Modellierung Bewegter Objekte in Dynamischen Umgebungen

Daniel Göhring
08.02.2006

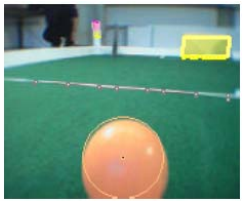
1

Motivation

- Wissen über Position von Objekten notwendig für Handlunsplanung und Kooperation

2

Die Sony-Liga

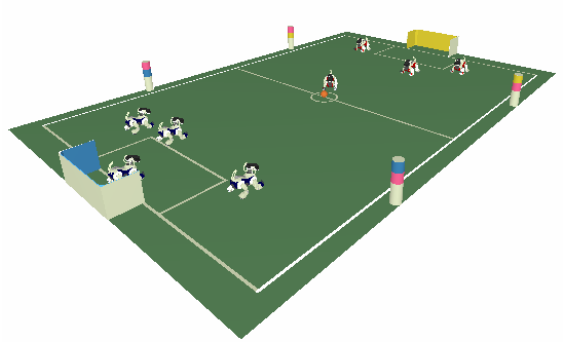
Die Roboter


Abb. 1.3: a) Roboter ERS-210 (blau) sowie ERS-210A (rot) b) ERS-7

Kategorie	ERS-210/210A	ERS-7
Kameraauflösung in Pixeln	176*144	208*160
Bildwiederholfrequenz (fps)	25	30
Kameraöffnung vertikal (Grad)	48	44
Kameraöffnung horizontal (Grad)	58	55
CPU (MHz)	200/400	576
RAM (MByte)	32	64

Tab. 1.1: Grobvergleich der Hardware von ERS-210/210A mit ERS-7

Das Spielfeld



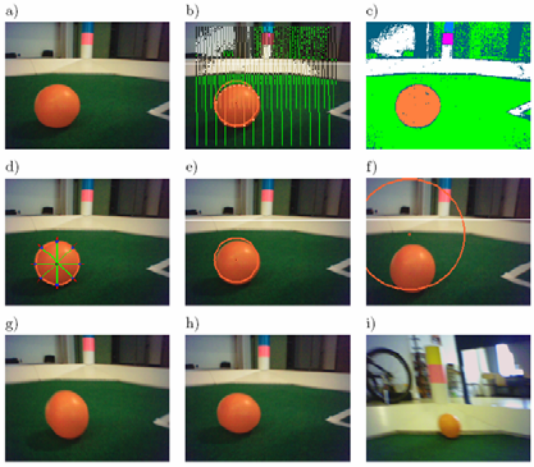
5

Beschränkungen

- Begrenzte Rechenleistung
- Nur kleiner Ausschnitt der Umwelt wahrnehmbar
- Verrauschte Sensordaten

6

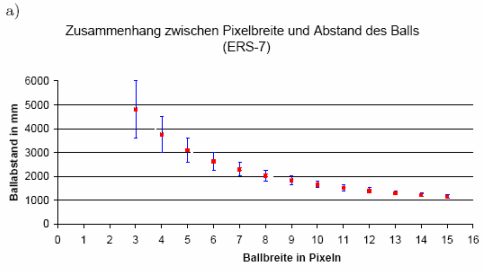
Schwierigkeiten in der Bildverarbeitung (Auszug)



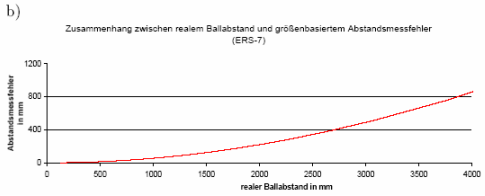
7

Fehler durch geringe Kameraauflösung

a)



b)



8

Grundlagen der Objektmodellierung

- Bayesfilter

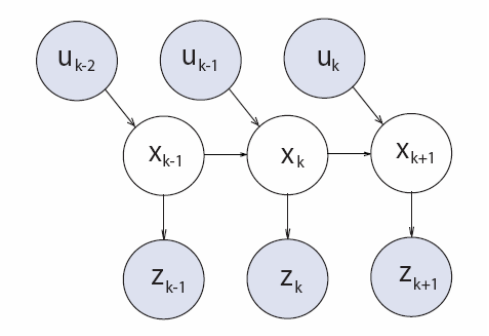
$$p(x|Z) = \frac{p(Z|x)P(x)}{p(Z)}$$

- Satz von Bayes:

- Objektzustand anhand der Sensordaten vorhersagen
  - aber nicht aller Sensordaten aus der Vergangenheit

9

Hidden Markov Modell



10

Satz von Bayes

$$Bel(x_t) = p(x_t|z_t, u_{t-1}, z_{t-1}, u_{t-2}, ..., z_1, u_0, z_0)$$

$$Bel(x_t) = \frac{p(z_t|x_t, u_{t-1}, ..., u_0, z_0)p(x_t|u_{t-1}, ..., u_0, z_0)}{p(z_t|u_{t-1}, ..., u_0, z_0)}$$

$$= \frac{p(z_t|x_t)p(x_t|u_{t-1}, ..., u_0, z_0)}{p(z_t|u_{t-1}, ..., u_0, z_0)}$$

$$= \frac{p(z_t|x_t)}{p(z_t|u_{t-1}, ..., u_0, z_0)} \int p(x_t|x_{t-1}, u_{t-1}, ..., z_0)p(x_{t-1}|u_{t-2}, ..., z_0)dx_{t-1}$$

Satz von Bayes (2)

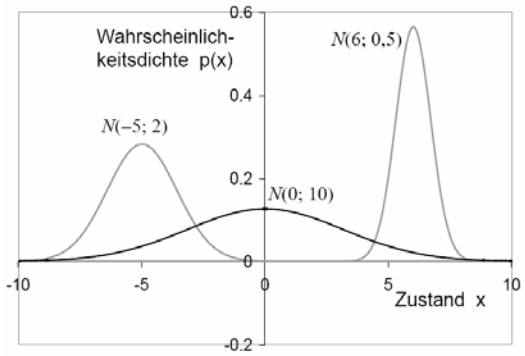
$$Bel^-(x_t) \longleftarrow \int p(x_t|x_{t-1}u_{t-1})Bel(x_{t-1})dx_{t-1}$$

$$Bel(x_t) \longleftarrow \eta p(z_t|x_t)Bel^-(x_t)$$

Resultat: Man erhält eine Berechnungsvorschrift in zwei Schritten:
Vorhersage (Prediction) und Aktualisierung (Update)

Repräsentation des Objektzustandes

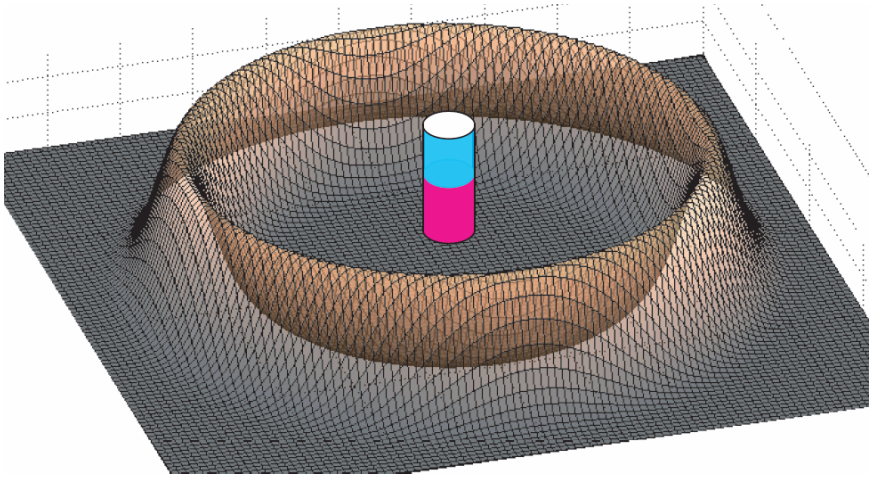
- Gaußverteilung:



$$P(X) = \frac{1}{\sqrt{(2\pi)^n * det(\Sigma)}} exp^{-\frac{1}{2}(X-\mu)^T(\Sigma)^{-1}(X-\mu)}$$

13

Veranschaulichung eines Gauß-Rotationskoerpers



14

Gaußfunktionen

- Fläche unter Funktion ist immer 1,
- Abgeschlossenheit bezüglich Addition und Multiplikation
- Eignung bei weißem Rauschen
- effizientee Berechnung und Repräsentation der
- Parameter ist die Kovarianzmatrix und der Mittelwert

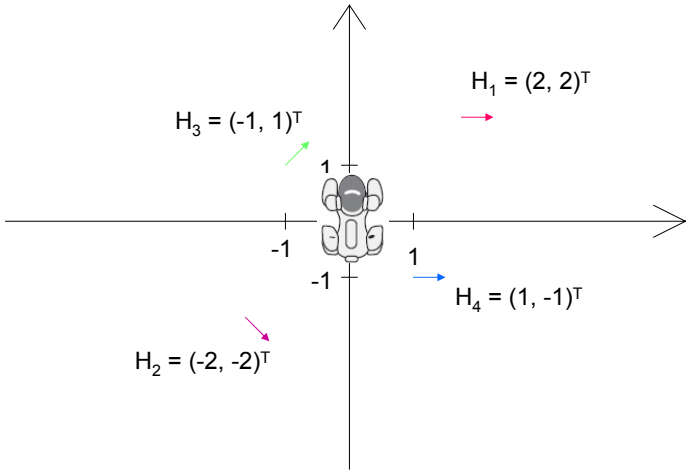
15

Kovarianzmatrix Berechnung

- 4 Hypothesen
- zwei Dimensionen: Position: x,y

16

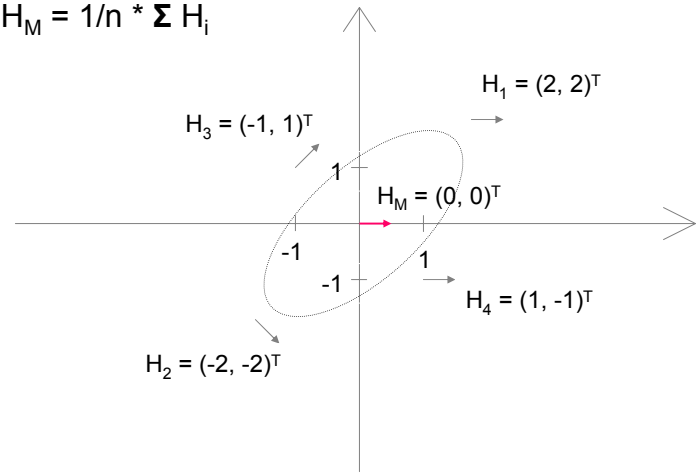
Beispielverteilung



17

Mittelwert

$H_M = 1/n * \Sigma H_i$



18

Kovarianzmatrix

$S = 1/n * \mathbf{\Sigma} (H_M - H_i) * (H_M - H_i)^T$

- Auf der Hauptdiagonale befinden sich die Varianzen von x und y, auf Nebendiagonale die Kovarianzen.

$$S = \begin{pmatrix} 2.5 & 1.5 \\ 1.5 & 2.5 \end{pmatrix}$$

Werte der Kovarianzmatrix können nicht direkt in Werte der Ellipse (Rotation, Halbachsenausdehnung) umgerechnet werden.

Kovarianzmatrix ist immer symmetrisch.

19

Zur Vollständigkeit: Korrelationsmatrix R

- jedes Element $r_{i,j}$ aus R berechnet sich bei gegebenen Elementen $c_{i,j}$ aus S als:

$r_{i,j} = c_{i,j} * c_{i,j} / (c_{i,i} * c_{j,j})$

$$= \begin{pmatrix} 1 & 0.36 \\ 0.36 & 1 \end{pmatrix}$$

20

Kalmanfilter

- dient der Filterung von Unimodalverteilungen
- Normalverteilungen vorausgesetzt
- Ideal für Zeitdiskrete, lineare Prozesse
- Minimieren den quadratischen Fehler zw. Originaldaten und Abschätzung
- Verwendung der ersten und zweiten Momente

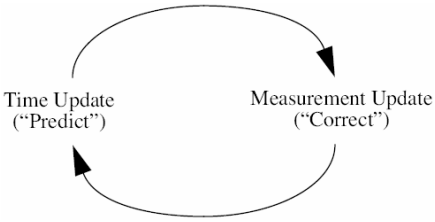
21

Berechnung

- Aus aktuellem, oder Initialzustand wird neuer Zustand vorausberechnet (prediction)
- Dieser a-priori-Zustand wird danach durch Vergleich mit den Sensordaten abgeglichen, der modellierte Objektzustand angepasst

22

Arbeitsweise Kalman



1. Schritt: Vorausberechnung der neuen Ballposition aus der alten Position (Modellierung)
2. Korrektur des berechneten Wertes durch die Messung

23

Das Kalmanmodell

1. Vorausberechnung (Prediction):

$$x_k = Ax_{k-1} + Bu_{k-1} + w_{k-1}$$

mit A gelangt man vom Zustand in k-1 zum Zustand in k
B repräsentiert die Kontrollinputmatrix

2. Messung (Measurement):

$$z_k = Hx_k + v_k$$

H beschreibt den Zusammenhang zwischen realem und gemessenem Zustand

w_k , v_k repräsentieren das Prozess- bzw. Messungsrauschen (weiß, normalverteilt)

24

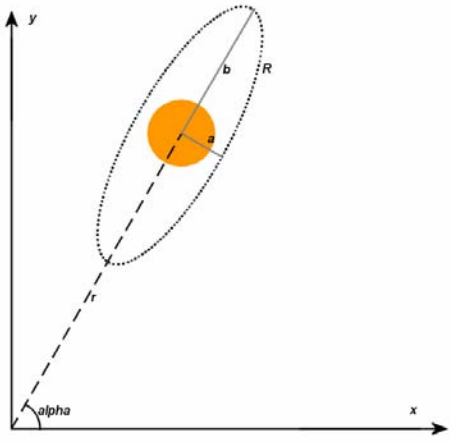
Kalmanmatrizen

- **Q** repräsentiert die Prozessfehlerkovarianzen, je kleiner die Werte von Q, je stabiler arbeitet der Filter
- **R** repräsentiert die Messfehlerkovarianzen, je kleiner die Werte von R, je reaktiver arbeitet der Filter
- P_k repräsentiert die Diskrepanz zwischen Messung und Berechnung in k, jeweils a priori und a posteriori, wird für jeden Schritt neu berechnet

Q, R müssen dem Modell mitgeteilt werden, beiben const., x_k wird Anfangs auf z_k initialisiert

25

Beispiel für Messfehlerkovarianzmatrizen **R**



- Entfernungsmessung für den Roboter schwieriger als Abschätzung des Winkels

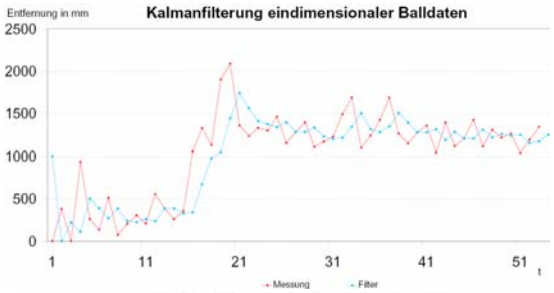
- Korrelation der einzelnen Messgrößen beachten

26

Auswirkung von Q bzw. R auf die Filterwirkung

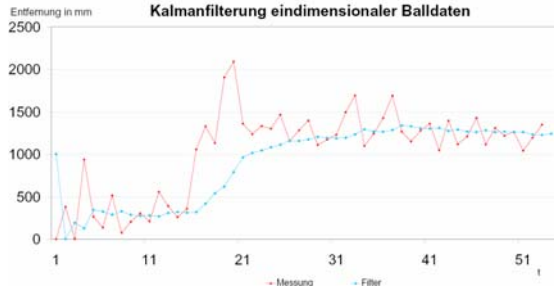
R klein

Q groß



R groß

Q klein



27

Berechnungsschleife

Prediction:

$$\bar{x}_k = Ax_{k-1} + Bu_{k-1}$$

$$\bar{P}_k = AP_{k-1}A^T + Q$$ //a-priori Fehlerkovarianzmatr.

Update:

$$K_k = \bar{P}_k H^T (H\bar{P}_k H^T + R)^{-1}$$

$$x_k = \bar{x}_k + K_k(z_k - H\bar{x}_k)$$

$$P_k = (I - K_k H)\bar{P}_k$$ //a-post. Fehlerkovarianzmatr.

K wird gemeinhin als das Kalmangain bez.

28

Vor-/Nachteile

- Messfehlerkov.-mat. R ist leicht aus Eingangsdaten zu ermitteln
- für verschiedene Ballentfernungen
- Prozessfehlerkov.-matrix hingegen schwierig
- Deckenkamera notwendig als Kontrolle von außen
- keine Repräsentation von Negativinformation möglich

29

Probleme (2)

- Messfehler nicht konstant
  - Entfernungsfehler steigt quadratisch mit wachsender Ballentfernung
  - Winkelfehler steigt bei Kopfbewegung
- also Messfehlerkovarianzmatrix **R** variabel gestalten

Wdh.: Update: $K_k = P^-_k H^T (H P^-_k H^T + \mathbf{R})^{-1}$

30

Implementationsbeispiel

- Eindimensionaler Kalmanfilter für x:

Bsp.: Für x:

$S_x^- = S_x + V_x$

$P_x^- = P_x + \mathbf{Q_x}$

$K_x = P_x^- * (P_x^- + \mathbf{R_x})^{-1}$

$S_x = S_x^- + K_x * (Z_x - S_x^-)$

$P_x = (I - K_x) * P_x^-$

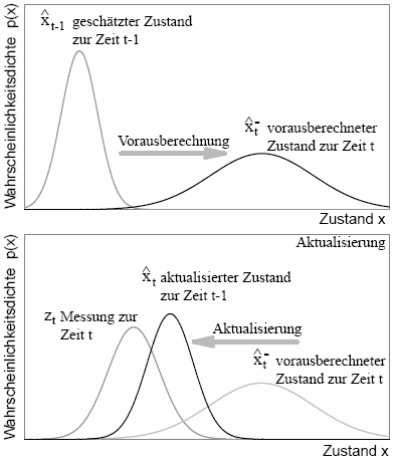
31

Vor-/Nachteile

- sehr effiziente Berechnung auch bei höherdimensionalen Räumen
- Aber nur lineare Prozesse, Erweiterungen: Extended KF (Linearisierung durch Taylorpolynome),
  Uncented Kalmanfilter, weitere statistische Betrachtungen zur Linearisierung nichtlin. Prozesse
- ein Kalmanfilter kann nur ein einzelnes Objekt modellieren (unimodalverteilungen)
- Multihypothesenmodelle, beliebige Verteilungen nur schwer oder gar nicht darstellbar

32

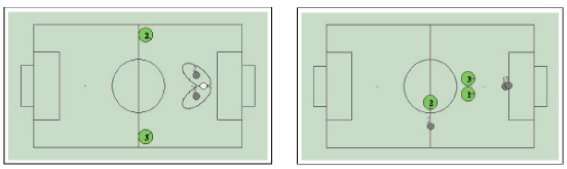
Arbeitsweise Schematisch



33

Anwendungsbeispiel Multiagentenmodellierung

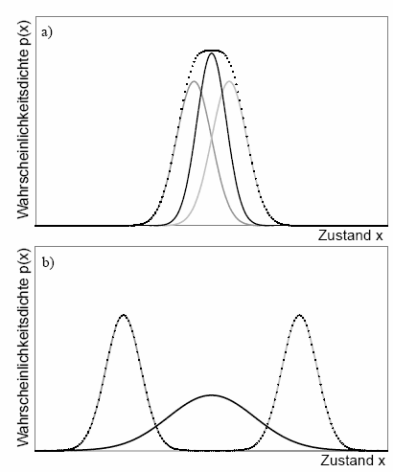
3 Agenten fusionieren ihr Wissen über die Ballposition



Quelle: Dietl, Gutmann (2002)

34

Kalmanfilter für die Fusion von Belief-Funktionen



35

Weitere Möglichkeiten zur Objektmodellierung

- Rasterbasierte Verfahren – keine Einschränkung für die Verteilungsfunktion, diskret, begrenzte Auflösung
- Multihypothesentracking – viele Kalmanfilter, jeder für eine Hypothese
- Partikelfilter: beliebige Wahrscheinlichkeitsdichtefunktionen werden durch Partikelverteilungen repräsentiert
- Kombinationen aus verschiedenen Verfahren für verschiedene Parameter des Zustandsraumes – Rao-Blackwellized Partikelfilter

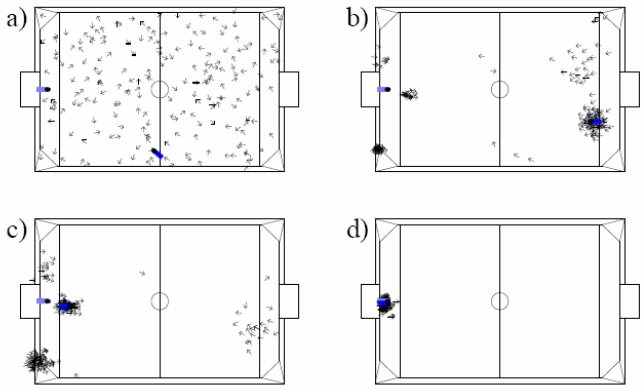
36

Monte Carlo Partikelfilter

- Jeder Partikel (entspricht Hypothese) hat Position und Wahrscheinlichkeit
- Anzahl der Partikel in einem bestimmten Gebiet gibt Auskunft über den Wert der Wahrscheinlichkeitsdichtefunktion an dieser Stelle

37

Beispiel für Konvergenz der Partikel



38

Ablauf

- Geg.: Partikelverteilung
- Propagierung der Partikel – Prediction
  - z.B. Propagierung der Geschwindigkeit und des Ortes nach Zeit t, dabei leichtes verrauschen
- Für jeden Evidence und für jeden Partikel
- Berechnung der Likelihoods für jeden Partikel
  - vergleiche für jede Hypothese, was gesehen wurde mit dem was hätte gesehen werden müssen

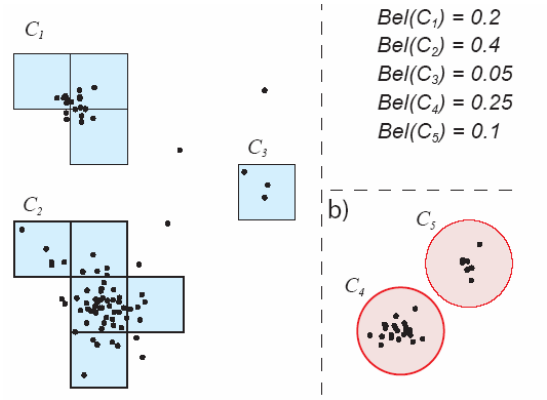
39

Ablauf (2)

- Resampling
  - Je nach Bewertung (Likelihood) werden die Partikel in den nächsten Zustand propagiert

40

Clusterung der Partikel zu Areas of Interest



41

Stand der Forschung

- Unterteilung des Zustandsraumes in Unterräume, die ggf. analytisch bzw. durch Repräsentanten lösbar sind

42

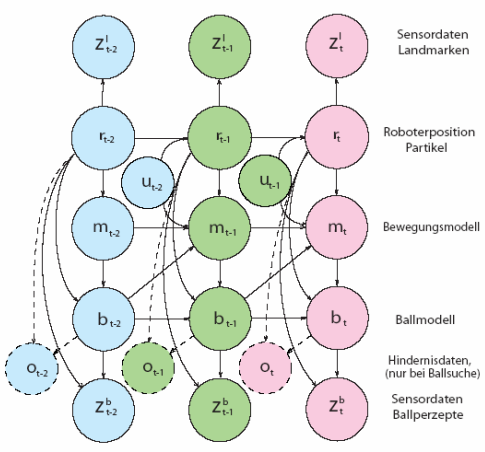
Maß für die Konvergenz

- Entropie H gibt Auskunft über die Konvergenz der Partikelverteilung
- Vorgehen, Rasterung des Spielfeldes in n Unterbereiche
- $H = \sum -p_i * \log p_i$

Wobei p_i die Anzahl der Partikel im Bereich i dividiert durch Gesamtpartikelzahl ist (relative Häufigkeit)

43

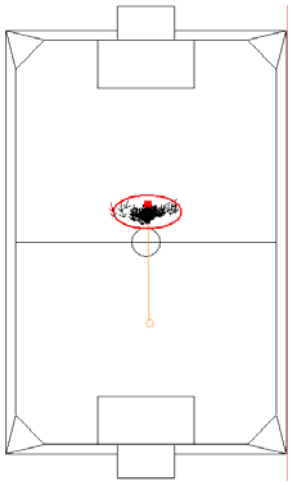
Darstellung als Bayesnetz



44

Berechnung der Hauptachsen einer statistischen Verteilung

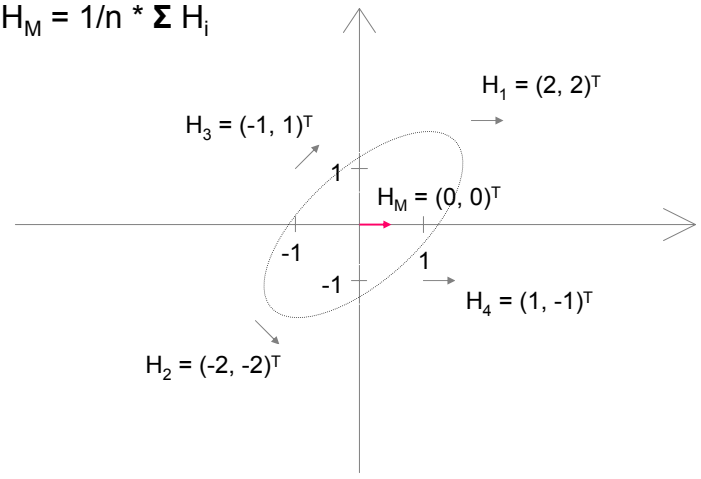
- Annahme – Normalverteilung
- Man moechte Betrag und Richtung der Hauptachsen wissen
- Ziel: Maximale Verringerung der Unsicherheit



45

1. Mittelwert

$H_M = 1/n * \Sigma H_i$



46

2. Berechnung der Eigenwerte für x und y

Ausgangsmatrix:

- $S = \begin{pmatrix} 2.5 & 1.5 \\ 1.5 & 2.5 \end{pmatrix}$

Gesucht: alle Eigenwerte λ,

wobei gilt: $\det (S - I \lambda) = 0$

47

2. Berechnung der Eigenwerte

- $\det \begin{pmatrix} 2.5 - \lambda & 1.5 \\ 1.5 & 2.5 - \lambda \end{pmatrix} = 0,$

- Also $(2.5 - \lambda)^2 - 1.5^2 = 0$

  $0 = \lambda^2 - 5\lambda + 4$

- Eigenwerte: $\lambda_1 = 2.5 + 1.5 = 4$

  $\lambda_2 = 2.5 - 1.5 = 1$

48

3. Berechnung der Eigenvektoren

- Einsetzen der Eigenwerte λ in die Kovarianzmatrix S, größte Eigenwerte zuerst.

$$(S - I\,\lambda) * \vec{V} = \vec{0}$$

49

3. Berechnung der Eigenvektoren (2)

Berechnung des 1. V, λ =4:

$$-1.5 * V_{1x} + 1.5 * V_{1y} = 0$$
$$1.5 * V_{1x} - 1.5 * V_{1y} = 0$$

$$V_{1x} = V_{1y}$$

Eigenvektor normiert (Betrag = 1): $V_1 = \begin{pmatrix} 0.707 \\ 0.707 \end{pmatrix}$

50

4. Normierung der Eigenvektoren

$$V_2 = \begin{pmatrix} -0.707 \\ 0.707 \end{pmatrix}$$

EVs sind Zeilen der Transf.-matrix

Transformationsmatrix $K = (V_1, V_2)^T$

$$K = \begin{pmatrix} 0.707 & 0.707 \\ -0.707 & 0.707 \end{pmatrix}$$

51

5. Transformation (optional)

$$S' = K * S * K^T$$

(K ist orthonormal, $K^T = K^{-1}$)

$$S' = \begin{pmatrix} 4*0.7 & 4*0.7 \\ -1*0.7 & 1*0.7 \end{pmatrix} * \begin{pmatrix} 0.7 & -0.7 \\ 0.7 & 0.7 \end{pmatrix}$$

52

Transformation (2)

- Nach der Transformation sind die Kovarianzen null

$$S' = \begin{pmatrix} 4 & 0 \\ 0 & 1 \end{pmatrix}$$

53

Zu Beachten

- Rotationsmatrix hat die Form

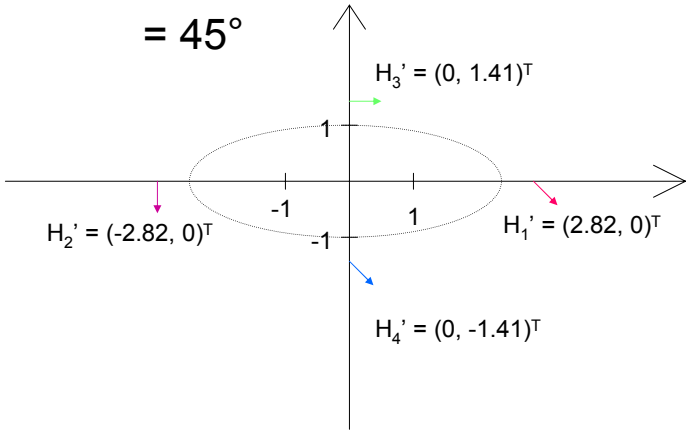
$$K = \begin{pmatrix} \cos(\alpha) & -\sin(\alpha) \\ \sin(\alpha) & \cos(\alpha) \end{pmatrix}$$

  Vorzeichen von K können zeilenweise vertauscht werden (Spiegelung der EVs)
- Ellipsenrotation von 200° = 20°, schnellere Erreichbarkeiten (Kosten) beachten

54

Nach der Transformation

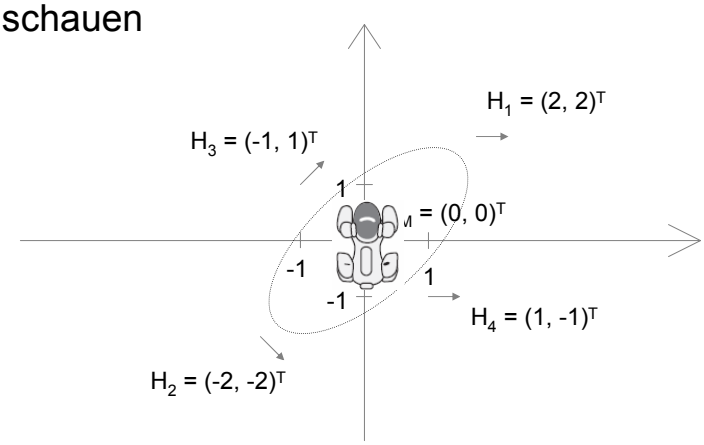
also: α = arccos(0.707) = arcsin(0.707)
= 45°



55

Also

- 45 Grad nach rechts schauen



56

Zusammenfassung

- Schwierigkeiten bei der Objektmodellierung:
- Eingangsdaten verrauscht durch begrenzte Auflösung, unzureichende Farbklassifizierung, verwackelte Kamerabilder durch Laufunruhen,
- Sichtwinkel begrenzt

57

Zusammenfassung (2)

- Objektmodellierung no
  - Weltmodell
  - Filterung verrauschter Eingangsdaten
  - Akkumulierung von Wissen aus verschiedenen Bildern (z.B. Berechnung der Ballgeschwindigkeit, Selbstlokalisierung)
  - Modellierung von Interdependenzen (Abhängigkeiten) verschiedener Objekte der Roboterumgebung

58

Zusammenfassung (3)

- Satz von Bayes Grundlage für viele wichtige Filterverfahren
- (Kalman, Monte Carlo)

- Forschung in Richtung Wissensverschmelzung durch verschiedene Agenten

59

- Vielen Dank!

60