Intelligente Systeme

Einführung

Prof. Dr. Rudolf Kruse Georg Ruß
Christian Moewes

{kruse,russ,cmoewes}@iws.cs.uni-magdeburg.de

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

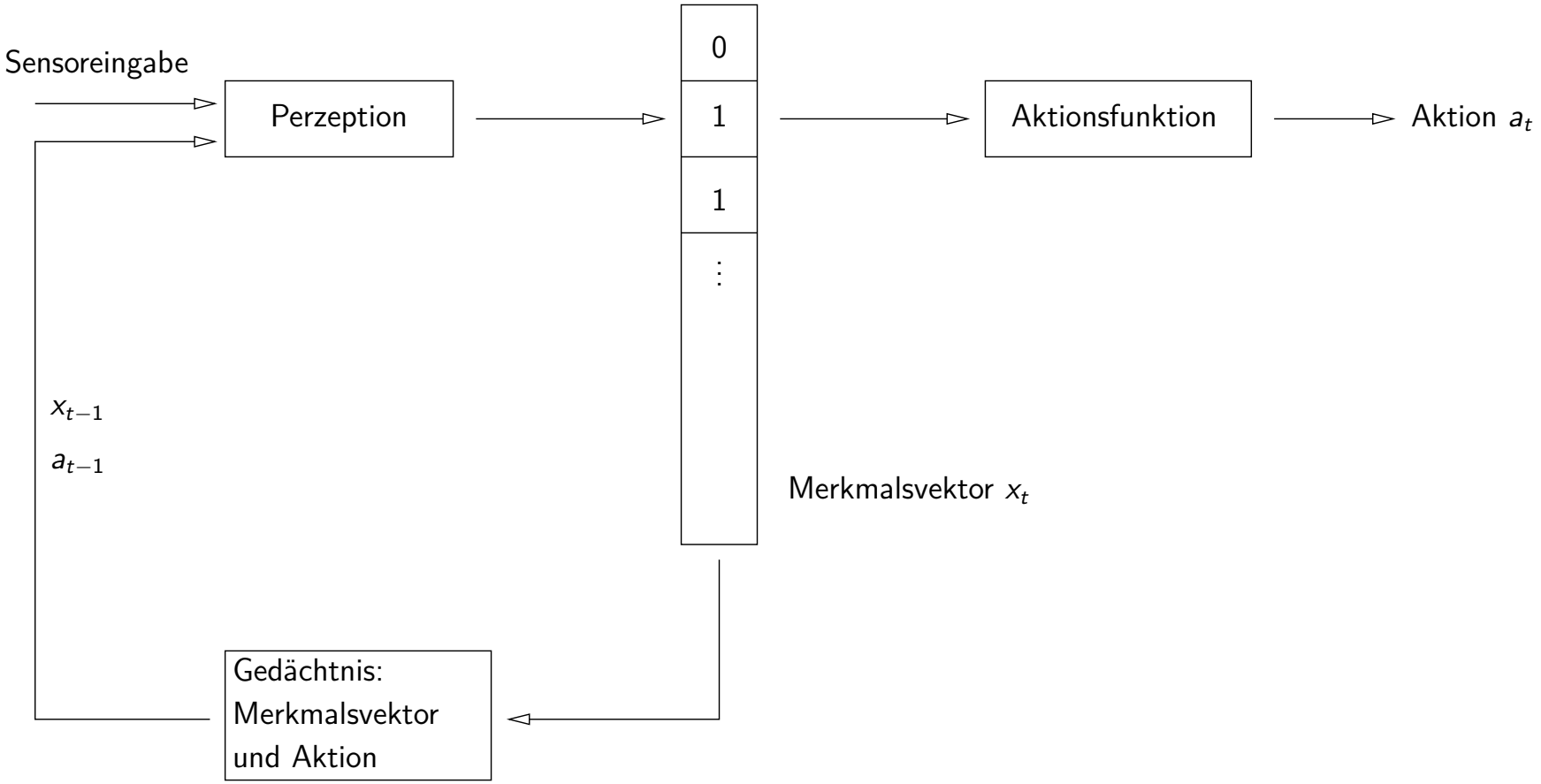
Zustandsautomaten

Zustandsautomaten

- Bisher:
  - S-R-Agenten mit unmittelbarer Reaktion auf Sensorreize
- Jetzt:
  - Ausnutzung von Sensorinformationen aus der Vergangenheit

Zustandsautomaten

- Darstellung der Umgebung mit Merkmalsvektoren
- Beispiel:

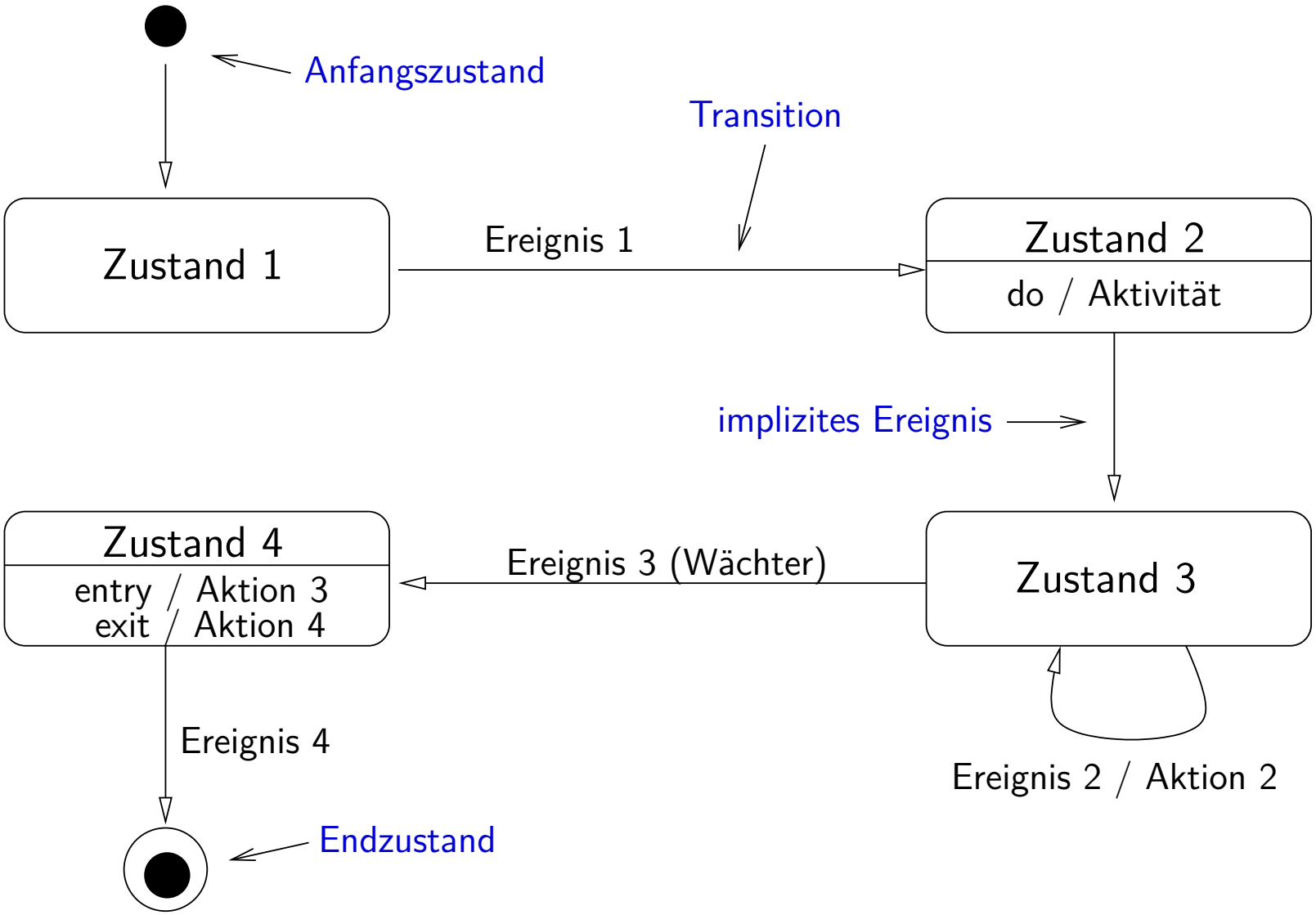


Zustandsautomaten

finite state machine / state chart diagram

- Anfangs-/Endzustand
- Zustände und Zustandsübergänge
- Objekt durchläuft Zustände
- *ein* Zeitpunkt = *ein* Zustand

Zustandsautomaten



Beispiel: Roboter in Gitterwelt (1)

- Roboter in Gitterwelt mit begrenzter Sensorinformation
  - Sensoreingabe zum Zeitpunkt t:
    - s_2^t, s_4^t, s_6^t, s_8^t (d.h. nur vier statt bisher acht Sensoren)
    - $s_i^t = 1 \leftrightarrow \mathrm{Feld}\ s_i^t$ ist nicht frei
  - Aufgabe: Wandverfolgung
  - Idee: den Merkmalsvektor des jeweils vorherigen Zeitpunkts nutzen

Beispiel: Roboter in Gitterwelt (2)

- Definition der Merkmalsvektoren:
  - $w_i^t = s_i^t$ für $i = 2, 4, 6, 8$
  - $w_1^t = 1 \leftrightarrow w_2^{t-1} = 1 \text{ and } a_{t-1} = \text{east}$
  - $w_3^t = 1 \leftrightarrow w_4^{t-1} = 1 \text{ and } a_{t-1} = \text{south}$
  - $w_5^t = 1 \leftrightarrow w_6^{t-1} = 1 \text{ and } a_{t-1} = \text{west}$
  - $w_7^t = 1 \leftrightarrow w_8^{t-1} = 1 \text{ and } a_{t-1} = \text{north}$
- Damit können eingeschränkte Sensorinformationen (teilweise) ausgeglichen werden!

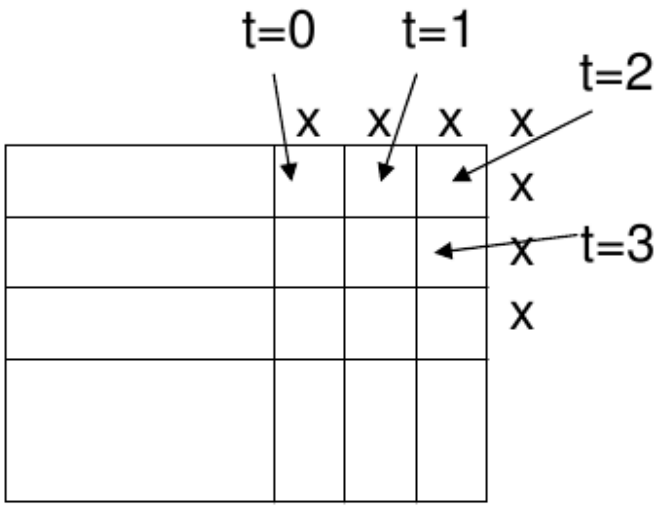
Beispiel: Roboter in Gitterwelt (3)

- Sinnvolle Aktionen zur Wandverfolgung:
  - $w_2^t \wedge \neg w_4^t \rightarrow \text{east}$
  - $w_4^t \wedge \neg w_6^t \rightarrow \text{south}$
  - $w_6^t \wedge \neg w_8^t \rightarrow \text{west}$
  - $w_8^t \wedge \neg w_2^t \rightarrow \text{north}$
  - $w_1^t \wedge \neg w_2^t \rightarrow \text{north}$
  - $w_3^t \wedge \neg w_4^t \rightarrow \text{east}$
  - $w_5^t \wedge \neg w_6^t \rightarrow \text{south}$
  - $w_7^t \wedge \neg w_8^t \rightarrow \text{west}$
  - alle $w_i = 0 \rightarrow \text{north}$

Beispiel: Roboter in Gitterwelt (4)

- Implementierung (Beispiel einer Bewegung):

	Sensoreingabe				Merkmalsvektor								Aktion
t	s_2^t	s_4^t	s_6^t	s_8^t	w_1^t	w_2^t	w_3^t	w_4^t	w_5^t	w_6^t	w_7^t	w_8^t	a_t
0	1	0	0	0	0	1	0	0	0	0	0	0	east
1	1	0	0	0	1	1	0	0	0	0	0	0	east
2	1	1	0	0	1	1	0	1	0	0	0	0	south
2'	*0*	*0*	*0*	*0*	*1*	*0*	*0*	*0*	*0*	*0*	*0*	*0*	*north*



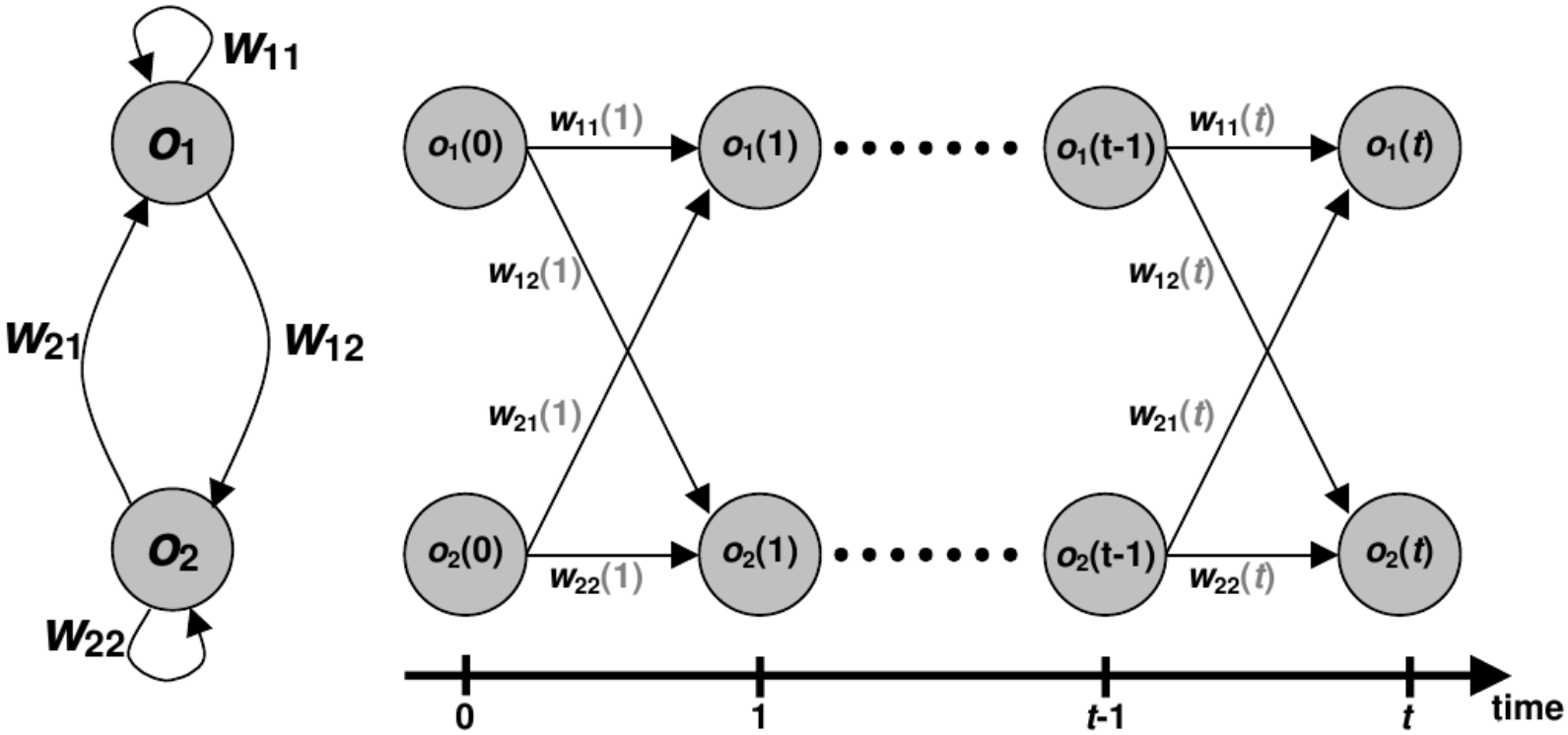
Anmerkung: Situation t=2‘ z.B. bei Sensorstörung (alle Sensoren s_i=0)

Rekurrente Neuronale Netze

- In rückgekoppelten neuronalen Netzen sind Verbindungen von oberen zu unteren Schichten erlaubt (Multilayerperzeptron: nur Verbindungen von “unten” nach “oben”).
- Durch die Rückkopplung kann Information gespeichert werden.
- Rückgekoppelte Netze können durch Modifikationen von Backpropagation-Verfahren trainiert werden.
- Rückgekoppelte Netze können komplexe Differentialgleichungssysteme approximieren.

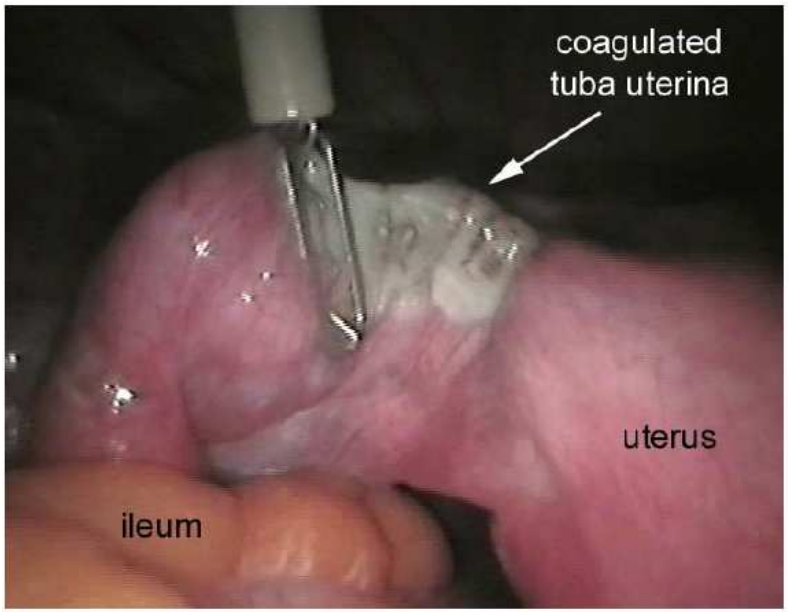
Rekurrente Netze: Lernverfahren

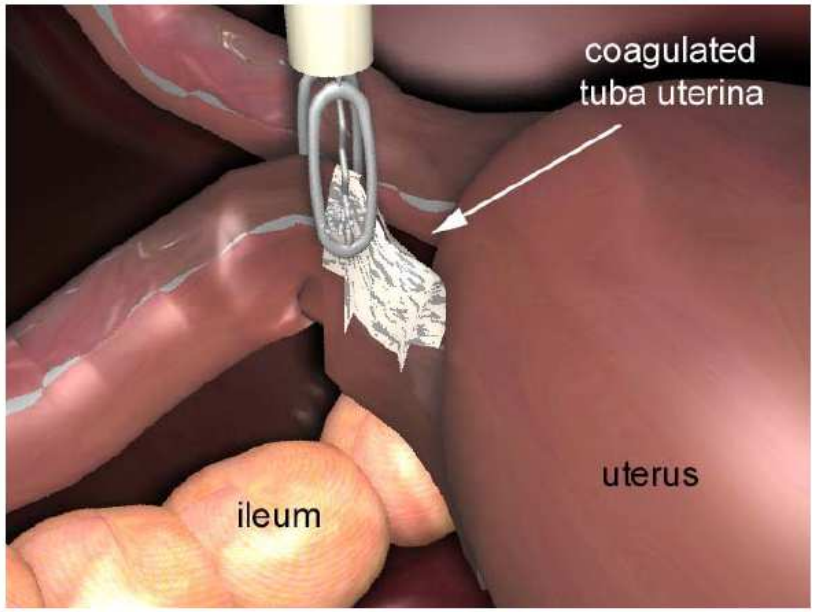
- Idee der Lernverfahren: Entfalten des Netzes über die Zeit

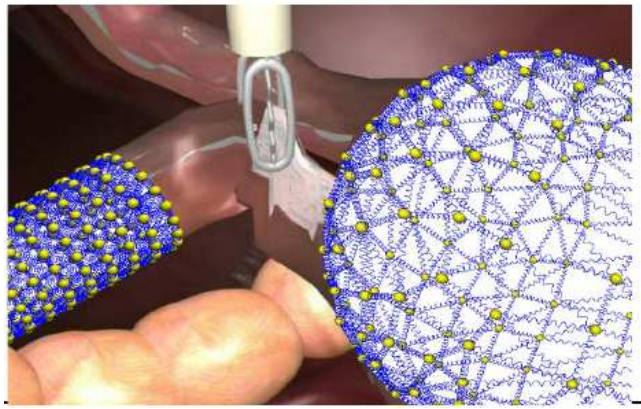


SUSILAP-G

SUrgical SImulator for LAParoscopy in Gynaecology





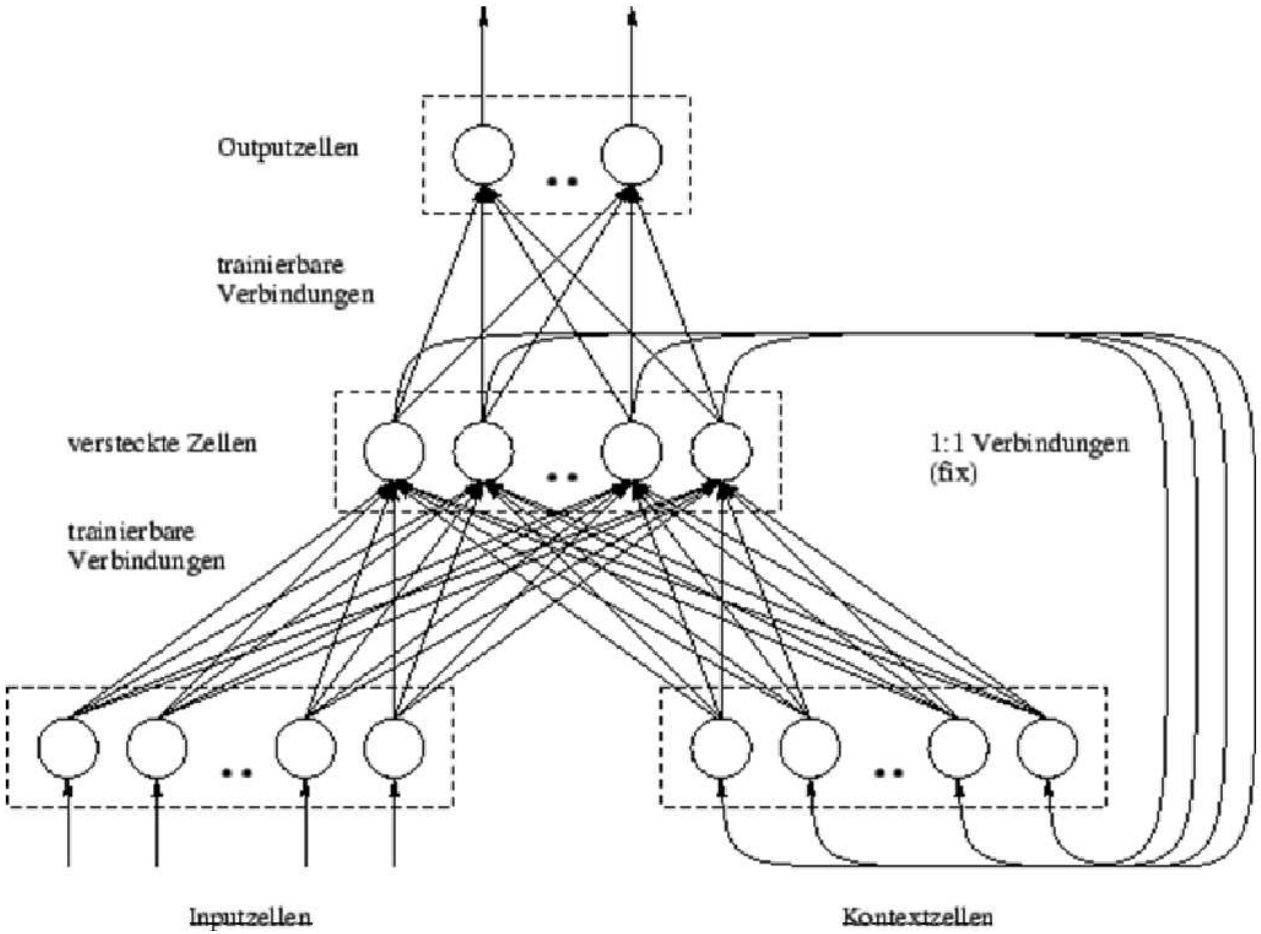


[Radetzky et al., 1999]

Einfache RNNs

Elman-Netze

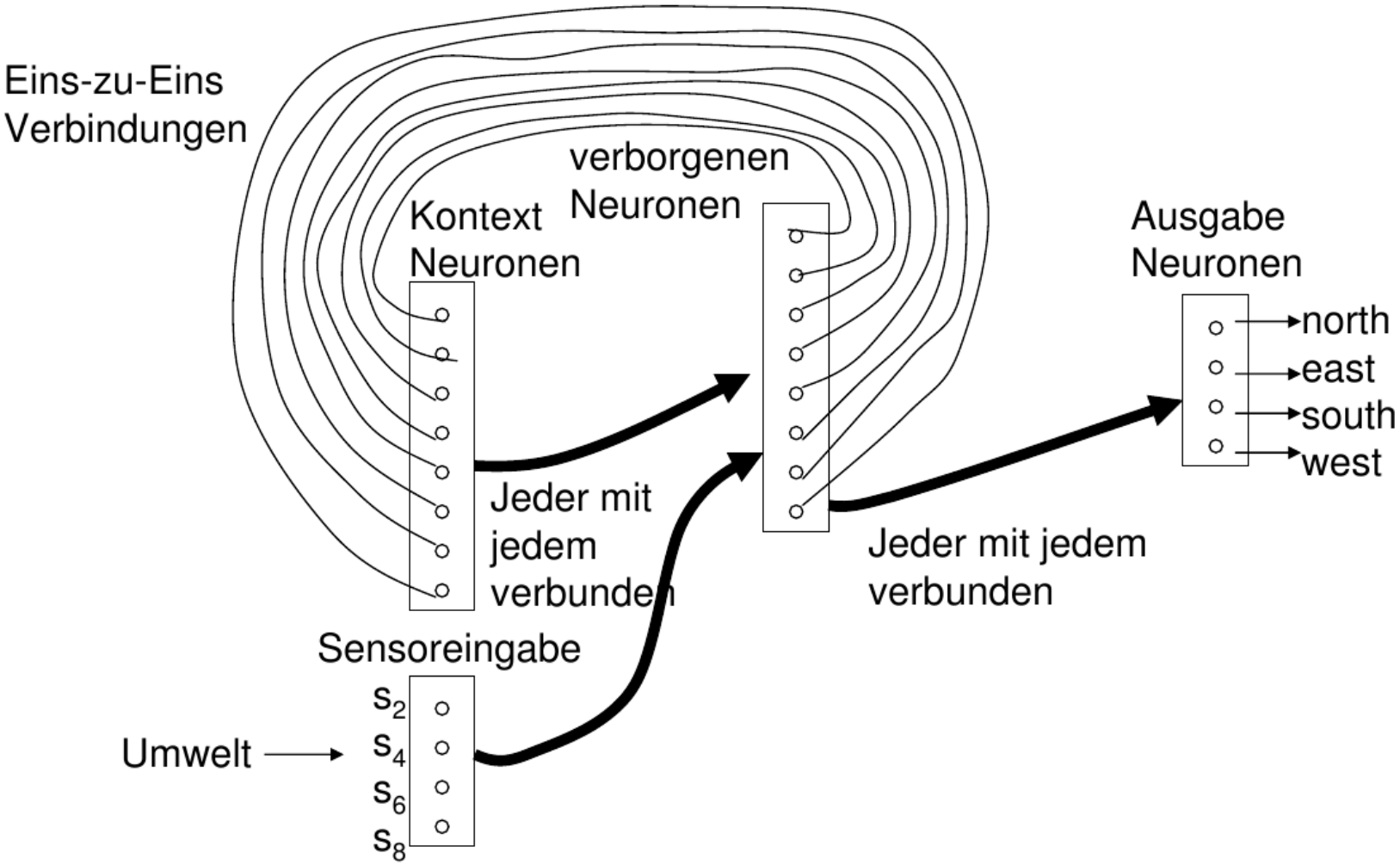
- Einführung einer Kontextschicht, die es ermöglicht, Informationen zu speichern



Beispiel: Elman-Netz für Roboter

- Elman Netz für Roboter in Gitterwelt:
  - 8-dimensionale Merkmalsvektoren (im Allgemeinen ist die Anzahl der Dimensionen unbekannt)
  - 4 Eingaben (Sensoren)
  - 4 Ausgaben (Richtungen; Ausgabe mit größtem Wert wird gewählt)
  - Dieses rückgekoppelte Netz kann auch mit Backpropagation trainiert werden.
  - Man kann diese Netze als "lernfähige" Automaten auffassen.

Beispiel: Elman-Netz

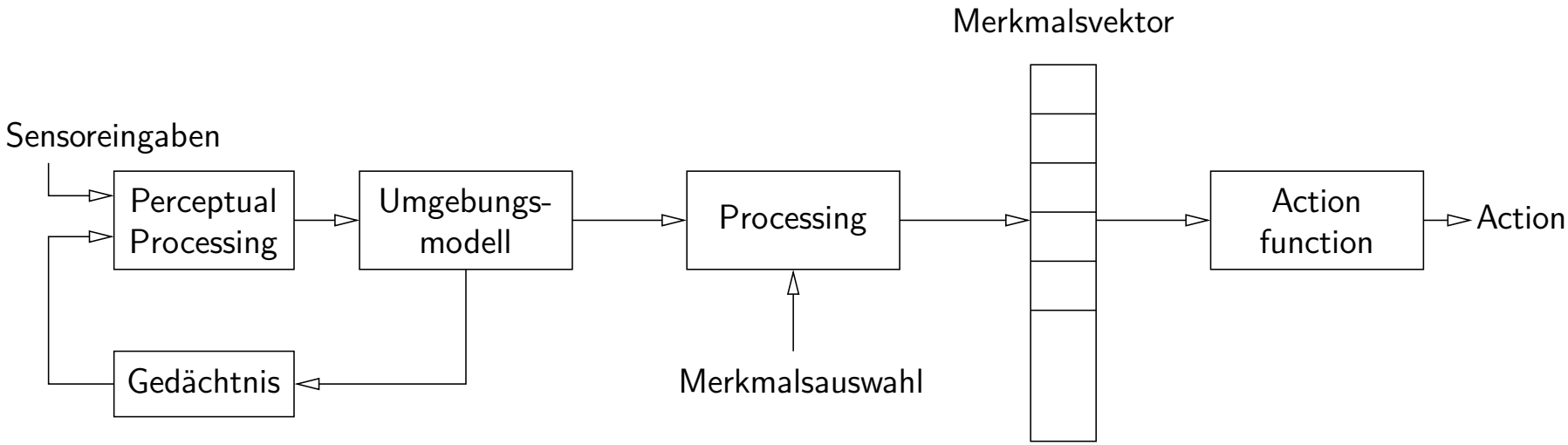


Nutzung von Umgebungsmodellen

- Bisher hatte der Roboter nur Informationen über einen sehr kleinen Ausschnitt seiner Umgebung:
  - unmittelbare Nachbarschaft
  - gespeichert in Merkmalsvektor
- Idee der Umgebungsmodelle:
  - Speicherung möglichst aller bereits gesammelter Informationen über die Umgebung
  - Nutzung geeigneter Datenstrukturen wie z.B. Landkarten, etc.

Nutzung von Umgebungsmodellen

Beispiel einer Implementierung für die Gitterwelt



Nutzung von Umgebungsmodellen

Modell der Umgebung

1	1	1	1	1	?	?
1	0	0	0	0	0	?
1	0	0	0	0	0	?
1	0	0	R	0	0	?
1	0	0	0	0	0	?
1	?	?	?	?	?	?
?	?	?	?	?	?	?

- 1: belegt, 0: frei, ?: unbekannt, R: Roboter
- Mögliche Aktion basierend auf diesen Informationen: *go west (or north)* and follow wall

Nutzung von Umgebungsmodellen

Aktionen

- Aktionen können z.B. über zwei-dimensionale Potentialfelder bestimmt werden.
- Potentialfelder werden aus der Überlagerung von anziehenden ("attractive") und abstoßenden ("repulsive") Komponenten gebildet.
- Die Bewegung des Roboters erfolgt in absteigende Richtung des Gradienten (lokale Minima!).
- Die Bewegungsrichtung kann vorberechnet oder online (in z.B. sich ändernden Umgebungen) bestimmt werden.

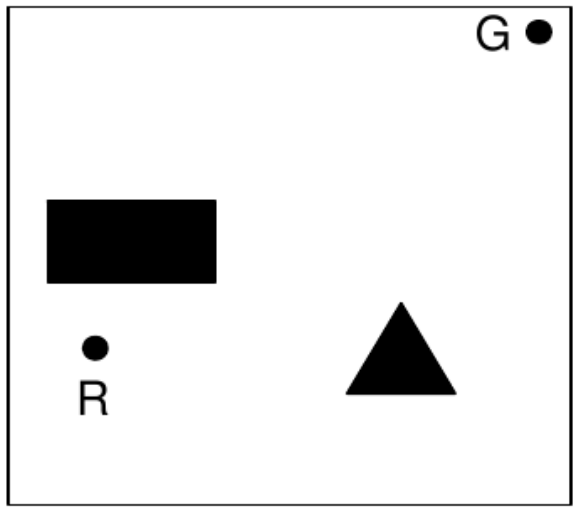
Nutzung von Umgebungsmodellen

Potentialfeld für das Gridworld-Beispiel

- Anziehende Komponente
  - Durch das Zielfeld erzeugt: $p_a(x^{(\rho)}) = k_1 \cdot d(x^{(\rho)})^2$
  - wobei k_1 ein konstanter Faktor und d der Abstand zum Zielfeld sind.
- Abstoßende Komponente(n):
  - Durch Hindernisse erzeugt: $p_r(x^{(\rho)}) = \frac{k_2}{d(x^{(\rho)})^2}$
  - wobei k_2 ein konstanter Faktor und d der Abstand zum Hindernis sind.
- Insgesamt: $p = p_a + p_r$

Potentialfelder

Beispiel der Umgebung eines Roboters



Roboter (R)

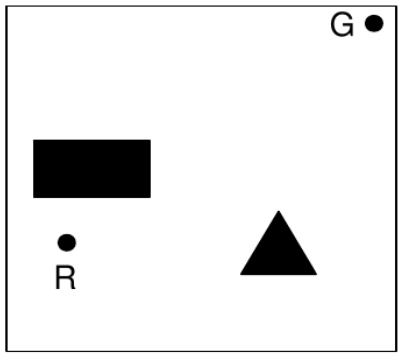
Ziel (G)

Hindernisse (♦)

Potentialfelder

Potentialfeldkomponenten

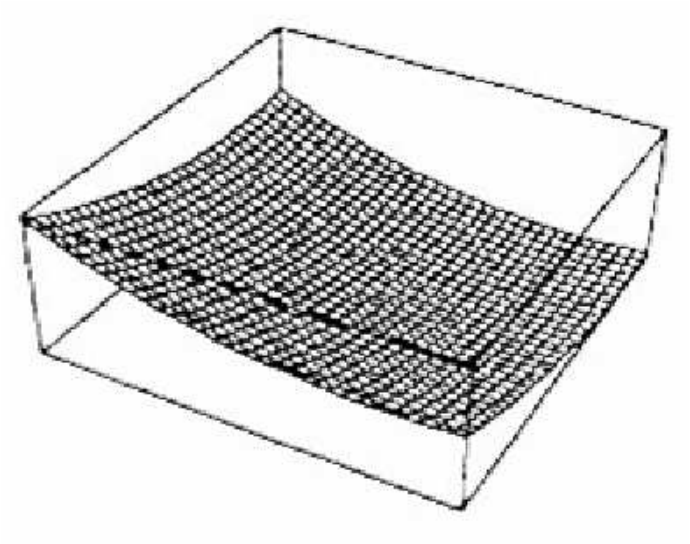
- Ziel (links)
- Hindernisse (rechts)

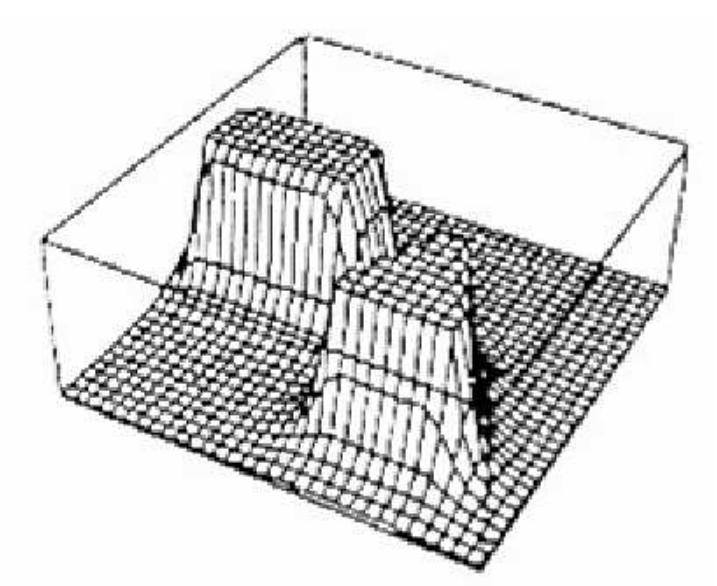


Roboter (R)

Ziel (G)

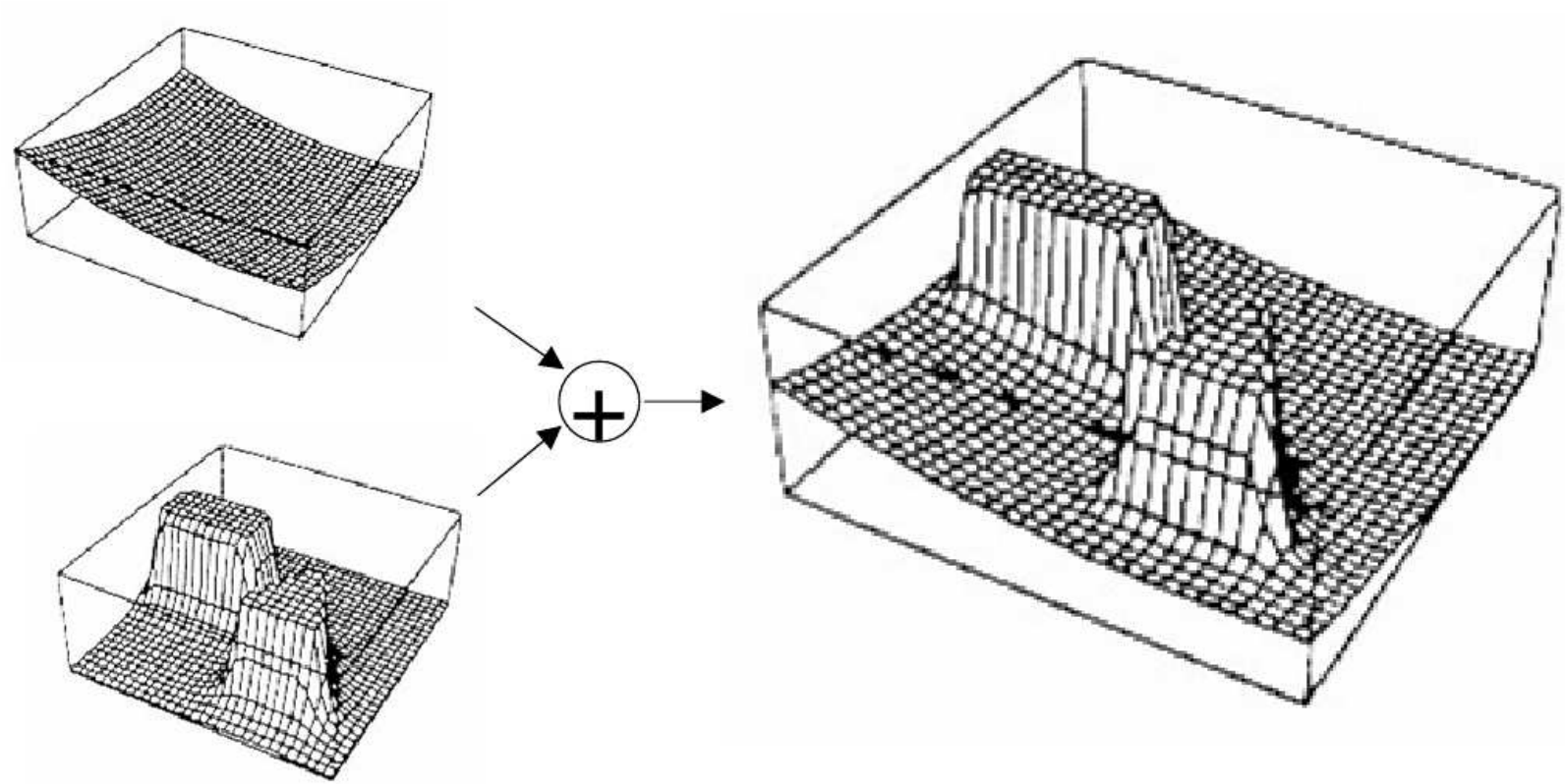
Hindernisse (♦)





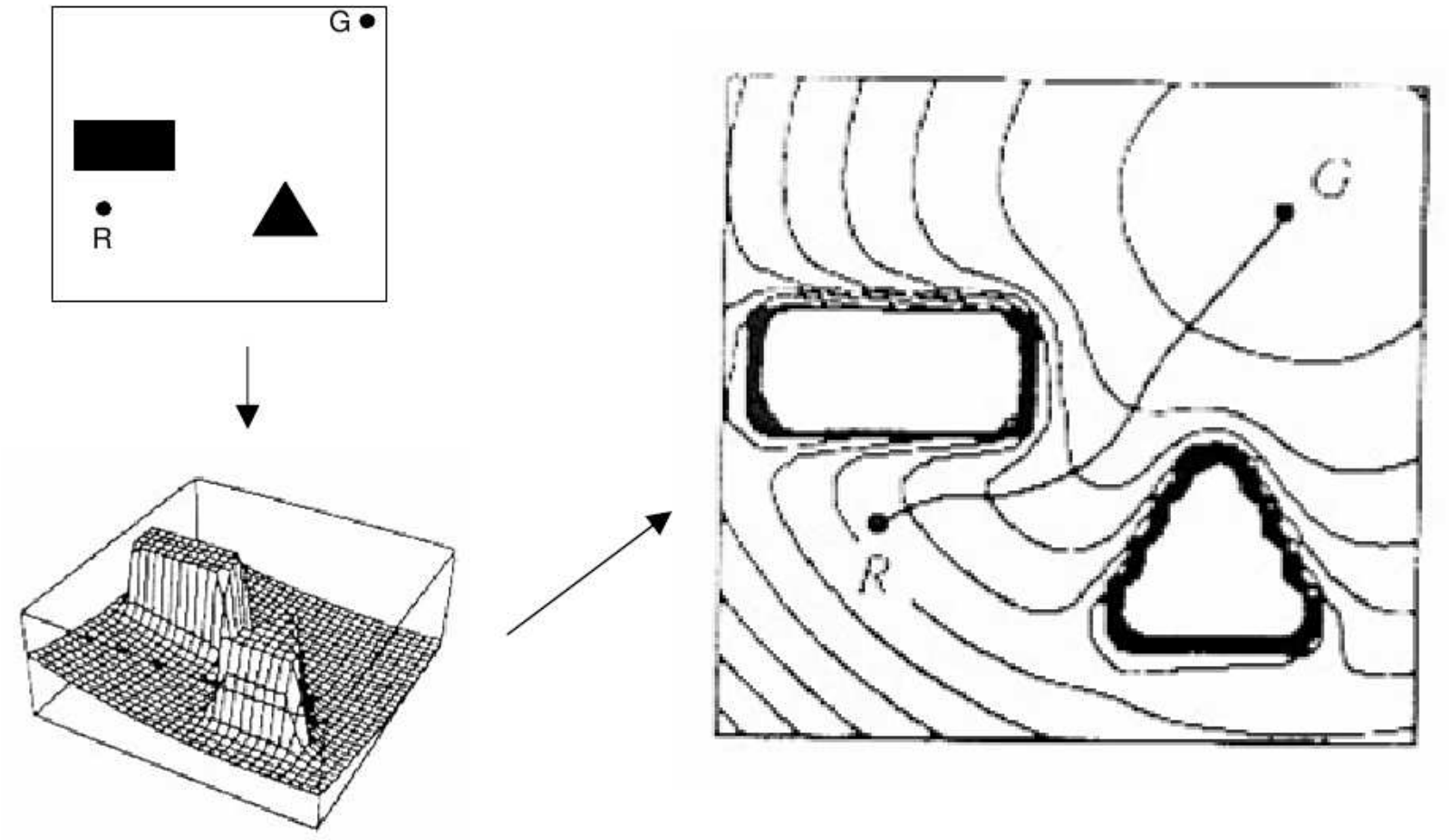
Potentialfelder

Gesamtes Potentialfeld



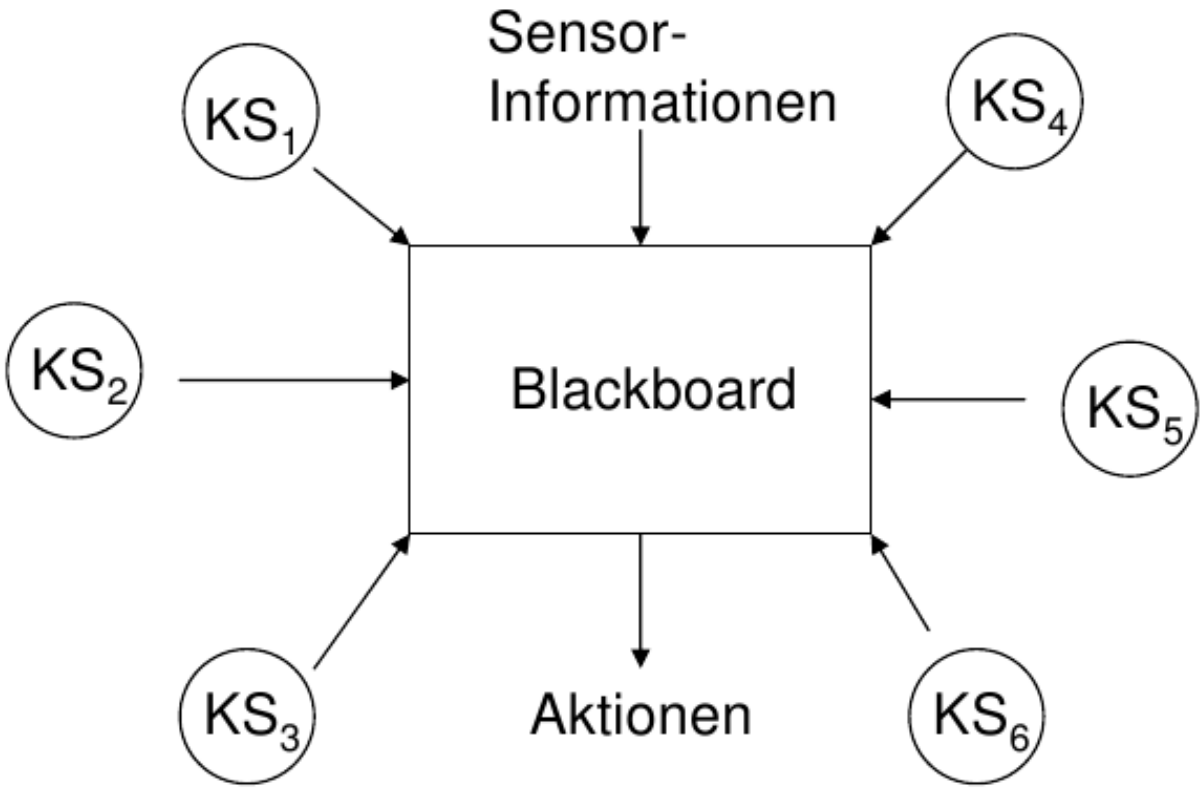
Potentialfelder

Äquipotentiallinien (für Gradientenverfahren)



Blackboard-Systeme

- Blackboard: Spezielle Datenstruktur
- Knowledge Source (KS): Programm, das das Blackboard lesen und verändern kann



Blackboard-Systeme

- Jede Knowledge Source hat
  - einen Bedingungsteil (berechnet Wert eines Merkmals) und
  - einen Aktionsteil (Programm, das die Datenstruktur ändert und/oder externe Aktionen durchführt)
- Wenn zwei Knowledge-Sourcen ausgeführt werden können, entscheidet ein *Konfliktlöser*, welche KS gewählt wird.
- Jede Knowledge Source dient als "Experte" des Blackboardteils, den sie überwacht.

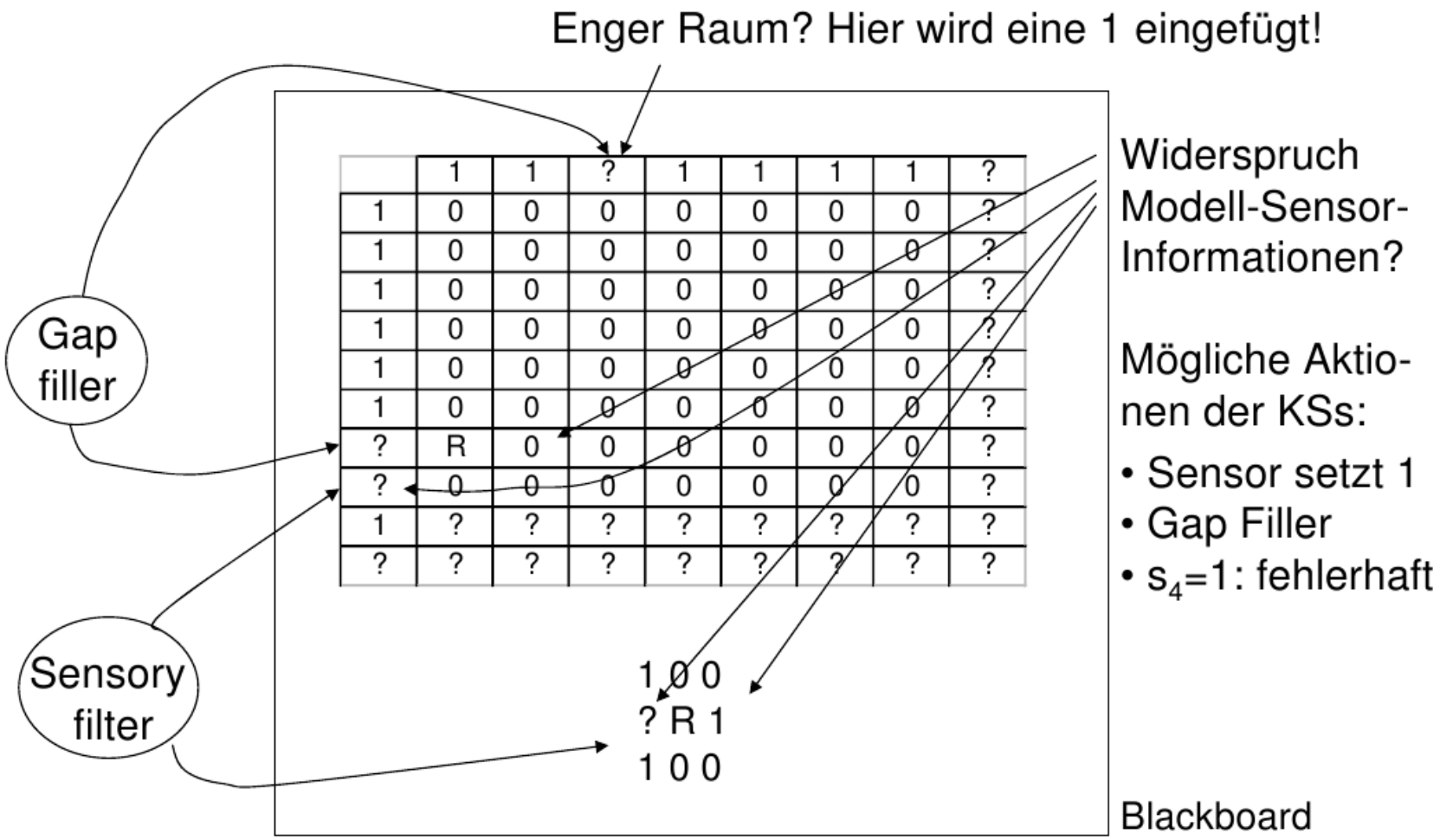
Blackboard-Systeme

Beispiel (1)

- Das Weltmodell im Roboter kann unvollständig oder falsch sein (wegen Sensorfehlern).
- Mögliche Knowledge Sourcen
  - Lückenfüller (Gap-Filler): Sucht nach engen Räumen (tight spaces) im gelernten Umgebungsmodell und markiert das Feld bzw. korrigiert ggf. vorhandene Fehler.
  - Sensorfilter (Sensory-Filter): Vergleicht Sensorinformationen mit dem gelernten Umgebungsmodell (Karte) und versucht Fehler zu beseitigen.

Blackboard-Systeme

Beispiel (2)



	1	1	?	1	1	1	1	?
1	0	0	0	0	0	0	0	?
1	0	0	0	0	0	0	0	?
1	0	0	0	0	0	0	0	?
1	0	0	0	0	0	0	0	?
1	0	0	0	0	0	0	0	?
1	0	0	0	0	0	0	0	?
?	R	0	0	0	0	0	0	?
?	0	0	0	0	0	0	0	?
1	?	?	?	?	?	?	?	?
?	?	?	?	?	?	?	?	?

Literatur I

Radetzky, A., Bartsch, W., Grospietsch, G., and Pretschner, D. P. (1999).
susilap-g: Ein Operationssimulator zum Training minimal-invasiver Eingriffe in der Gynäkologie.
Zentralblatt für Gynäkologie, 121(2):110–116.