Tracking

- Einführung
- Allgemeiner Systemaufbau
- Objektlokalisation: Template-Matching
- Prädiktionsfilter: Kalman

Birgit Möller & Denis Williams
AG Bioinformatik & Mustererkennung
Institut für Informatik
Martin-Luther-Universität Halle-Wittenberg

Einführung

Tracking von Objekten

Zielsetzung:

Verfolgung der Bewegung eines Objektes mit Hilfe einer Kamera

durch Auswertung von Bildfolgen

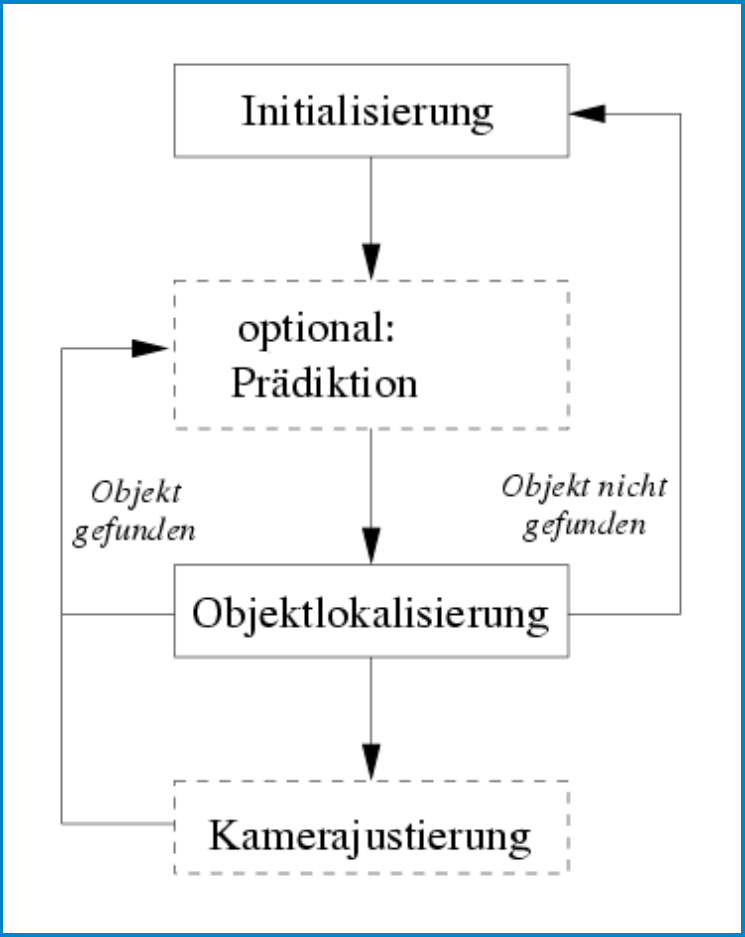
Anwendungen:

- Beobachtung des Straßenverkehrs

  (stationäre, statische Kamera und bewegte Objekte)
- Sicherheitsanlagen (stationäre, nicht-statische Kamera und bewegte Objekte)
- autonome Fahrzeuge (nicht-stationäre Kamera und bewegte Objekte)
- mobile Roboter (nicht-stationäre Kameras in beliebigen Umgebungen)
- . . .

Systemaufbau

Allgemeiner Systemaufbau:



- suche Zielobjekt gemäß Modell
- schätze Objektposition im nächsten Bild:
  - Prädiktionsfilter
  - Condensation-Algorithmus
  - . . .
- lokalisiere Objekt im aktuellen Bild:
  - Templates
  - Active Contours
  - . . .
- positioniere Kamera neu

Bemerkungen

- Probabilistische / nicht-probabilistische Modelle:

  eine absolute Position pro Zeitschritt vs. mehrere potenzielle Positionen

- Single- und Multi-Objektverfolgung:

  Tracking eines Objektes vs. paralleles Tracking mehrerer Objekte

- Objektlokalisation anhand geeigneter Merkmale:

  Farbe, Form, Struktur, $\cdots$

- Prädiktion durch zeitliche Modellierung der Bewegung

- Nachführung der Kamera bei großen Bewegungsradien der Objekte:

  ”Halte Objekt möglichst im Zentrum des Bildausschnitts.”

Objektlokalisation: Template-Matching

- einfachster Ansatz zur Lokalisation
- Grundidee:

  ”Gegeben ein Beispielmuster (*Template S*) des gesuchten Signals, suche in unbekanntem Muster *f* einen Ausschnitt, der zum Template passt!”
- Signal *f* in der Regel direkt durch Bild- oder Audiosignal bzw. geeigneten Merkmalsvektor gegeben
- Template *S* entspricht einem Signalausschnitt gleicher Dimension

Problem:

Template-Matching erfordert möglichst punktgenaue Übereinstimmung!!!

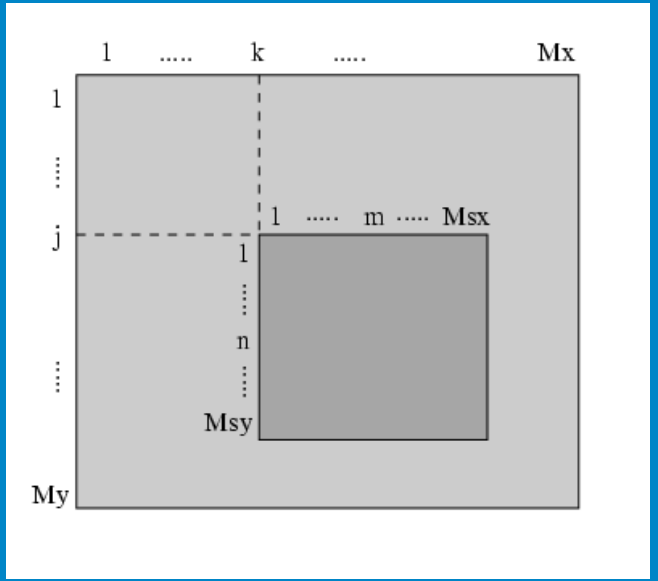
(geeignet bei definierten Lichtverhältnissen und Objektkonstanz)

Template-Matching auf Bildern

Gegeben:

- $M_x \times M_y$-dimensionales Bildsignal f
- $M_{s_x} \times M_{s_y}$-dimensionales Template S
- im Allgemeinen gilt:

  $M_x > M_{s_x}$ und $M_y > M_{s_y}$
- Matching: suche Position (j, k) im Bild f mit der größten Übereinstimmung (Faltungsoperation!)



Gesucht: geeignetes **Abstandsmaß**, das zu minimieren ist!

Abstandsmaße

- City-Block:

$$\epsilon^1_{j,k} = \sum_{m=1}^{M_{sx}} \sum_{n=1}^{M_{sy}} |f_{(j+m,k+n)} - S_{m,n}| \quad \text{(minimieren)}$$

- Quadratischer Abstand:

$$\epsilon^2_{j,k} = \sum_{m=1}^{M_{sx}} \sum_{n=1}^{M_{sy}} (f_{(j+m,k+n)} - S_{m,n})^2 \quad \text{(minimieren)}$$

- Kreuzkorrelation:

$$R_{j,k} = \sum_{m=1}^{M_{sx}} \sum_{n=1}^{M_{sy}} f_{(j+m,k+n)} \cdot S_{m,n} \quad \text{(maximieren)}$$

- normalisierte Kreuzkorrelation:

$$R'_{j,k} = \frac{R_{j,k}}{\sqrt{\sum_{m=1}^{M_{sx}} \sum_{n=1}^{M_{sy}} f^2_{(j+m,k+n)}} \sqrt{\sum_{m=1}^{M_{sx}} \sum_{n=1}^{M_{sy}} S^2_{m,n}}} \quad \text{(maximieren)}$$

Normalisierung

Alternative Normalisierung:

- Energie von Bild und Template:

$$\sum_{x=j}^{M_{S_x}} \sum_{y=k}^{M_{S_y}} f_{(x,y)}^2 = 1 \quad \text{bzw.} \quad \sum_{x=1}^{M_{S_x}} \sum_{y=1}^{M_{S_y}} S_{(x,y)}^2 = 1$$

Weitere Probleme:

- Templates sind i.A. weder rotations- noch skalierungsinvariant

  (vorherige Transformationen, Normalisierungen, etc. notwendig)
- bei partiellen Verdeckungen Zerlegung des Templates in Sub-Templates

  $\Rightarrow$ Match bei Übereinstimmung in $k \geq \theta_k$ Sub-Templates

Aufwandsreduktion

Ansätze zur Aufwandsreduktion:

- Template-Matching entspricht einer **Faltung**

  $\Rightarrow$ Transformation in den Frequenzraum (FFT): Faltung $\rightarrow$ Multiplikation

- Auflösungspyramiden:
  - Subsampling von Bild und Template
  - Suche nach Matches auf unterster Ebene
  - Beschränkung des Matchings auf der nächsten Ebene auf die n besten Positionen aus vorherigem Schritt

- Prädiktionsfilter:

  Beschränkung des Suchbereichs durch Schätzung der neuen Objektposition

Prädiktionsfilter - Kalman

Allgemeine Grundidee:

rekursive Modellierung eines mit einem Zufallsprozeß überlagerten linearen Systems, um aus dem Verhalten bis zum aktuellen Zeitpunkt Aussagen über das zukünftige Verhalten machen zu können

”Features”:

- Interpretation von verrauschten Meßdaten in Realzeit
- Implizite Modellierung von Zufallskomponenten für sichere Vorhersagen
- ermöglicht Aussagen über Vorhersagefehler

Ansatz:

”Gegeben ein Modell für die Bewegung und die Meßdaten über den Zustand des Systems bis zum Zeitpunkt k, sowie ferner Annahmen über zufällige Störeinflüsse, mache eine Vorhersage für den Systemzustand zum Zeitpunkt $k + 1$!”

Modellierung

Modellierung des linearen Systems:

$$\begin{aligned} \vec{x}_{k+1} &= \Phi_k \cdot \vec{x}_k + \vec{\omega}_k \\ \vec{z}_k &= H_k \cdot \vec{x}_k + \vec{\nu}_k \end{aligned}$$

$\vec{x}_k$:= Systemzustand zum Zeitpunkt k ($n \times 1$-dimensional)

Φ_k := Zustandsübergangsmatrix $\vec{x}_k \longrightarrow \vec{x}_{k+1}$ ($n \times n$-dimensional)

$\vec{\omega}_k$:= statistischer Systemanteil, weißes, unkorreliertes Rauschen bekannter Kovarianz

$$E\{\vec{\omega}_i \vec{\omega}_k\} = \delta_{ik} \cdot Q_k$$

$\vec{z}_k$:= Meßdaten des Zeitpunktes k ($m \times 1$-dimensional)

H_k := spezifiziert Zusammenhang zwischen $\vec{x}_k$ und $\vec{z}_k$ ($m \times n$-dimensional)

$\vec{\nu}_k$:= Meßfehler ($n \times 1$-dimensionaler Zufallsvektor)

$$\begin{aligned} E\{\vec{\nu}_i \vec{\nu}_k\} &= \delta_{ik} \cdot R_k \\ E\{\vec{\omega}_k \vec{\nu}_i\} &= 0 \quad , \forall k, i \end{aligned}$$

Modellierung - Beispiel

Beispiel: 1D-Bewegung eines PKW

- Systemgleichungen: lineare Bewegungsgleichungen der Physik
- Messung: Abstand per Radar
- Zufallsprozeß: Luftwiderstand
- Messfehler: Störsignale, Zeitmessung

Also:

$$
\begin{aligned}
\vec{x}_k &= [y_k, \dot{y}_k, \ddot{y}_k]^T \\
y_{k+1} &= y_k + \Delta t \cdot \dot{y}_k + \frac{\Delta t^2}{2} \cdot \ddot{y}_k \\
\dot{y}_{k+1} &= \dot{y}_k + \Delta t \cdot \ddot{y}_k \\
\Rightarrow \Phi_k &= \begin{pmatrix} 1 & \Delta t & \frac{\Delta t^2}{2} \\ 0 & 1 & \Delta t \\ 0 & 0 & 1 \end{pmatrix}
\end{aligned}
$$

Modellierung - Beispiel

- Zufallsprozess: Wind

$$Q = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

- messbarer Zustand $\vec{z}_k$:
  Position y_k des PKW über Radar, also $H_k = (1\ 0\ 0)$

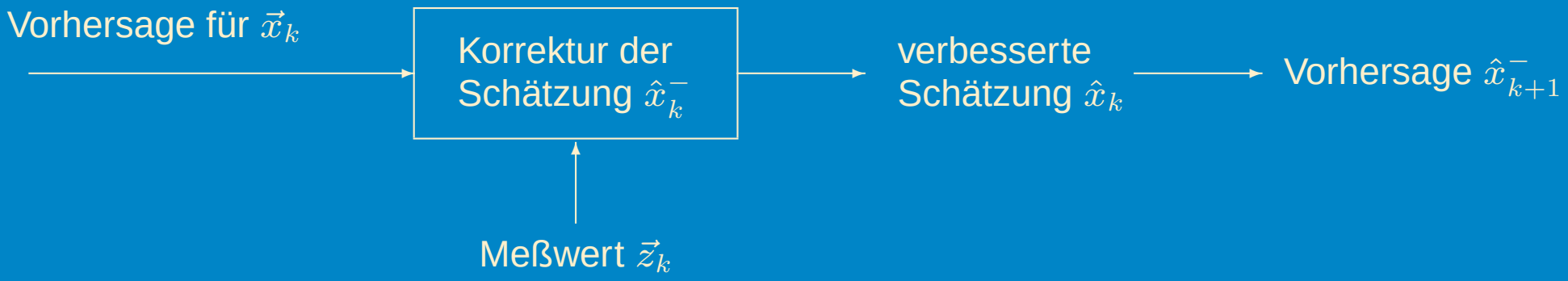
- Messfehler:

$$N(0,1) \cdot \frac{y_k}{10}$$

Frage jetzt: Wie arbeitet der Kalman-Filter?

Arbeitsweise Kalmanfilter

Grundprinzip:



Notation:

- $\vec{x}_k$:= tatsächlicher Systemzustand (nicht bekannt!)
- $\hat{x}_k^-$:= geschätzter Zustand vor Analyse (*a-priori Schätzung*)
- $\hat{x}_k$:= geschätzter Zustand nach der Korrektur (*a-posteriori Schätzung*)

Arbeitsweise Kalmanfilter

Herleitung der Filtergleichungen

- gegeben a-priori Schätzfehler zum Zeitpunkt k

$$e_k^- = (\vec{x}_k - \hat{x}_k^-)$$

mit Kovarianzmatrix

$$P_k^- = E\{e_k^- e_k^{-T}\} = E\{(\vec{x}_k - \hat{x}_k^-)(\vec{x}_k - \hat{x}_k^-)^T\}$$

- Ziel: korrigiere Zustandsschätzung und Kovarianz anhand des Messfehlers

$$\hat{x}_k = \hat{x}_k^- + K_k \cdot (\vec{z}_k - H \cdot \hat{x}_k^-) \quad (1)$$

- der Korrekturfaktor K_k heisst auch **Kalmanfaktor**

Arbeitsweise Kalmanfilter

Ziel: finde optimales K_k für eine neue Zustandsschätzung $\hat{x}_k$

$\Rightarrow$ Minimierung des mittleren quadratischen Schätzfehlers $P_k = E\{e_k e_k^T\}$

Ansatz:

$$\underset{K_k}{\text{argmin}}\ P_k = \underset{K_k}{\text{argmin}}\ E\{(\vec{x}_k - \hat{x}_k)(\vec{x}_k - \hat{x}_k)^T\}$$

Es gilt:

$$\vec{z}_k = H_k \cdot \vec{x}_k + \vec{\nu}_k$$

Aus Einsetzung in Gleichung (1) folgt dann

$$\hat{x}_k = \hat{x}_k^- + K_k \cdot (H_k \vec{x}_k + \vec{\nu}_k - H_k \hat{x}_k^-)$$

Damit gilt schließlich:

$$P_k = E\{(\vec{x}_k - \hat{x}_k^- - K_k \cdot (H_k \vec{x}_k + \vec{\nu}_k - H_k \hat{x}_k^-)) \cdot (\vec{x}_k - \hat{x}_k^- - K_k \cdot (H_k \vec{x}_k + \vec{\nu}_k - H_k \hat{x}_k^-))^T\}$$

Arbeitsweise Kalmanfilter

- Umformung liefert nach diversen ”Rumrechnereien”... :-)

$$P_k = (I - K_k H_k) \cdot P_k^- \cdot (I - K_k H_k)^T + K_k R K_k^T \tag{2}$$

- Minimierung von P_k durch Ableiten und Nullsetzen ergibt

$$K_k = \frac{P_k^- \cdot H_k^T}{H_k P_k^- H_k^T + R_k}$$

- Einsetzen des Kalmanfaktors in Gleichung (2) liefert

$$P_k = (I - K_k H_k) \cdot P_k^-$$

  als korrigierte Fehlerkovarianz

- analog dazu folgt aus Gleichung (1) eine korrigierte Zustandsschätzung $\hat{x}_k$

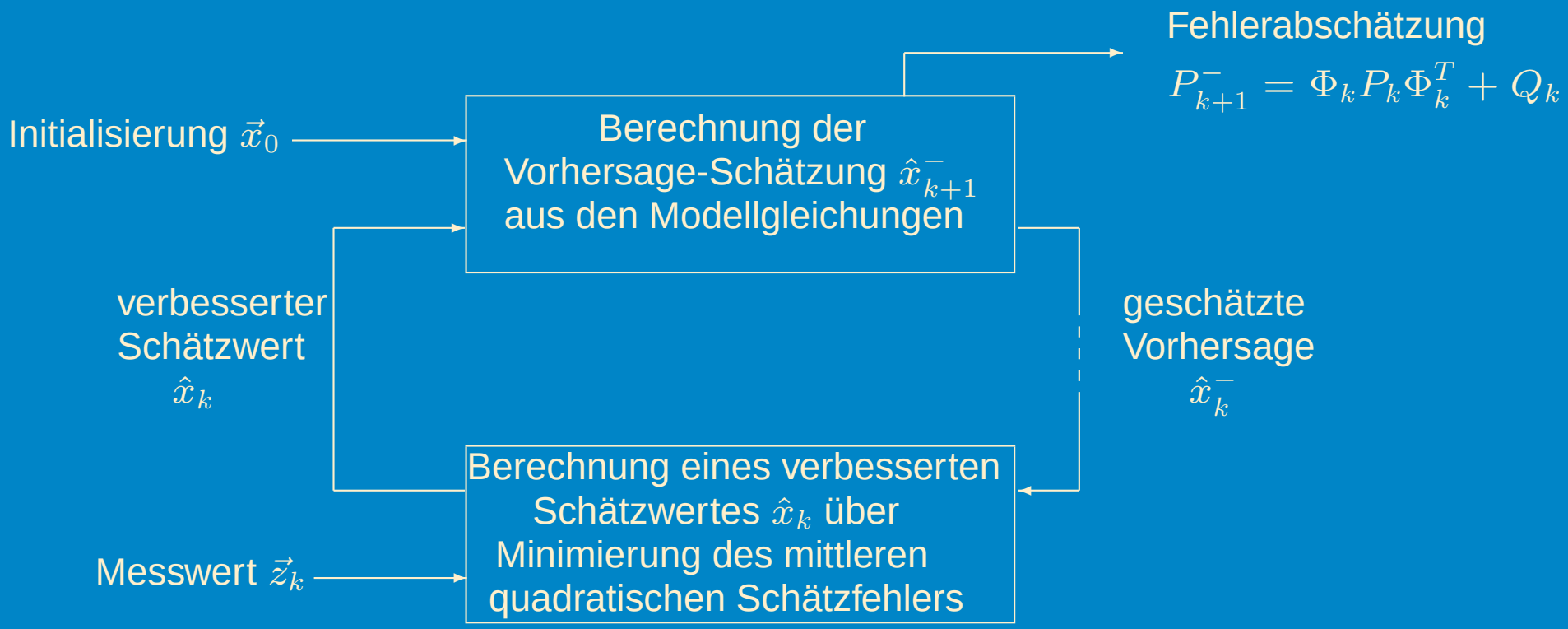
Arbeitsweise Kalmanfilter

- Vorhersage des Systemzustandes für den Zeitpunkt $t+1$:

$$
\begin{aligned}
\hat{x}_{k+1}^{-} &= \Phi_k \cdot \hat{x}_k + \vec{\omega}_k \\
P_{k+1}^{-} &= E\{(\vec{x}_{k+1} - \hat{x}_{k+1}^{-})(\vec{x}_{k+1} - \hat{x}_{k+1}^{-})^T\} \\
&= E\{(\Phi_k \vec{x}_k + \vec{\omega}_k - \Phi_k \hat{x}_k - \vec{\omega}_k)^2\} \\
&= E\{\Phi_k e_k + \vec{\omega}_k^2\} \\
&= \Phi_k \cdot E\{e_k^2\}\Phi_k^T + E\{\vec{\omega}_k^2\} \\
&= \Phi_k \cdot P_k \Phi_k^T + Q_k
\end{aligned}
$$

- $\vec{\omega}_k$ ist mittelwertfrei und unkorreliert und kann daher entfallen

Überblick



- je mehr Zustände beobachtet werden, desto sicherer ist die Vorhersage
- Filter erfordert passendes Modell (fehlerhaftes Modell wird nicht ausgeglichen!)
- numerische Probleme bei langer Laufzeit möglich
- beschränkt auf *lineare* Systeme $\rightarrow$ **Extended Kalman**