel: Wandverfolgung (IV)



Roboter, der an Position P startet, bewegt sich entgegen Uhrzeigersinn am Objekt entlang

Roboter, der an Position S startet, bewegt sich im Uhrzeigersinn an der äußeren Begrenzung entlang

Auswertung der Sensoreingaben

für beiden Phasen *perception processing* und *action computation* werden oft Boolesche Algebren verwendet

so gilt:

$$x_4 = s_1 \vee s_8 \text{ und go north}$$
$$\Leftrightarrow$$
$$(\bar{x_1} \wedge \bar{x_2} \wedge \bar{x_3} \wedge \bar{x_4}) \vee (x_4 \wedge \bar{x_1}) = 1$$

Auswertung der Sensoreingaben

Geeignete Repräsentationsform für Aktionen sind Regelsysteme der Form $c_j \rightarrow a_j$, wobei

c_j der Bedingungsteil und

a_j der Aktionsteil sind

In unserem Beispiel erhält man folgende Regeln:

$$\begin{aligned} x_4 \wedge \bar{x_1} &\rightarrow \text{go north} \\ x_1 \wedge \bar{x_2} &\rightarrow \text{go east} \\ x_2 \wedge \bar{x_3} &\rightarrow \text{go south} \\ x_3 \wedge \bar{x_4} &\rightarrow \text{go west} \\ 1 &\rightarrow \text{go north} \end{aligned}$$

Regelsysteme und Boolesche Algebren kann man gut anhand von Netzwerken implementieren

Wandverfolgung

Alternativer Ansatz zu S-R-Agenten: Einführung von Subsumptions-Modulen

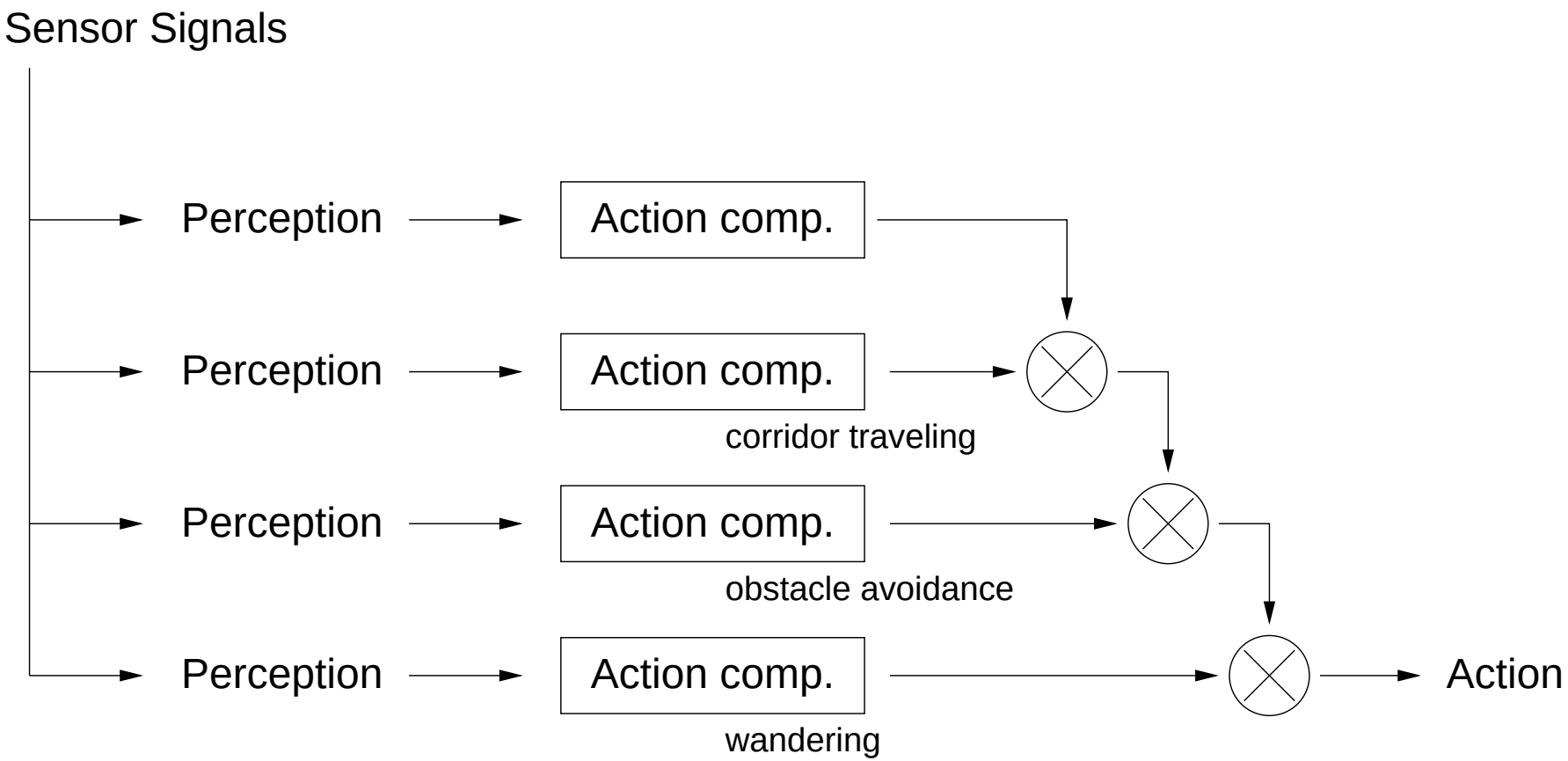
jedes Modul enthält Sensorinformationen direkt von Umwelt

sind spezifizierten Voraussetzungen des Moduls erfüllt, wird Programm ausgeführt

„höhere“ Module subsumieren „tiefere“, d.h. falls Voraussetzung eines höheren Moduls erfüllt, wird tieferes Modul durch höheres ersetzt

enditemize

Wandverfolgung



Beispiel aus unserer Forschung: Sicherheitskritische Systeme

Fehlfunktion kann zu schweren Unfällen, Umwelt- oder physischen Schäden oder Todesopfern führen

Fehlentscheidungen solcher Systeme können für gewöhnlich nicht korrigiert werden

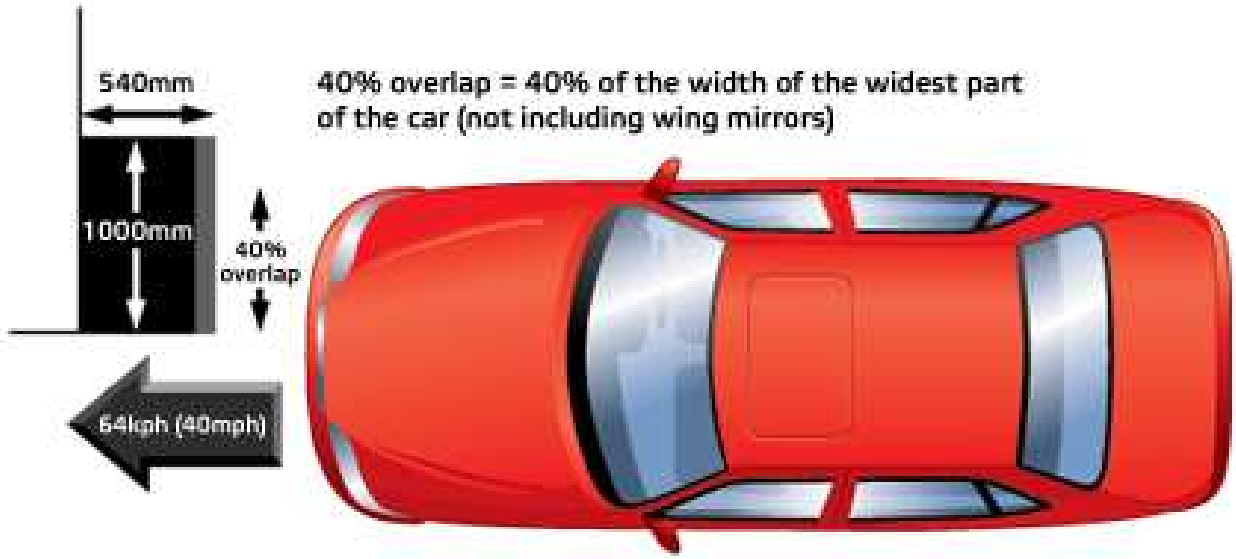
Airbag-Zündung

Airbag soll nur in schweren Crashs zünden

Krankheitsdiagnose

kein kranker Patient soll als gesund diagnostiziert werden

Frontal-Crashes



Quelle: `http://www.euroncap.com/tests/frontimpact.aspx`

gegen starre Wand (engl. wall crash)

mit relativem Winkel von 30° zwischen 2 Autos (angular crash),

gegen deformierbare Barriere mit Versatz (offset deformable barrier (ODB) crash)

bei $v = 15\,\mathrm{km/h}$ (Allianz Zentrum für Technik (AZT) crash)

Crash-Sensorik

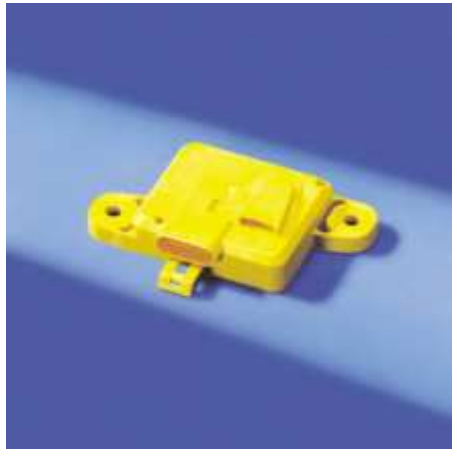
Airbag-
steuergerät

Sensor-
positionen

Insassen-
klassifikations-
system

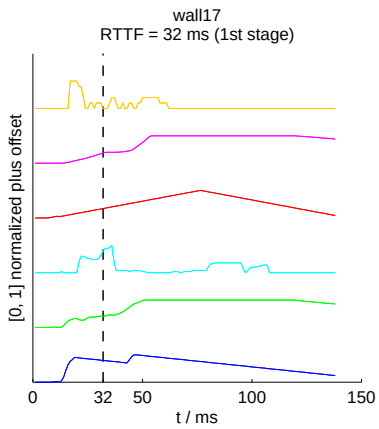
“Early Crash”
Sensor

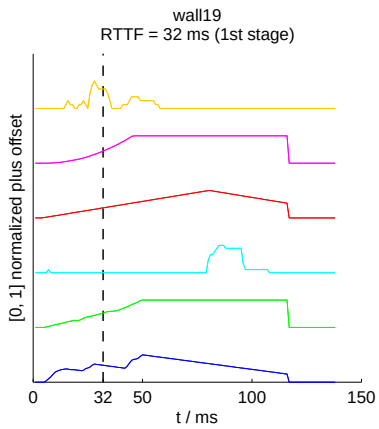
Beschleunigungs-
sensor

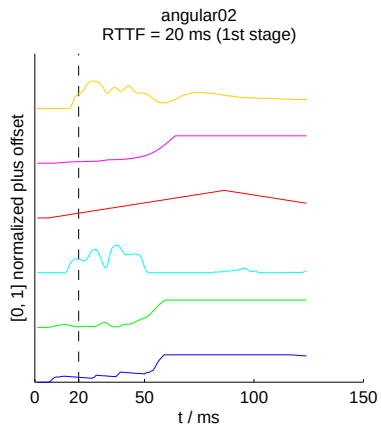


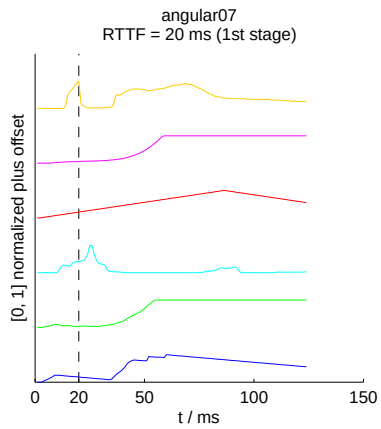
Drucksensor

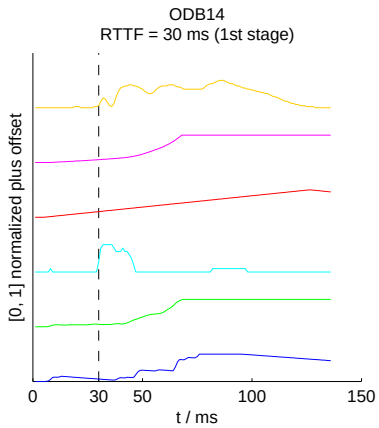
Beispielhafte Crash-Signale

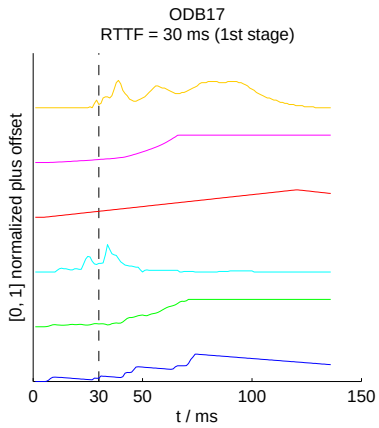


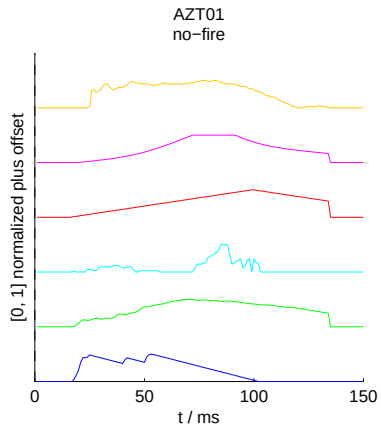


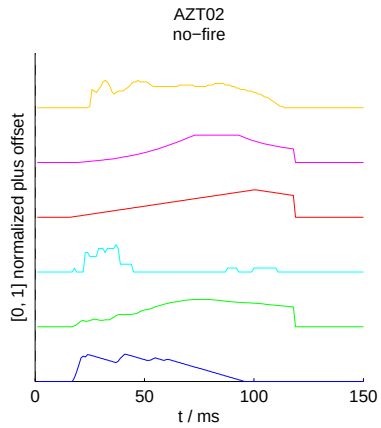












Stand der Technik

Verwendung von regelbasierten Agenten **(manuell erstellt)**

Zwei-Klassen-Problem: positive (Crash, krank) und negative (kein Crash, gesund) Beispiele (Fahrten, Patienten)

jedes Beispiel: beschrieben durch numerischen Attributwerten

Reale Regelbasis: Erkennen von Wand-Crashes

```
 1: (X01 >= 221) & (X02 >= 164) => deploy airbag
 2: (X01 >= 215) & (X03 >= 22) => deploy airbag
 3: (X04 >= 225) & (X02 >= 188) => deploy airbag
 4: (X01 >= 177) & (X02 >= 248) => deploy airbag
 5: (X04 >= 176) & (X02 >= 236) => deploy airbag
 6: (X04 >= 171) & (X02 >= 231) & (X05 >= 74) & (X06 >= 14) => deploy airbag
 7: (X04 >= 164) & (X07 >= 3) & (X06 >= 9) => deploy airbag
 8: (X02 >= 224) & (X03 >= 26) => deploy airbag
 9: (X01 >= 149) & (X08 >= 106) => deploy airbag
10: (X01 >= 144) & (X03 >= 32) => deploy airbag
11: (X04 >= 150) & (X07 >= 4) => deploy airbag
12: (X07 >= 2) & (X02 >= 255) => deploy airbag
13: (X07 >= 5) & (X09 >= 131) => deploy airbag
14: (X010 >= 255) & (X07 >= 3) & (X05 >= 231) => deploy airbag
15: (X010 >= 255) & (X07 >= 3) & (X011 >= 77) & (X06 >= 10) => deploy airbag
16: (X07 >= 3) & (X06 >= 24) & (X09 >= 134) => deploy airbag
17: (X010 >= 255) & (X07 >= 2) & (X02 >= 50) & (X03 >= 22) => deploy airbag
18: (X010 >= 255) & (X07 >= 2) & (X02 >= 188) & (X03 >= 11) => deploy airbag
19: (X010 >= 255) & (X07 >= 2) & (X02 >= 188) & (X05 >= 255) & (X06 >= 26) => deploy airbag
20: (X07 >= 2) & (X02 >= 179) & (X03 >= 14) => deploy airbag
21: (X07 >= 2) & (X02 >= 176) & (X08 >= 90) => deploy airbag
22: (X010 >= 255) & (X07 >= 2) & (X01 >= 128) & (X08 >= 93) & (X06 >= 15) => deploy airbag
23: (X010 >= 255) & (X07 >= 1) & (X01 >= 131) & (X08 >= 103) & (X06 >= 23) => deploy airbag
24: (X07 >= 2) & (X01 >= 133) & (X08 >= 137) => deploy airbag
25: (X01 >= 131) & (X08 >= 105) & (X03 >= 24) => deploy airbag
26: (X010 >= 226) & (X07 >= 3) & (X012 >= 13) & (X06 >= 15) & (X03 >= 3) => deploy airbag
27: (X07 >= 2) & (X01 >= 115) & (X06 >= 52) & (X03 >= 19) => deploy airbag
28: (X010 >= 138) & (X07 >= 3) & (X01 >= 94) & (X03 >= 9) & (X09 >= 113) => deploy airbag
```