

Nachrichtentechnik 2

Multimodale Daten-Fusion (Kalmanfilter)

Jan Langer (jan@langernetz.de) – 22376
Sven Kühn (sven@svenkuehn.de) – 22279
Daniel Kriesten (daniel@amico-online.de) – 22262
Robert Lange (rola@hrz.tu-chemnitz.de) – 22243

16. Juli 2002

Inhaltsverzeichnis

1	Herleitung	2
2	Auswertung	4
3	Schluß	4
4	Matlab-Quelltext	6

Zusammenfassung

Der Kalmanfilter ist ein mathematisches Modell, welches mit Hilfe probabilistischer Methoden Messwerte verarbeitet. Dabei wird die Methode der minimalen quadratischen Abweichung genutzt. Der Filter verwendet vergangene Messwerte, um den weiteren Werteverlauf vorherzusagen. Predizierte Werte und Messwerte werden verglichen, um kontinuierlich die Vorhersagegenauigkeit zu verbessern. Der Kalmanfilter kann auch Werte, die sich aus der Messgröße herleiten, predizieren (z.B. Geschwindigkeit und Beschleunigung aus einer Wegmessung). Selbst wenn eine genaue Beschreibung des Modells nicht möglich ist, können vergangene, gegenwärtige und zukünftige Werte bestimmt werden.

1 Herleitung

Abbildung 1 stellt den grundlegenden Aufbau des dem Kalmanfilter zugrundeliegenden Zustandsmodells dar. Die Eingangsgrößen $\vec{u}(t)$ beeinflussen über die Matrix B den durch $\vec{x}(t)$ und $\dot{x}(t)$ gekennzeichneten inneren Zustand des Filters. Dieser innere Zustand kann durch $\vec{v}(t)$ gestört werden und ist über die Matrix C am Ausgang $\vec{y}(t)$ zu beobachten.

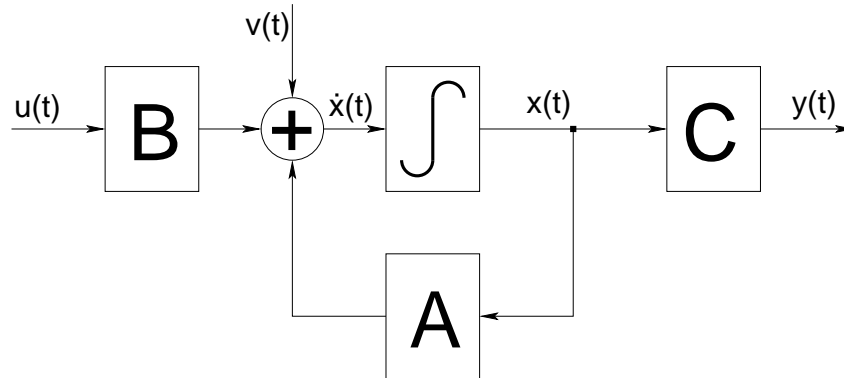


Abbildung 1: Zustandsmodell (Blockdiagramm)

Es folgt nun die Ermittlung der Matrizen A , B und C anhand der physikalischen Aufgabenstellung. Durch den Wegfall der Eingangsgrößen ergibt sich Matrix B zu Null. Das in der Aufgabenstellung geforderte Bewegungsmodell konstanter Beschleunigung erfordert Position $s(t)$, Geschwindigkeit $v(t)$ und Beschleunigung $a(t)$ für jede der beiden Dimensionen x und y . Der interne Zustand des Modells wird durch den Zustandsvektor x dargestellt. x ergibt sich aus Position, Geschwindigkeit sowie Beschleunigung in Richtung der Dimensionen x und y .

$$x = \left(s_x \quad v_x \quad a_x \quad s_y \quad v_y \quad a_y \right)^T$$

Dieser Zustandsvektor wird dann über die Matrix C auf den internen Ausgangsvektor $y_i = \begin{pmatrix} s_x \\ s_y \end{pmatrix}$ abgebildet. Damit ergibt sich die Matrix C zu:

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

Ausgehend von den Gleichungen

$$s = s_0 + vt + \frac{1}{2}at^2$$

$$v = v_0 + at$$

und der Diskretisierung der Zeit t mit Hilfe des diskreten Zeitschrittes T erhält man folgende Matrix A , deren Funktion es ist, den vorangegangenen auf den aktuellen Zeitschritt abzubilden:

$$\dot{x}(t) = A\dot{x}(t) + v(t)$$

$$A = \begin{pmatrix} 1 & T & \frac{T^2}{2} & 0 & 0 & 0 \\ 0 & 1 & T & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & T & \frac{T^2}{2} \\ 0 & 0 & 0 & 0 & 1 & T \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

Mit dem gegebenen Zeitschritt $T = 0.1s$ erhält man folgende Matrix:

$$A = \begin{pmatrix} 1 & 0.1 & 0.005 & 0 & 0 & 0 \\ 0 & 1 & 0.1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0.1 & 0.005 \\ 0 & 0 & 0 & 0 & 1 & 0.1 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

Die Ausgangswerte des Zustandsmodells ($y_i(k)$) werden nun mit den Messwerten ($y_a(k)$) verglichen. Die Differenz wird mit den Kalmangain K gewichtet und als Störung dem Zustandsmodell zugeführt.

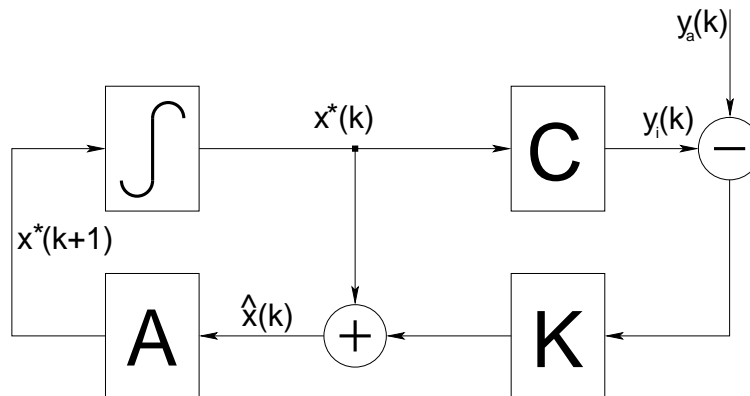


Abbildung 2: Kalmanfilter (Blockdiagramm)

Die Matrix K wird durch die Messunsicherheit R und die Prozessunsicherheit Q dynamisch angepasst. Dabei dient die geschätzte Fehler-Kovarianz P als Zwischengröße. Damit erhält man nun die fünf Gleichungen des Kalman-Filters, unterteilt in 2 Phasen:

Prediktion:

$$\begin{aligned} x^*(k+1) &= A(k) \cdot \hat{x}(k) \\ P^*(k+1) &= A(k) \cdot \tilde{P}(k) \cdot A^T(k) + Q(k) \end{aligned}$$

Fusion:

$$\begin{aligned} K(k+1) &= P^*(k+1) \cdot C^T(k+1) (C(k+1) \cdot P^*(k+1) \cdot C^T(k+1) + R(k+1))^{-1} \\ \hat{x}(k+1) &= x^*(k+1) + K(k+1) (y_a(k+1) - C(k+1) \cdot x^*(k+1)) \\ \tilde{P}(k+1) &= P^*(k+1) - K(k+1) \cdot C(k+1) \cdot P^*(k+1) \end{aligned}$$

In unserem Versuch wurden die folgenden Werte für die charakteristischen Matrizen R und Q angenommen:

$$R = \begin{pmatrix} 2.25 & 0 \\ 0 & 1 \end{pmatrix}$$

$$Q = \begin{pmatrix} \frac{T^5}{20} & \frac{T^4}{8} & \frac{T^3}{6} & 0 & 0 & 0 \\ \frac{T^4}{8} & \frac{T^3}{3} & \frac{T^2}{2} & 0 & 0 & 0 \\ \frac{T^3}{6} & \frac{T^2}{2} & T & 0 & 0 & 0 \\ 0 & 0 & 0 & \frac{T^5}{20} & \frac{T^4}{8} & \frac{T^3}{6} \\ 0 & 0 & 0 & \frac{T^4}{8} & \frac{T^3}{3} & \frac{T^2}{2} \\ 0 & 0 & 0 & \frac{T^3}{6} & \frac{T^2}{2} & T \end{pmatrix} \cdot \sigma^2 \quad (\sigma = 3)$$

Des Weiteren wurde als Anfangswert der Matrix P ein beliebig grosser Wert angenommen, um im weiteren Verlauf durch den sich nach dem Einschwingvorgang ergebenden Wert ersetzt zu werden.

2 Auswertung

Nachdem die vorgegebene Detektionsliste mit der Matlab-Funktion verarbeitet wurde, wird der ungefilterte und der gefilterte Positionsgraph im Diagramm 3 dargestellt. Man kann deutlich sehen, daß die gefilterte Kurve wesentlich ruhiger verläuft und versucht, die Schwankungen der Meßgröße auszugleichen.

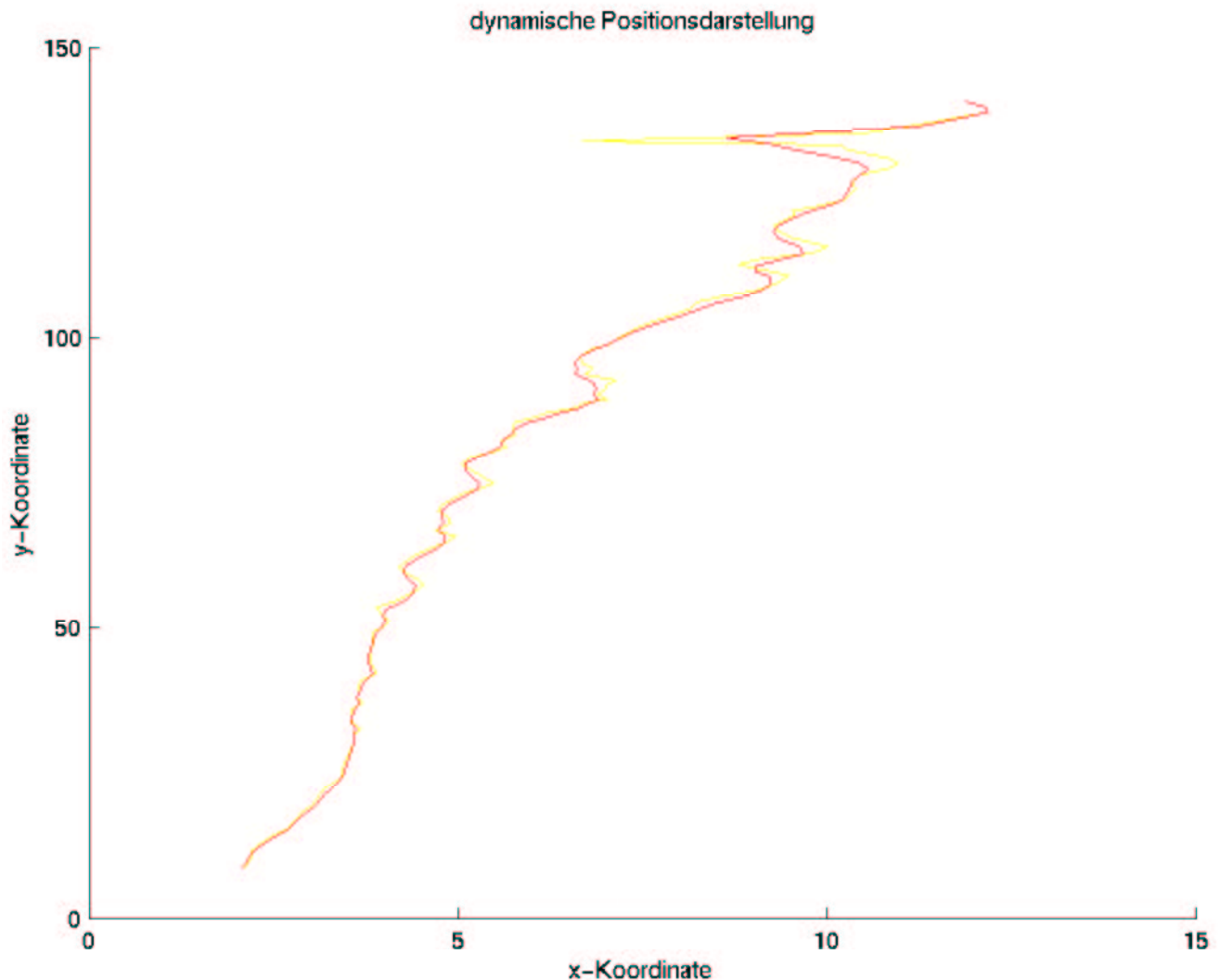


Abbildung 3: Darstellung der Position der des Objektes (gefiltert (rot) und ungefiltert (gelb))

In Diagramm 4 kann man zusaetzlich zur Position des Objektes auch den zeitlichen Verlauf des Betrages der Geschwindigkeit und Beschleunigung sehen.

3 Schluß

Schon an diesem simplen Beispiel ist leicht die grosse Flexibilitaet des Kalmanfilters zu erkennen. Die Anwendungsmöglichkeiten erstrecken sich über Filterung (nicht nur auf Glättung beschränkt) der Eingangswerte bis zur Bestimmung von abgeleiteten Größen (wie Beschleunigung und Geschwindigkeit) aus den Eingangswerten. Hinzu kommt die Möglichkeit der fast grenzenlosen Einstellbarkeit des Filterverhaltens durch die geeignete Wahl der entsprechenden Systemmatrizen.

Somit ist der Kalman-Filter in vielen Anwendungsgebieten einsetzbar, in denen aus Eingangsgrößen über bestimmte Beziehungen Ausgangsgrößen herzuleiten sind.

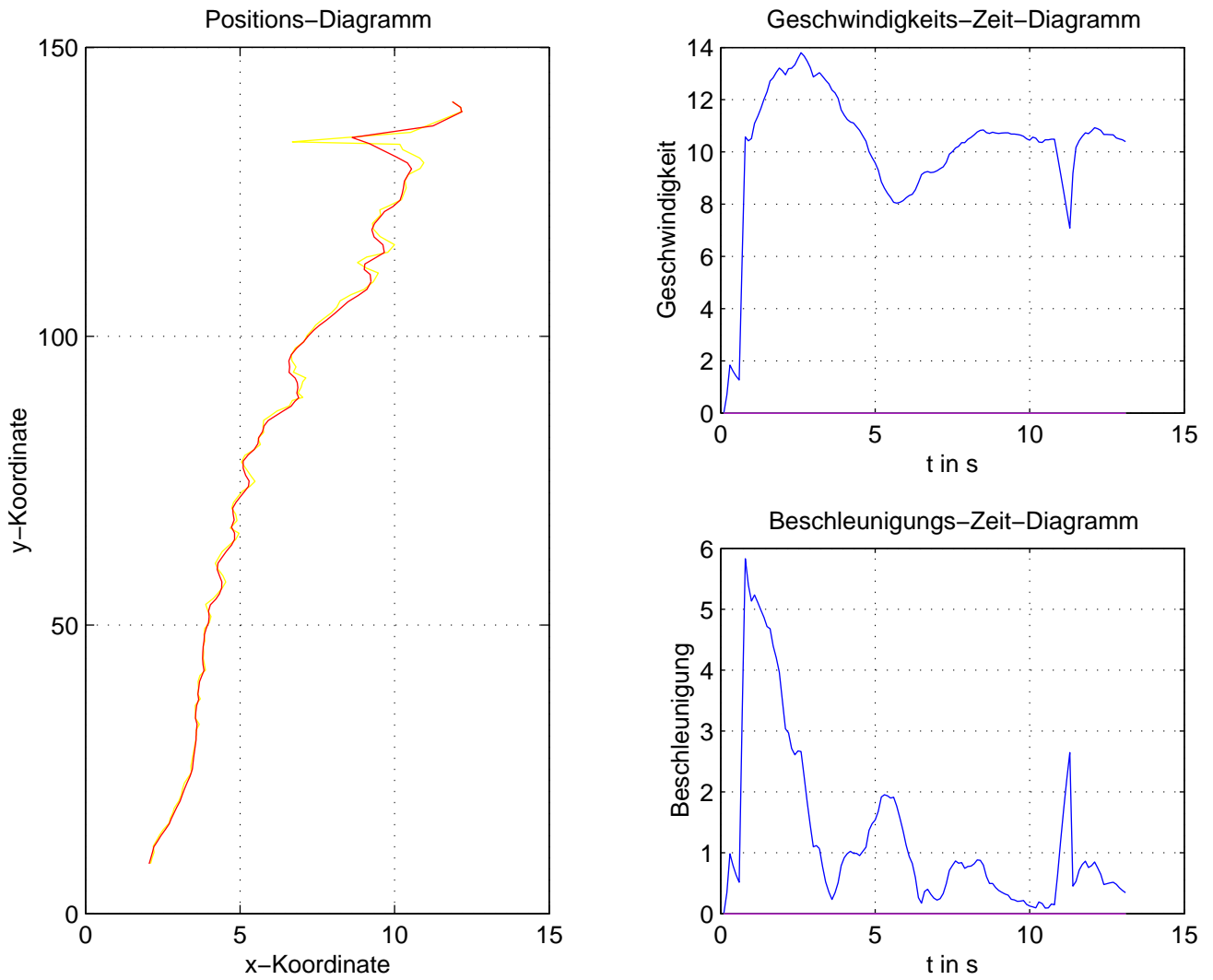


Abbildung 4: Verläufe aller drei Zustandsgrößen

4 Matlab-Quelltext

```
% Kalman-Filter
% Praktikum NT2 - Gruppe 3
% Jan Langer, Robert Lange, Sven Kuehn, Daniel Kriesten

% Konstanten
sigma = 3;
timedelay = 0.05;
a_skalierung = 3;
hoehenlinien = [.80 .20 .05];

% aus Datei lesen
A = load('initA.imp');
C = load('initC.imp');
P = load('initPO.imp');
R = load('initR.imp');
Q = load('initQ.imp');
daten = load('Data.imp');    % z entspricht dem y (Messwert) - Messwerte

anzahl = size(daten, 1);
HL = zeros (2, 120, anzahl);

% ummodelln fuer Initialisierungsphase
starx = [daten(1,1) 0 0 daten(1,2) 0 0]'; % Startwert ist unser erster Filterwert
yin = starx;
starP = P;
Q = Q*(sigma^2);    % die Q-Matrix mit sigma skalieren

% Ausgangsvektor anlegen
xout = zeros (anzahl, 6);
xvout = zeros (anzahl);
xaout = zeros (anzahl);
xout(1,:) = starx';

% Berechnung
for i=2:anzahl

    % aktuelles Wertepaar extrahieren (ausser wenn Null)
    if (daten(i,1)~=0 & daten(i,2)~=0)
    else
        daten(i,1) = xout(i-1,1);
        daten(i,2) = xout(i-1,4);
    end
    yin = daten(i,:);

% Berechnen
% Prediktion
K = starP * C' * ( C * starP * C' + R)^-1;
hatx = starx + K * ( yin - C * starx);
hatP = starP - K*C*starP;

% Fusion
starP = A * hatP * A' + Q;
starx = A * hatx;

% unser Endergebnis, das wir ausgeben wollen
```

```
xout(i,:) = hatx';
xvout(i) = sqrt(hatx(2)^2 + hatx(5)^2);
xaout(i) = sqrt(hatx(3)^2 + hatx(6)^2);

% Hoehenlinien berechnen
HLtmp = GetHL (hatP, xout(i,:)', hoehenlinien, 1, 4)';
HL(:,:,i) = HLtmp;

end

% Ausgabe

figure('Renderer','OpenGL')
%grid minor
hold on
plot (daten(:,1), daten(:,2), 'y-');
xlabel('x-Koordinate')
ylabel('y-Koordinate')
title('dynamische Positionsdarstellung')
for i=2:anzahl
    axis([0 15 0 150])
    plot (xout(i-1:i,1), xout(i-1:i,4), 'r-');
    handle0 = plot (xout(i,1), xout(i,4), 'ro');
    handle1 = plot ([xout(i, 1) (xout(i, 1) + xout(i, 2))], ...
                   [xout(i, 4) (xout(i, 4) + xout(i, 5))], 'g-');
    handle2 = plot ([xout(i, 1) (xout(i, 1) + a_skalierung * xout(i, 3))], ...
                   [xout(i, 4) (xout(i, 4) + a_skalierung * xout(i, 6))], 'b-');
    handle3a = plot (HL(1,1:40,i), HL(2,1:40,i));
    handle3b = plot (HL(1,41:80,i), HL(2,41:80,i));
    handle3c = plot (HL(1,81:120,i), HL(2,81:120,i));

    pause(timedelay)

    delete(handle0)
    delete(handle1)
    delete(handle2)
    delete(handle3a)
    delete(handle3b)
    delete(handle3c)
end
hold off

% Ausgabe der Weg/Zeit-, Geschwindigkeit/Zeit- und
% Beschleunigung/Zeit-Diagramme

figure
subplot(2, 2, [1 3]); plot (daten(:,1), daten(:,2), 'y-', xout(:,1), xout(:,4), 'r-')
xlabel('x-Koordinate')
ylabel('y-Koordinate')
grid on
title('Positions-Diagramm')
subplot(2, 2, 2); plot ((1:size(daten,1))/10, xvout)
xlabel('t in s')
ylabel('Geschwindigkeit')
grid on
title('Geschwindigkeits-Zeit-Diagramm')
subplot(2, 2, 4); plot ((1:size(daten,1))/10, xaout)
```

```
xlabel('t in s')
ylabel('Beschleunigung')
grid on
title('Beschleunigungs-Zeit-Diagramm')

clear sigma timedelay A C P R Q K daten hatP hatx i xout yin
clear starP starx a_skalierung hoehenlinien
clear handle0 handle1 handle2 handle3a handle3b handle3c
clear HL HLtmp anzahl xvout xaout

% ENDE
```