

Hidden Markov Modelle

Prof. Dr.-Ing. Rüdiger Dillmann

Prof. Dr.-Ing. J. Marius Zöllner



Forschungszentrum Karlsruhe
in der Helmholtz-Gemeinschaft



Universität Karlsruhe (TH)
Forschungsuniversität • gegründet 1825

- Motivation
- Diskreter Markov Prozess
- Hidden Markov Modelle
- Die 3 grundlegenden Probleme
- Lösungen der Probleme
- Anwendungsbeispiele
z.B. Gestenerkennung, Ampelerkennung

Signale in der realen Welt sind

- verrauscht (erfassbar)
- besitzen nicht-deterministische Eigenschaften

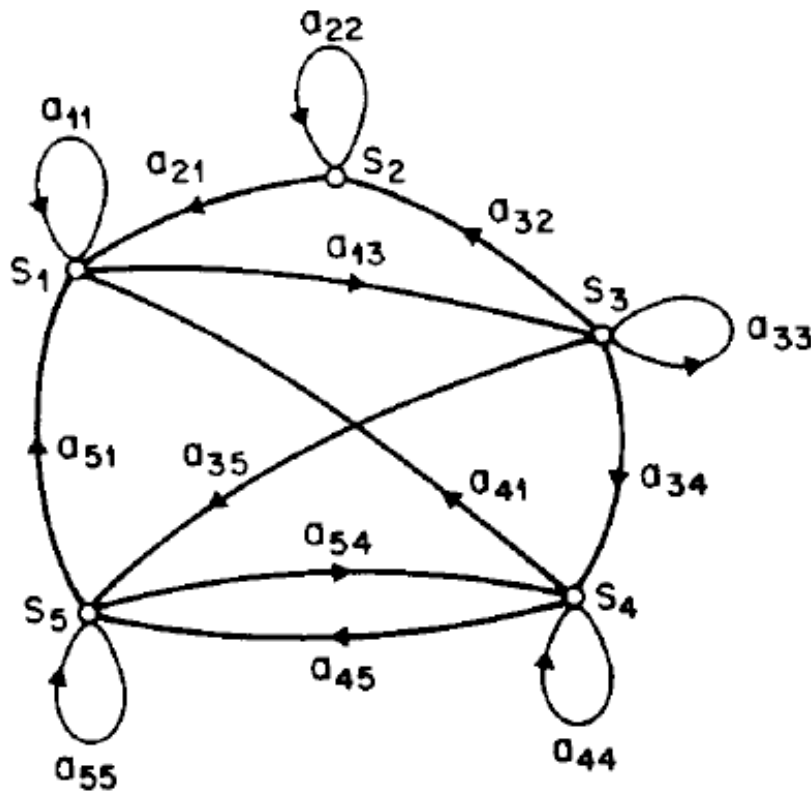
Stochastische Modelle von Signalen

- z.B. Markov Prozess

Nutzen von stochastischen Signalmodellen

- Erkennung der Signale (Sprache, Gesten, ...)
- Theoretische Beschreibung von signalverarbeitenden Systemen
- Simulationen
- Arbeiten in der Praxis oft außerordentlich gut

Diskreter Markov Prozess



N diskrete Zustände:

$$S = \{S_1, S_2, \dots, S_N\}$$

Zeitpunkte der Zustandsübergänge:

$$t = 1, 2, \dots$$

Aktueller Zustand zur Zeit t:

$$q_t$$

Beispiel: Wetter

Jeden Mittag wird das Wetter beobachtet:

Zustand 1: Regen (oder Schnee)

Zustand 2: bewölkt

Zustand 3: sonnig

Übergangswahrscheinlichkeiten:

$$A = \{a_{ij}\} = \begin{bmatrix} 0.4 & 0.3 & 0.3 \\ 0.2 & 0.6 & 0.2 \\ 0.1 & 0.1 & 0.8 \end{bmatrix}$$

Frage: Wenn es heute sonnig ist, wie wahrscheinlich ist es, dass das Wetter der nächsten 7 Tage {sonnig, sonnig, Regen, Regen, sonnig, bewölkt, sonnig} ist?

Beispiel: Wetter

Beobachtung: $O = S_3, S_3, S_3, S_1, S_1, S_3, S_2, S_3$

Lösung:

$$\begin{aligned} P(O|\text{Modell}) &= P(S_3, S_3, S_3, S_1, S_1, S_3, S_2, S_3|\text{Modell}) \\ &= P(S_3) \cdot P(S_3|S_3) \cdot P(S_3|S_3) \cdot P(S_1|S_3) \\ &\quad \cdot P(S_1|S_1) \cdot P(S_3|S_1) \cdot P(S_2|S_3) \cdot P(S_3|S_2) \\ &= \pi_3 \cdot a_{33} \cdot a_{33} \cdot a_{31} \cdot a_{11} \cdot a_{13} \cdot a_{32} \cdot a_{23} \\ &= 1 \cdot 0.8 \cdot 0.8 \cdot 0.1 \cdot 0.4 \cdot 0.3 \cdot 0.1 \cdot 0.2 \\ &= 1.536 \times 10^{-4} \end{aligned}$$

wobei

$$\pi_i = P(q_1 = S_i), \quad 1 \leq i \leq N$$

Beschränkter Horizont:

$$\begin{aligned} P(q_{t+1} = S_j | q_t = S_i, q_{t-1} = S_k, \dots) \\ = P(q_{t+1} = S_j | q_t = S_i) \end{aligned}$$

Die Wahrscheinlichkeit einen Zustand zu erreichen ist nur von seinem direkten Vorgängerzustand abhängig.

Zeitinvarianz:

Die Wahrscheinlichkeit eines Zustandsübergangs ist unabhängig von der Zeit.

Hidden Markov Modelle (HMM)

Bisher:

- Ereignisse (Zustände) direkt beobachtbar

HMM:

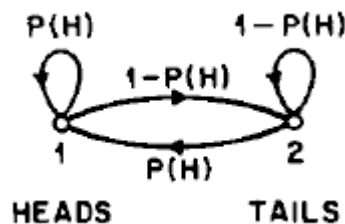
- Beobachtung ist eine stochastische Funktion des Zustands
→ Zustände können nur indirekt beobachtet werden

HMMs sind ein doppelt stochastischer Prozess:

Der zugrunde liegende stochastische Prozess kann nur indirekt durch eine andere Menge von stochastischen Prozessen beobachtet werden, die eine Beobachtungssequenz produziert.

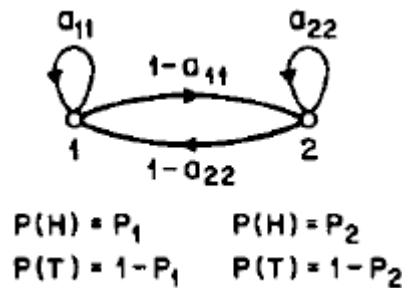
Beispiel: Münze werfen

- Jemand wirft hinter einem Vorhang eine oder mehrere Münze(n)
- Er sagt nicht, was er genau tut, sondern teilt nur das Ergebnis mit (Kopf oder Zahl)
- Beobachtung: $O = O_1 O_2 O_3 \dots O_T$
 $= HHT \dots H$



$O = HHTTHTHTTTH\dots$
 $S = 11221211221\dots$

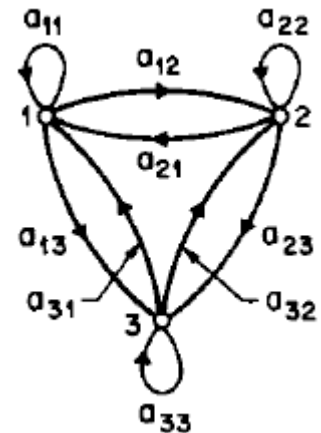
eine unausgewogene Münze



$O = HHTTHTHTTTH\dots$
 $S = 21122212212\dots$

zwei unausgewogene Münzen

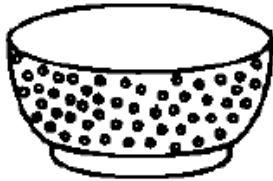
	STATE		
	1	2	3
P(H)	P_1	P_2	P_3
P(T)	$1-P_1$	$1-P_2$	$1-P_3$



$O = HHTTHTHTTTH\dots$
 $S = 31233112313\dots$

drei unausgewogene Münzen

Beispiel: Ziehen aus Urnen



URN 1

$$P(\text{RED}) = b_1(1)$$

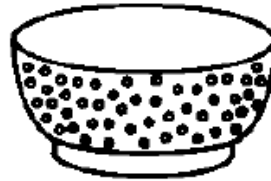
$$P(\text{BLUE}) = b_1(2)$$

$$P(\text{GREEN}) = b_1(3)$$

$$P(\text{YELLOW}) = b_1(4)$$

\vdots

$$P(\text{ORANGE}) = b_1(M)$$



URN 2

$$P(\text{RED}) = b_2(1)$$

$$P(\text{BLUE}) = b_2(2)$$

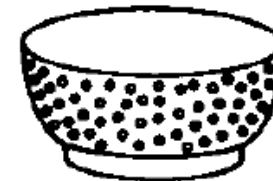
$$P(\text{GREEN}) = b_2(3)$$

$$P(\text{YELLOW}) = b_2(4)$$

\vdots

$$P(\text{ORANGE}) = b_2(M)$$

...



URN N

$$P(\text{RED}) = b_N(1)$$

$$P(\text{BLUE}) = b_N(2)$$

$$P(\text{GREEN}) = b_N(3)$$

$$P(\text{YELLOW}) = b_N(4)$$

\vdots

$$P(\text{ORANGE}) = b_N(M)$$

$O = \{\text{GREEN, GREEN, BLUE, RED, YELLOW, RED, , BLUE}\}$

Definition: Hidden Markov Model (HMM)

Ein HMM ist ein Fünf-Tupel $\lambda = \{S, V, A, B, \Pi\}$:

S Menge der Zustände $S = \{S_1, S_2, \dots, S_N\}$

q_t Zustand zur Zeit t

V Menge der Ausgabezeichen $V = \{v_1, \dots, v_M\}$

A Matrix der Übergangswahrscheinlichkeiten $A = (a_{ij})$
 a_{ij} ist die Wahrscheinlichkeit, dass S_j nach S_i kommt

B Menge der Emissionswahrscheinlichkeiten
 $b_i(k)$ ist die Wahrscheinlichkeit, v_k im Zustand S_i zu beobachten

Π Die Verteilung der Anfangswahrscheinlichkeiten
 π_i ist die Wahrscheinlichkeit, dass S_i der erste Zustand ist

Die 3 grundlegenden Probleme

Problem 1

- Gegeben: Modell $\lambda = \{S, V, A, B, \Pi\}$
- Gesucht: Wahrscheinlichkeit $P(O|\lambda)$
d.h. für die Ausgabe $O = O_1 O_2 \dots O_T$

Problem 2

- Gegeben: Ausgabesequenz O und Modell λ
- Gesucht: wahrscheinlichste Zustandsfolge Q , die O erklärt

Problem 3

- Gegeben: Ausgabesequenz O und Suchraum für Modelle
- Gesucht: Anpassung der Parameter $\lambda = \{S, V, A, B, \Pi\}$
s.d. O besser erklärt wird

Bedeutung dieser Probleme

Problem 1: Evaluationsproblem

- Wie gut erklärt ein Modell eine Beobachtungssequenz?

Problem 2: Dekodierungsproblem

- Finden der „korrekten“ Zustandssequenz (verborgen)
→ Optimalitätskriterium?

Problem 3: Lern- oder Optimierungsproblem

- Optimierung der Modellparameter (Training)

Beispiel: Worterkenner

- Aufbau von HMMs für jedes zu erkennende Wort (P3)
- Verstehen der aufgebauten Modelle (P2) und damit sinnvolle Verbesserungen (kann auch für Erkennung genutzt werden)
- Erkennen unbekannter Wörtern durch Finden des besten Modells dafür (P1)

Die 3 grundlegenden Probleme

Problem 1

- Gegeben: Modell $\lambda = \{S, V, A, B, \Pi\}$
- Gesucht: Wahrscheinlichkeit $P(O|\lambda)$
d.h. für die Ausgabe $O = O_1 O_2 \dots O_T$

Problem 2

- Gegeben: Ausgabesequenz O und Modell λ
- Gesucht: wahrscheinlichste Zustandsfolge Q , die O erklärt

Problem 3

- Gegeben: Ausgabesequenz O und Suchraum für Modelle
- Gesucht: Anpassung der Parameter $\lambda = \{S, V, A, B, \Pi\}$
s.d. O besser erklärt wird

P1: Naiver Ansatz

- Gegeben eine feste Zustandsfolge q_1, q_2, \dots, q_T gilt

$$\begin{aligned}
 P(O|Q, \lambda) &= \prod_{t=1}^T P(O_t|q_t, \lambda) \\
 &= b_{q_1}(O_1)b_{q_2}(O_2) \cdots b_{q_T}(O_T)
 \end{aligned}$$

- Wahrscheinlichkeit einer Folge: $q_1 \rightarrow q_2 \rightarrow \dots \rightarrow q_T$

$$P(Q|\lambda) = \pi_{q_1} a_{q_1 q_2} a_{q_2 q_3} \cdots a_{q_{T-1} q_T}$$

- Lösung durch Produktregel

$$P(O, Q|\lambda) = P(O|Q, \lambda)P(Q|\lambda)$$

$$P(O|\lambda) = \sum_{\text{alle } Q} P(O|Q, \lambda)P(Q|\lambda)$$

$$\begin{aligned}
 &= \sum_{q_1, q_2, \dots, q_T} \pi_{q_1} b_{q_1}(O_1) a_{q_1 q_2} b_{q_2}(O_2) \\
 &\quad \cdots a_{q_{T-1} q_T} b_{q_T}(O_T)
 \end{aligned}$$

Aufwand: $O(2T \cdot N^T)$

P1: Bessere Lösung

Problem:

- Naiver Algorithmus sehr ineffizient
→ Exponentieller Aufwand

Lösung:

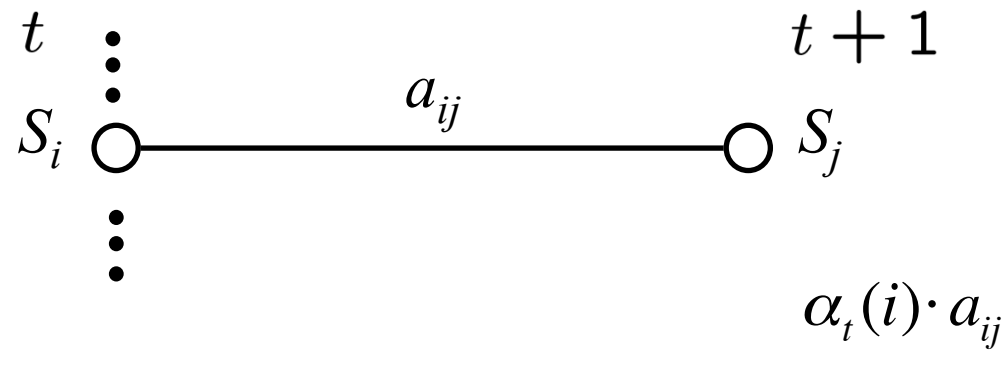
- Vorwärts- oder Rückwärtsalgorithmus
- Idee: Berechne Teilresultate (in Tabellen) → Lösung ohne nochmals alle Berechnungen zu starten
 - Definiere z.B. vorwärts $\alpha_t(i) = P(O_1 O_2 \cdots O_t, q_t = S_i | \lambda)$
(rückwärts analog s. später)
 - Vollständige Berechnung durch Induktion

P1: Vorwärts-Algorithmus - Induktion

1. Initialisierung:

$$\alpha_1(i) = \pi_i b_i(O_1), \quad 1 \leq i \leq N$$

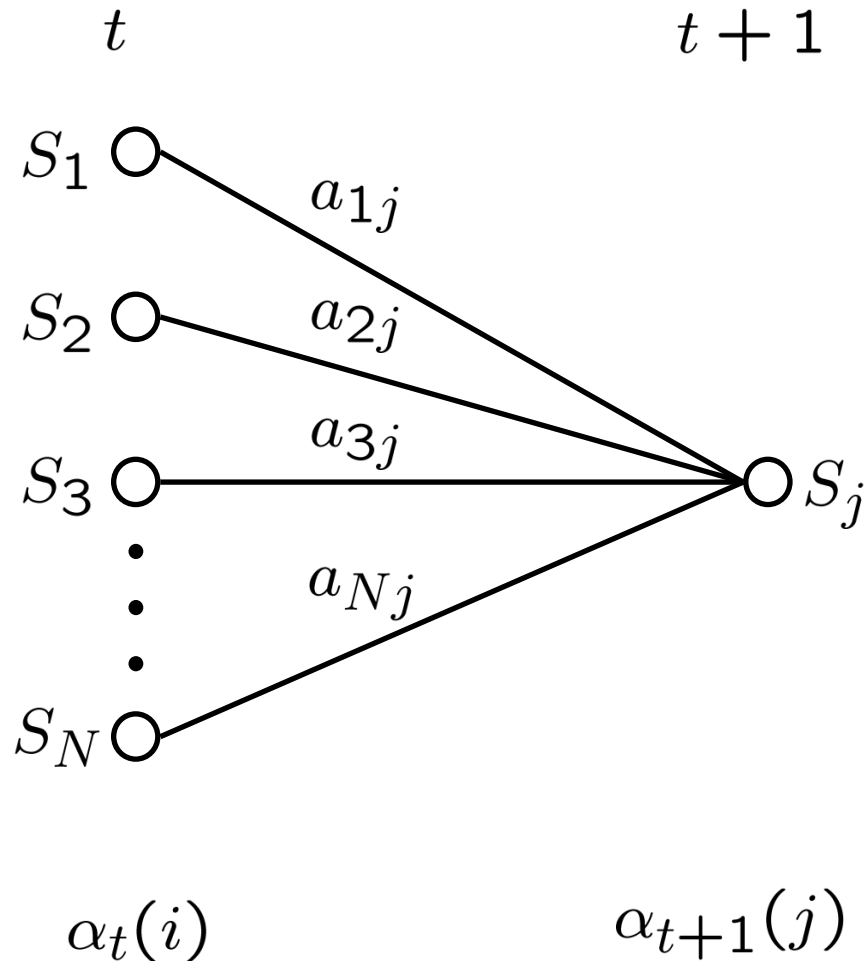
2. Induktion: $\alpha_t(i) \rightarrow \alpha_{t+1}(j)$



ist die Wahrscheinlichkeit,
dass $O_1 O_2 \dots O_t$ beobachtet wurde
und S_i = Zustand zum Zeitpunkt t

ist die Wahrscheinlichkeit,
dass $O_1 O_2 \dots O_t$ beobachtet wurde
und S_j = Zustand zum Zeitpunkt $t+1$
nach Zustand S_i zum Zeitpunkt t

P1: Vorwärts-Algorithmus - Induktion (Fortsetzung)



- Für die Berechnung von $\alpha_{t+1}(j)$ in $t+1$ müssen
 - die Pfade über alle Vorgängerzustände S_i zum Zeitpunkt t betrachtet werden und so über alle $\alpha_t(i)a_{ij}$ aufsummiert werden
 - und mit der Emissionswahrscheinlichkeit für O_{t+1} multipliziert werden

P1: Vorwärts-Algorithmus (Gesamt)

1. Initialisierung:

$$\alpha_1(i) = \pi_i b_i(O_1), \quad 1 \leq i \leq N$$

2. Induktion: $\alpha_t(i) \rightarrow \alpha_{t+1}(j)$

$$\alpha_{t+1}(j) = \left[\sum_{i=1}^N \alpha_t(i) a_{ij} \right] b_j(O_{t+1}), \quad 1 \leq t \leq T - 1$$
$$1 \leq j \leq N$$

3. Terminierung:

$$P(O|\lambda) = \sum_{i=1}^N \alpha_T(i)$$

Aufwand: $O(N^2T)$

Analog zum Vorwärts-Algorithmus:

■ Definiere $\beta_t(i) = P(O_{t+1}O_{t+2} \cdots O_T | q_t = S_i, \lambda)$

- als Wahrscheinlichkeit, dass $O_{t+1}O_{t+2} \cdots O_T$ beobachtet wird und S_i = Zustand zum Zeitpunkt t
- Wert wird später für Lernalgorithmus benötigt

■ Induktion

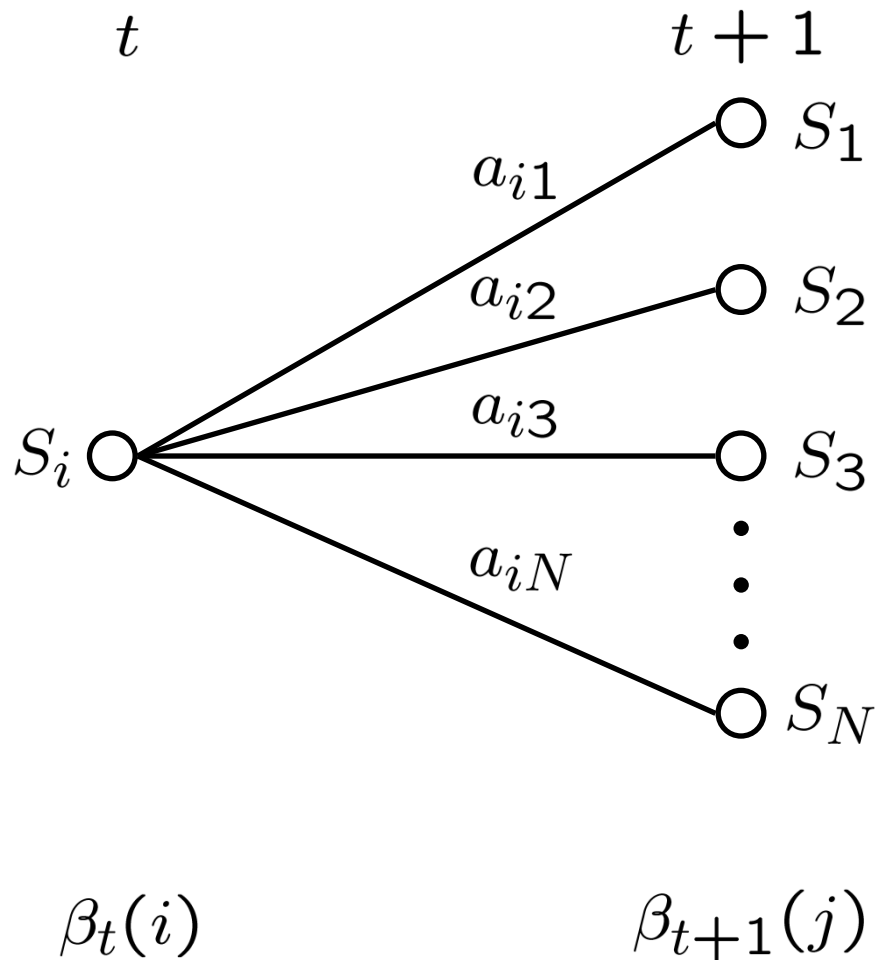
1. Initialisierung: $\beta_T(i) = 1, 1 \leq i \leq N$

2. Induktionsschritt:

$$\beta_t(i) = \sum_{j=1}^N a_{ij} b_j(O_{t+1}) \beta_{t+1}(j),$$

$$t = T - 1, T - 2, \cdots, 1, 1 \leq i \leq N$$

Rückwärts-Algorithmus – Induktionsschritt (Erklärung)



- Für die Berechnung von $\beta_t(i)$ müssen die Pfade über alle Nachfolgerzustände S_j zum Zeitpunkt $t+1$ betrachtet werden und entsprechend aufsummiert werden

Die 3 grundlegenden Probleme

Problem 1

- Gegeben: Modell $\lambda = \{S, V, A, B, \Pi\}$
- Gesucht: Wahrscheinlichkeit $P(O|\lambda)$
d.h. für die Ausgabe $O = O_1 O_2 \dots O_T$

Problem 2

- Gegeben: Ausgabesequenz O und Modell λ
- Gesucht: wahrscheinlichste Zustandsfolge Q , die O erklärt

Problem 3

- Gegeben: Ausgabesequenz O und Suchraum für Modelle
- Gesucht: Anpassung der Parameter $\lambda = \{S, V, A, B, \Pi\}$
s.d. O besser erklärt wird

P2: Optimalitätskriterium

Definition der „optimalen“ Zustandsfolge?

Ein Ansatz: Wahl der Zustände q_t , die unabhängig voneinander am wahrscheinlichsten sind

Definiere: Wahrscheinlichkeit, zum Zeitpunkt t im Zustand S_i zu sein

$$\gamma_t(i) = P(q_t = S_i | O, \lambda)$$

$$\gamma_t(i) = \frac{\alpha_t(i)\beta_t(i)}{P(O|\lambda)} = \frac{\alpha_t(i)\beta_t(i)}{\sum_{i=1}^N \alpha_t(i)\beta_t(i)}$$

Lösung: $q_t = \arg \max_{1 \leq i \leq N} \gamma_t(i)$ für alle $1 \leq t \leq T$

P2: Optimalitätskriterium

Problem: Bei **nicht vollständig vernetztem** HMM ergibt dies unter Umständen keinen gültigen Pfad!
(z.B. bestes $q_t = S_i$ und $q_{t+1} = S_j$ aber $a_{ij} = 0$)

Besser: Wahl der insgesamt besten Zustandsfolge über Maximierung von:

$$P(Q|O, \lambda)$$

→ Entspricht der Maximierung von $P(Q, O|\lambda)$
(Anwendung der Produktregel)

P2: Viterbi-Algorithmus

Idee ist ähnlich zum Vorwärts-Algorithmus:

- Vorwärtsvariable speichert maximale Wahrscheinlichkeit mit der ein Zustand erreicht wird:

$$\delta_t(i) = \max_{q_1, q_2, \dots, q_{t-1}} P(q_1 q_2 \cdots q_t = S_i, O_1 O_2 \cdots O_t | \lambda)$$

- Induktionsschritt

$$\delta_{t+1}(j) = \left[\max_i \delta_t(i) a_{ij} \right] b_j(O_{t+1})$$

Unterschied zum Vorwärts-Algorithmus:
Maximierung statt Summation

- Rückwärtsvariable speichert wahrscheinlichsten Vorgängerknoten: $\psi_t(j)$ (argmax s. nächste Folie)

P2: Viterbi-Algorithmus

1. Initialisierung:

$$\delta_1(i) = \pi_i b_i(O_1), \quad 1 \leq i \leq N$$
$$\psi_1(i) = 0$$

2. Rekursion:

$$\delta_t(j) = \max_{1 \leq i \leq N} [\delta_{t-1}(i) a_{ij}] b_j(O_t), \quad 2 \leq t \leq T, \quad 1 \leq j \leq N$$
$$\psi_t(j) = \arg \max_{1 \leq i \leq N} [\delta_{t-1}(i) a_{ij}], \quad 2 \leq t \leq T, \quad 1 \leq j \leq N$$

P2: Viterbi-Algorithmus

3. Terminierung:

Wähle Zielkonten mit maximaler Wahrscheinlichkeit

$$P^* = \max_{1 \leq i \leq N} [\delta_T(i)]$$

$$q_T^* = \arg \max_{1 \leq i \leq N} [\delta_T(i)]$$

4. Backtracking (Bestimmen) der Zustandsfolge:

$$q_t^* = \psi_{t+1}(q_{t+1}^*), \quad t = T - 1, T - 2, \dots, 1$$

Die 3 grundlegenden Probleme

Problem 1

- Gegeben: Modell $\lambda = \{S, V, A, B, \Pi\}$
- Gesucht: Wahrscheinlichkeit $P(O|\lambda)$
d.h. für die Ausgabe $O = O_1 O_2 \dots O_T$

Problem 2

- Gegeben: Ausgabesequenz O und Modell λ
- Gesucht: wahrscheinlichste Zustandsfolge Q , die O erklärt

Problem 3

- Gegeben: Ausgabesequenz O und Suchraum für Modelle
- Gesucht: Anpassung der Parameter $\lambda = \{S, V, A, B, \Pi\}$
s.d. O besser erklärt wird

P3: Lern- oder Optimierungsproblem

- Schwierigstes der drei Probleme
 - Kein analytischer Lösungsweg bekannt
 - Bei gegebener endlicher Beobachtungssequenz O gibt es keinen optimalen Weg, um die Modellparameter zu schätzen
- Lokale Maximierung von $P(O|\lambda)$ mit iterativer Prozedur, z.B. Baum-Welch-Algorithmus

P3: Baum-Welch-Algorithmus

Gegeben:

- Trainingssequenz O_{training} und
- Hypothesenraum für Modelle $\lambda = \{S, V, A, B, \Pi\}$

Gesucht:

- Modell, das die Daten am besten erklärt:

$$\bar{\lambda} = \arg \max_{\lambda} P(O|\lambda)$$

Ansatz:

- Hypothesenraum wird so gewählt, dass die Anzahl der Zustände vorgegeben wird.
- Lediglich die stochastischen Modellparameter werden angepasst:

$$\bar{\lambda} = \{\bar{A}, \bar{B}, \bar{\Pi}\}$$

P3: Baum-Welch-Algorithmus

- Beginne mit zufälligem Modell λ , berechne

$$P(O_{\text{training}}|\lambda)$$

- Bestimme die erwartete Anzahl von Zustandsübergängen (aus und zwischen Zuständen) und Symbolausgaben
- Neuschätzung der Übergangs- und Emissionswahrscheinlichkeiten: Berechnung eines neuen Modells
- Iteriere so lange bis (lokales) Maximum erreicht ist

Wiederholung Formelzeichen

Vorwärts-Algorithmus:

$$\alpha_t(i) = P(O_1 O_2 \cdots O_t, q_t = S_i | \lambda)$$

Rückwärts-Algorithmus:

$$\beta_t(i) = P(O_{t+1} O_{t+2} \cdots O_T | q_t = S_i, \lambda)$$

Wahrscheinlichkeit für einen Zustand (Optimalitätskriterium aus Problem 2):

$$\gamma_t(i) = P(q_t = S_i | O, \lambda)$$

$$\gamma_t(i) = \frac{\alpha_t(i) \beta_t(i)}{P(O | \lambda)} = \frac{\alpha_t(i) \beta_t(i)}{\sum_{i=1}^N \alpha_t(i) \beta_t(i)}$$

P3: Baum-Welch-Algorithmus

- Definiere $\xi_t(i, j)$ als die Wahrscheinlichkeit eines Zustandsübergangs von Zustand i nach j zum Zeitpunkt t , bei gegebenem O und λ :

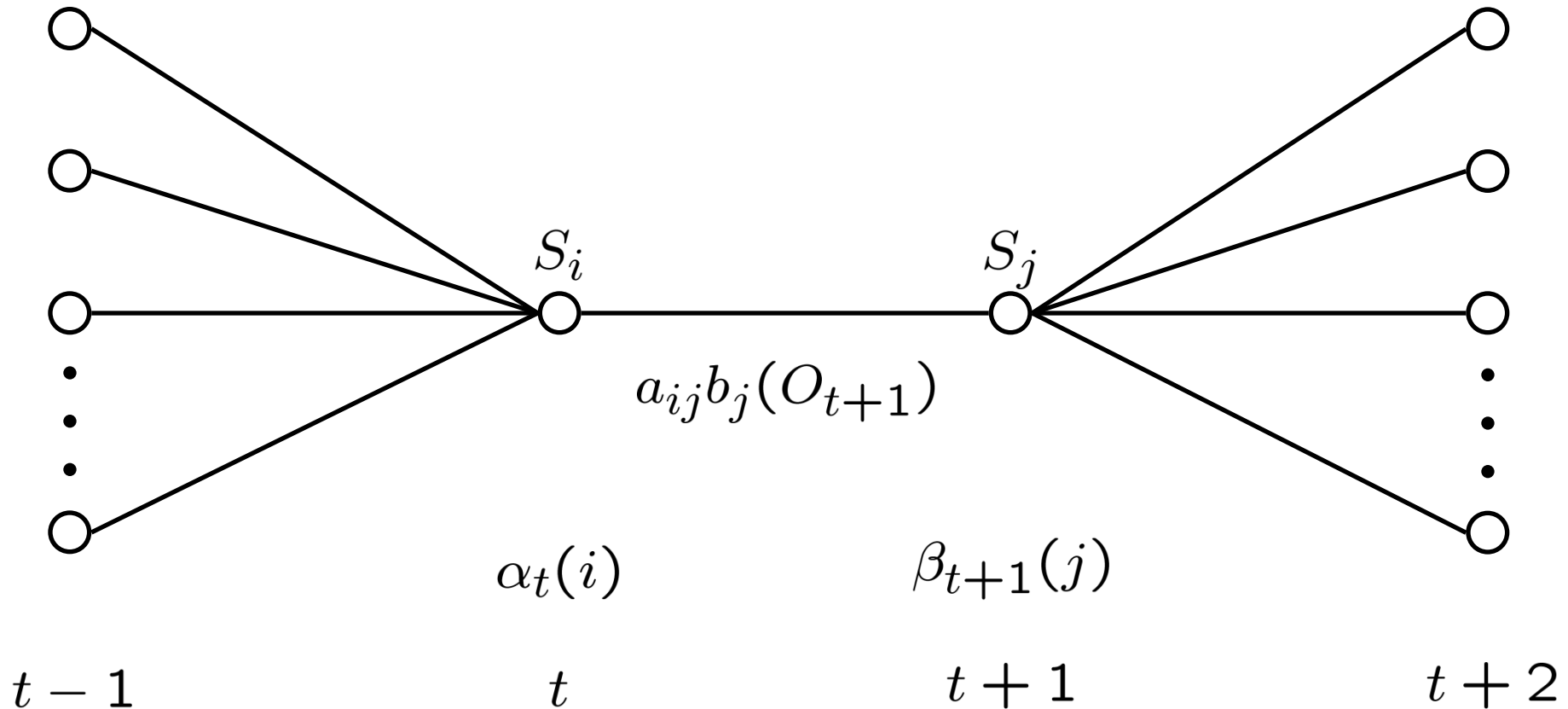
$$\xi_t(i, j) = P(q_t = S_i, q_{t+1} = S_j | O, \lambda)$$

$$\begin{aligned} \xi_t(i, j) &= \frac{\alpha_t(i) a_{ij} b_j(O_{t+1}) \beta_{t+1}(j)}{P(O | \lambda)} \\ &= \frac{\alpha_t(i) a_{ij} b_j(O_{t+1}) \beta_{t+1}(j)}{\sum_{i=1}^N \sum_{j=1}^N \alpha_t(i) a_{ij} b_j(O_{t+1}) \beta_{t+1}(j)} \end{aligned}$$

- Definition von $\gamma_t(i)$ als die Wahrscheinlichkeit, zum Zeitpunkt t im Zustand S_i zu sein, bei gegebenem O und λ :

$$\gamma_t(i) = \sum_{j=1}^N \xi_t(i, j)$$

P3: Baum-Welch-Algorithmus



P3: Baum-Welch-Algorithmus

- Z.B. Erwartungswert der Anzahl der Zustandsübergänge von Zustand S_i aus (über alle Zeitpunkte):

$$\sum_{t=1}^{T-1} \gamma_t(i)$$

- Z.B. Erwartungswert der Anzahl der Zustandsübergänge von Zustand S_i nach S_j (über alle Zeitpunkte):

$$\sum_{t=1}^{T-1} \xi_t(i, j)$$

- Algorithmus berechnet neue Parameter $\{\bar{A}, \bar{B}, \bar{\Pi}\}$ auf Basis dieser (solcher) Erwartungswerte

P3: Baum-Welch-Algorithmus

$$\bar{\pi}_i = \gamma_1(i)$$

= Erwartete Häufigkeit des Zustands S_i
zur Zeit $t = 1$

$$\bar{a}_{ij} = \frac{\sum_{t=1}^{T-1} \xi_t(i, j)}{\sum_{t=1}^{T-1} \gamma_t(i)}$$

= Erwartete # Zustandsübergänge von S_i nach S_j

Erwartete # Zustandsübergänge von S_i

$$\bar{b}_j(k) = \frac{\sum_{t=1, O_t=v_k}^{T-1} \gamma_t(j)}{\sum_{t=1}^{T-1} \gamma_t(j)} = \frac{\text{Erwartete \# (Zustand } S_j \text{ , Beobachtung } v_k \text{)}}{\text{Erwartete \# Zustand } S_j}$$

- Allgemeiner Fall für mehrere Trainingssequenzen: zähler- und nennerweise Summierung über Sequenzen, [1], S. 273

P3: Baum-Welch-Algorithmus - Bewertung

- Es gilt:

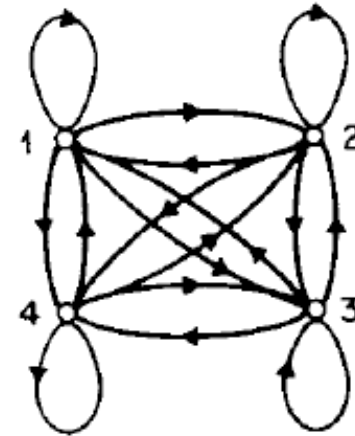
$$P(O|\bar{\lambda}) \geq P(O|\lambda)$$

- Damit verbessert sich das Modell auf der Menge der Trainingsdaten
- Realisiert EM (Expectation Maximization) Ansatz
- Training wird abgebrochen, wenn nur noch minimale Verbesserungen
- Es kann kein globales Maximum garantiert werden

Arten von Hidden Markov Modellen

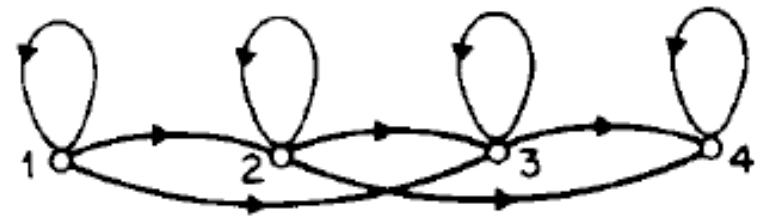
■ Ergodisches Modell

- Jeder Zustand kann von jedem anderen Zustand aus in einer endlichen Anzahl von Schritten erreicht werden

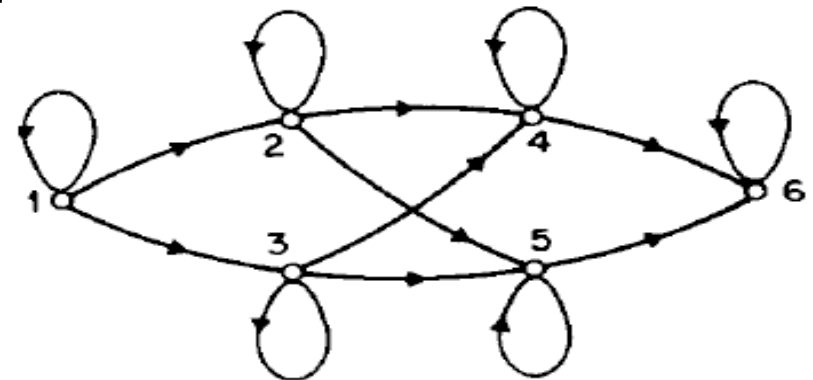


■ Links-nach-rechts-Modell (Bakis-Modell)

- Der Zustandsindex wird mit der Zeit größer oder bleibt gleich



■ Links-nach-rechts-Modell mit parallelen Pfaden



Typ der Inferenz	induktiv	↔	deduktiv
Ebenen des Lernens	symbolisch	↔	subsymbolisch
Lernvorgang	überwacht	↔	unüberwacht
Beispielgebung	inkrementell	↔	nicht inkrementell
Umfang der Beispiele	umfangreich	↔	gering
Hintergrundwissen	empirisch	↔	axiomatisch

Beispiele:

- Spracherkennung
- Gestenerkennung z.B. in Robotik
- Autonome Fahrzeuge Ampelzustandsschätzung
- Bioinformatik (Genomanalyse)

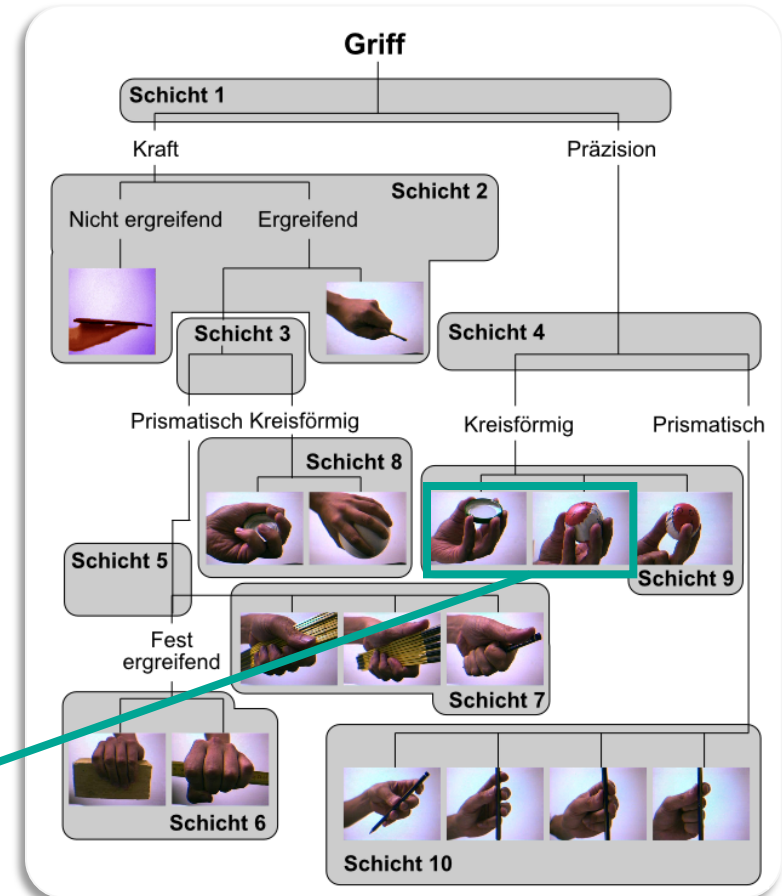
... überall dort wo man zugrunde liegende stochastische Prozesse nur indirekt (zusätzliche stochastischen Prozesse) beobachten (erfassen) kann ...

Gestenerkennung



Beispiel für Klassifikationen von Griffen und Gesten (nicht mit HMM)

- Z.B. durch Bildung hierarchischer Griffklassen [Cutkosky89] und Multi-Klassen-Klassifikation



Dynamische Gesten

Einsatz:

- Kommandierung durch Handbewegung

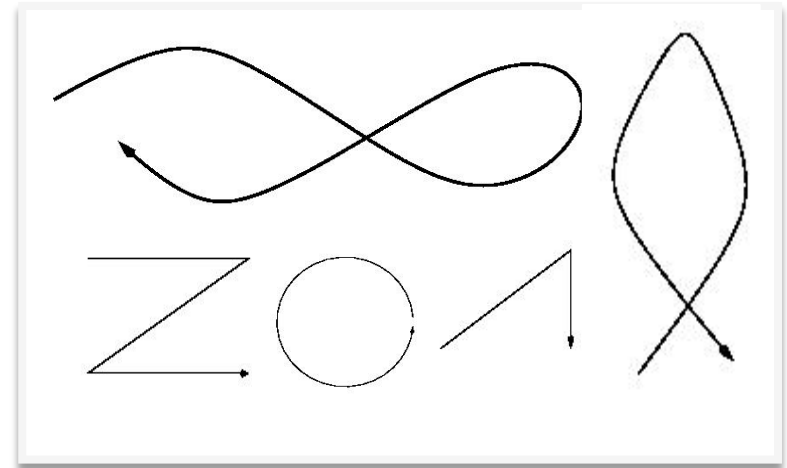
Erkennung: Bildgestützt auf Basis verschiedene geometrischer Merkmale:

- **Geradensegmente**
- Kreisbahnen
- Kurven
- Unterschiedliche Bahnlängen

Fragestellung: Welche Geste wurde beobachtet?

HMM? → Stochastische Prozesse:

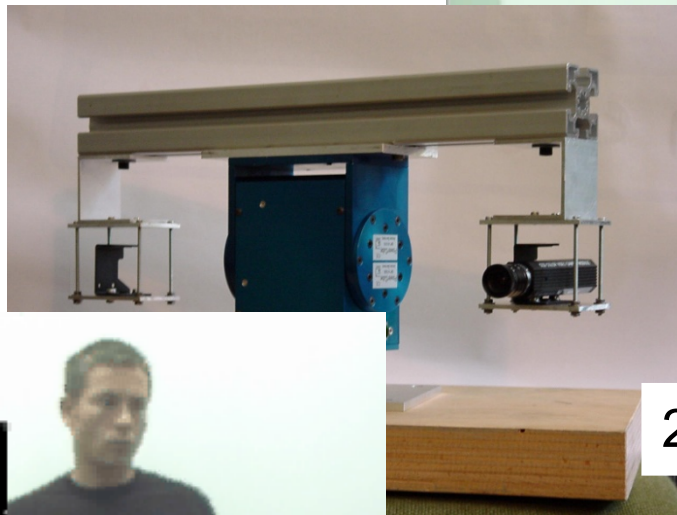
- Geste (jeweils x Zustände)
- Beobachtung (als Bewegungsmerkmale)



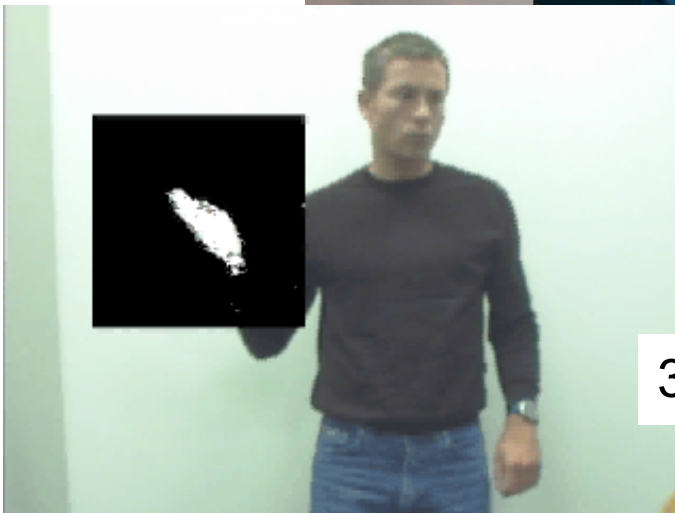
Erkennung dynamischer Gesten



1. Vorführung



2. Bildaufnahme: Kamera

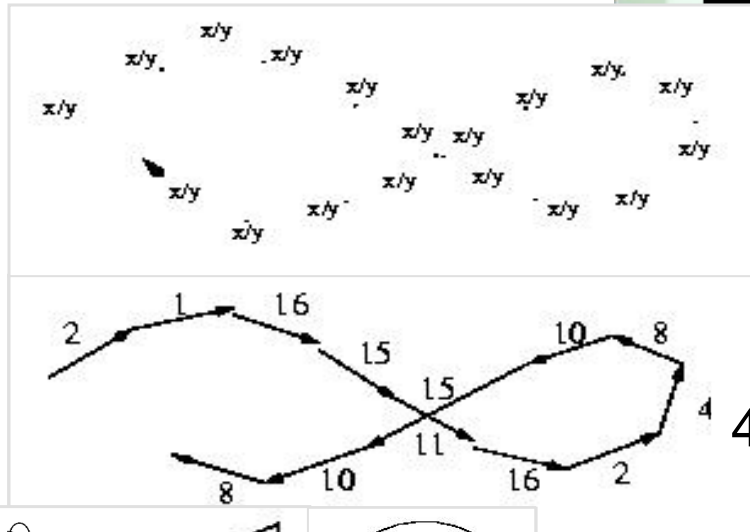


3. Bildverarbeitung: Handtracking und Aufnahme der Bewegungsbahn des Schwerpunkts der Hand

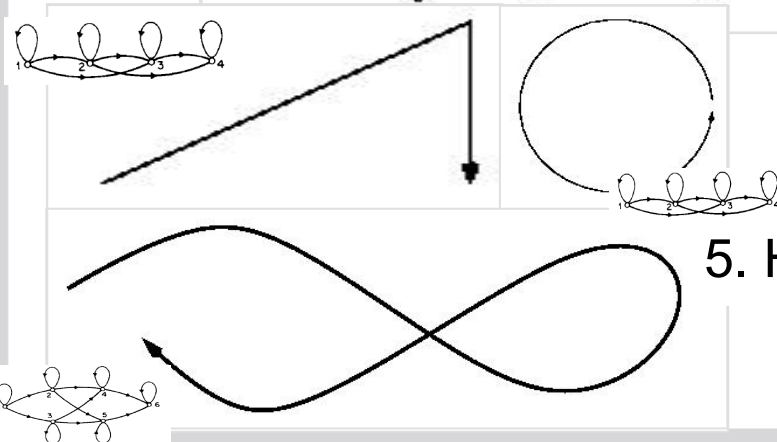
Erkennung dynamischer Gesten



3. Bildverarbeitung
→ Bahn



4. Quantisierung: Bahn reduzieren mit
Kode-Buch 16 Bewegungsvektoren
(entspricht der Beobachtung von 16
Zeichen)



5. HMM: Pro Geste trainiere ein Modell. Bestes
Modell mit Viterbi bestimmen
(Vorsicht! Sinnvolle Filterung nötig → Lit.)

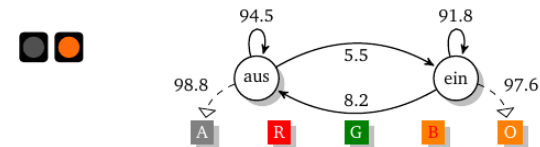
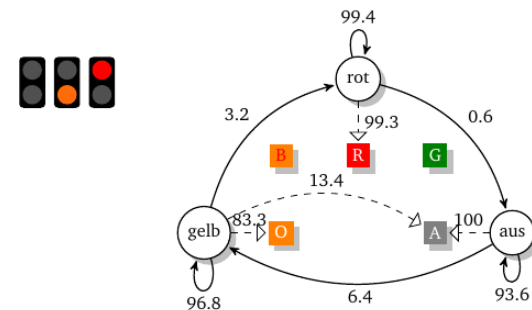
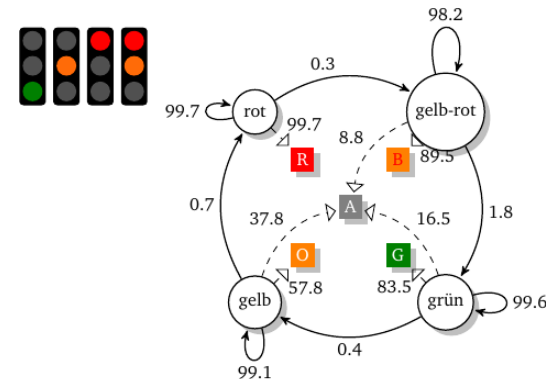
Einige Herausforderungen:

- Bildbasierte Erkennung
- Verfolgung von Ampeln auch bei Wechsel des Zustands (Farbe)
- Robuste Schätzung des Zustands
- Erkennung des Typs



Ampelerkennung (ein Ansatz)

- Bildbasierte Segmentierung
- Tracking
 - Schätzung der Ampelposition in Weltkoordinaten
 - Multi-Target-Tracker mit Kalman-Filter
- Temporale Fusion
 - Ampelfarbe als Beobachtung
 - **HMM zur Zustandsschätzung**
 - **Eliminierung von Fehlerkennungen**
 - **Unterscheidung von Ampeltypen**



■ Diskreter Markov Prozess

- Markov Bedingung

■ Hidden Markov Modelle

- Definition

■ Die 3 grundlegenden Probleme:

- Evaluationsproblem → Vorwärts-Algorithmus
- Dekodierungsproblem → Viterbi-Algorithmus
- Lern- oder Optimierungsproblem → Baum-Welch-Algorithmus

■ Anwendungen:

- Z.B: Gestenerkennung (Vorgehensweise)

- [1] *L.R. Rabiner*: **A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition**. Proceedings of the IEEE, Vol. 77, No. 2, Februar 1989.
- [2] *L.R. Rabiner, B.H. Juang*: **An Introduction to Hidden Markov Models**. IEEE ASSP Magazine, Januar 1986.
- [3] *S. Russel, P. Norvig*: **Artificial Intelligence: A Modern Approach**. Prentice Hall, 2nd Edition, 2003.

- [4] *M. Ehrenmann et al.*: **Dynamic Gestures as an Input Device for Directing a Mobile Platform.**
Proceedings of the 2001 International Conference on Robotics and Automation (ICRA), Seoul, Korea, Mai 2001.
- [5] *H.-K. Lee, J.H. Kim*: **An HMM-Based Threshold Model Approach for Gesture Recognition.** IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 21, No. 10, Oktober 1999.
- [6] D. Nienhüser: **Kontextsensitive Erkennung und Interpretation fahrrelevanter statischer Verkehrselemente**, Dissertation 2014

- [7] ***M.R. Cutkosky: On grasp choice, Grasp Models and the design of Hands for Manufacturing Tasks.*** IEEE Transactions on Robotics and Automation, Vol. 5, Issue 3, June 1989.