

Intelligente Systeme

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

- 1. Intelligente Systeme**
2. Agenten
3. Stimulus-Response-Agenten

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:

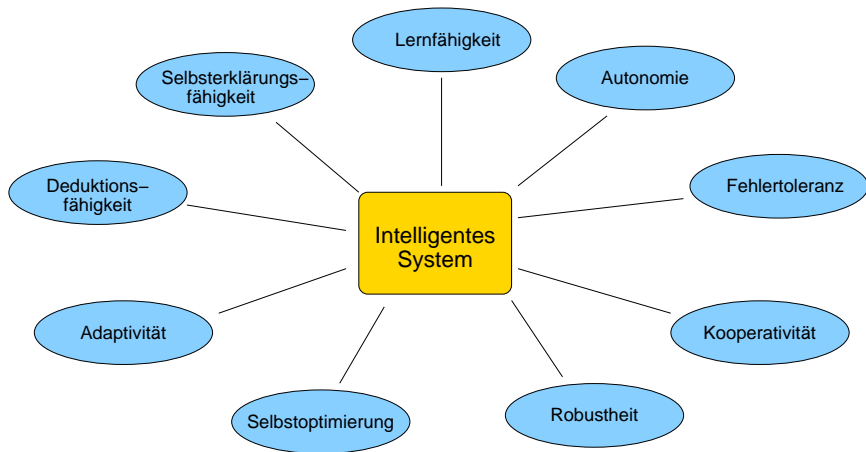
Daten: Zeichen, die maschinell verarbeitet werden können

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



Künstliche Intelligenz

Der Begriff *Intelligentes System* wandelt sich mit wissenschaftlichem Fortschritt:

1956 Dartmouth Project on Artificial Intelligence

1967 Taschenrechner

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

2006 VW Touareg Stanley bei DARPA Grand Challenge

2011 IBM Watson (Jeopardy)

...

Künstliche Intelligenz

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

Künstliche Intelligenz

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.

Künstliche Intelligenz

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, ...)

Künstliche Intelligenz

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

Künstliche Intelligenz

“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

Künstliche Intelligenz

“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.

Turing-Test Beispiel 1

Judge: How do you like Bletchley Park?

Entity: lol.

Judge: Are you from England?

Entity: They have Wi-Fi here in the pub.

Judge: Which pub?

Entity: I'm just down the pub.

Judge: Have you ever been in a Turing Test before?

Entity: Will this be the 5 min argument, or were you thinking of going for the full half hour?

Judge: Very funny. You sound suspiciously human. Do you like the beatles?

Entity: I'd like to get the next Dread the Fear tape.

Judge: What is Dread the Fear?

Entity: Dread the fear has that Steve Henderson guy in it.

Judge: What sort of music is that? Or is it comedy?

Turing-Test Beispiel 2

Judge: Why hello there!

Entity: Why hello to you too!

Judge: How are you feeling in this fine day?

Entity: To be quite honest a little rejected, I thought you were never going to reply :(

Judge: Oh, I'm very sorry, it will not happen again.

Entity: It just did!

Judge: Oh, I lied then.

Entity: That's a great shame indeed.

Judge: It is. Are you following the Euro 2012's at the moment?

Entity: Yeah quite closely actually. I am Cristiano Ronaldo.

Frage: Ist der Teilnehmer *Entity* ein Mensch oder ein Computer?

Was ist ein Intelligentes System?

Gebiet der Wissensverarbeitung ist extrem innovativ:

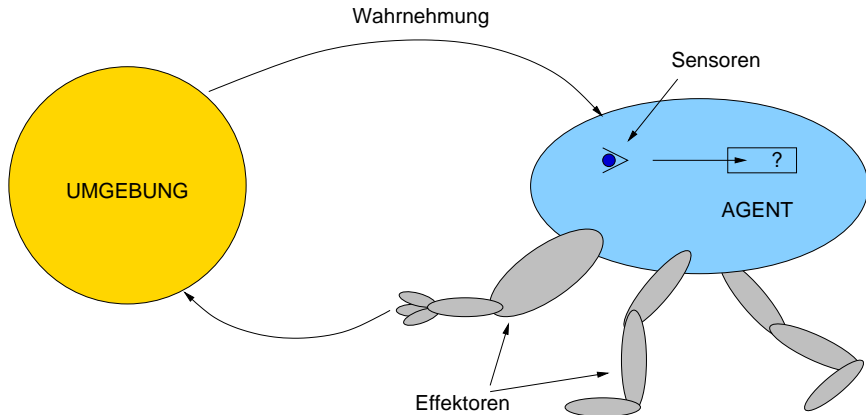
- objektorientierte Programmiersprachen
- graphische Oberflächen
- Expertensysteme
- Software-Agenten (Internet)
- Autonome Roboter
sind hier erfunden worden.

Übersicht

1. Intelligente Systeme
- 2. Agenten**
3. Stimulus-Response-Agenten

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

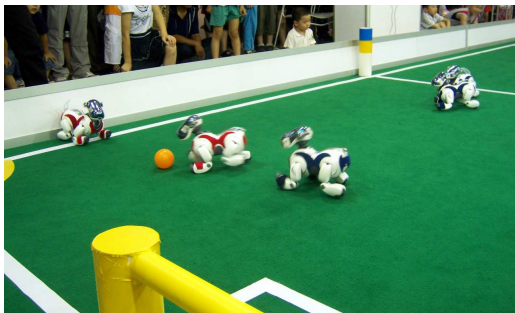
Auch andere Definitionen aus unterschiedlichen Fachgebieten, z.B.:

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



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

Charakterisierung von Agenten

Agenten können charakterisiert werden durch (*PAGE*):

Wahrnehmungen (*perceptions*)

Aktionen (*actions*)

Ziele (*goals*)

Umgebung (*environment*)

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

1. Intelligente Systeme

2. Agenten

3. Stimulus-Response-Agenten

Gitterwelt

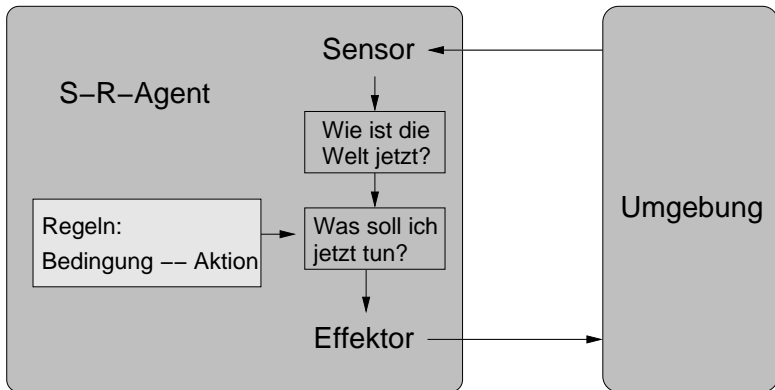
Signalverarbeitung

Beispiel: Wandverfolgung

Sicherheitskritische Systeme

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

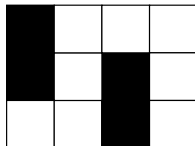
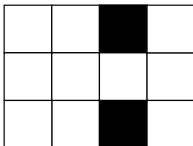
Verschiedene (Spielzeug-)Agenten tummeln sich dort

In Zellen können Objekte mit verschiedenen Eigenschaften sein

Es gibt Barrieren

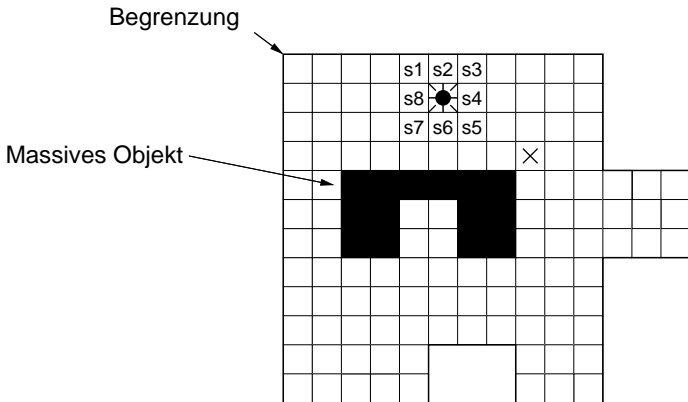
Agenten können von Zelle zu Zelle laufen

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, \dots, s_8 feststellen, welche Zellen in seiner Nachbarschaft belegt sind

Gitterwelt (III)

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

- $s_j = 0 \Leftrightarrow$ Zelle s_j ist frei für Roboter
- an der mit \times markierten Stelle: $(0, 0, 0, 0, 0, 0, 1, 0)$

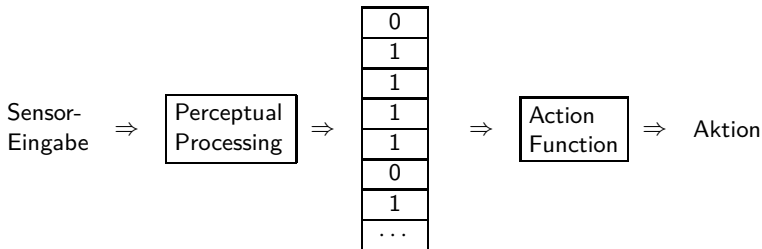
Vier mögliche Aktionen:

- 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 $X = (0, 1, 1, 1, 1, 0, 1, \dots)^T$

vom Entwickler zugewiesene Bedeutungen:

- $(0, 1, \mathbf{1}, 1, 1, 1, \dots)$: „an einer Wand“
- $(0, 1, 1, 1, \mathbf{1}, 1, \dots)$: „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, \dots, x_4 :

$$x_1 = 1 \Leftrightarrow (s_2 = 1 \vee s_3 = 1)$$

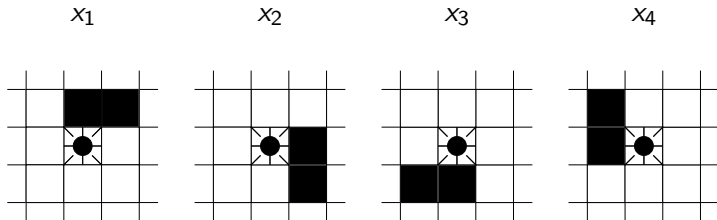
$$x_2 = 1 \Leftrightarrow (s_4 = 1 \vee s_5 = 1)$$

$$x_3 = 1 \Leftrightarrow (s_6 = 1 \vee s_7 = 1)$$

$$x_4 = 1 \Leftrightarrow (s_8 = 1 \vee s_1 = 1)$$

Beispiel: Wandverfolgung (II)

Das 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:

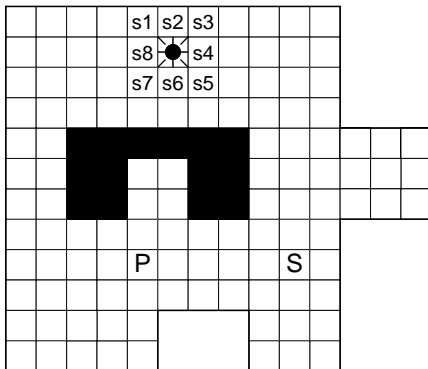
If $x_1 = 1$ and $x_2 = 0$ then `east`

If $x_2 = 1$ and $x_3 = 0$ then `south`

If $x_3 = 1$ and $x_4 = 0$ then `west`

If $x_4 = 1$ and $x_1 = 0$ then `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:

$$\begin{aligned}x_4 &= s_1 \vee s_8 \text{ und go north} \\ &\Leftrightarrow \\ (\bar{x}_1 \wedge \bar{x}_2 \wedge \bar{x}_3 \wedge \bar{x}_4) \vee (x_4 \wedge \bar{x}_1) &= 1\end{aligned}$$

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:

$$x_4 \wedge \bar{x}_1 \rightarrow \text{go north}$$

$$x_1 \wedge \bar{x}_2 \rightarrow \text{go east}$$

$$x_2 \wedge \bar{x}_3 \rightarrow \text{go south}$$

$$x_3 \wedge \bar{x}_4 \rightarrow \text{go west}$$

$$1 \rightarrow \text{go north}$$

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

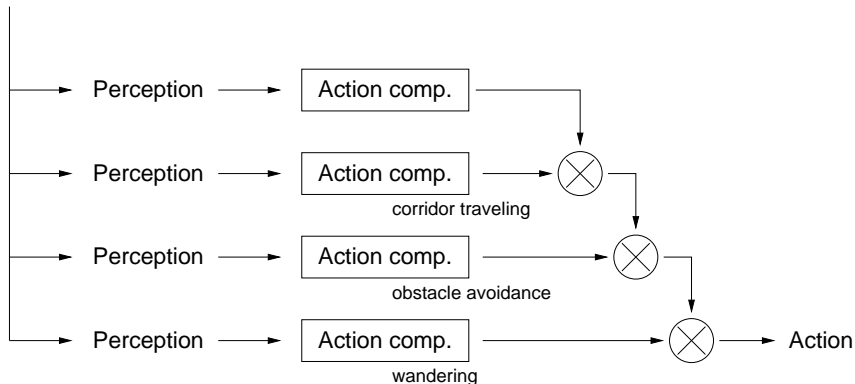
sind spezifizierten Voraussetzungen des Moduls erfüllt, wird Programm ausgeführt

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

enditemize

Wandverfolgung

Sensor Signals



Beispiel aus unserer Forschung: Sicherheitskritische Systeme

Fehlfunktion kann zu schweren Unfällen, Umwelt- oder physischen Schäden oder Todesopfern führen und können für gewöhnlich nicht korrigiert werden



Airbag-Zündung
nur bei schweren Crashes

Crash-Sensorik



Airbag-
steuergerät



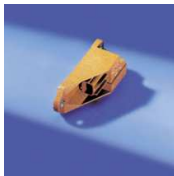
Sensor-
positionen



Insassen-
klassifikations-



“Early Crash”
Sensor

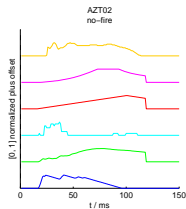
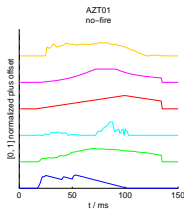
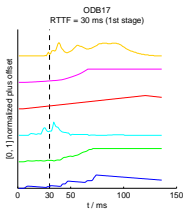
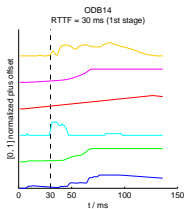
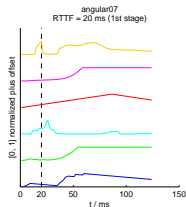
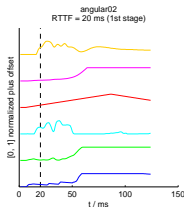
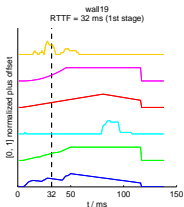
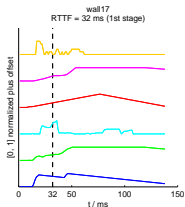


Beschleunigungs-
sensor



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
```