Technische Universität Braunschweig

Institut für Regelungstechnik

Masterarbeit

Untersuchung von Algorithmen zur Identifikation der Parameter nichtlinearer Fahrdynamikmodelle

Wuqiang Sun
Matrikel-Nr.:4353838

Betreuer	:	Prof. Dr.-Ing. M. Maurer
	:	Musterassi
Eingereicht am	:	30. September 2015

Erklärung

Hiermit versichere ich, die vorliegende Arbeit selbständig und ohne fremde Hilfe angefertigt zu haben. Die verwendete Literatur und sonstige Hilfsmittel sind vollständig angegeben.

Braunschweig, 30. September 2015

Inhaltsverzeichnis

Abbildungsverzeichnis

Symbolverzeichnis

1. Kennzeichnung durch Art der Schreibweise

$x_t,\ u_t,\ y_t$ etc.	Augenblickswerte
$w_t^i,\ \tilde{w}_t^i$, etc.	nicht normierter/normierter Wert

2. Symbole

Abkürzung	*Variable*	*Einheit*
t	Zeitpunkt	s
T	Endzeit des Zeitabschnitts	s
F	Kraft	N
l	Lange	m
J	Trägheitsmoment	$\mathrm{kg/m^2}$
V	Geschwindigkeit	m/s
a	Beschleunigung	$\mathrm{m/s^2}$
$\dot{v}$	Beschleunigung	$\mathrm{m/s^2}$
φ	Gierwinkel des Fahrzeugs	rad
ω	Gierrate des Fahrzeugs	rad/s
$\dot{\omega}$	Winkelbeschleunigung des Fahrzeugs	$\mathrm{rad/s^2}$
α	Schräglaufwinkel des Reifens	rad
β	Schwimmwinkel des Reifens	rad
δ	Lenkwinkel des vorderes Reifens	rad
h	Höhe des Schwerpunkts von Fahrzeug	m
g	Schwerbeschleunigung	$\mathrm{m/s^2}$
Cs	Reifensteifigkeit	rad^{-1}
x_t	Zustandsvektor des Systems am Zeitpunkt t	
u_t	Eingangsvektor des Systems am Zeitpunkt t	
y_t	Ausgangsvektor des Systems am Zeitpunkt t	
w_t	Prozessrauschen des Systems am Zeitpunkt t	

Ξ	Varianzmatrix von Prozessrauschen
v_t	Messrauschen des Systems am Zeitpunkt t
Θ	Varianzmatrix von Messrauschen
p	Parametervektor des Systems
c^T	Ausgangsmatrix des Systems
θ	Parameter von der Funktion f
x_t^i	**i**te Partikel vom Partikel-Filter für Zustand x
$x_{t\|T}^i$	**i**te Partikel vom Partikel-Smoother für Zustand x
w_t^i	Gewicht für **i**te Partikel x_t^i in Partikel-Filter
$\tilde{w}_t^i$	Normierter Gewicht für **i**te Partikel x_t^i

3. Indizes

L	Abstand zwischen vorderer Achse und hinterer Achse
l_f	Abstand zwischen vorderer Achse und Schwerpunkt
i_r	Abstand zwischen vorderer Achse und Schwerpunkt
V_x^{Ref}	Geschwindigkeit des Fahrzeugs am Reference-Punkt in Langrichtung
V_y^{Ref}	Geschwindigkeit des Fahrzeugs am Reference-Punkt in Querrichtung
V_x^{IMU}	Geschwindigkeit des Fahrzeugs am Mess-Punkt in Langrichtung
V_y^{IMU}	Geschwindigkeit des Fahrzeugs am Mess-Punkt in Querrichtung
$\dot{V}_x^{Ref}$	Beschleunigung des Fahrzeugs am Reference-Punkt in Langrichtung
$\dot{V}_y^{Ref}$	Beschleunigung des Fahrzeugs am Reference-Punkt in Querrichtung
$\dot{V}_x^{IMU}$	Beschleunigung des Fahrzeugs am Mess-Punkt in Langrichtung
$\dot{V}_y^{IMU}$	Beschleunigung des Fahrzeugs am Mess-Punkt in Querrichtung
a_x	Beschleunigung des Fahrzeugs am Schwerpunkt in Langrichtung
F_{zf}	Druckkraft vorderes Reifens
F_{zr}	Druckkraft hinteres Reifens
F_{yf}	Seitenkraft vorderes Reifens
F_{yr}	Seitenkraft hinteres Reifens
β_f	Schwimmwinkel vorderes Reifens
β_r	Schwimmwinkel hinteres Reifens
α_f	Schräglaufwinkel vorderes Reifens
α_r	Schräglaufwinkel hinteres Reifens
C_{sf}	Reifensteifigkeit vorderes Reifens
C_{sr}	Reifensteifigkeit hinteres Reifens
J_y	Trägheitsmoment des Fahrzeugs um Querrichtung
x_0	Anfangszustand des Systems

Kurzfassung

In Rahmen der Projekt “Autonomes Fahren“ von Automotive Abteilung der Institute für Verkehrssystemtechnik von DLR Braunschweig ist die Regelung von “Trajektorie Folgen“ für autonomes Fahren einer wichtiger Bestandteil. Zur Regelung des Fahrzeugs muss das Fahrzeug genau modelliert werden. Die ungenaue Parameter kann die Regelung stark stören. Nun ist die Parameter-Identifikation notwendig. Die Fahrzeugsystem ist normalerweise nichtlinear System, dafür ist viele Identifikationsmethode wie “Least-Sqaure-Verfahren“ nicht geeignet. Hierfür ist die EM- Algorithmus als der Identifikationsmethode ausgewählt. Dieser Artikel demostriert wie die Parameter-Identifikation mit Expectation-Maximization-Identifikation für Fahrzeugsystem funktioniert.

In diesem Arbeit sind folgender Inhalte aufgelistet:

- Modellierung des Fahrzeugsystems
- Grundkenntnisse von EM-Identifikation
- Implementierung der EM-Identifikation in Matlab
- Überprüfung der EM-Identifikation bei Simulation
- EM-Identifikation für Testfahrzeug

Abstract

The trajectory tracking controller is a central component of the automated vehicle research at the Institute of Transportation Systems, DLR Braunschweig. A precise vehicle model is required for high control performance. As several parameters are not known a-priori and others may vary over time, a parameter identification method is designed, which uses test drive recordings to estimate the best fitting parameter set for a nonlinear lateral dynamics model. Due to the model's non-linearity, typical least-squares fitting methods are insufficient. Therefore an "Expectation-Maximization" procedure as an approximation to Maximum-Likelihood-Estimation is implemented. This work successfully demonstrates the identification of vehicle parameters for simulation data and for recordings from a physical test drive.
Content:

- Modeling of vehicle lateral dynamics
- Principle of the Estimation-Maximization (EM) Algorithm
- Implementation of the EM-Algorithm in Matlab
- Evaluation of EM-Algorithm on simulated data
- Parameter identification for physical test vehicle

1 Einleitung

Die Automotive Abteilung vom Institute für Verkehrssystemtechnik vom DLR Braunschweig entwickelt sichere und hoch automatisierte Fahrzeuge. Das Ziel ist der Aufbau eines autonomen Fahrzeugs, welches auf der Straße selbständig fahren kann. Das bedeutet, es plant die Routen, erkennt Hindernisse oder andere Autos, analysiert das Szenario und führt die dafür optimale Aktion(wie Spurwechsel oder Bremsen) aus.
Die Architektur eines autonomen Fahrzeugsystem besteht aus drei Steuerungsschichten für selbständiges Fahren(siehe Abbildung 1.1). Sie sind nach der Aufgabenstellung von Oben nach Unten als Routenplanung, Trajektorie-Planung und Regelung des Fahrzeugs bestimmt. Die Entscheidung der oberen Ebene wird durch dem Wunsch des Fahrers herstellt. Nach der gegebener Entscheidung bildet die zweite Ebene eine Trajektorie ab und die dritte Ebene versucht das Fahrzeug nach dieser Trajektorie zu regeln. Jede Schicht erfasst bestimmte Kennwerte von Sensoren (wie z.B. GPS, Kamera, Radar und Inertial Navigation System) und erstellt daraus Stellwert um das Fahrzeug zu steuern.

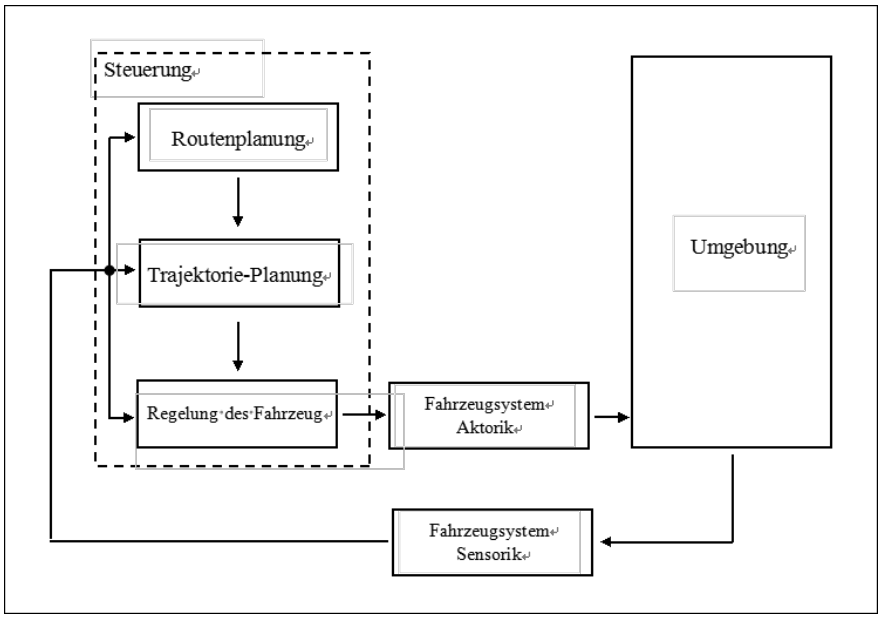


Abbildung 1.1: Schichtenarchitektur der Steuerung für autonomes Fahren

1.1 Aufbau des Testfahrzeugs

Für Tests der Funktionen des autonomen Fahrens baut das Institute einen VW Passant um. Viele Sensoren und Steuergeräte werden an das Auto angebracht. Der Aufbau der Sensorik ist in Abbildung 1.2 und Abbildung 1.3 zu sehen.

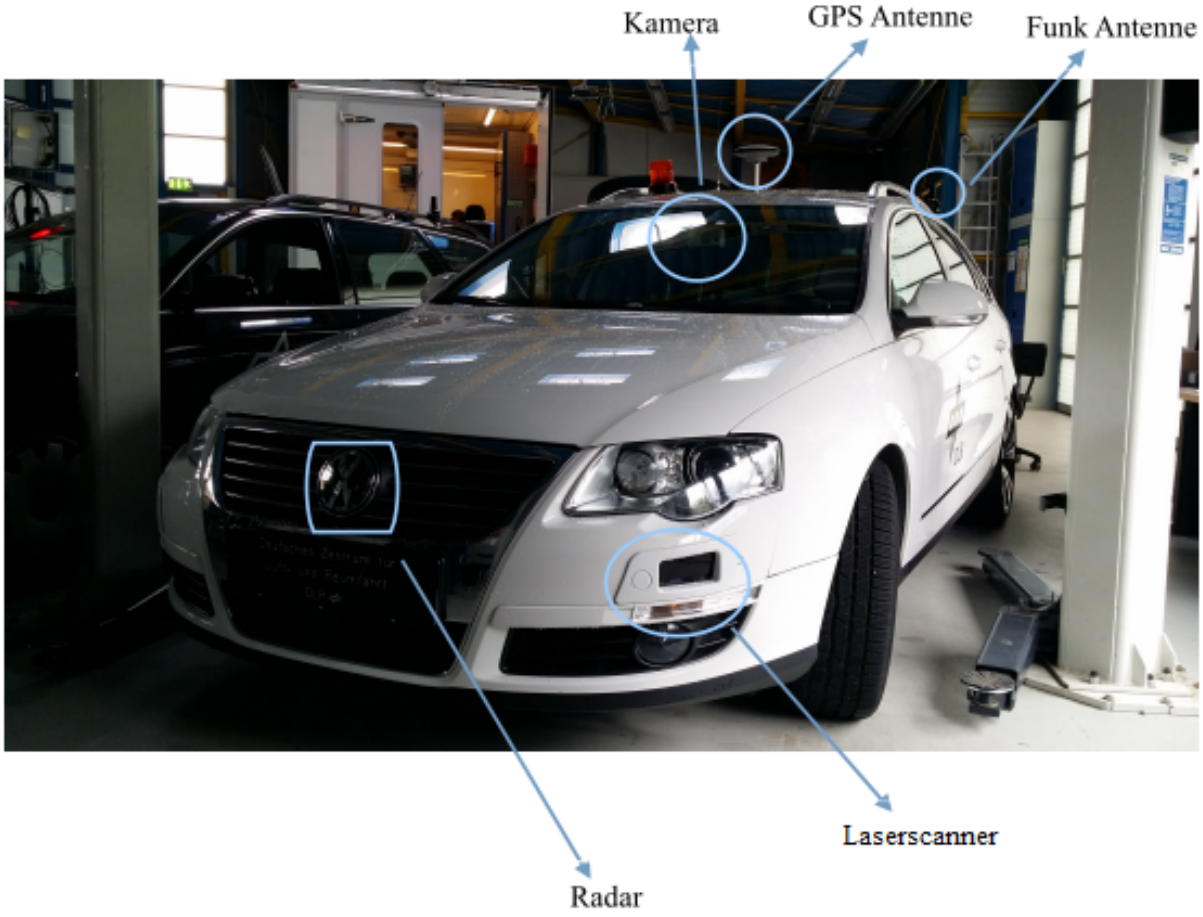


Abbildung 1.2: Aufbau der Sensorik des Fahrzeugs

Die obere Abbildung zeigt die eingebaute Sensorik am Fahrzeug. Das Radar, die Kamera und der Laserscanner überwachen den Bereich vor dem Fahrzeug. Jedes Geräte hat seine eigene Aufgaben. Der Radar überwacht den Abstand und die relative Geschwindigkeit zu vorderen Objekten. Der Laserscanner gibt den präzisen Abstand zum vorderen Objekt und korrigiert dadurch die Messung vom Radar. Die Kamera ist geeignet zur Erfassung der Geometrie des Objekts und der Information der Straße. Drei Sensoriken arbeiten zusammen, damit mit unterschiedlichen Zuständen umgegangen werden kann z.B.: Bei schlechtem Wetter oder Dunkelheit kann sich die Fähigkeit der Kamera verschlechtern, nun spielt der Radar eine

wichtigere Rolle. Auf der Seite des Dachs steht eine Funkantenne. Es ist für die Kommunikation mit der Außenwelt wie etwa zur Basisstation oder anderen Fahrzeugen. Zusätzlich ist die GPS Antenne nötig, zwei Antennen werden separat am mittleren Punkt der Vorder- und Hinterachse eingebaut. Diese greifen zyklisch das GPS-Signal für die Positionierung in der Navigation ab.

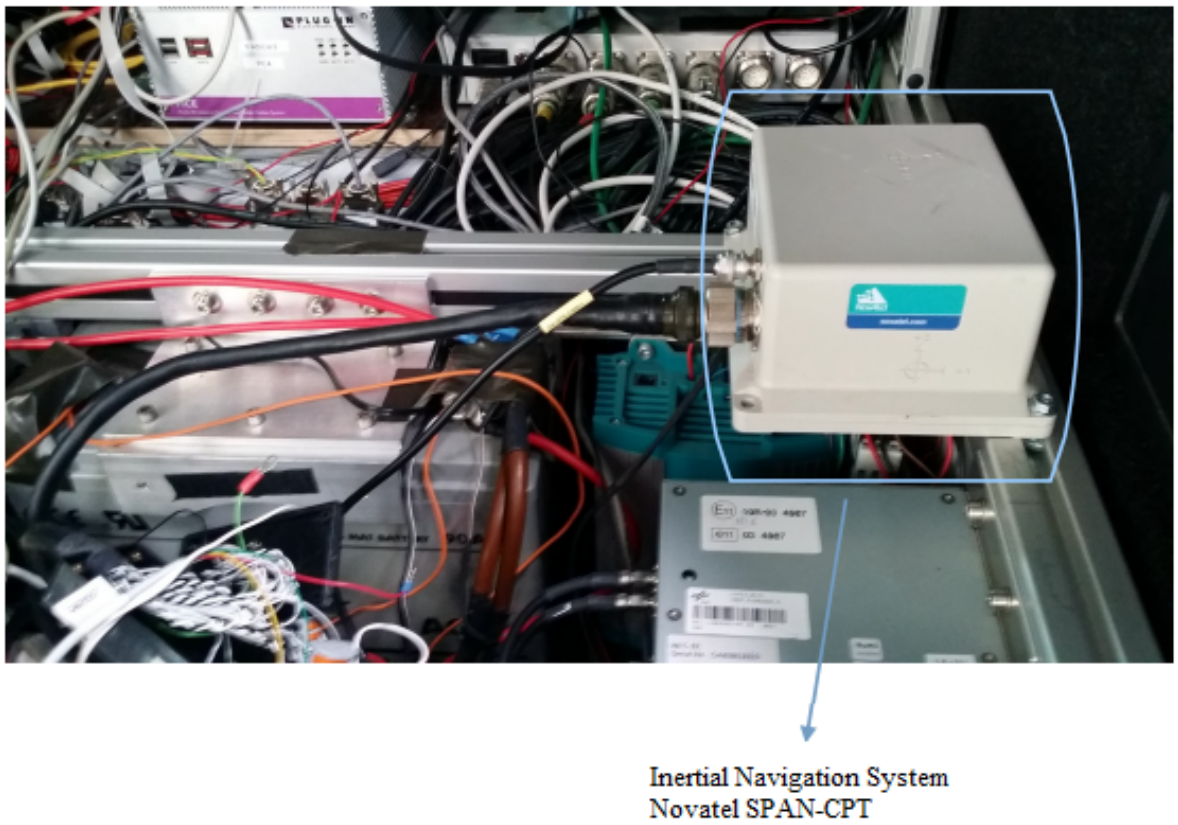


Abbildung 1.3: Einbau der Inertial Navigation System in Kofferraum

In Kofferraum, neben dem Mittelpunkt hinteren Achse befindet sich ein Inertial Navigation System. Das Inertial Navigation System misst die Beschleunigung, Geschwindigkeit und Winkelrate in drei Richtung (X, Y, Z). Der Referenz-Messpunkt ist der Mittelpunkt der hinteren Achse. Das bedeutet: Die gemessenen Daten müssen zum Referenz-Punkt umgerechnet werden.
Außerdem wird eine Messung des Lenkwinkels vom Lenkungsservo ausgeliefert. Diese Messung überwacht den aktuellen Lenkwinkel und bringt ihn zum Datenrekorder.

1.2 Aufgabenstellung

Die Aufgabenstellung hat mit der dritten Ebene (Regelung des Fahrzeugsystems) zu tun. Der Regler auf der dritten Ebene beschäftigt sich mit dem Verfolgen der Trajektorie, die von der zweiten Ebene gegeben wird. Zur Bestimmung des Reglers muss das Systemmodell untersucht werden. Das bedeutet: Der Aufbau eines geeigneten mathematischen Modells ist die Voraussetzung. Bei der Modellierung kann es passieren, dass einige Parameter wie Reifensteifigkeit, Trägheitsmoment und Gewichtsposition nicht von Datenbuch genau ermittelbar sind. Zum Beispiel kann das Trägheitsmoment und die Gewichtsposition von der Ladung und seiner Massenverteilung beeinflusst werden. Die Reifensteifigkeit hängt vom Luftdruck und Reibungskoeffizient der Straße ab. Solche ungenauen Parameter erzeugen die große Abweichung von Regelgröße. Wenn diese Ungenauigkeit zu groß würde, kann die Regelung zum Misserfolg führen.
Um das mathematisches Modell so exakt wie möglich zu erstellen, ist die Identifikation für solche ungenauen Parameter erforderlich. Die Identifikation dynamischer Systeme basiert auf der Modellbildung des Systems. Im abgebildeten Modell erhält man einige Parameter, welcher aus physikalischen Prozesskoeffizienten oder Grunddaten des Prozesses berechnet werden. Solche Parameter haben oft große Unsicherheit. Hierbei verwendet man gemessene Signale und ermittelt das zeitliche Verhalten des Systems. Um dann mit einer geeigneten Identifikationsmethode die ungenauen Parameter zu identifizieren, bis seine Unsicherheit akzeptierbar ist(siehe Isermann, 2008, S. 333).
Zur Systemidentifikation ist der erster Schritt der Aufbau eines mathematischen Modells. Dabei ist wichtig welche Eigenschaften die Analyse des Modells zeigt: Linearität(linear oder nichtlinear ?), Stabilität(stabil oder nicht stabil ?), Zeitverhalten(zeitvariant oder zeitinvariant) sowie wie groß der Beitrag von Parametern ist und welche wichtig zu identifizieren sind?(wichtig oder vernachlässig? Solche Eigenschaften sind die entscheidenden Bedingungen der Bestimmung der Identifikationsmethode.
Zur Bestimmung einer richtigen Identifikationsmethode gibt es viele mathematische Werkzeuge. Wie zum Beispiel: Mit dem "Least-Square-Verfahren"(siehe Ljung, 1998, S. 176-180)(siehe in Abbildung 1.4) kann einmalig alle Parameter einer Übertragungsfunktion eines Systems in zeitdiskreter Form ($G(z)$) ermittelt werden. Diese Methode ist geeignet für ein lineares und stabiles System. Für ein nichtlineares System sind ein paar Methoden vorhanden, z.B.: der extended-Kalman-Filter(EKF)(siehe Van Der Merwe, 2004, S. 35) oder der Sigma-Point-Kalamn-Filter(SPKF)(siehe Van Der Merwe, 2004, S. 49). Sie erachten den identifizierten Parameter als eine neue Zustandsgröße und präditiert nach der Messung des Systems seinen realen Wert(siehe Van Der Merwe, 2004, S. 43, Wan, Van Der Merwe u. a., 2000, Wenzel u. a., 2006)(siehe in Abbildung 1.5). Die Identifikation mit EKF oder SPKF kann nicht nur

nach der Messung den Parameter einmalig(offline) stattfinden, sondern auch den Parameter nach der Zeit(online) identifizieren. Das ist geeignet für einige Systeme, die zeitabhängige Parameter haben.

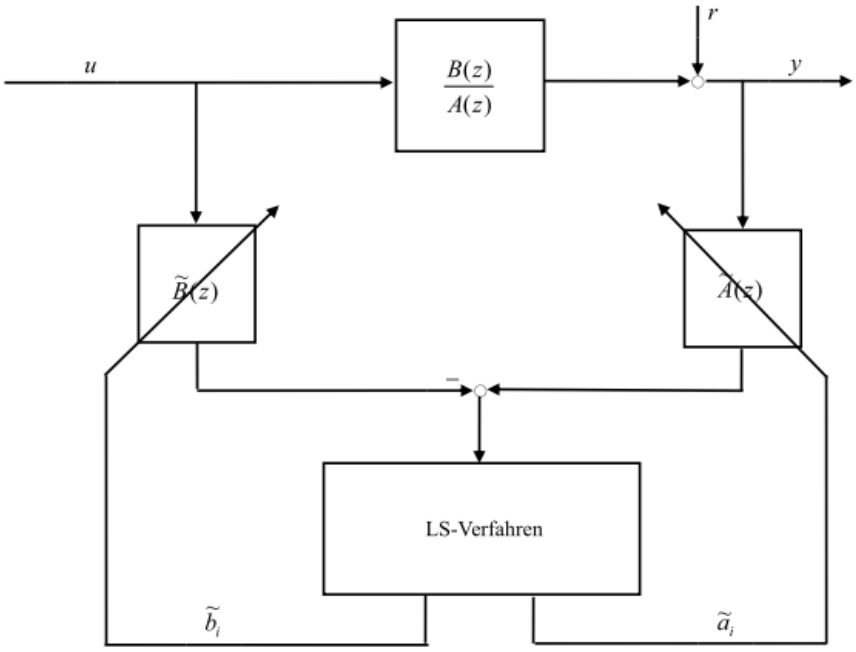


Abbildung 1.4: BSB von "LS" Verfahren

Ein System $G(z) = \frac{B(z)}{A(z)} = \frac{\sum_{j=1}^{m} b_j z^{-j}}{\sum_{i=1}^{n} a_i z^{-i}}$ ist in diskreter Form. Seiner Ausgang y ist unter weißem Rauschen r gestört. Das Ein- und Ausgangssignal ist in LS-Algorithmus eingegangen. Die Ergebnisse sind als korrigierte Parameter "$\hat{a}_0 \dots \hat{a}_n$" und "$\hat{b}_0 \dots \hat{b}_m$" vom System $G(z)$ zurückgegeben.

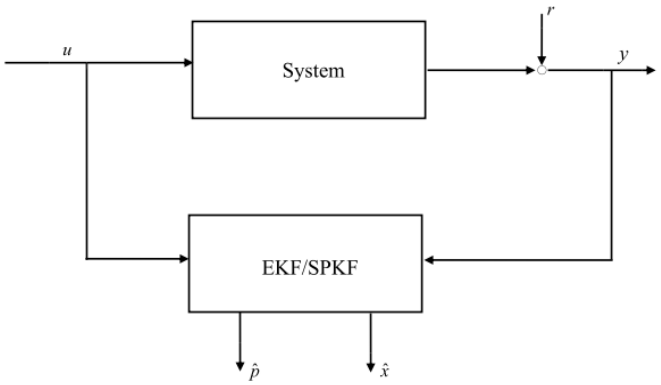


Abbildung 1.5: BSB von Systemidentifikation mit Kalman-Filter

Bei der Identifikation mit dem Kalman-Filter ist der Systemzustand durch das Einfügung von identifizierten Parameter erweitert. Das bedeutet für den neueren Zustand $x^* = [x \quad p]^T$. Der Kalman-Filter beschäftigt sich mit der Einschätzung dieses neuen Zustands, damit die Varianzmatrix vom System P gegen das Minimum konvergiert. Weil das erweiterte Systemmodell nichtlinear ist, soll hier EKF oder SPKF zum Einsatz gebracht werden.
Eine weitere Identifikationsmethode heißt "Expectation-Maximization" (EM) Methode(siehe Abbildung 1.6). Diese Methode versucht die Likelihood Funktion zu erreichen, die sich auf die identifiziertem Parameter bezieht. Der EM Algorithmus berechnet diese Likelihood Funktion unter der Umgebung von aktuellen aber ungenauen Parametern. Durch die Bestimmung der Extrawerte dieser Likelihood Funktion legt sie den wahrscheinlichsten Wert des Parameters fest.(Schön, 2009) Die EM Methode ist eine offline Methode, das bedeutet die Identifikation ist nach der Messung auszuführen. Seine Nutzung ist für die Anwendung der Kalman-Filter oder Partikel-Filter erforderlich.(Wills u. a., 2008)

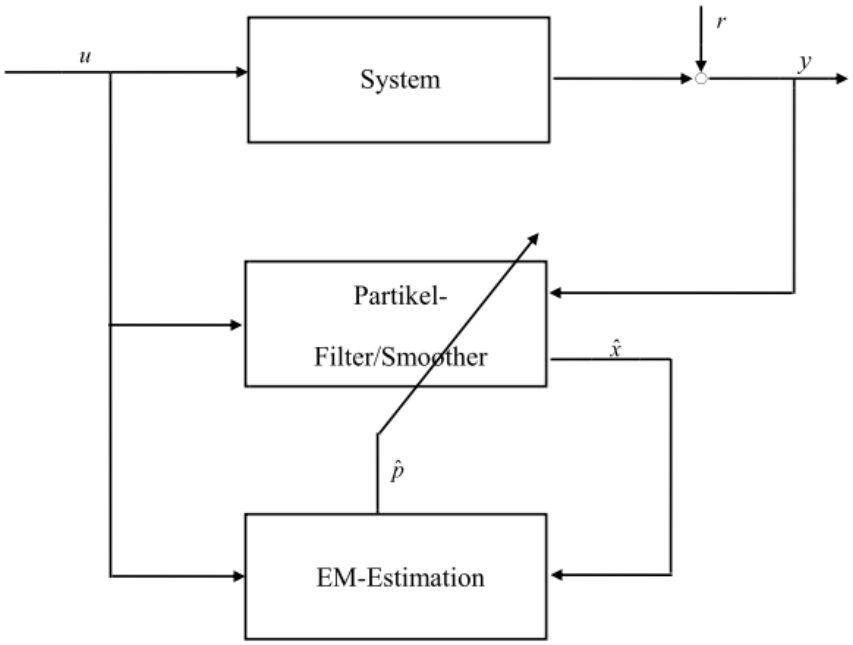


Abbildung 1.6: BSB von Systemidentifikation mit EM-Algorithmus

Bei der Systemidentifikation mit dem EM-Algorithmus schätzt der Partikel-Filter/Smoother den Systemzustand ein und der Systemzustand wird in stochastische Form übersetzt. Solch Systemzustand in stochastisch Form wird dann von EM-Algorithmus zum Aufbau der Likelihood Funktion eingebracht. Der Parameter wird mittels der Likelihood Funktion korrigiert und zurück zum Partikel-Filter/Smoother gegeben.
Die Zielstrecke (des Fahrzeugs) ist ein nichtlineares dynamisches System. Hierbei ist die

EM Methode als Identifikationsmethode ausgewählt um den Parameter zu identifizieren. Es gibt vier Parameter(Reifensteifigkeit von vorderem und hinterem Rad, der Quotient zwischen Masse und Trägheitsmoment, Abstand zwischen Schwerpunkt und vorderem Rad) die als unsichere Parameter dienen. Solche Parameter müssen zur Identifikation eingebracht werden. Umfang dieser Abschlussarbeit ist die:

- Modellierung des Fahrzeugsystems und Bestimmung der Vorgehensweise von Partikel-Filter/Smoother sowie EM-Algorithmus.
- Implementieren von Partikel-Filter/Smoother sowie EM-Algorithmus in das Fahrzeugmodell in Matlab-Code.
- Test des Effekts von EM-Identifikation bei Simulation. Führt nötige Einstellung aus.
- Parameter-Identifikation für reales Fahrzeug.

In folgenden Kapiteln wird für das Thema “Parameter-Identifikation mit EM Methode für Fahrzeugmodelle“ die bezüglichen Grundlagen, Performance und Ergebnisse von Simulation und Testfahrten stetig beschrieben und diskutieren.

2 Grundlagen

Die Identifikation für ein dynamisches Modell sind normalerweise in zwei Schritte durchzuführen. Der erste Schritt ist die Modellierung für die Zielstrecke, das abgebildetes Modell wird in den folgenden Schritt eingebracht. Der zweite Schritt ist die Parameter-Identifikation. Mit geeigneten Identifikationsmethoden werden Systemparameter in Verbindung mit Messung des Systems und Prädiktion des Systemmodells identifiziert.
In Rahmen der Parameter-Identifikation für Fahrzeugmodelle ist hier der EM-Algorithmus mit Partikel-Methode ausgewählt. Über die Partikel-Methode ist der Partikel-Filter/Smoother angewendet, er sagt die Zustandsgrößen nach der Messung und dem Systemmodell vorher und liefert dem EM-Algorithmus solche Zustandsgrößen im Form von diskretem Verteilungsfunktion. Der EM-Algorithmus übernimmt diese und bildet die sekundäre optimale Funktion ab um damit die "Likelihood (ML)" Funktion anzunähern. Der wahrscheinlichste Parameter tritt bei maximalem Wert dieser sekundären optimalen Funktion auf.(Wills u. a., 2008). Weil die Identifikation nicht einmalig erledigt werden kann, muss oberer Prozess mehrmalig durchgeführt werden.
Im diesem Kapital wird die Vorgehensweise und das Prinzip der EM-Identifikation auf theoretischer Ebene vorgestellt und erklärt.

2.1 Modellierung des Fahrzeugs

Das Fahrzeug wird als ein Einspurmodell angenommen, In Quer- und Längsrichtung ist das Fahrzeugmodell zu betrachten. Durch der Modellierung wird eine dynamische Gleichung abstrahiert, welche als Systemfunktion erachtet wird. Wegen der unbekannter Gewichtspostion wird der Mittelpunkt der hinteren Achse als Referenz-Punkt definiert. Die folgende Abbildung zeigt das Fahrzeugmodell in Querrichtung. Das Messsystem am Fahrzeug besteht

aus einer GPS-IMU-Kombination, die die Geschwindigkeit, Beschleunigung und Winkelgeschwindigkeit in drei Richtung(X, Y und Z), die Postion (Richtung in Norden und Osten) sowie den Azimut ermitteln kann.

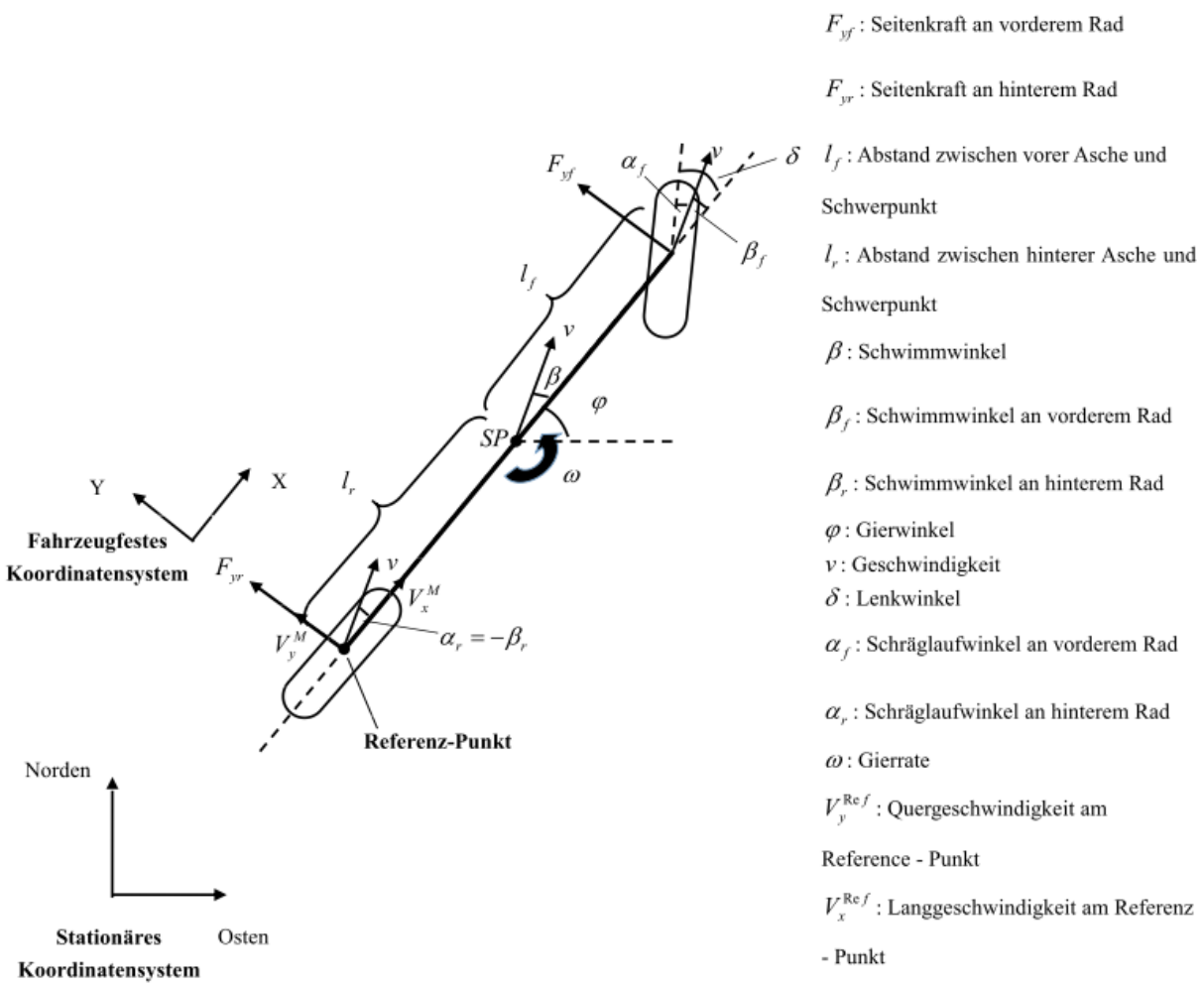


Abbildung 2.1: Fahrzeugmodell in Querrichtung

Die Modellierung beginnt mit der Dynamik der Gierrate, die Gierrate entsteht aus der Querkraft des Fahrzeugs. Die Gleichung ist:

$$\dot{\omega} = \frac{1}{J}\left(l_f F_{yf} - l_r F_{yr}\right) \tag{2.1}$$

Die vordere und hintere Seitenkraft sind:

$$F_{yf} = F_{zf}\alpha_f C_{sf} \tag{2.2}$$

$$F_{rf} = F_{zr}\alpha_r C_{sr} \tag{2.3}$$

Die Seitenkraft F_y eines Reifens hängt nichtlinear vom Schräglaufwinkel und senkrechter Kraft ab(siehe Pacejka, 2005, S. 1). Die Abbildung 2.2 zeigt ein typisches Diagramm der Seitenführungskraft in Abhängigkeit vom Schräglaufwinkel.

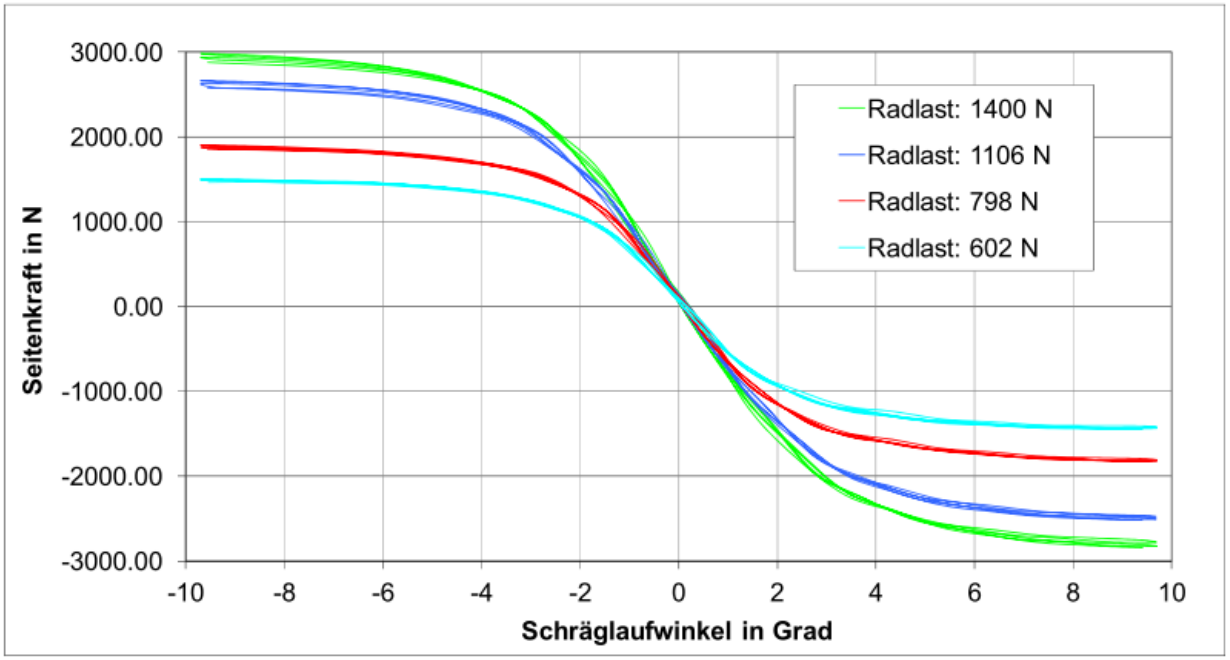


Abbildung 2.2: Seitenführungskraft in Abhängigkeit vom Schräglaufwinkel (Quelle: Continental)

Die obere Abbildung sagt aus: Die Darstellung weist eine Linearität auf, wenn der Schräglaufwinkel ein kleinen Wert besitzt. Diese Linearität dient als Führung vom Schräglaufwinkel und senkrechter Kraft zur Seitenkraft. Man definiert die Koeffizient als C_{sf} und C_{sr}, sie werden auch als Reifensteifigkeit benannt. Dieser Koeffizient kann die senkrechte Kraft und Schräglaufwinkel in den Seitenkraft umsetzen.

Der Kraft F_{zf} und F_{zr} sind die senkrechte Kraft vom vorderen und hinteren Rad, welche zwischen dem Reifen und Boden ist. α_f und α_r sind Schräglaufwinkel des Reifens, es ist

der Winkel zwischen der Achse des Reifens und Richtung der Geschwindigkeit. Der vordere Schräglaufwinkel ist die Differenz zwischen Lenkwinkel und Schwimmwinkel des Reifens. Der hintere Schräglaufwinkel ist als in Gegenrichtung des Schwimmwinkels des hinteren Reifens definiert. Die beiden Schräglaufwinkel betragen:

$$\alpha_f = \delta - \beta_f \tag{2.4}$$

$$\alpha_r = -\beta_r \tag{2.5}$$

Der Schwimmwinkel ist nicht direkt messbar, er ist aus der messbaren Lang- und Quergeschwindigkeit umzurechnen. Durch die Messung der Geschwindigkeit am Referenz-Punkt lautet der Schwimmwinkel des hinteren Reifens:

$$\beta_r = \arctan(\frac{V_y^{Ref}}{V_x^{Ref}}) \approx \frac{V_y^{Ref}}{V_x^{Ref}} \tag{2.6}$$

Wenn der Schwimmwinkel des hinteren Reifens bekannt ist, ist der Schwimmwinkel des vorderen Reifens mittels Länge des Fahrzeugs und Gierrate ermittelbar. Der lautet:

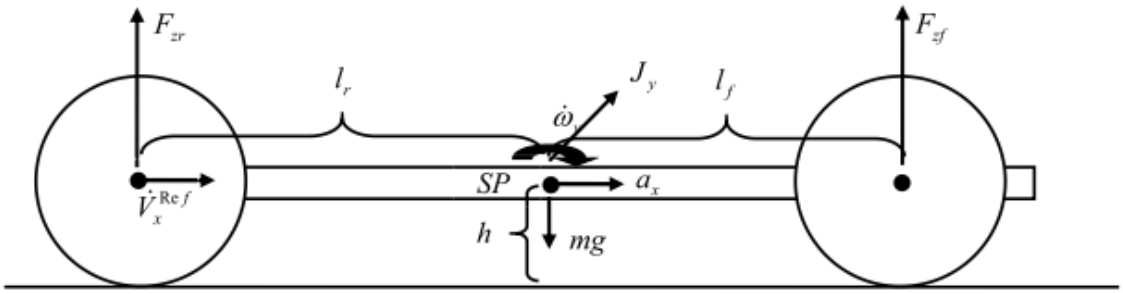
$$\beta_f = \arctan(\frac{V_y^{Ref} + L\omega}{V_x^{Ref}}) \approx \frac{V_y^{Ref} + L\omega}{V_x^{Ref}} \tag{2.7}$$

Nun sind die beiden Schräglaufwinkel:(siehe Rajamani, 2011, S. 29)

$$\alpha_f \approx \delta - \frac{V_y^{Ref} + L\omega}{V_x^{Ref}} \tag{2.8}$$

$$\alpha_r \approx -\frac{V_y^{Ref}}{V_x^{Ref}} \tag{2.9}$$

Das gesamte Gewicht des Fahrzeugs wird auf vier Reifen verteilt. Aber die Position des Schwerpunkts befindet sich nicht am Mittelpunkt der Längsachse. Deshalb sind die Verteilungen von Gewicht auf vorderem und hinterem Rad nicht gleich. Außerdem muss die Neigung des Fahrzeugs bei Fahrt mit beachtet werden, die Gewichtsverteilung ist bei Fahrt daher zeitvariant. Für die Längsrichtung des Fahrzeugmodells wird die folgende Abbildung erstellt:



h : Höhe des Schwerpunkts

mg : Schwerkraft

l_f : Abstand zwischen vorer Asche und Schwerpunkt

F_{zf} : Druckkraft an vorderem Rad

l_r : Abstand zwischen hinterer Asche und Schwerpunkt

F_{zr} : Druckkraft an hinterem Rad

a_x : Beschleunigung am Schwerpunkt in Längrichtung

$\dot{\omega}_y$: Winkelbeschleunigung der Neigung

$\dot{V}_x^{Ref}$: Beschleunigung am Referenz-Punkt in Längrichtung

J_y : Trägheitsmoment in Y-Richtung

Abbildung 2.3: Fahrzeugmodell in Längsrichtung

Nach der oberen Abbildung stellt sich das Kärftegleichgewicht wie folgt auf:

$$J_y \dot{\omega}_y = l_r F_{zr} - l_f F_{zf} - hma_x \tag{2.10}$$

Es ist bekannt, die Gewichtskraft des Fahrzeugs mit: $mg = F_{zf} + F_{zr}$. Die Winkelbeschleunigung $\dot{\omega}_y$ ist hierbei als 0 angenommen. Dann es ergibt sich:

$$F_{zf} = \frac{l_r mg - hma_x}{L} \tag{2.11}$$

$$F_{zr} = \frac{l_f mg + hma_x}{L} \tag{2.12}$$

Die Differentialgleichung der Gierrate ist nun hergeleitet über:

$$\dot{\omega} = \frac{C_{sf} l_f (l_r mg - hma_x)}{JL}(\delta - \frac{V_y^{Ref} + L\omega}{V_x^{Ref}}) + \frac{C_{sr} l_r (l_f mg + hma_x)}{JL} \frac{V_y^{Ref}}{V_x^{Ref}} \tag{2.13}$$

Nun werden die Längs- und Querbeschleunigung am Referenz-Punkt in Betracht gezogen. Die Beziehung der Beschleunigung am Schwerpunkt und am Referenz-Punkt sind in unteren stehender Formel zu sehen (für die Herleitung siehe in Anhang 1).

$$\dot{V}_x^{Ref} = \frac{1}{m}\sum F_x + \omega V_y^{Ref} + \omega^2 l_r \tag{2.14}$$

$$\dot{V}_y^{Ref} = \frac{1}{m}\sum F_y - \omega V_x^{Ref} - \dot{\omega} l_r \tag{2.15}$$

Weil die Formel $a_x = \frac{1}{m}\sum F_x$ existiert, wird die Gleichung 2.14 umgestellt in:

$$a_x = \dot{V}_x^{Ref} - \omega V_y^{Ref} - \omega^2 l_r \tag{2.16}$$

Mit der Formel 2.15, der Summe aller Seitenkräfte $\sum F_y$ wird die Querbeschleunigung am Schwerpunkt berechnet. Es ist die Summe von Seitenkraft an vorderen und hinteren Reifen. Das bedeutet: $\sum F_y = F_{zf}\alpha_f C_{sf} + F_{zr}\alpha_r C_{sr}$. Zuletzt ergibt sich über $\dot{V}_y^{Ref}$ die DGL:

$$\dot{V}_y^{Ref} = \frac{C_{sf}(l_r mg - hma_x)}{mL}(\delta - \frac{V_y^{Ref} + L\omega}{V_x^{Ref}}) - \frac{C_{sr}(l_f mg + hma_x)}{mL}\frac{V_y^{Ref}}{V_x^{Ref}} - \omega V_x^{Ref} - \dot{\omega} l_r \tag{2.17}$$

Der Gierwinkel ist das Integral der Gierrate, somit: $\dot{\varphi} = \omega$. Wird diese Formel mit den Formeln 2.13 ,2.16, 2.17 verbunden, entsteht:

$$a_x = \dot{V}_x^{Ref} - \omega V_y^{Ref} - \omega^2 l_r \tag{2.18}$$

$$\dot{\omega} = \frac{m}{J}\frac{C_{sf} l_f (l_r g - ha_x)}{L}(\delta - \frac{V_y^{Ref} + L\omega}{V_x^{Ref}}) + \frac{m}{J}\frac{C_{sr} l_r (l_f g + ha_x)}{L}\frac{V_y^{Ref}}{V_x^{Ref}} \tag{2.19}$$

$$\dot{V}_y^{Ref} = \frac{C_{sf}(l_r g - ha_x)}{L}(\delta - \frac{V_y^{Ref} + L\omega}{V_x^{Ref}}) - \frac{C_{sr}(l_f g + ha_x)}{L}\frac{V_y^{Ref}}{V_x^{Ref}} - \omega V_x^{Ref} - \dot{\omega} l_r \tag{2.20}$$

$$\dot{\varphi} = \omega \tag{2.21}$$

Soweit ist die Modellierung des Fahrzeugs fertiggestellt. Das nichtlineare Fahrzeugmodell kann als Funktion $\dot{x} = f(x,\ u,\ p)$ abstrahiert werden. x ist der Zustandsvektor, der als $x = \begin{bmatrix} V_y^{Ref} & \omega & \varphi \end{bmatrix}^T$ definiert wird. $u = \begin{bmatrix} \dot{V}_x^{Ref} & \delta & V_x^{Ref} \end{bmatrix}^T$ ist der Eingangsvektor, $p = [C_{sf} \quad C_{sr} \quad m/J \quad l_f]^T$ ist der Parametervektor, welcher den unbestimmten Parameter enthält. Diese Parameter werden im nächsten Schritt identifiziert.

Nach dem Aufbau des Testfahrzeugs ist bekannt, dass alle Zustandsgröße des Fahrzeugmodells messbar sind, deshalb lautet die Ausgangsgröße y:

$$y_t = c^T x_t \tag{2.22}$$

mit c^T:

$$c^T = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{2.23}$$

2.2 Parameter-Identifikation des dynamisches Systems mit EM-Algorithmus

Der EM-Algorithmus versucht die Likelihood Funktion eines probabilistischen Systems mit ungenauem Parameter anzunähern. (Schön, 2009) Ein System, welches mit der Funktion $x_{t+1} = f(x_t,\ u_t)$ und $y_t = h(x_t,\ u_t)$ beschrieben wird, hat den latenten Zustand x_t. Es ist unmöglich, die Likelihood Funktion eines Systems direkt zu bestimmen, wenn dieses System latente Zustandsgröße hat(Schön, 2009). Die EM-Algorithmus ist ein iterativer Prozess, der in jeder Iteration die sekundäre optimale Funktion auf der Basis von aktuellen ungenauen Parametern bestimmt. Diese sekundäre optimale Funktion ist ähnlich zur real Likelihood Funktion. In jeder Iteration wird der Parameter nach der ML identifiziert. Diese Iteration wird immer noch durchgeführt, bis der Parameter den realen Wert erreicht. Das bedeutet, der EM-Algorithmus unterteilt die Maximum Likelihood Methode in zwei Schritt: Berechnung der "Maximum Expection" von Likelihood Funktion und Identifizierung der Parameter durch Ermittelung des maximalen Werts von dieser Likelihood Funktion.(Schön, 2009)
Vor der Vorstellung des EM-Algorithmus ist es nötig eine kurze Einleitung über ML Methode vorzuführen.

2.2.1 Maximum Likelihood Methode

Für ein System mit dem Zustandsraum mit Parameter θ, Prozess- und Messrauschen w_t, v_t ist definiert:

$$x_{t+1} = f_\theta(x_t,\ u_t) + w_t \qquad w_t \sim N(0,\ \Xi) \tag{2.24}$$

$$y_t = h_\theta(x_t,\ u_t) + v_t \qquad v_t \sim N(0,\ \Theta) \tag{2.25}$$

Die Likelihood Funktion $L_\theta(x_1,\ x_2\ \dots\ x_t \dots\ x_T)$ ist definiert als eine Verteilungsfunktion, die die Verteilungsdichte von Zufallsvariablen $x_{1:T} = \{x_1,\ x_2\ \dots\ x_t\ \dots\ x_T\}$ unter Bedingung vom Parameter θ im Logarithmus beschreibt.(Krengel, 1988) Nach dieser Definition ist die Likelihood Funktion eines Systems die Likelihood Funktion von eine Messreihe $y_{1:T} = \{y_1,\ y_2\ \dots\ y_t \dots\ y_T\}$ unter der Beeinflussung von Parameter θ. So lautet die Likelihood Funktion vom System:

$$L_\theta(y_{1:T}) = \log(p_\theta(y_{1:T})) \tag{2.26}$$

Die Identifikation für Parameter θ ist der maximale Wert der Likelihood Funktion unter dem variierten Parameter θ, der Parameter bei maximalem Wert des Likelihood Funktion $\hat{\theta}^{ML}$ ist der wahrscheinlichste Wert. Dieses Verfahren lautet in mathematischer Formel:

$$\hat{\theta}^{ML} = arg \underbrace{max}_{\theta} L_{\theta}(y_{1:T}) \tag{2.27}$$

2.2.2 EM Algorithmus

Wenn die direkte Ermittelung der LM Funktion nicht möglich ist, kann durch der EM-Algorithmus mittels des aktuell ungenauen Parameters die wahrscheinlichste Funktion gestaltet werden. Es gibt zwei Schritten: In erster Schritte ist die Variable $x_{1:T}$ als latente Variable und $y_{1:T}$ als beobachtbare Variable definiert. Die Funktion $q(x_{1:T})$ ist eine beliebige Verteilungsfunktion für $x_{1:T}$. Die Formel gibt es unter: (siehe Särkkä, 2013, S. 182)

$$\log p_{\theta}(y_{1:T}) \geq F_{\theta}\left[q(x_{1:T})\right] \tag{2.28}$$

Die Funktion $q(x_{1:T})$ bildet die untere Grenze der ML Funktion ab, sie lautet:(siehe Särkkä, 2013, S. 183)

$$F_{\theta}\left[q(x_{1:T})\right] = \int q(x_{1:T}) \log \frac{p_{\theta}(x_{1:T}, y_{1:T})}{q(x_{1:T})} dx_{1:T} \tag{2.29}$$

Die Bestimmung des Maximums der Funktion $F_{\theta}\left[q(x_{1:T})\right]$ ist die Vorbereitung für nächste Schritte. Die Ermittelung des Maximums der Funktion $F_{\theta}\left[q(x_{1:T})\right]$ ist als erster Schritt "E-Step" der EM Algorithmus bezeichnet. Ist das Maximum der Funktion $F_{\theta}\left[q(x_{1:T})\right]$ als $q^{(m+1)}(x_{1:T})$ definiert, dann bringt diese $q^{(m+1)}(x_{1:T})$ wieder in die Funktion $F(q)$ ein. Die untere Grenze der Likelihood Funktion wird dadurch abgebildet und identifiziert den Parameter θ_{m+1} durch Bestimmung des Maximum von $F_{\theta}\left[q^{(m+1)}(x_{1:T})\right]$. Damit ist auch der zweiter Schritte "M-Step" abgeschlossen.
Bringt man den aktuell identifizierten Parameter θ_{m+1} in die nächste Iteration und führt die ´´E-Step´´ und ´´M-Step´´ noch einmal durch, so läuft die Iteration bis die Änderung des Parameter gering genug ist. Der EM-Algorithmus ist in folgenden abstrahiert:(siehe Särkkä, 2013, S. 182)

- Beginnen mit Initialisierungszustand: $q^{(0)}$, θ_0
- For $m = 0, 1, 2$
  - E-Step: $q^{(m+1)} = arg \underbrace{max}_{q^{(m)}} F_{\theta_m} \left[q^{(m)}\right]$
  - M-Step: $\theta_{m+1} = arg \underbrace{max}_{\theta} F_{\theta} \left[q^{(m+1)}\right]$

  End

Für E-Step ist schon bewiesen, dass die Funktion $F_{\theta_m}[q]$ maximal ist, wenn $q^{m+1}(x_{1:T}) = p_{\theta_m}(x_{1:T}|y_{1:T})$ ist.(siehe Särkkä, 2013, S. 182). Dann wird $p_{\theta_m}(x_{1:T}|y_{1:T})$ in den M-Step eingebracht. Es ergibt sich:

$$F_{\theta_m}[q] = \int p_{\theta_m}(x_{1:T}|y_{1:T}) \log \frac{p_\theta(x_{1:T},\ y_{1:T})}{p_{\theta_m}(x_{1:T}|y_{1:T})} dx_{1:T} \tag{2.30}$$

$$= \int p_{\theta_m}(x_{1:T}|y_{1:T}) \log p_\theta(x_{1:T},\ y_{1:T}) dx_{1:T} - \int p_{\theta_m}(x_{1:T}|y_{1:T}) \log p_{\theta_m}(x_{1:T}|y_{1:T}) dx_{1:T} \tag{2.31}$$

$$\propto \int p_{\theta_m}(x_{1:T}|y_{1:T}) \log p_\theta(x_{1:T},\ y_{1:T}) dx_{1:T} \tag{2.32}$$

Werden die oberen Ergebnisse als neue Funktion $Q(\theta,\ \theta_m)$ definiert, ändert sich auch der EM-Algorithmus zu:(siehe Särkkä, 2013, S. 183)

- Beginnen mit Initialisierungszustand: $q^{(0)}$, θ_0
- For $m = 0, 1, 2$
  - E-Step: Bestimmung von $Q(\theta,\ \theta_m)$
  - M-Step: $\theta_{m+1} = arg \underbrace{max}_{\theta} Q(\theta,\ \theta_m)$

  End

Die folgende Darstellung zeigt wie sich der Parameter θ mit zunehmender Iteration verhaltet.

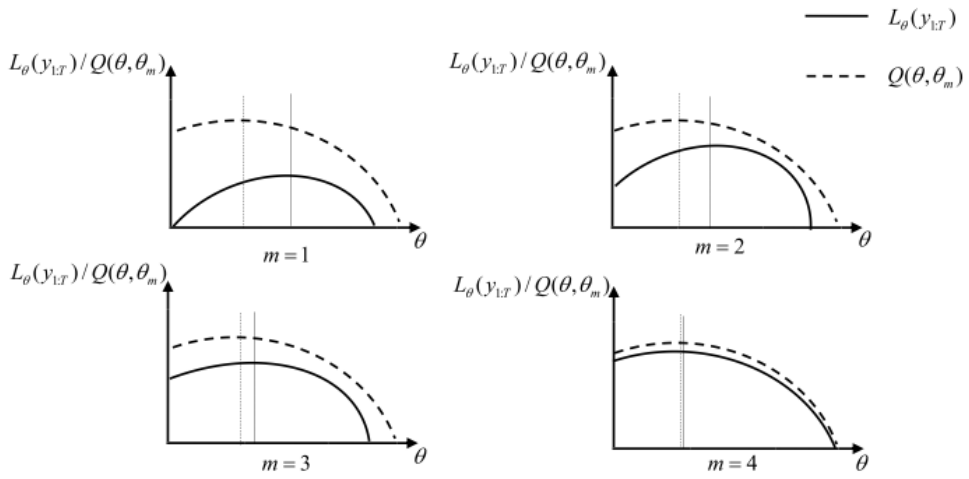


Abbildung 2.4: Funktionsweise des EM-Algorithmus

In oberer Darstellung ist zu sehen: In jeder Iteration versucht die Funktion $Q(\theta,\ \theta_m)$ die Likelihood-Funktion anzunähern. Bis zur 4ten Iteration hat die Funktion $Q(\theta,\ \theta_m)$ total angekommen. Die Parameter θ wird endlich bei realem Wert bestimmt.
Das Systemmodell mit Formel 2.24 und 2.25 ist ein typischer Markov-Prozess(siehe Krengel, 1988, S. 2). Für einen Markov-Prozess erfüllt der Likelihood Funktion die untere Gleichung(Herleitung siehe in Anhang 2):

$$p_\theta(y_{1:T}) = p_\theta(y_1)\prod_{t=2}^{T} p_\theta(y_t|y_{t-1}) \tag{2.33}$$

$$\Rightarrow L_\theta(y_{1:T}) = \log p_\theta(y_{1:T}) = \sum_{t=2}^{T} \log p_\theta(y_t|y_{t-1}) + \log p_\theta(y_1) \tag{2.34}$$

Zum Aufbau der Likelihood Funktion des Systemmodell sind nicht nur die beobachtbare Variable $y_{1:T}$sondern auch die latente Variable $x_{1:T}$ zu betrachten. Deshalb ist die Likelihood Funktion $L_\theta(x_{1:T},\ y_{1:T})$ gleich:

$$L_\theta(x_{1:T},\ y_{1:T}) = \log p_\theta(x_{1:T},\ y_{1:T}) = \log p_\theta(x_1) + \sum_{t=2}^{T} \log p_\theta(x_t|x_{t-1}) + \sum_{t=1}^{T} \log p_\theta(y_t|x_t) \tag{2.35}$$

Damit ist die Funktion $Q(\theta,\ \theta_m)$ dann:

$$Q(\theta,\ \theta_m) = I_1(\theta,\ \theta_m) + I_2(\theta,\ \theta_m) + I_3(\theta,\ \theta_m) \tag{2.36}$$

Wo:

$$I_1(\theta,\ \theta_m) = \int p_{\theta_m}(x_1|y_{1:T}) \log p_\theta(x_1) dx_1 \tag{2.37}$$

$$I_2(\theta,\ \theta_m) = \sum_{t=2}^{T} \int p_{\theta_m}(x_t|y_{1:T}) \log p_\theta(x_t|x_{t-1}) dx_{t-1} \tag{2.38}$$

$$I_3(\theta,\ \theta_m) = \sum_{t=1}^{T} \int p_{\theta_m}(x_t|y_{1:T}) \log p_\theta(y_t|x_t) dx_t \tag{2.39}$$

2.2.3 Berechnung der Funktion $Q(\theta,\ \theta_m)$ mit der Partikel Methode

Nun zeigt sich ein Problem: Wie können diese drei Terme der Funktion $Q(\theta,\ \theta_m)$: $I_1(\theta,\ \theta_m)$, $I_2(\theta,\ \theta_m)$, $I_3(\theta,\ \theta_m)$ berechnet werden? Hierbei muss die Verteilungsfunktion in diskreter Form umgesetzt werden (Erklärung siehe in Anhang 3). Diese Maßnahme ermöglicht die Approximation der Verteilung(siehe Särkkä, 2013, S. 119). Wie zum Beispiel:

$$p(x_t|y_{1:T}) \approx \sum_{i=1}^{N} \tilde{w}_t^i \delta(x_t - x_t^i) \tag{2.40}$$

Diese Formel beschreibt: wie durch zufällige Verteilung des gewichteten Partikels seine Verteilung approximiert werden kann. In oberer Formel sind die gewichteten Partikel aus Formel: $\{x_t^i,\ \tilde{w}_t^i\}$. N ist die gesamte Zahl der Partikel, x_t^i ist der Wert des Partikels am Zeitpunkt t, jeder Partikel hat einen normiertes Gewicht $\tilde{w}_t^i$ mit der Eigenschaft:

$$\left\{ \tilde{w}_t^i \mid \tilde{w}_t^i \in [0 \quad 1], \quad \sum_{i=1}^{N} \tilde{w}_t^i = 1 \right\} \tag{2.41}$$

Mit der Dirac-Funktion $\delta(x_t - x_t^i)$ entscheidet sich die Verteilungsdichte und mit welchem Wert sie auftritt.
Jetzt lautet der Berechnungsformel für die Funktion $Q(\theta,\ \theta_m)$:

$$Q(\theta,\ \theta_m) = I_1(\theta,\ \theta_m) + I_2(\theta,\ \theta_m) + I_3(\theta,\ \theta_m) \tag{2.42}$$

Wo:

$$I_1(\theta,\ \theta_m) \approx \sum_{i=1}^{N} \tilde{w}_1^i \log p_\theta(x_1^i) \tag{2.43}$$

$$I_2(\theta,\ \theta_m) \approx \sum_{t=2}^{T} \sum_{i=1}^{N} \tilde{w}_t^i \log p_\theta(x_t^i | x_{t-1}) \tag{2.44}$$

$$I_3(\theta,\ \theta_m) \approx \sum_{t=1}^{T} \sum_{i=1}^{N} \tilde{w}_t^i \log p_\theta(y_t | x_t^i) \tag{2.45}$$

2.2.4 Partikel Methode

Die Partikel Methode besteht aus zwei Teilen: Partikel-Filter und Partikel-Smoother. Der Partikel-Filter schätzt gemäß der gestörten Messgröße alle Zustandsgröße ein. Der Partikel-Smoother verbessert die Verteilung des Partikels, damit die Einschätzung nicht nur von vorheriger Messung sondern auch von nachheriger Messung abhängig ist. Alle Partikel $\{x_t^i,\ \tilde{w}_t^i\}$ von der ganzen Messreihe werden dann zur Berechnung der Funktion $Q(\theta,\ \theta_m)$ eingesetzt.

Partikel Filter

Viele Kalman-Filter für die Einschätzung des Zustands basieren auf der gaußschen Approximation, aber manchmal wenn die gaußsche Approximation für Filterung nicht geeignet ist, ist z.B: Partikel-Filter, der auf der "sequential importane reampling" basiert, eine Alternative.(siehe Särkkä, 2013, S. 116)

Zuerst wird angenommen, dass der Zustand des Systems der unteren Aussage folgt:

$$x_t \sim p(x_t | x_{t-1}) \tag{2.46}$$

$$y_t \sim p(y_t | x_t) \tag{2.47}$$

Diese Aussage definiert die aktuelle latente Zustandsgröße x_t bezüglich der letzten latenten Zustandsgröße x_{t-1}. Diese Beziehung existiert auch zwischen Messgröße y_t und latente Zustandsgröße x_t.

Die Funktionsweise des Partikel-Filters geht um eine grundlegende Formel(siehe Särkkä,

2013, S. 117):

$$E\left[g(x)|y_{1:T}\right] = \int g(x)p(x|y_{1:T})dx \tag{2.48}$$

Der x ist irgendeine Größe, die mit einer Wertreihe $y_{1:T}$ zu tun hat. Diese Formel beschreibt den Erwartungswert einer Funktion $g(x)$ unter der Bedingung von Wertreihe $y_{1:T}$. Damit ist das Integral das Produktion von der Funktion $g(x)$ und der hinterer Verteilungsfunktion $p(x|y_{1:T})$.

Hierbei ist für ein Systemmodell die Funktion $g(x)$ in Formel 2.45 von Systemfunktion $f(x,\ u)$ ersetzt, denn es gilt:

$$E\left[f(x_t,\ u_t)|y_{1:T}\right] = \int f(x_t,\ u_t)p(x_t|y_{1:T})dx_t \tag{2.49}$$

In diskreter Formel (Formel 2.38) gibt es:

$$E\left[f(x_t,\ u_t)|y_{1:T}\right] \approx \sum_{i=1}^{N} \tilde{w}_t^i f(x_t^i,\ u_t) \tag{2.50}$$

Mit der oberer Formel kann das Integral in die Berechnung von der Summe alle gewichteten Partikel umsetzen. Dies macht die Berechnung realisierbar.

Generell funktioniert der Partikel-Filter so: Am Anfang erzeugt der Pratikel-Filter eine Menge vom Partikeln mit dem gleichen Gewicht, durch die Systemfunktion $x_{t+1} = f(x_t,\ u_t)$ wird jeder Partikel für den nächsten Zeitpunkt berechnet($x_{t+1}^i = f(x_t^i,\ u_t)$). Gleichzeitig wird für jeden Partikel sein eigenes Gewicht $\tilde{w}_t^i$ nach der Abweichung zwischen$h(x_t^i,\ u_t)$ und y_t ermittelt. Es kann passieren, dass Partikel mit sehr geringem Gewicht mit zunehmender Zeit immer leichter werden. Solche Partikel sind allmählich sinnlos für die Berechnung. Zum Löschen dieser Partikel dient das Resampling-Verfahren. Das Resampling-Verfahren ermöglicht die Filterung von unwichtigem Partikel und stellt aufgrund der verbleibend Partikel die neuen Partikel. Die generell Funktionsweise des Partikel-Filters ist in folgender Abbildung gezeigt.

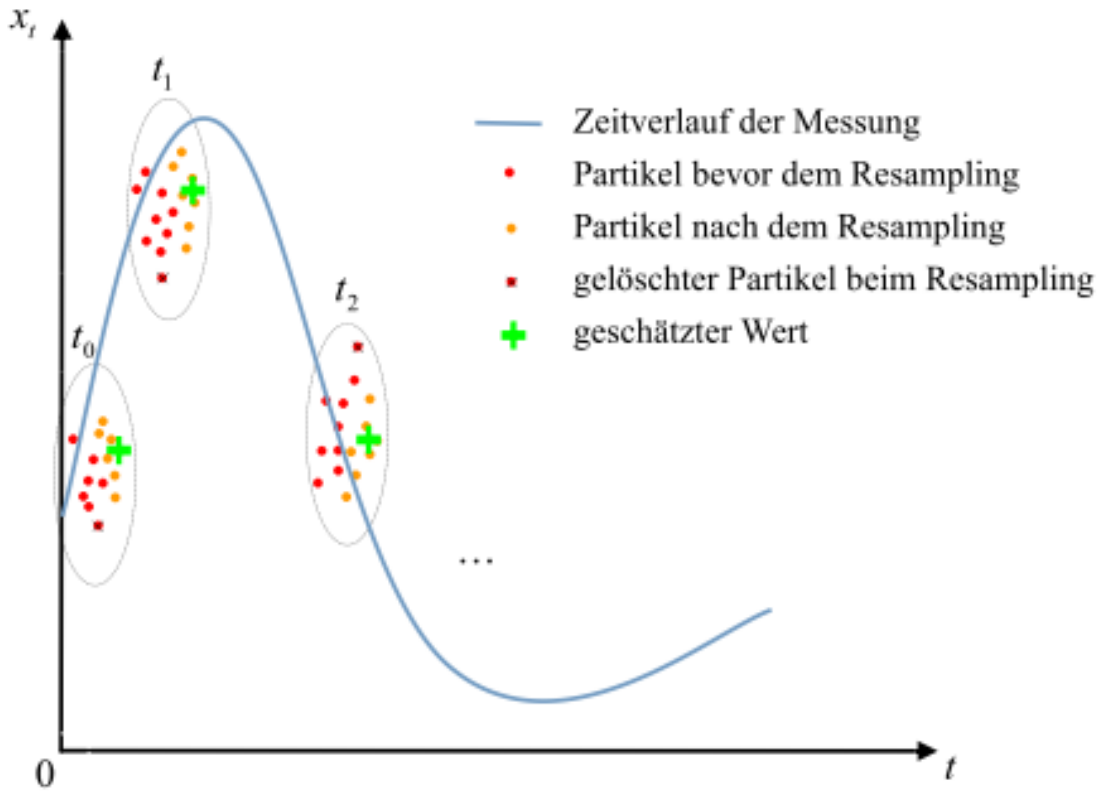


Abbildung 2.5: Funktionsweise des Partikel-Filters

Die obere Abbildung zeigt die Funktionsweise des Partikel-Filters. Jeder Partikel wird in die Systemfuntkion $x_{t+1}^i = f(x_t^i,\ u_t)$ gebracht. Danach wird das Resampling ausgeführt. Der neue Schätzwert ist der Mittelwert der Partikel nach dem Resampling.
Der Partikel-Filter kann nur vorwärts ausgeführt werden, nach der Struktur des Markov-Prozess ist die Messung nach der aktuelle Zeitpunkt t für x_t sinnlos. Deshalb steht die Verteilungsfunktion $p(x_t|y_{1:T})$ gleichwertig in $p(x_t|y_{1:t})$. Dann ergibt sich:

$$p(x_t|y_{1:t}) \approx \sum_{i=1}^{N} \tilde{w}_t^i \delta(x_t - x_t^i) \tag{2.51}$$

Doch wie kann das normierte Gewicht ermittelt werden? Vor der Herleitung der Berechnungsformel von $\tilde{w}_t^i$, wird hier der Begriff "importance distribution" eingebracht. Es wird bezeichnet als: "$\pi(x_t|y_{1:t})$". Darum erfüllt die Verteilung der Partikel $x_t^i : x_{t|T}^i \sim \pi(x_t|y_{1:t})$

Die verkettete Verteilungsdichte $p(x_{1:t}|y_{1:t})$ kann so umgeformt werden:

$$p(x_{1:t}|y_{1:t}) = p(x_{1:t}|y_t,\ y_{1:t-1}) = \frac{p(y_t|x_{1:t},\ y_{1:t-1})p(x_{1:t}|y_{1:t-1})}{p(y_t|y_{1:t-1}} \tag{2.52}$$

$$\propto p(y_t|x_{1:t},\ y_{1:t-1})p(x_{1:t}|y_{1:t-1}) \tag{2.53}$$

$$= p(y_t|x_t)p(x_t|x_{1:t-1},\ y_{1:t-1})p(x_{1:t}|y_{1:t-1}) \tag{2.54}$$

Nach der Eigenschaft der Markov-Kette ist : $p(x_t|x_{1:t-1},\ y_{1:t-1}) = p(x_t|x_{t-1})$ selbstverständlich. Es gilt weiter:

$$p(x_{1:t}|y_{1:t}) \propto p(y_t|x_t)p(x_t|x_{t-1})p(x_{1:t}|y_{1:t-1}) \tag{2.55}$$

Jetzt gibt es für das Gewicht jedes einzelnen Partikels die Gleichung:

$$w_t^i \propto \frac{p(x_{1:t}^i|y_{1:t})}{\pi(x_{1:t}^i|y_{1:t})} \tag{2.56}$$

$$= \frac{p(y_t|x_t^i)p(x_t^i|x_{t-1}^i)}{\pi(x_t^i|x_{1:t-1}^i,\ y_{1:t})} \cdot \underbrace{\frac{p(x_{1:t-1}^i|y_{1:t-1})}{\pi(x_{1:t-1}^i|y_{1:t-1})}}_{w_{t-1}^i} \tag{2.57}$$

Hier ist $\pi(x_{1:t-1}^i|y_{1:t-1})$ als $p(x_t^i|x_{1:t-1}^i)$ definiert(siehe Särkkä, 2013, S. 125). Das alte Gewicht w_{t-1}^i ist deswegen mit dem Resampling gleich $\frac{1}{N}$. Damit beträgt das Gewicht:

$$w_t^i = p(y_t|x_t^i) \tag{2.58}$$

Hier beschreibt die Gaußverteilung die Verteilungsdichte $p(y_t|x_t^i)$, das bedeutet:

$$p(y_t|x_t^i) = N(y_t; h(x_t^i,\ u_t),\ \Theta) \tag{2.59}$$

$$= \frac{1}{\sqrt{2\pi\Theta}} exp\left[-\frac{(y_t - h(x_t^i,\ u_t) - v_t)^2}{2\Theta}\right] \tag{2.60}$$

Der Symbol R ist die Varianz des Messrauschens v_t. So lautet w_t^i:

$$w_t^i = N(y_t; h(x_t^i,\ u_t),\ \Theta) \tag{2.61}$$

Normiert dieses Gewicht:

$$\tilde{w}_t^i = \frac{w_t^i}{\sum_{i=1}^N w_t^i} \tag{2.62}$$

Das Resampling-Verfahren ermöglicht die Filterung der unwichtigen Partikel. Die neue Menge den Partikel wird von verbleibenden Partikel erzeugt und jedem neuen Partikel wird das Gewicht $\frac{1}{N}$ zugeteilt. Für das Resampling stehen viele Algorithmen wie z.B.: "Multinomial"; "Stratified" oder "Residual" zur Verfügung(Lite). Eine detaillierte Vorstellung ist in Anhang 4 zu sehen.
Jetzt wird der Ablauf des Partikel-Filters abstrahiert (Sileshi u. a., 2013). Hier werden die Partikel vor und nach dem Resampling separat als: $\left\{x_{t-1|t}^i \quad \tilde{w}_{t-1|t}^i\right\}_{i=1}^N$ und $\left\{x_{t|t}^i \quad \tilde{w}_{t|t}^i\right\}_{i=1}^N$ bezeichnet. Zusätzlich sind prädiktierte Zustandsgröße vom Partikel-Filter mit der Symbol $\bar{x}$ gezeigt. Der Prozess lautet(siehe Särkkä, 2013, S. 123):

- Initialisierung des Partikels $\left\{x_{0|0}^i \quad \tilde{w}_{0|0}^i\right\}_{i=1}^N$ mit $\tilde{w}_{0|0}^i = \frac{1}{N}$
- For $t = 1, 2 \ldots T$
  - For $i = 1, 2 \ldots N$
    - Prädiktion neuer Partikel: $x_{t-1|t}^i = f(x_{t-1|t-1}^i,\ u_{t-1}) + w_t$
    - Bestimmung des Gewichts: $w_{t-1|t}^i = p(y_t | x_{t-1|t}^i)$

    End
  - Normierung des Gewichts: $\tilde{w}_{t-1|t}^i = \frac{w_{t-1|t}^i}{\sum_{i=1}^N w_{t-1|t}^i}$
  - Resampling: $\left\{x_{t|t}^i \quad \tilde{w}_{t|t}^i\right\}$ mit $\tilde{w}_{t|t}^i = \frac{1}{N}$
  - Einschätzung der Zustandsgröße: $\bar{x}_t = \frac{1}{N} \sum_{i=1}^N x_{t|t}^i$

  End

Partikel Smoother

So weit es vom Partikel-Filter ausgekommt, ist die Prädiktion der Zustandsgröße nur auf der Messung bis zum aktuellen Schritt: $y_{1:t}$ basiert. Aber wenn die Prädiktion auf der ganzen Messung $y_{1:T}$ notwendig ist, braucht es hier einen Smoother.(siehe Särkkä, 2013, S. 165) Hier ist der "Reweighting particle smoother" angewendet. Der "Reweighting particle smoother" ist auch als "marginal particle smoother" benannt. Es berechnet das Gewicht von einzelnen

Partikeln auf Basis vom Gewicht, welches vom Partikel-Filter angegeben wird. Das bedeutet, der Partikel Smoother startet das "Smoothing" nach dem Durchlauf des Partikel-Filters und es führt die Prädiktion rückwärts aus.
Wie gesagt, der Smoother basiert auf der ganzen Messung, das heißt: Statt der Verteilungsdichte $p(x_t|y_{1:t})$ ist $p(x_t|y_{1:T})$ betrachtet. In diskreter Approximation beträgt $p(x_t|y_{1:T})$:

$$p(x_t|y_{1:T}) \approx \sum_{i=1}^{N} \tilde{w}_{t|T}^{i} \delta(x_t - x_t^i) \tag{2.63}$$

Das Ziel ist die Bestimmung des normierten Gewichts vom Smoother $\tilde{w}_{t|T}^{i}$. Diese Herleitung beginnt mit Bayes-Smoothing-Gleichung(siehe Särkkä, 2013, S. 135). Die Lautet:

$$p(x_{t+1}|y_{1:t}) = \int p(x_{t+1}|x_t) p(x_t|y_{1:t}) dx_t \tag{2.64}$$

$$p(x_t|y_{1:T}) = p(x_t|y_{1:t}) \cdot \int \frac{p(x_{t+1}|x_t) p(x_{t+1}|y_{1:T})}{p(x_{t+1}|y_{1:t})} dx_{t+1} \tag{2.65}$$

Mit der Annahme:

$$p(x_{t+1}|y_{1:T}) \propto \sum_{i=1}^{N} \tilde{w}_{t+1|T}^{i} \delta(x_{t+1} - x_{t+1}^i) \tag{2.66}$$

$$p(x_{t+1}|y_{1:t}) \propto \sum_{j=1}^{N} \tilde{w}_{t+1}^{j} p(x_{t+1}|x_t^j) \tag{2.67}$$

$$p(x_t|y_{1:t}) \propto \sum_{l=1}^{N} \tilde{w}_{t}^{l} \delta(x_t - x_t^l) \tag{2.68}$$

Es gilt:

$$p(x_t|y_{1:T}) = p(x_t|y_{1:t}) \cdot \int \frac{p(x_{t+1}|x_t)p(x_{t+1}|y_{1:T})}{p(x_{t+1}|y_{1:t})} dx_{t+1} \tag{2.69}$$

$$\approx p(x_t|y_{1:t}) \cdot \int \frac{p(x_{t+1}|x_t)}{p(x_{t+1}|y_{1:t})} \sum_{i=1}^{N} \left[\tilde{w}_{t+1|T}^i \delta(x_{t+1} - x_{t+1}^i)\right] dx_{t+1} \tag{2.70}$$

$$= p(x_t|y_{1:t}) \cdot \sum_{i=1}^{N} \frac{p(x_{t+1}^i|x_t)}{p(x_{t+1}^i|y_{1:t})} \tag{2.71}$$

$$\approx p(x_t|y_{1:t}) \cdot \sum_{i=1}^{N} \tilde{w}_{t+1|T}^i \frac{p(x_{t+1}^i|x_t)}{\sum_{j=1}^{N} \tilde{w}_t^j p(x_{t+1}^i|x_t^j)} \tag{2.72}$$

$$\approx \sum_{l=1}^{N} \tilde{w}_t^l \delta(x_t - x_t^l) \cdot \sum_{i=1}^{N} \tilde{w}_{t+1|T}^i \frac{p(x_{t+1}^i|x_t)}{\sum_{j=1}^{N} \tilde{w}_t^j p(x_{t+1}^i|x_t^j)} \tag{2.73}$$

Daraus ergibt sich:

$$w_{t|T}^l = \sum_{i=1}^{N} \tilde{w}_{t+1|T}^i \frac{\tilde{w}_t^l p(x_{t+1}^i|x_t^l)}{\sum_{j=1}^{N} \tilde{w}_t^j p(x_{t+1}^i|x_t^j)} \tag{2.74}$$

Normierung:

$$\tilde{w}_{t|T}^l = \frac{w_{t|T}^l}{\sum_{l=1}^{N} w_{t|T}^l} \tag{2.75}$$

Der Prozess vom Partikel-Smoother lautet(siehe Särkkä, 2013, S. 167):

- Startet der Partikel-Filter und speichert alle Partikel mit Gewicht ergibt sich: $\left\{x_{t|t}^i \quad \tilde{w}_{t|t}^i\right\}$
- Dies initialisiert die Partikel für das Smoothing $\left\{x_{T|T}^i \quad \tilde{w}_{T|T}^i\right\}$ am Endzeitpunkt T mit $\tilde{w}_{T|T}^i = \frac{1}{N}$
- For $t = 1, 2 \ldots T$
  - For $l = 1, 2 \ldots N$
    - Bestimmung des Gewichts: $w_{t+1|T}^l = \sum_{i=1}^{N} \tilde{w}_{t+1|T}^i \frac{\tilde{w}_t^i p(x_{t+1}^i|x_t^l)}{\sum_{j=1}^{N} \tilde{w}_t^j p(x_{t+1}^i|x_t^j)}$

  End

- Normierung des Gewichts: $\tilde{w}^l_{t|T} = \frac{w^l_{t|T}}{\sum_{l=1}^N w^l_{t|T}}$
- Resampling: $\left\{x^i_{t|T} \quad \tilde{w}^i_{t|T}\right\}$ mit $\tilde{w}^i_{t|T} = \frac{1}{N}$
- Einschätzung der Zustandsgröße: $\bar{x}_{t|T} = \frac{1}{N}\sum_{i=1}^N x^i_{t|T}$

End

2.2.5 Zusammenfassung von Parameter-Identifikation mit EM-Algorithmus und Partikel-Methode

Nach den gegebenen Partikeln mit Gewicht die sich aus dem Partikel-Smoother ergeben, ist die Formel (2.43)-(2.45) umgesetzt in:

$$I_1(\theta,\ \theta_m) \approx \sum_{i=1}^{N} \tilde{w}^i_{0|T} \log p_\theta(x^i_0) \tag{2.76}$$

$$I_2(\theta,\ \theta_m) \approx \sum_{t=1}^{T-1}\sum_{l=1}^{N}\sum_{i=1}^{N} \underbrace{\frac{\tilde{w}^i_{t+1|T}}{\sum_{j=1}^N \tilde{w}^j_t p_{\theta_m}(x^i_{t+1}|x^j_t)}}_{\varphi^i_t} \tilde{w}^l_t p_{\theta_m}(x^i_{t+1}|x^l_t) \log p_\theta(x^i_{t+1}|x^l_t) \tag{2.77}$$

$$I_3(\theta,\ \theta_m) \approx \sum_{t=1}^{T}\sum_{i=1}^{N} \tilde{w}^i_{t|T} \log p_\theta(y_t|x^i_t) \tag{2.78}$$

Zur Vereinfachung der Funktion $I_2(\theta,\ \theta_m)$ wird hier der Term $\frac{\tilde{w}^i_{t+1|T}}{\sum_{j=1}^N \tilde{w}^j_t p_{\theta_m}(x^i_{t+1}|x^j_t)}$ mit dem Symbol φ^i_t ersetzt.
So weit ist der EM-Algorithmus mit der Partikel Methode zu abstrahieren:
Ein gegebenes Systemmodell, welches Formel 2.24, 2.25 aufweist, hat den Parameter θ zur Identifikation. Hier ist der Parameterwert in der jeweiligen Iteration als $\theta_m$bezeichnet. So ist am Anfang der Parameter θ_0, für den Ablauf lautet die ganze Identifikation:

- Modellierung des dynamischen Systems und Bestimmung der Systemfunktion
- Initialisieren den Parameter θ_0 und der Varianz von System- und Messrauschen.
- For $m = 1, 2...$
  - Starten des Partikel-Filters für das Modell mit Parameter θ_m und dann speichern alle Partikel $\left\{x^i_{t|t} \quad \tilde{w}^i_{t|t}\right\}_{i=1}^N$

- Starten der Partikel-Smoothers für das Modell mit Parameter θ_m und den Partikel vom Partikel-Filter $\left\{x^i_{t|t} \quad \tilde{w}^i_{t|t}\right\}_{i=1}^{N}$. Dann speichern alle Partikel vom Smoother: $\left\{x^i_{t|T} \quad \tilde{w}^i_{t|T}\right\}_{i=1}^{N}$
- Mit dem Partikel $\left\{x^i_{t|T} \quad \tilde{w}^i_{t|T}\right\}_{i=1}^{N}$ ermittelt der Funktion $Q(\theta,\ \theta_m)$
- $\theta_{m+1} = arg\ \underbrace{max}_{\theta} Q(\theta,\ \theta_m)$

End

2.3 Fazit

Soweit ist die Vorstellung der theoretischen Kenntnisse über Parameter-Identifikation mit EM-Algorithmus abgeschlossen. Das Fahrzeug wird in Längs- und Querrichtung modelliert und dient als Systemmodell. Vier Parameter bei Fahrzeug, welche nicht genau bestimmt werden können, werden zur Identifikation eingebracht. Die EM-Algorithmus versucht unter aktuellen ungenauen Parameter die Likelihood Funktion so weit wie möglich anzunähern. Hierfür wird für den Aufbau der Verteilungsfunktion zwischen Messgröße und Zustandsgröße die Partikel-Methode eingesetzt. Diese Verteilungsfunktion ist durch viele gewichtete Partikel in diskreter Form dargestellt. Die Partikel-Methode schätzt mit dem Partikel-Filter/Smoother die Zustandsgrößen ein. Dabei wird die Verteilungsfunktion des Zustands in diskreter Form zur Identifikation mit EM-Algorithmus bereitgestellt. Im nächsten Kapital wird die Performance der EM-Identifikation für das Fahrzeug demonstriert.

3 Hauptteil

Um die ungenauen Parameter vom Fahrzeugsystem zu identifizieren wird der komplexe EM-Algorithmus implementiert. Nach dem abgebildeten Fahrzeugmodell sieht das Fahrzeug die Zielstrecke. Der EM-Algorithmus erfasst die Prädiktion vom Modell und die Ausgangsgröße vom Modell aus der Simulation oder Messung. Damit die Parameter identifiziert werden können, muss die Parameter-Identifikation mit dem EM-Algorithmus in Matlab-Code implementiert werden. In Hauptteil dieses Kapitels wird die Ausführung der Parameter-Identifikation mit dem EM-Algorithmus der Simulation und realer Messungen demonstriert.

3.1 Implementierung der EM-Identifikation in Matlab

Die Parameter-Identifikation ist in einem Matlab-Skript programmiert. Im Skript wird die Ausgangsgröße des Systems aus der Simulation und realen Messwerten herstellt. Dann initialisiert und startet die Parameter-Identifikation die einzelnen Schritte der Identifikation wie: Partikel-Filter, Partikel-Smoother und EM-Algorithmus, die in Matlab als Funktion gestaltet sind. Die Abbildung 3.1 zeigt die Programm-Architektur für die Parameter-Identifikation:

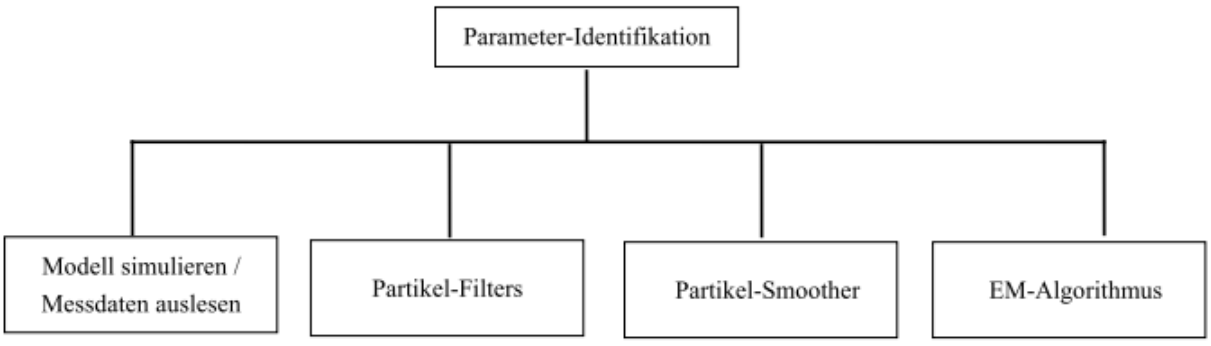


Abbildung 3.1: Programm-Architektur für Parameter-Identifikation

Die Programm-Architektur für die Parameter-Identifikation besteht aus vier Blöcken, der jeweilige Block ist relativ selbstständig zu anderen Blöcken. In Folgenden wird jeder einzelne

Block vorgestellt.

Abbildung der Simulation des Fahrzeugmodells

Zum Test, ob der Identifikation funktioniert, kann durch der Simulation die virtuellen Messwerte herstellt werden. Um damit die Parameter aus der Identifikation mit vorgegebenen Parameter der Simulation zu vergleichen. Das Prinzip der Simulation ist die Lösung der DGL vom Fahrzeugmodell mit den hergeleiteten Formeln 2.18 - 2.21. Dadurch ist die DGL des Fahrzeugmodells begründet. Die DGL ist als eine Matlab Funktion "`dx=vmodel_nolinear(x, u, p)`" gestaltet. Hier entsprechen die Eingangsparameter "x", "u", "p" separat dem Zustandsvektor, Eingangsvektor und Parametervektor. Zur Lösung der DGL stehen viele Algorithmus wie "Euler", "Heun" oder "Runge-Kutta" zur Verfügung. Es ist in diesem Fall das "Runge-Kutta " Verfahren(Vorstellung siehe in Anhang 5) angewendet. Für die Realisierung vom "Runge-Kutta " Verfahren wird die Matlab Funktion "`[t, x] = runge_kutta4_vmodel(fun_handle, tstart, h, n, x0, u, p, err)`" erstellt, für die genaue Beschreibung siehe unter:

- Ziel: Lösen der DGL vom Fahrzeugmodell mit "Runge-Kutta " Verfahren
- Eingang:
  1. **fun_handle:** Handle der Systemfunktion
  2. **tstart:** Anfangszeit
  3. **h:** gesamte Länge der Zeitspanne
  4. **n:** Zeitintervall
  5. **x0:** Anfangszustand
  6. **u:** Eingangsvektor in ganzer Zeitspanne
  7. **p:** Parametervektor
  8. **err:** Prozessrauschen in ganzer Zeitspanne
- Ausgang:
  1. **t:** Zeitspanne
  2. **x:** Zustandsgröße in ganzer Zeitspanne

Nach den Formeln 2.22 und 2.23 gibt das System die Ausgangsgrößen aus. Die komplexe Simulation wird in Matlab-Code geschrieben.
Statt der Simulation sind für reale Parameter Messdaten vom Testfahrzeug auszulesen. Dieses Verfahren wird in Kapital 3.3 vorgestellt

Partikel-Filter

Zur Realisierung der Funktion vom Partikel-Filters ist dafür die Matlab Funktion erstellt. Die Funktion "`[x_out, x_Partikel, Partikel_w, t]=PartikelFilter(fun_handle,C_T, Messwert,input,t_span,x0,N,V,R,Q,para)`" liefert den geschätzten Zustand des Systems sowie alle Partikel der ganzen Zeitdauer mit ihrem Gewicht. Die Beschreibung dieser Funktion siehe:

- Ziel: Realisierung des Partikel-Filters
- Eingang:
  1. **fun_handle:** Handle der Systemfunktion
  2. **C_T:** Ausgangsmatrix
  3. **Messwert:** gemessene Ausgangsgrößen der Zielstrecke
  4. **input:** Eingangssignal
  5. **t_span:** Zeitspanne der Messung
  6. **x0:** Anfangszustand
  7. **N:** Anzahl des Partikels
  8. **V:** Varianz der Verteilung des Partikels am Anfang
  9. **R:** Varianz von Messrauschen
  10. **Q:** Varianz von Prozessrauschen
  11. **para:** Parametervektor $[C_{sf} \quad C_{sr} \quad m/J \quad l_f]^T$
- Ausgang:
  1. **x_out:** geschätzte Zustandsgröße
  2. **x_Partikel:** alle Partikel in ganzer Zeitspanne, hier wird der Partikel vor und nach dem Resampling für jeden Zeitpunkt gespeichert.
  3. **Partikel_w:** Gewicht vom jeweiligen Partikel vor dem Resampling
  4. **t:** Zeitspanne

Das "Stratified" Verfahren ist hierfür als das Resampling-Verfahren ausgewählt. Das Flussdiagramm der Funktion ist folgender Abbildung zu sehen.

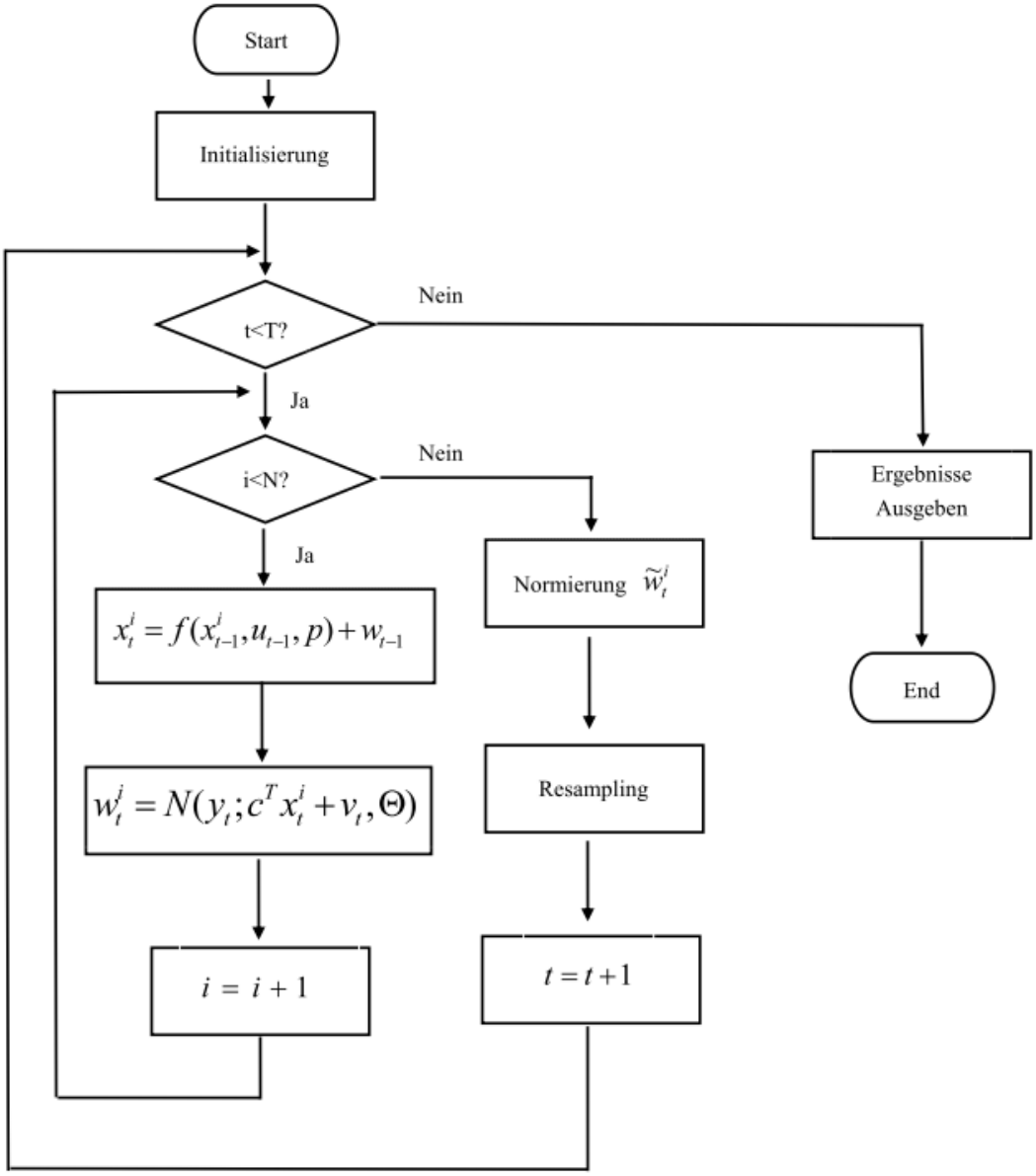


Abbildung 3.2: Flussdiagramm für Partikel-Filter

Partikel-Smoother

Der Partikel-Smoother übernimmt die Partikel aus dem Partikel-Filter und schätzt den Zustand x_t des System nach $p(x_t|y_{1:T})$ ab. Die Berechnungsformel ist Formel 2.74, zur Realisierung im Programm gibt es ein Problem: Wegen des Resamplings beim Partikel-Filter hat die Reihenfolge nach dem Resampling eine neu Anordnung(siehe Abbildung 3.3).

Die obere Abbildung zeigt wie der Partikel im Partikel-Filter und -Smoother transportiert wird. Hier bezeichnet man den Partikel in Partikel-Filter vor und nach dem Resampling

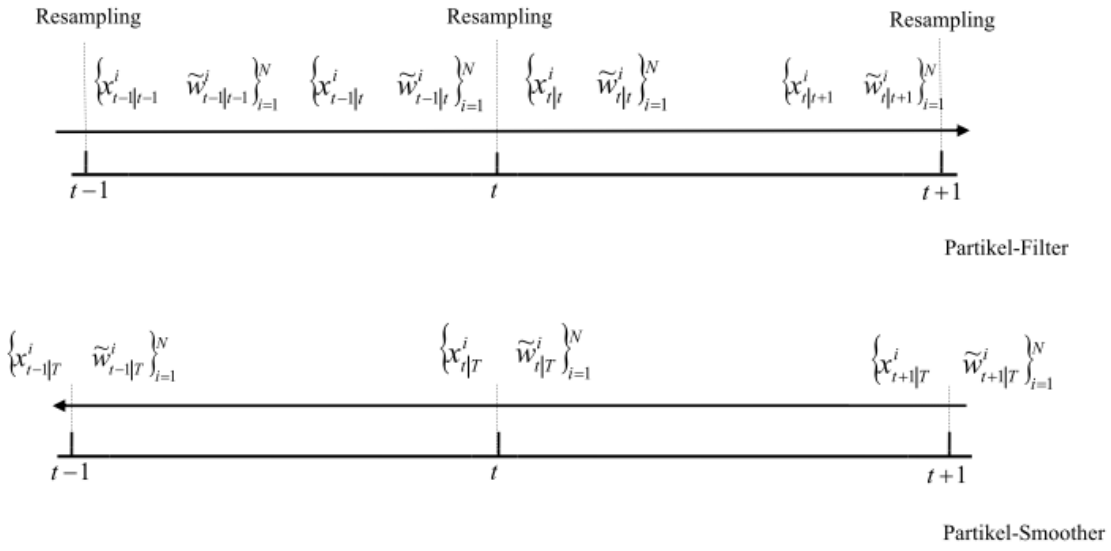


Abbildung 3.3: Partikel im Partikel-Filter und -Smoother

separat als $x^i_{t-1|t}$ und $x^i_{t|t}$. In der Formel (2.74) sind die Variable x^l_t und x^j_t separat mit $x^l_{t|t}$ und $x^j_{t|t}$ ersetzt. Zweifellos sind der zugehörigen Gewichte "$\tilde{w}^l_{t|t}$" und "$\tilde{w}^j_{t|t}$" gleich $\frac{1}{N}$. Dann wird die Formel (2.74) weitergeleitet in:

$$w^l_{t|T} = \sum_{i=1}^{N} \tilde{w}^i_{t+1|T} \frac{p(x^i_{t|t+1}|x^l_{t|t})}{\sum_{j=1}^{N} p(x^i_{t|t+1}|x^j_{t|t})} \tag{3.1}$$

Nach erweitern der Formel (3.1) bildet sich aus dieser Formel der Term $\varphi^i_t = \frac{\tilde{w}^i_{t+1|T}}{\sum_{j=1}^{N} p(x^i_{t|t+1}|x^j_{t|t})}$. Dann lautet die Formel (2.74):

$$w^l_{t|T} = \sum_{i=1}^{N} \varphi^i_t p(x^i_{t|t+1}|x^l_{t|t}) \tag{3.2}$$

Die Variable φ^i_t wird zur Berechnung der Funktion $Q(\theta,\ \theta_m)$ im nächsten Schritte vom Partikel-Smoohter ausgegeben. Dann wird das Flussdiagramm für die Funktion des Partikel-Smoother hergestellt:

Nach dem oberen Flussdiagramm ist die Matlab-Funktion programmiert, welche als: `[x_p_w_s, x_out_smoothing, P_x_s, Psi]= Partikelsmoother(fun_handle,x_Partikel,Partikel_w, Tab,Para,input,Q)` hergestellt wird. Die Beschreibung von dieser Funktion siehe:

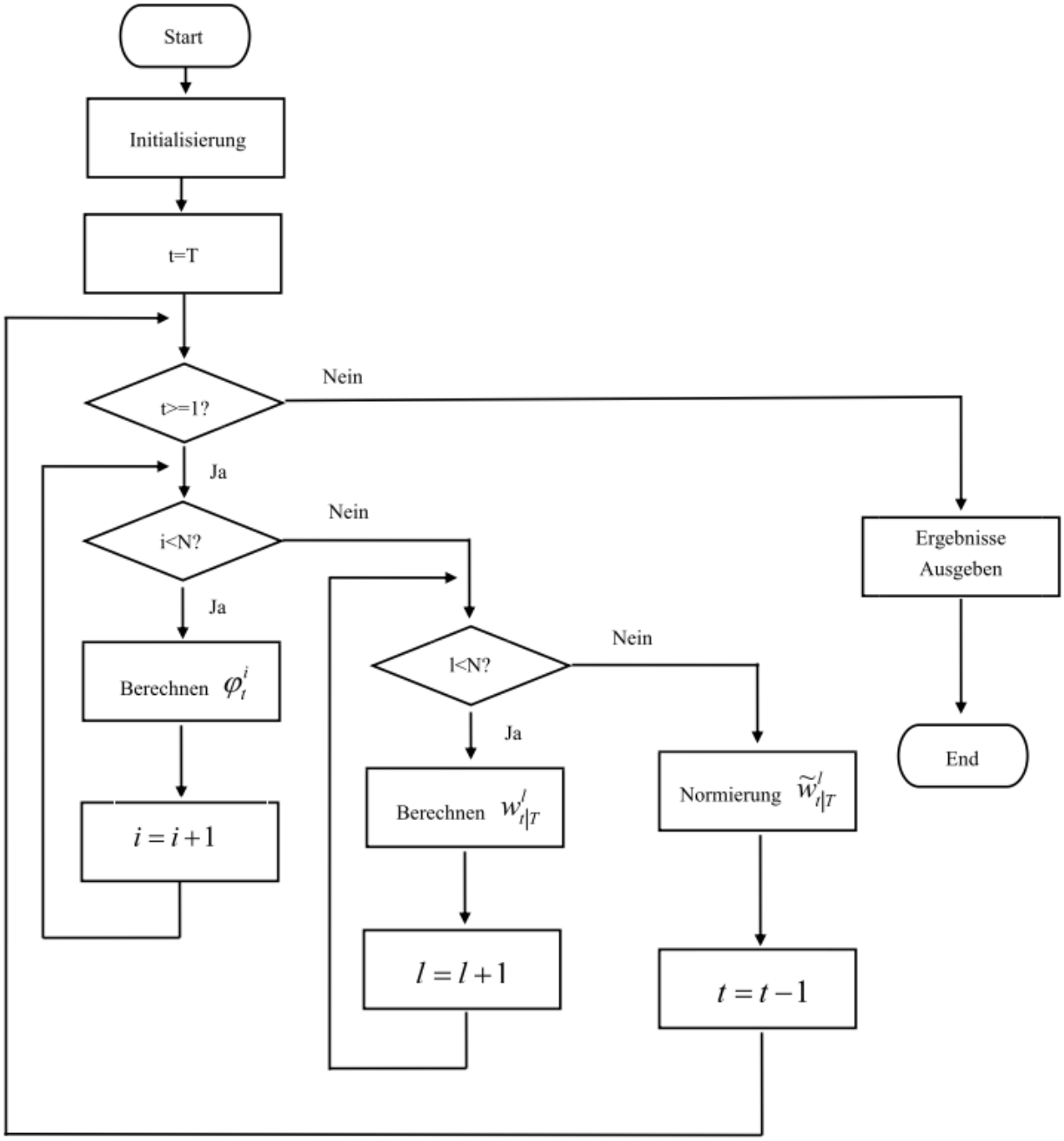


Abbildung 3.4: Flussdiagramm der Funktion des Partikel-Smoothers

- Ziel: Realisierung des Partikel-Smoothers
- Eingang:
  1. **fun_handle:** Handle der Systemfunktion
  2. **x_Partikel:** Partikel aus Partikel-Filter

3. **Partikel_w:** Gewicht von jeweiligem Partikel
4. **Tab:** Zeitintervall
5. **Para:** Parametervektor $[C_{sf} \quad C_{sr} \quad m/J \quad l_f]^T$
6. **input:** Eingangssignal
7. **Q:** Varianz vom Prozessrauschen

- Ausgang:
  1. **x_p_w_s:** Gewicht vom jeweiligen Partikel $\tilde{w}^i_{t|T}$
  2. **x_out_smoothing:** geschätzte Zustandsgröße
  3. **P_x_s:** alle Partikel in ganzer Zeitspanne $x^i_{t|T}$
  4. **Psi:** Variable φ^i_t

Funktion $Q(\theta,\ \theta_m)$

Zur Berechnung der Funktion $Q(\theta,\ \theta_m)$ dient hier die programmierte Matlab Funktion "`Q_out=Kostfkt(fun_handle,x_Partikel,T,N,Tab,Theta,Para,input,Q,err,Psi,d)`" . Die Ausgabe sind die Ergebnisse von $Q(\theta,\ \theta_m)$. Bei extremen Werten der Funktion $Q(\theta,\ \theta_m)$ ist entsprechend θ als wahrscheinlichster Parameter ausgesucht.
Das Systemmodell, welches in Funktion $Q(\theta,\ \theta_m)$ eingesetzt wird, ist das Fahrzeugmodell. Die ungenauen Parameter sind im Vektor $p = [C_{sf} \quad C_{sr} \quad m/J \quad l_f]^T$ gestaltet. Hier ist $\theta = p$, das bedeutet von den Formeln (2.76) - (2.78) ist nur die Formel (2.77) sinnvoll. Die anderen zwei Funktionen $I_1(\theta,\ \theta_m)$ und $I_3(\theta,\ \theta_m)$ sind von Parameter θ unabhängig. Die restliche Formel (2.77) lautet:

$$Q(\theta,\ \theta_m) = I_2(\theta,\ \theta_m) \approx \sum_{t=1}^{T-1} \sum_{l=1}^{N} \sum_{i=1}^{N} \varphi^i_t \tilde{w}^l_{t|t} p_{\theta_m}(x^i_{t|t+1}|x^l_{t|t}) \ln p_\theta(x^i_{t|t+1}|x^l_{t|t}) \tag{3.3}$$

Mit $\tilde{w}^l_{t|t} = \frac{1}{N}$ eingesetzt zerlegt sich der logarithmische Term in obiger Formel, dann wird er weiter entwickelt zu:

$$Q(\theta,\ \theta_m) \approx \sum_{t=1}^{T-1}\sum_{l=1}^{N}\sum_{i=1}^{N} \varphi_t^i \frac{1}{N} p_{\theta_m}(x_{t|t+1}^i | x_{t|t}^l) \ln\left\{ \frac{1}{\sqrt{2\pi\Xi}} exp\left[-\frac{(x_{t|t+1}^i - f_\theta(x_{t|t}^l,\ u_t) - w_t)^2}{2\Xi} \right] \right\} \tag{3.4}$$

$$\approx \sum_{t=1}^{T-1}\sum_{l=1}^{N}\sum_{i=1}^{N} \varphi_t^i \frac{1}{N} p_{\theta_m}(x_{t+1}^i | x_t^l) \left[\ln\frac{1}{2\Xi} - \frac{(x_{t|t+1}^i - f_\theta(x_{t|t}^l,\ u_t) - w_t)^2}{2\Xi} \right] \tag{3.5}$$

$$\propto -\sum_{t=1}^{T-1}\sum_{l=1}^{N}\sum_{i=1}^{N} \varphi_t^i p_{\theta_m}(x_{t+1}^i | x_t^l) \frac{(x_{t|t+1}^i - f_\theta(x_{t|t}^l,\ u_t) - w_t)^2}{2\Xi} \tag{3.6}$$

Mit der Entwickelung von $p_{\theta_m}(x_{t+1}^i | x_t^l)$ ist sie in die Formel einzubringen. Außerdem weil Matlab die "`fminserach`"-Funktion zur Bestimmung des Minimums anbietet, so müssen die Ergebnisse von Funktion $Q(\theta,\ \theta_m)$ als Inverse angenommen werden. Deshalb beträgt $Q(\theta,\ \theta_m)$:

$$Q(\theta,\ \theta_m) \propto \sum_{t=1}^{T-1}\sum_{l=1}^{N}\sum_{i=1}^{N} \varphi_t^i p_{\theta_m}(x_{t+1}^i | x_t^l) \frac{(x_{t|t+1}^i - f_\theta(x_{t|t}^l,\ u_t) - w_t)^2}{2\Xi} \tag{3.7}$$

$$\propto \sum_{t=1}^{T-1}\sum_{l=1}^{N}\sum_{i=1}^{N} \varphi_t^i \frac{1}{\sqrt{2\pi\Xi}} exp\left[\frac{(x_{t|t+1}^i - f_{\theta_m}(x_{t|t}^l,\ u_t) - w_t)^2}{2\Xi} \right] \frac{(x_{t|t+1}^i - f_\theta(x_{t|t}^l,\ u_t) - w_t)^2}{2\Xi} \tag{3.8}$$

$$\propto \sum_{t=1}^{T-1}\sum_{l=1}^{N}\sum_{i=1}^{N} \varphi_t^i exp\left[\frac{(x_{t|t+1}^i - f_{\theta_m}(x_{t|t}^l,\ u_t) - w_t)^2}{2\Xi} \right] \frac{(x_{t|t+1}^i - f_\theta(x_{t|t}^l,\ u_t) - w_t)^2}{2\Xi} \tag{3.9}$$

Damit ist das Flussdiagramm zu erstellen, hierbei sind die Ergebnisse zu jedem Zeitpunkt als $Q_t(\theta,\ \theta_m)$ gezeichnet, dann ergibt sich:

$$Q(\theta,\ \theta_m) \propto \sum_{t=1}^{T-1} Q_t(\theta,\ \theta_m) \tag{3.10}$$

$$\Rightarrow Q_t(\theta,\ \theta_m) = \sum_{l=1}^{N}\sum_{i=1}^{N} \varphi_t^i exp\left[\frac{(x_{t|t+1}^i - f_{\theta_m}(x_{t|t}^l,\ u_t) - w_t)^2}{2\Xi} \right] \frac{(x_{t|t+1}^i - f_\theta(x_{t|t}^l,\ u_t) - w_t)^2}{2\Xi} \tag{3.11}$$

Weiter wird definiert: $Q_t(\theta,\ \theta_m) = \sum_{l=1}^{N} Q_t^l(\theta,\ \theta_m)$ und $Q_t^l(\theta,\ \theta_m)$ lautet:

$$Q_t^l(\theta,\ \theta_m) = \sum_{i=1}^{N} \underbrace{\varphi_t^i exp\left[\frac{(x_{t|t+1}^i - f_{\theta_m}(x_{t|t}^l,\ u_t) - w_t)^2}{2\Xi}\right] \frac{(x_{t|t+1}^i - f_{\theta}(x_{t|t}^l,\ u_t) - w_t)^2}{2\Xi}}_{\Psi_t^{i,l}} \tag{3.12}$$

$$= \sum_{i=1}^{N} \Psi_t^{i,l} \tag{3.13}$$

zusätzlich sind die Ergebnisse von Funktion $f_{\theta}(x_{t|t}^l,\ u_t) + w_t)$ und $f_{\theta_m}(x_{t|t}^l,\ u_t) + w_t$ separat als $x_{t|t+1}^{l-}$ und $x_{t|t+1}^{l*-}$ definiert. Im Folgenden ist das Flussdiagramm zu sehen:

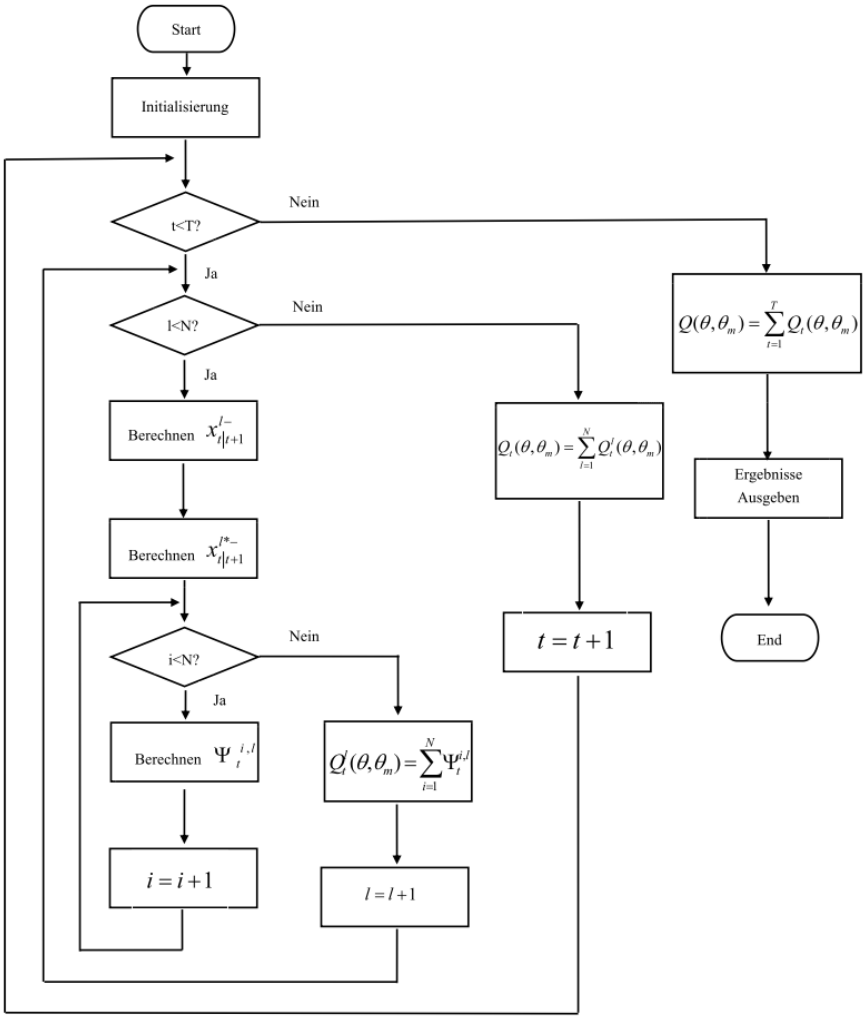


Abbildung 3.5: Flussdiagramm der Funktion $Q(\theta,\ \theta_m)$

Nach dem oberen Flussdiagramm wird die Funktion "`Q_out=Kostfkt(fun_handle,x_Partikel,T,N,Tab,Theta,Para,input,Q,err,Psi,d)`" fertiggestellt. Für die Beschreibung siehe:

- Ziel: Berechnung der Funktion $Q(\theta,\ \theta_m)$
- Eingang:
  1. **fun_handle:** Handle der Systemfunktion
  2. **x_Partikel:** Partikel aus Partikel-Filter
  3. **T:** Gesamte Zeitspanne
  4. **N:** Menge der Partikel
  5. **Tab:** Zeitintervall
  6. **Theta:** Parametervektor zu variieren θ
  7. **Para:** aktuelle Parametervektor θ_m
  8. **input:** Eingangssignal
  9. **Q:** Varianz von Prozessrauschen
  10. **err:** Prozessrauschen
  11. **Psi:** φ_t^i
  12. **d:** Dimension vom Systemmodell
- Ausgang:
  1. **Q_out:** Ergebnisse der Funktion $Q(\theta,\ \theta_m)$

M-Step von EM-Identifikation

Es ist eine Funktion nötig, die die minimalen Ergebnisse von Funktion $Q(\theta,\ \theta_m)$ aussucht. Bzw. um die "M-Step" von EM-Identifikation zu ermöglichen. Matlab bietet die "`fminsearch`" Funktion an, sie kann den minimalen Punkt bei einer angegebenen Funktion bestimmen und gibt gelegentlich die Variable am minimalen Punkt aus. Eine typische Anwendung von dieser Funktion ist:

```
x = fminsearch(@(x) myfun(x,a),[0,1])
```

Hierbei ist die Funktion "`myfun(x,a)`" mit der Variable x genutzt, welche als variierende Größe angenommen wird. Die Durchsuchung beginnt mit der Anfangsgröße $[0, 1]$ und nach der Durchsuchung gibt sie die Variable x aus bei der die Funktion "$myfun(x, a)$" minimal ist.

Hierbei wird zur EM-Identifikation auch eine Funktion spezifiziert, welche mit der Name

"`Para_out= EM_estimation(fun_handle,x_Partikel,Tab,Para,input,Q,Psi,v)`" benannt wird. Diese Funktion identifiziert in jedem Ablauf nur einen Parameter. Für die Beschreibung siehe fort folgend:

- Ziel: Bestimmung den wahrscheinlichsten Parameter
- Eingang:
  1. **fun_handle:** Handle der Systemfunktion
  2. **x_Partikel:** Partikel aus Partikel-Filter
  3. **Tab:** Zeitintervall
  4. **Para:** aktueller Parametervektor θ_m
  5. **input:** Eingangssignal
  6. **Q:** Varianz von Prozessrauschen
  7. **Psi:** φ_t^i
  8. **v:** Nummer des Parameters. Das bedeutet, vte Parameter wird zur EM-Estimation eingebracht.
- Ausgang:
  1. **Para_out:** bestimmter Parameter

EM-Identifikation

Bisher sind alle Unterfunktionen für die EM Identifikation abgebildet. Zur Organisation der EM-Identifikation sind alle solche Unterfunktion zusammengesetzt. Zur Identifikation wird eine Maßnahme Namens ´´ganz Schritt Verfahren´´ eingesetzt. Bei jeder Durchsuchung wird nur ein Parameter im Parametervektor variiert, die restliche drei sind festgelegt. Der identifizierter Parameter erneuert den Parametervektor und wird in die Identifikation für den nächsten Parameter eingesetzt. So ist die Durchsuchung bei jedem Parameter einmal ausgeführt. Eine Maßnahme darf nicht vergessen werden: Nach jedem Update des Parametervektor muss die Partikel-Methode für die nächste Identifikation einmal durchlaufen gelassen werden.
Der ganze Ablauf besteht aus zwei Schritte: EM-Identifikation mit Partikel-Methode und Überprüfung der Parameter. Nach der EM-Identifikation wird die Änderung der Parameter überprüft , ob es im vordefinierten Bereich konvergiert. Wenn die Änderung sich außerhalb dieses Bereichs befinden, wird der ganze Ablauf noch ein mal durchgeführt. Sonst bricht das Programm ab und gibt das Ergebnisse aus. Im Folgenden ist das Flussdiagramm von

EM-Identifikation zu sehen:

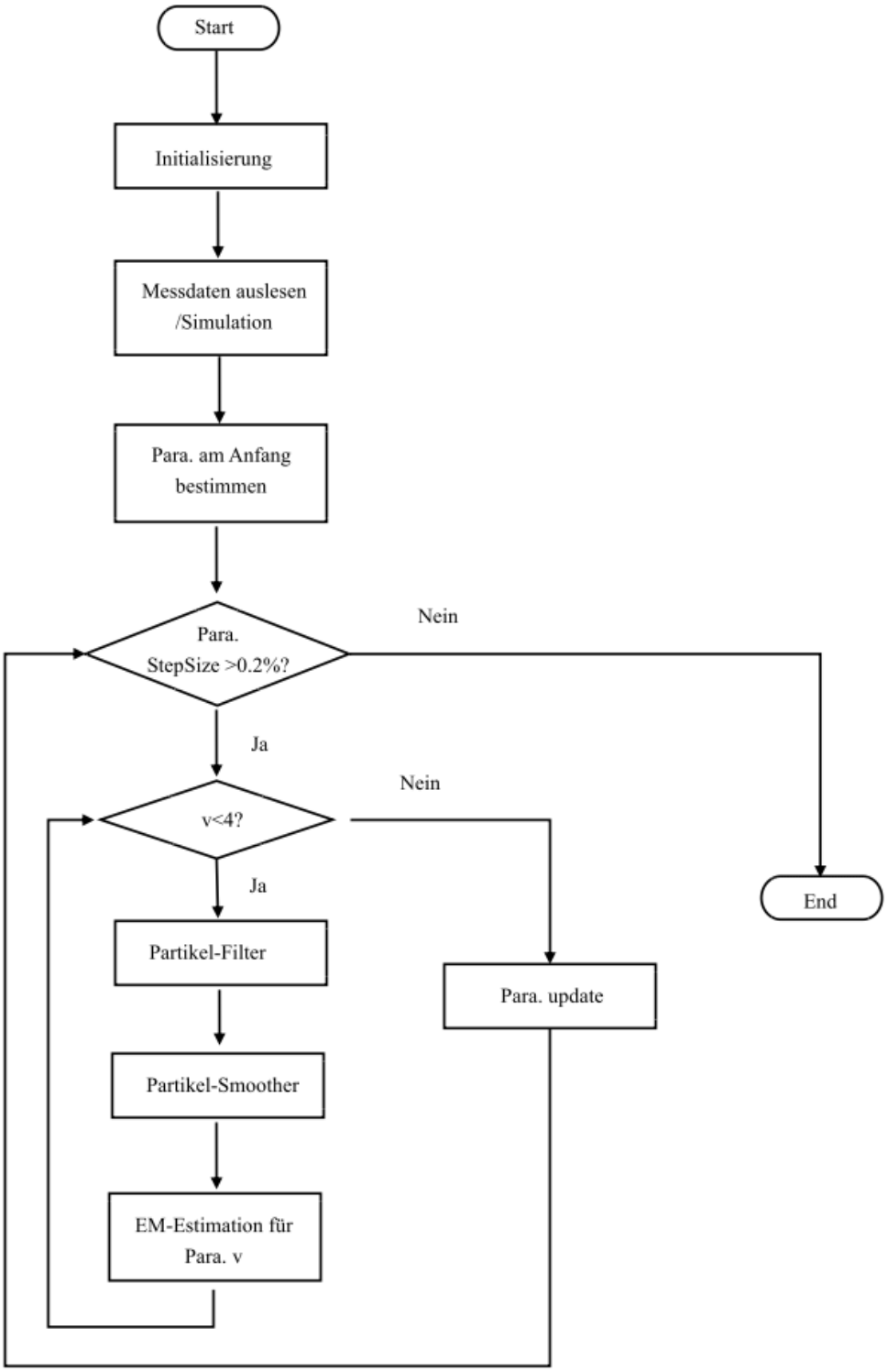


Abbildung 3.6: das Flussdiagramm von EM-Identifikation

Eine Initialisierung ist vor der Identifikation notwendig. Die folgenden Variablen müssen bereitgestellt werden:

- Ausgangsgröße aus Simulation(für Simulation) **y_rel_rk** oder Messreihe aus Testfahrt(für reale Fahrt) **y_rel_m**
- Varianzmatrix von Prozessunsicherheit **Q**
- Varianzmatrix von Messunsicherheit **R**
- Zahl der Partikel **N**
- Varianz der Verteilung der Partikel am Anfang **V**
- aktueller Parametervektor **para_akt**
- Zeitspanne der abgeschnittenen Messung **t_span**

Das Array "**Para_update**" ist für die Speicherung des Parametervektors hergestellt. Der ermittelte Parametervektor in jeder Iteration wird in dieses Array hinein gelegt.

3.2 Parameter-Identifikation des Fahrzeugmodells mit Simulation

Die Simulation des Fahrzeugs erzeugt die virtuelle Messreihe, welche die Messreihe aus dem realen Fahrzeug ersetzen kann. Der Vorteil ist: Der Parametervektor des Fahrzeugs kann selbst definiert werden, das heißt, durch Vergleichen der Parametervektoren des Fahrzeugs mit identifizierten Parametern kann die Qualität der Identifikation ausgewertet werden.
Dieses Unterkapital besteht aus drei Bereichen:

1. Test des Partikel-Filters/Smoothers in der Simulation
2. Plot der Diagramm der Funktion $Q(\theta,\ \theta_m)$
3. Parameter Identifikation in der Simulation

Vor dem Start der Simulation werden relative Parameter des Fahrzeugs eingetragen, die Parameter kommen aus dem Datenbuch und vorheriger Messungen an dem Fahrzeug.

- Schwerkraftbeschleunigung: $g = 9.81m/s^2$
- Höhe des Schwerpunkts des Fahrzeugs: $h = 0.5m$
- Masse des Testfahrzeugs: $m = 1707kg$
- Trägheitsmoment in Z-Achse des Testfahrzeug: $J = 2741.9kg \cdot m^2$
- Abstand zwischen Gewichtspunkt und vorderer Achse: $l_f = 1.014m$
- Abstand zwischen hinterer- und vorderer Achse: $L = 2.69m$
- Reifensteifigkeit des vorderen Reifens: $C_f = 70000N/rad$
- Reifensteifigkeit des hinteren Reifens: $C_r = 65100N/rad$

Damit lauten die einzelnen Element im Parametervektor p :

$$C_{sf} = \frac{2C_f}{mg} = 8.361rad^{-1} \tag{3.14}$$

$$C_{sr} = \frac{2C_r}{mg} = 7.781rad^{-1} \tag{3.15}$$

$$m/J = \frac{m}{J} = 0.62m^{-2} \tag{3.16}$$

$$l_f = 1.014m \tag{3.17}$$

$$p = [C_{sf}, \quad C_{sr}, \quad m/J, \quad l_f]^T = [8.36, \quad 7.78, \quad 0.62, \quad 1.014]^T \tag{3.18}$$

Es fehlt noch das Eingangsignal für den Simulator, der das Fahrzeugmodell anregen kann. Hier wird eine Datenreihe vom Eingangssignalen aus Messungen von Testfahrten (z.B. Spurwechsel) eingefügt um den Simulator möglichst die Realität simulieren zu lassen. Der Zeitverlauf des Eingangssignals siehe folgend:

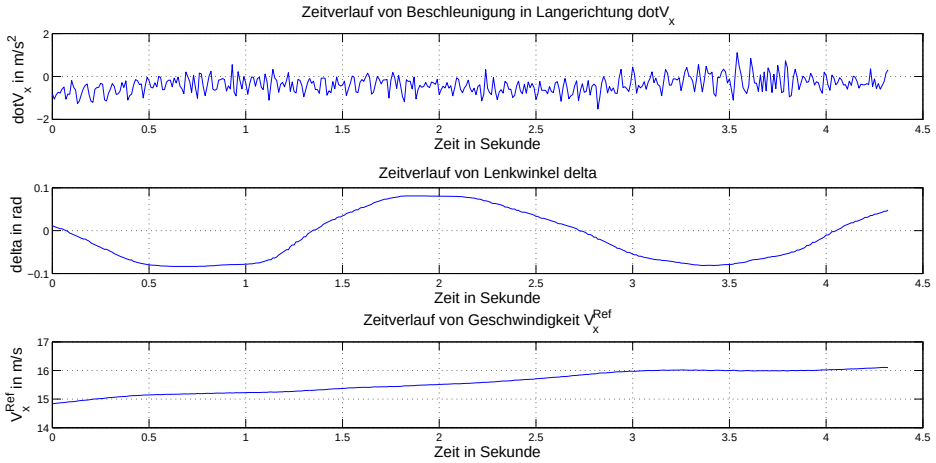


Abbildung 3.7: Zeitverlauf des Eingangssignals

Die Zeitdauer von der abgeschnittenen Messung ist 13.81s mit dem Zeitintervall 0.01s. Der Anfangszustand kann auch aus der Messung abgeschrieben werden. Es lautet:

$$x_0 = \left[V_{y0}^M, \quad \omega_0, \quad \varphi_0\right]^T = [0.2285, \quad -0.4463, \quad 0.1199]^T \tag{3.19}$$

Die Unsicherheit des Messgeräts muss hier berücksichtigt werden, diese Beeinflussung muss auch in den Simulator eingebracht werden. Die Messunsicherheit ist als weißes und mittelwertfreies Rauschen angenommen. Die Steuerung des Rauschen ist definiert als:

- $err_{V_y^{Ref}} = 0.005 m/s$
- $err_\omega = 0.02 \times \frac{\pi}{180} rad/s$

- $err_\varphi = 0.005 \times \frac{\pi}{180} rad$

Jetzt wird der Simulator gestartet, der Zeitverlauf der Ausgangsgröße ist in der unteren Abbildung zu sehen:

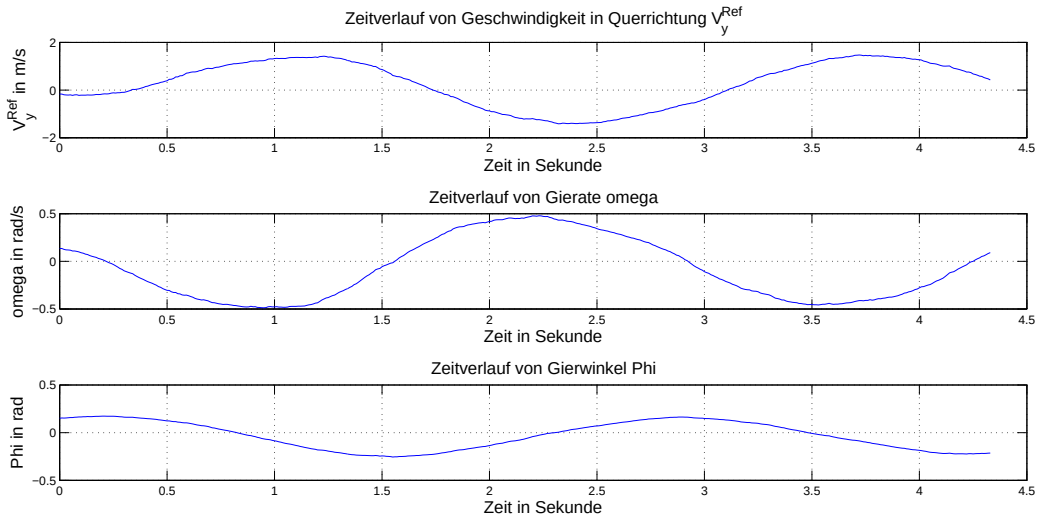


Abbildung 3.8: Zeitverlauf von Ausgangsgröße des Simulators

Soweit ist die virtuelle Messreihe vorhanden. Die Messreihe wird in nächsten Schritt vom Partikel-Filter genutzt. Der Partiekel-Filter wird die Zustandsgrößen einschätzen und in stochastischer Form ausgeben.

Test des Partikel-Filters/Smoothers in der Simulation

Zur Überprüfung des Partikel-Filters/Smoothers ist die virtuelle Messung als Referenz-Messung in den Partikel-Filter eingegeben. Zur Initialisierung des Partikel-Filters sind eingestellt:
Der Rechenaufwand muss bei Einsatz vom EM-Identifikation gespart werden, sonst ist die Zeitdauer zu lang. Die Zahl der Partikel $N = 50$ und die Zeitspanne wird um 500 Zeitpunkte weiter verkürzt. Damit unter solchen Bedingungen es nicht den Effekt der Identifikation beeinflusst, ist die Zeitdauer der Identifikation zu reduzieren.
Die Varianzmatrix der Messunsicherheit Θ wird nach angegebener Unsicherheit des Messgeräts oder Messwerts bestimmt, diese wird eingeschätzt als: $\Theta = diag\left(\left[err^2_{V_y^{Ref}}, \quad err^2_\omega, \quad err^2_\varphi\right]\right)$.

Die Varianzmatrix der Systemunsicherheit Ξ wird als $\Xi = \Theta$ definiert. Dann werden die Ausgaben vom Partikel-Filter/Smoother in den drei Diagramm unterhalb dargestellt:

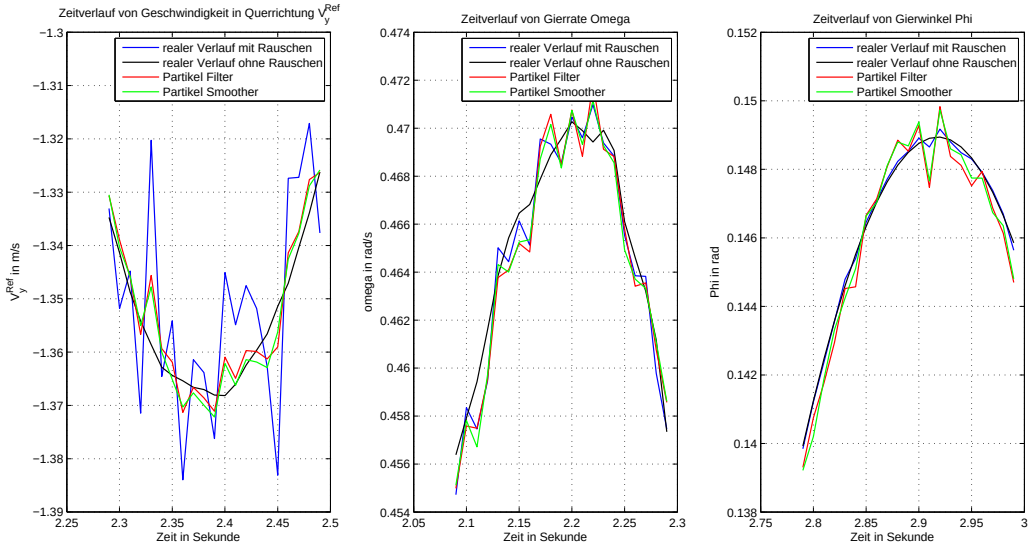


Abbildung 3.9: Einschätzung des Zustands mit Partikel-Filter/Smoother

Dann die RMSE für Partikel-Filter und Smoother wird berechnet, zu sehen in folgender Tabelle:

Zustandsgröße	RMSE von V_y^{Ref}	RMSE von ω	RMSE von φ
Partikel-Filter	0.0052	0.0015	0.0006
Partikel-Smoother	0.0048	0.0014	0.0006

Tabelle 3.1: Vergleich des Partikel-Filters mit Partikel-Smoother in RMSE

Die Darstellung zeigt, dass der Partikel-Smoother eine genauere und glattere Einschätzung ausgibt.

Funktion $Q(\theta,\ \theta_m)$

Nun wird der Parametervektor als $p = [7.942, \quad 7.386, \quad 0.56, \quad 0.91]^T$ versetzt, die Varianzmatrix der Systemunsicherheit Ξ muss neu ausgewertet werden und diese Matrix kommt aus der Systemunsicherheit. z.B.: Wie viel Abweichung der Systemantwort hat der ungenauere Parameter bewirkt? Die Auswirkung kann nur eingeschätzt werden. Von der Formel über

Funktion $Q(\theta,\ \theta_m)$ kann man erfahren, dass die Matrix Ξ die Ergebnisse von Funktion beeinflussen kann. Eine unrichtige Matrix Ξ kann die Ergebnisse der Identifikation abweichend beeinflussen. Eine zuverlässige Methode für die Einschätzung von Ξ ist die Abweichung zwischen Systemantwort mit versetztem Parameter und realem Parameter. Mit dieser Methode wird der $\Xi = diag\left([0.02^2,\quad 0.003^2,\quad 0.003^2]\right)$ eingeschätzt.
Dann führt der Partikel-Filter/Smoother die Einschätzung nach diesen Parametervektor erneut aus. Die Ausgabe wird in die Funktion "Kostfkt" eingelegt. θ ist die Variable die den Verlauf der Ergebnisse der Funktion $Q(\theta,\ \theta_m)$ darstellt, diese wird nach θ geplottet:

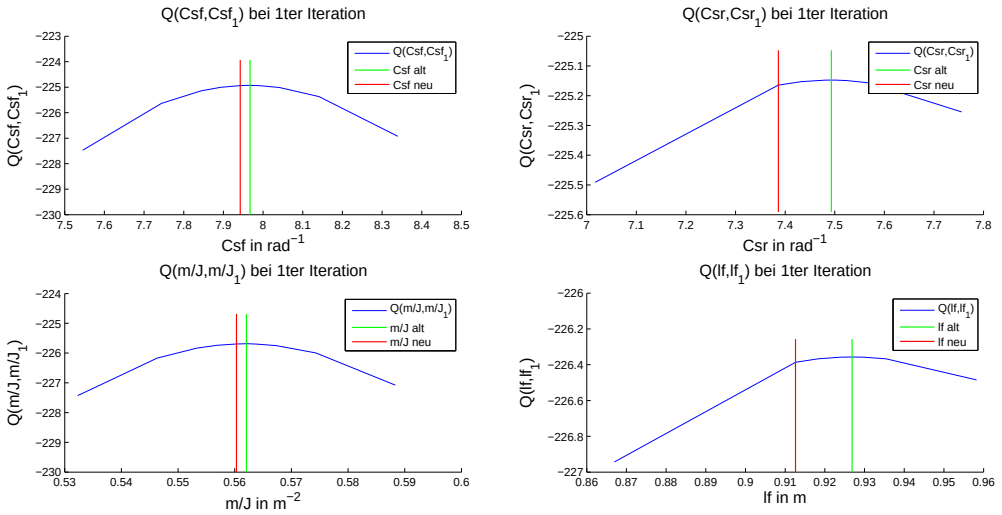


Abbildung 3.10: Der Verlauf der Funktion $Q(\theta,\ \theta_m)$ nach θ bei der 1. Iteration

Die obere Darstellung weist deutlich auf, dass der externe Punkt der Funktion $Q(\theta,\ \theta_m)$ zum Referenz Wert neigt.

Parameter Identifikation im der Simulation

Dafür lässt man die Iteration weiter laufen, bis der Parameter so genau ist, dass das Iterationsverfahren abbricht. Die untere Abbildung zeigt dieses Konvergenz.

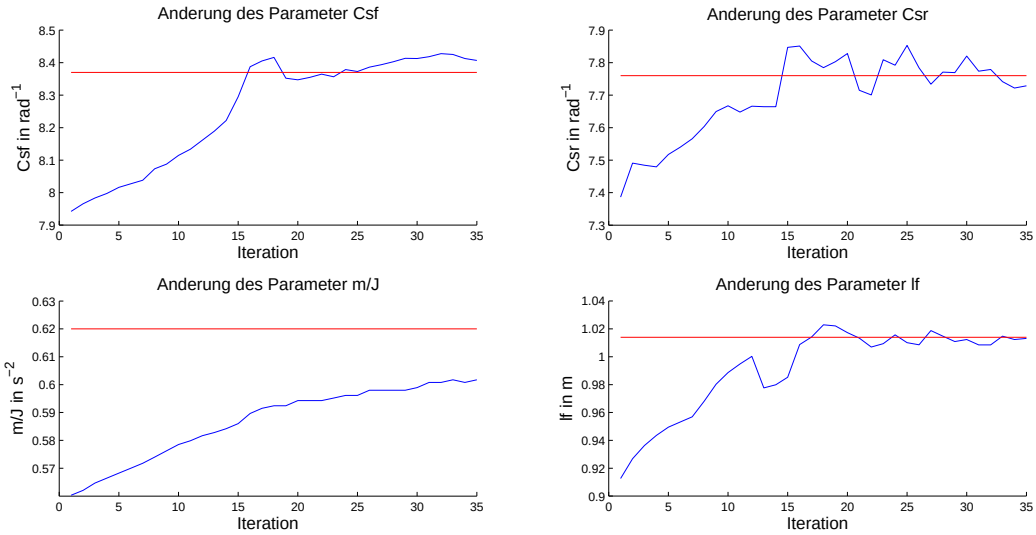


Abbildung 3.11: Parameter-Identifikation mit EM-Identifikation

Die Parameter werden identifiziert als:

Parameter	C_{sf}	C_{sr}	$\frac{m}{J}$	l_f
Reference-Wert	$8.36rad^{-1}$	$7.78rad^{-1}$	$0.62m^{-2}$	$1.014m$
identifizierter Wert	$8.41rad^{-1}$	$7.73rad^{-1}$	$0.60m^{-2}$	$1.013m$
$rel\%$	0.59%	0.64%	3.2%	0.1%

Tabelle 3.2: Ergebnisse von EM-Identifikation

Soweit ist der Test der Parameter-Identifikation in der Simulation fertig. Die Identifikation zeigt effektiv: alle Parameter konvergieren endlich am Referenz-Wert. Aber für den 3ten Parameter m/J weist sie große Abweichungen auf. Durch Betrachtung des Fahrzeugmodells ist abstrahiert: der 3te Parameter m/J und 4te Parameter l_f auf der Systemantwort haben eine sehr geringe Auswirken. Eine kleine Änderung in Matrix Ξ kann diese Parameter der Ungenauigkeit identifizieren.

3.3 Parameter-Identifikation des Fahrzeugmodells bei realer Testfahrt

Nun wird der Algorithmus der Identifikation bei einem realem Testfahrzeug angewendet. Die Fahrbahn des Testfahrt ist eine geradeaus Trajektorie mit Spurwechsel. Die maximale Geschwindigkeit beträgt 70km/h. Für die Identifikation ist normalerweise eine höhere Geschwindigkeit geeignet. Deshalb wird eine Strecke mit höherer Geschwindigkeit zur Parameter-Identifikation zugeschnitten(Geschwindigkeit von 35km/h bis 70 km/h). Die gesamte Trajektorie wird in zwei Sequenzen verteilt. Die erste Sequenz heißt "Training", die ist zur Identifikation einzusetzen, die zweite heißt ´´Evaluationssequenz´´, welche zur Überprüfung der Ergebnisse von Identifikation einzubringen ist.
Die Navigation hat die Koordinaten der gefahrener Strecke eingeloggt. Mit diesen Logdaten wird die Fahrbahn dargestellt:

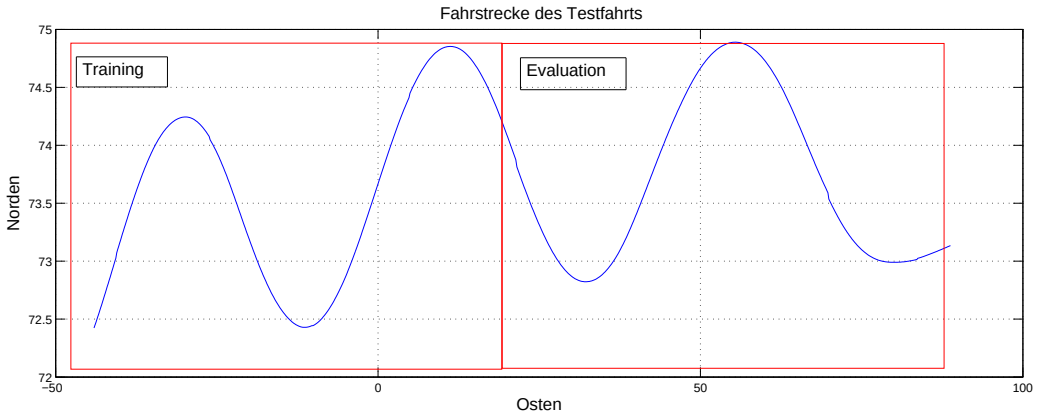


Abbildung 3.12: Fahrbahn der Testfahrt

Messdaten Bearbeitung

Statt der Simulation ist das Auslesen der Ausgangsgrößen und Eingangssignale aus der realen Messreihe durchgeführt. Das Messsystem besteht aus einer Navigationsgerät und einem Inertial Navigationssystem. Beide Messgeräte messen nach ihren eigenen Koordinationssystemen, für die Navigation ist das Koordinationssystem nach der Station abgebildet. Norden und Osten entspricht der Y- und X-Richtung. Andererseits ist das Koordinatensystem von Inertial Navi auf dem Messpunkt des Fahrzeug befestigt. Die beiden Geräte liefern zahlreiche Messeinträge. Hier sind die brauchbaren Ein- und Ausgangsgrößen des Fahrzeugmodells zusehen:

- von Navigation:
  1. Geschwindigkeit in nördliche Richtung: V_{north}
  2. Geschwindigkeit in östliche Richtung: V_{east}
  3. Azimut des Fahrzeugs nach Norden: $Azimuth$
- von Inertial Navigationssystem:
  1. Gierrate: ω
  2. Beschleunigung in X-Richtung des Koordinatensystems vom Inertial Navi.: $\ddot{x}^{IMU}$
  3. Beschleunigung in Y-Richtung des Koordinatensystems vom Inertial Navi.: $\ddot{y}^{IMU}$

Aus den Einträgen sind die nun gezeigten Messwert ermittelt:

$$Die\ Beschleunigung\ vom\ Gierwinkel : \dot{\omega}(t) = \frac{\omega(t+1) - \omega(t)}{\Delta T} \tag{3.20}$$

$$Der\ Gierwinkel\ am\ Messpunkt : \varphi_{IMU} = \pi - \pi \frac{Azimuth}{180} \tag{3.21}$$

Zusätzlich ist der Lenkwinkel des Lenkrads zu betrachten. Dieser Lenkwinkel δ wird direkt an der vorderen Achse gemessen. Das bedeutet die Zeitverzögerung von Lenkrad zur vorderen Achse ist nicht zu beachten. Die Kalibrierung der Lenkwinkelmessung muss nach Auslesen der Messdaten ausgeführt werden. Hierfür wird ein Offset auf den Messwert addiert.

Weil der Messpunkt sich nicht am Referenz-Punkt befindet ist es notwendig diesen auf den Referenz Punkt umzurechnen. Für die relative Postion des Messgeräts zum Referenz Punkt siehe die untere Abbildung:

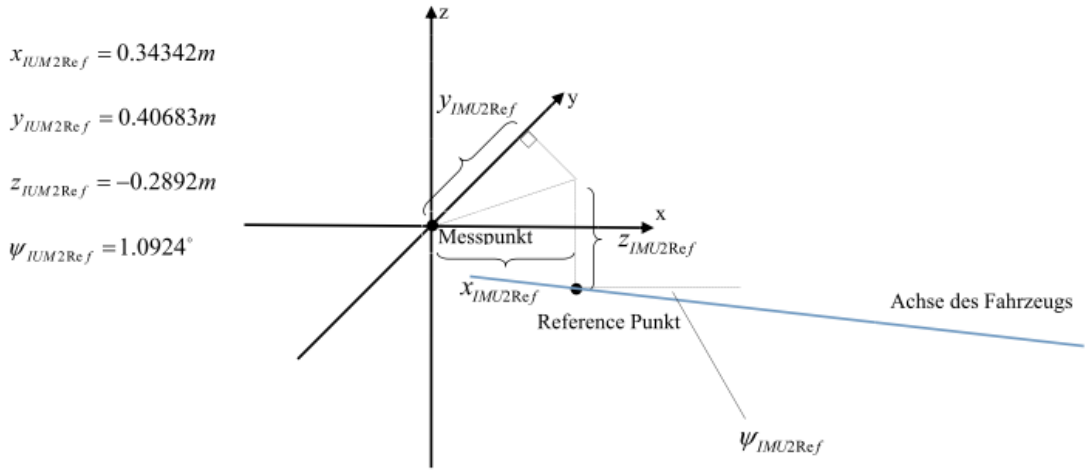


Abbildung 3.13: Relative Postion von Messpunkt zur Referenzpunkt

Die obere Abbildung zeigt, dass es neben der relevanten Position zwischen Messpunkt und Referenz-Punkt auch noch den Winkel zwischen ihren Koordinatensystem gibt. Das bedeutet, die Umsetzung der Messgrößen geht nicht nur mit der Umsetzung der Position(Verschiebung) sondern auch mit der Umsetzung eines Winkels(Drehung). Die Umsetzungsformeln lauten:

$$\varphi_{Ref} = \varphi_{IMU} + \Psi_{IMU2COR} \tag{3.22}$$

$$V_x^{IMU} = V_{east}\cos(\varphi_{Ref}) + V_{north}\sin(\varphi_{Ref}) \tag{3.23}$$

$$V_y^{IMU} = -V_{east}\sin(\varphi_{Ref}) + V_{north}\cos(\varphi_{Ref}) \tag{3.24}$$

$$V_x^{Ref} = V_x^{IMU} - \omega \cdot x_{IMU2Ref} \tag{3.25}$$

$$V_y^{Ref} = V_y^{IMU} + \omega \cdot y_{IMU2Ref} \tag{3.26}$$

$$X^{Ref} = X^{IMU} + x_{IMU2Ref} \cdot \cos(\varphi_{Ref}) - y_{IMU2Ref} \cdot \sin(\varphi_{Ref}) \tag{3.27}$$

$$Y^{Ref} = Y^{IMU} + x_{IMU2Ref} \cdot \sin(\varphi_{Ref}) + y_{IMU2Ref} \cdot \cos(\varphi_{Ref}) \tag{3.28}$$

$$\ddot{x}^{Ref} = \ddot{x}^{IMU}\cos(\Psi_{IMU2Ref}) + \ddot{y}^{IMU}\sin(\Psi_{IMU2Ref}) - x_{IMU2Ref} \cdot \omega^2 - y_{IMU2Ref} \cdot \dot{\omega} \tag{3.29}$$

$$\ddot{y}^{Ref} = -\ddot{x}^{IMU}\sin(\Psi_{IMU2Ref}) + \ddot{y}^{IMU}\cos(\Psi_{IMU2Ref}) - y_{IMU2Ref} \cdot \omega^2 - x_{IMU2Ref} \cdot \dot{\omega} \tag{3.30}$$

Die Ein- und Ausgangsgrößen sowie der Anfangszustand des Systems wird hergestellt mit:

$$u = \begin{bmatrix} \dot{V}_x^{Ref} & \delta & V_x^{Ref} \end{bmatrix}^T \tag{3.31}$$

$$x_0 = \begin{bmatrix} V_y^{Ref}(0) & \omega(0) & \varphi_{Ref}(0) \end{bmatrix}^T \tag{3.32}$$

$$y = \begin{bmatrix} V_y^{Ref} & \omega & \varphi_{Ref} \end{bmatrix}^T \tag{3.33}$$

Alle Verfahren vom Auslesen und Bearbeitung des Messdaten sind in der Matlab Funktion "`[T t_span input x0 Messwert]=Messdaten_lesen`" gestaltet. Diese Funktion ist wie folgt beschrieben:

- Ziel: Auslesen und Bearbeitung des Messdaten
- Eingang: Keine
- Ausgang:
  1. **T:** Zeitintervall der Messreihe
  2. **t_span:** Zeitspanne
  3. **input:** Eingangsgröße des Fahrzeugmodells
  4. **x0:** Anfangszustand des Fahrzeugmodells
  5. **y:** Ausgangsgröße des Fahrzeugmodells

Die Zeitspanne von ´´Training´´ wird von $30.00s$ bis $35.00s$ mit 500 Zeitpunkt ausgewählt. Wird die Funktion aufgerufen verrechnet sie die Eingangssignale und Messwerte. Die Darstellung ist folgende:

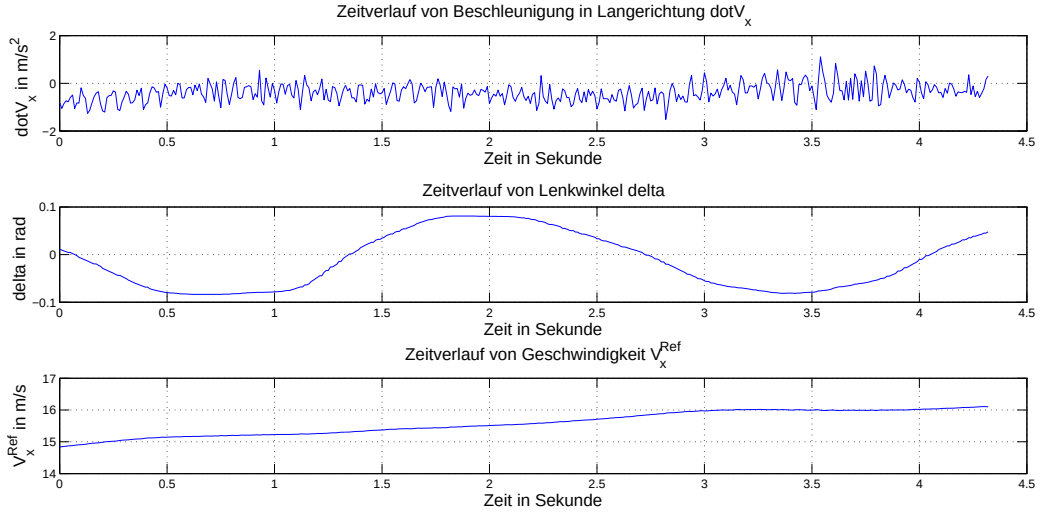


Abbildung 3.14: Zeitverlauf des Eingangssignals

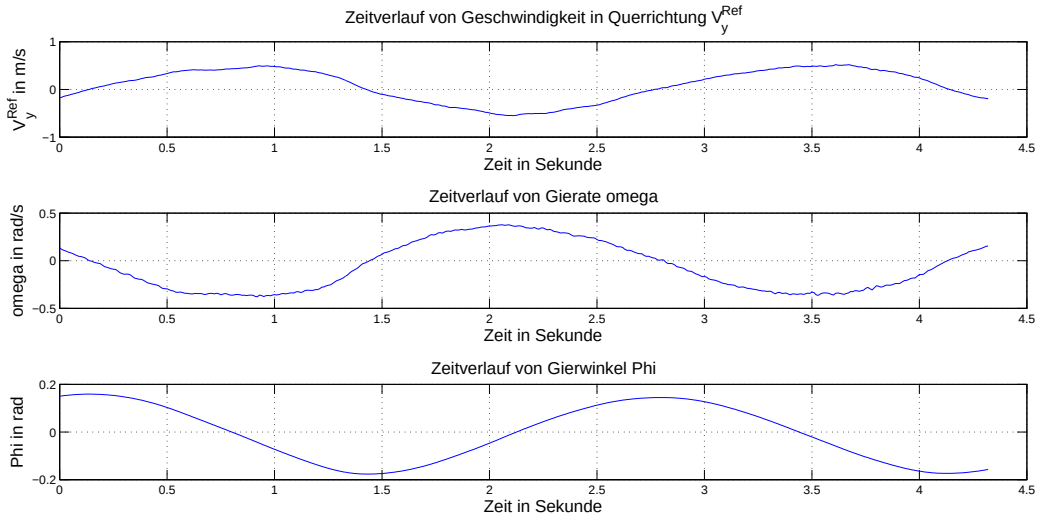


Abbildung 3.15: Zeitverlauf des Ausgangssignals

Bestimmung der Anfangsbedingung

Nach dem Auslesen sind die Eingangssignale und Anfangszustände bestimmt. Weiter gilt es noch die Parametervektoren mit einem geeigneten Parametervektor auszustatten, dies kann bei der Identifikation viel Zeit sparen. Das Trägheitsmoment J, die Masse des Fahrzeugs m und l_f sind vorgegeben:

$$J = 2400kg \cdot s^2 \tag{3.34}$$
$$m = 1800kg \tag{3.35}$$
$$l_f = 1.08m \tag{3.36}$$

Außerdem sind auch einige Werte vorgegeben:

$$L = 2.71m \tag{3.37}$$
$$h = 0.5m \tag{3.38}$$
$$g = 9.8m/s^2 \tag{3.39}$$

Zur Einschätzung der Reifensteifigkeit muss die Seitenkraft F_y, Druckkraft der Reifen F_z und der Schräglaufwinkel α abgebildet werden. Mittels der vorhandenen Messwerte ist dies machbar. Die Berechnungsformeln lauten:

$$F_{yf} = \ddot{y}^{Ref} m + \dot{\omega} m l_r + \frac{J\dot{\omega}}{l_r} \tag{3.40}$$
$$F_{yr} = \ddot{y}^{Ref} m + \dot{\omega} m l_r - \frac{J\dot{\omega}}{l_f} \tag{3.41}$$
$$F_{zf} = \frac{l_r mg - hm\dot{v}_x}{L} \tag{3.42}$$
$$F_{zr} = \frac{l_f mg + hm\dot{v}_x}{L} \tag{3.43}$$
$$\alpha_f = \delta - arctan(\frac{V_y^{Ref} + L\omega}{V_x^{Ref}}) \tag{3.44}$$
$$\alpha_r = -arctan(\frac{V_y^{Ref}}{V_x^{Ref}}) \tag{3.45}$$

Die Beziehung zwischen Schräglaufwinkel und dem Verhältnisse $\frac{F_y}{F_z}$ ist dargestellt als:

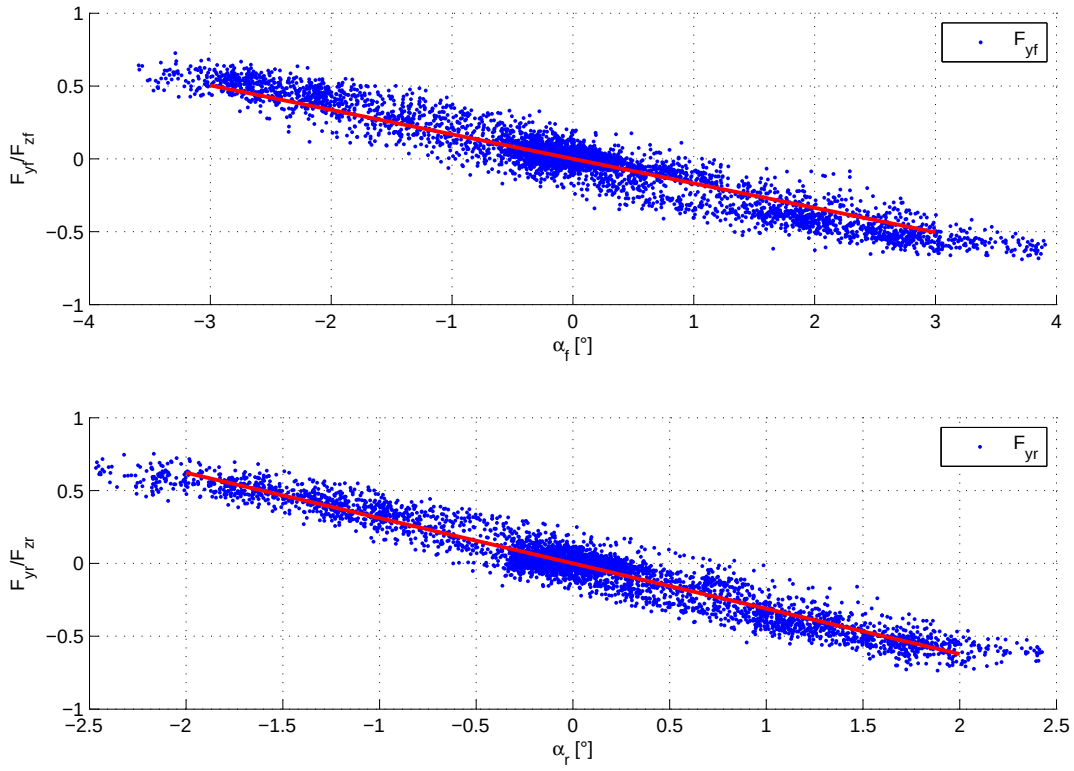


Abbildung 3.16: Darstellung der Beziehung zwischen Schräglaufwinkel und $\frac{F_y}{F_z}$

Der blaue Punkt in der oberen Darstellung ist die Seitenkraft des Reifens die von der Messung ermittelt wird. Mittels der Verteilung solcher Punkte wird eine Gerade dargestellt(rote Gerade in oberer Abbildung), dabei ist die Reifensteifigkeit die Steigung dieser Geraden. Sie wird nach der obigen Darstellung eingeschätzt als:

$$C_{sf} = 0.1681/\circ = 9.62 rad^{-1} \tag{3.46}$$

$$C_{sr} = 0.3111/\circ = 17.82 rad^{-1} \tag{3.47}$$

Die Messung vom Lenkwinkel wird mit einem Faktor k_δ und einem Offset off_δ kalibriert. Er lautet:

$$\delta = k_\delta \cdot \delta_{Mess} + \text{off}_\delta \tag{3.48}$$

Bei Testfahrt werden diese beide Koeffizient als $k_\delta = 0.88$ und $\text{off}_\delta = 0.0056 rad$ bestimmt. Es fehlt noch die Varianzmatrix vom Prozessrauschen Ξ. Die Bestimmung der Elemente in Ξ kommt darauf an, wie viel Abweichung der ungenauen Parameter für die einzelnen zugeführt Ausgangsgröße hat. Werden die ungenauen Parameter in die Simulation eingebracht und die Ergebnisse der Simulation mit der Messung verglichen ist die Matrix Ξ einzuschätzen als:

$$\Xi = \begin{bmatrix} 0.0108^2 & 0 & 0 \\ 0 & 0.0031^2 & 0 \\ 0 & 0 & 0.00125^2 \end{bmatrix} \tag{3.49}$$

Die andere Varianzmatrix Θ kann durch den Vergleich der Amplitude des Rauschens zwischen Simulation und Messung hergestellt werden. Dadurch wird Θ bestimmt als:

$$\Theta = \begin{bmatrix} 0.005^2 & 0 & 0 \\ 0 & (3.49 \times 10^{-4})^2 & 0 \\ 0 & 0 & (3.49 \times 10^{-5})^2 \end{bmatrix} \tag{3.50}$$

Ergebnisse der EM-Identifikation

Nun kann die Identifikation gestartet werden bis die 266 Iterationen der einzelnen Parameter in den kleinen Bereich konvergieren. Dann bricht die Identifikation ab und gibt die Darstellung der Änderung den Parameter aus:

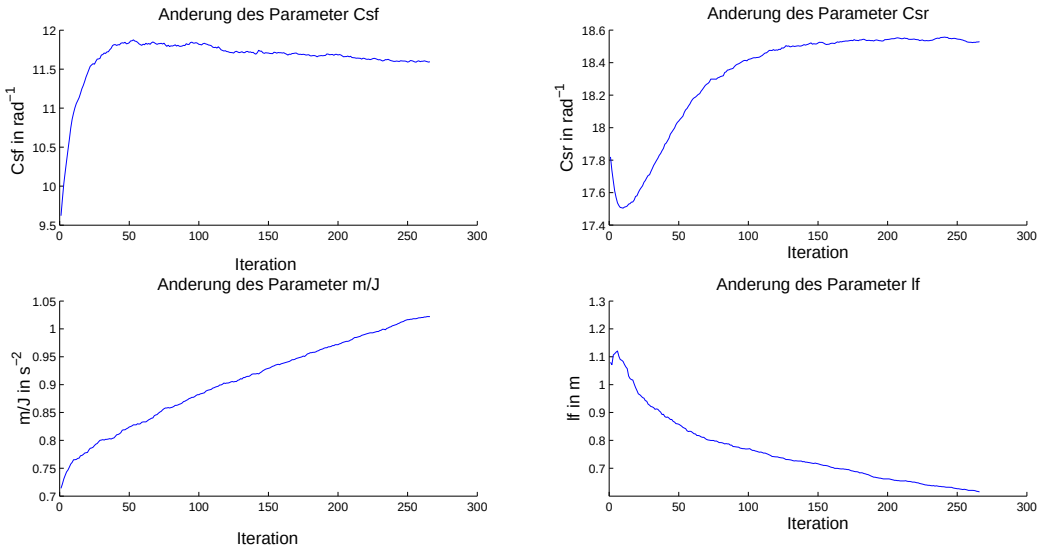


Abbildung 3.17: Konvergenz den Parameter bei der Identifikation

Den Parameter werden identifiziert als:

$$C_{sf} = 11.59 rad^{-1} \tag{3.51}$$

$$C_{sr} = 18.53 rad^{-1} \tag{3.52}$$

$$m/J = 1.02 m^{-2} \tag{3.53}$$

$$l_f = 0.62 m \tag{3.54}$$

Der Parametervektor ist:

$$p = [11.59 \quad 18.53 \quad 1.02 \quad 0.62]^T \tag{3.55}$$

Bringt man diese Ergebnisse wieder in die Simulation ein, aber dieses Mal wird die Zeitintervall von Evaluation eingefügt. Die Vergleichung seiner Ausgabe mit der vorherigen Ausgabe ist es wie unten dargestellt zu sehen:

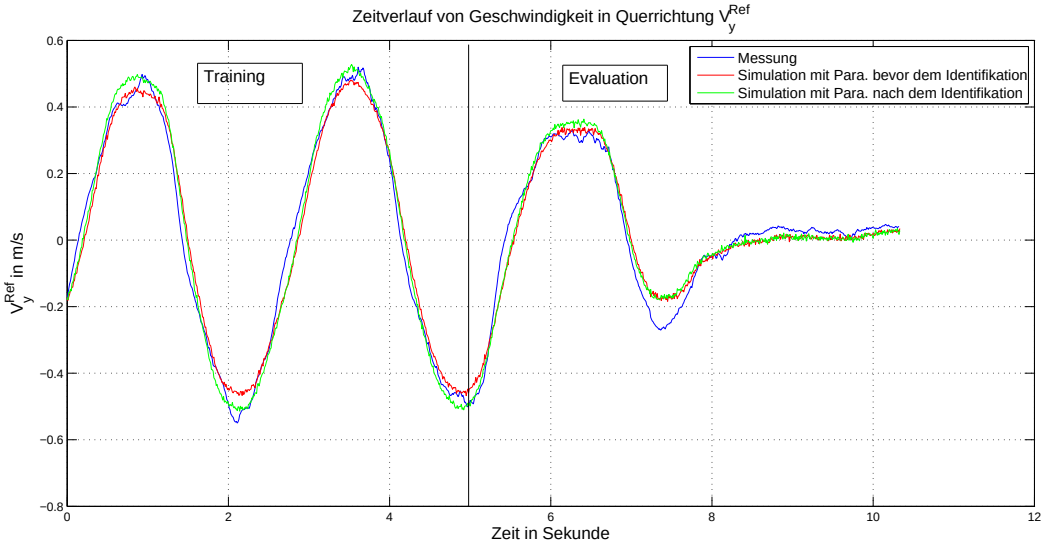


Abbildung 3.18: Zeitverlauf von Geschwindigkeit in Querrichtung V_y^{Ref}

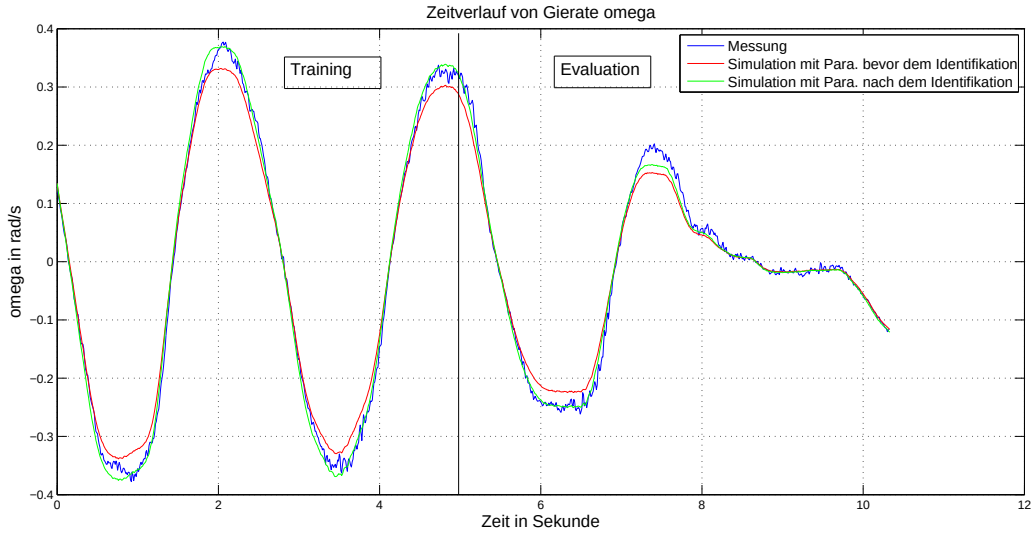


Abbildung 3.19: Zeitverlauf von Gierate ω

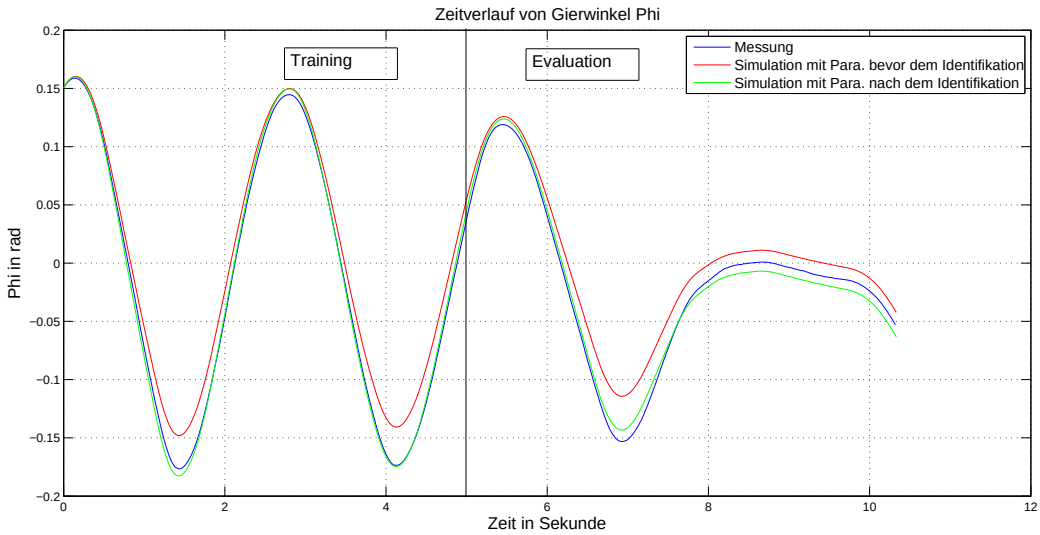


Abbildung 3.20: Zeitverlauf von Gierwinkel φ

Die Vergleichung von RMSE siehe untere Tabelle:

Zustandsgröße	V_y^{Ref}	ω	φ
RMSE Messung/Modell bevor der Identifikation	$0.0447 m/s$	$0.0219 rad/s$	$0.0196 rad$
RMSE Messung/Modell nach der Identifikation	$0.0407 m/s$	$0.0138 rad/s$	$0.0058 rad$

Tabelle 3.3: Ergebnisse von EM-Identifikation

In dieser Vergleichung weist den Modell nach der Identifikation ein kleiner Ungenauigkeit auf. Es hat die Genauigkeit nicht nur im Zeitintervall von “Training“ sondern auch in der Sequenz von ´´Evaluation´´ verbessert.

Die Beziehung zwischen Schräglaufwinkel und dem Verhältnis $\frac{F_y}{F_z}$ ist nach der Ausgabe der Simulation mit identifizierten Parametern dargestellt:

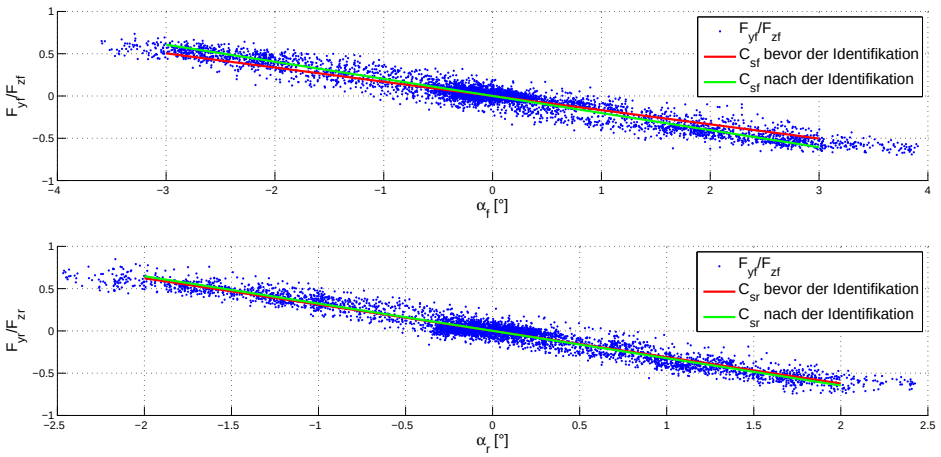


Abbildung 3.21: Darstellung der Beziehung zwischen Schräglaufwinkel und $\frac{F_y}{F_z}$

Die Ergebnisse der Identifikation werden auch in den Darstellungen von $\dot{y}$ und $\dot{\omega}$ mit Messwerten verglichen. Die Zeitabschnitt enthaltet die Sequenzen von "Training" und "Evaluation"

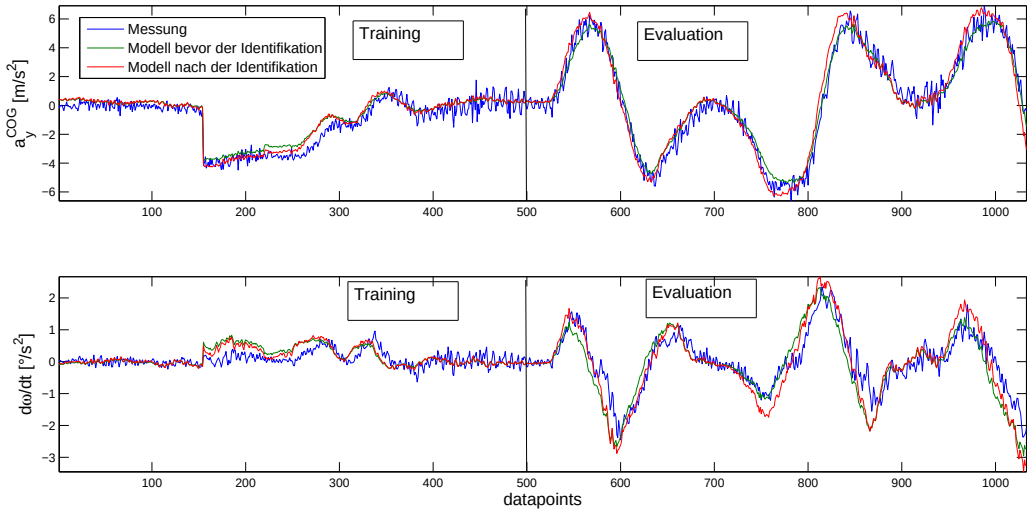


Abbildung 3.22: Vergleichung in $\dot{y}$ und $\dot{\omega}$

Die oberen Darstellungen zeigen: In der "Training" Sequenz weist der Parameter nach der Identifikation eine gute Korrektur in Vergleich mit der Parameter bevor der Identifikation auf. Aber in der "Evaluation" Sequenz ist diese Verbesserung nicht eindeutig.

3.4 Fazit

In diesem Kapital wird die EM-Identifikation in Matlab implementiert und bei der Simulation/Testfahrt demonstriert. Bei der Implementierung muss das Programm für den Rechenaufwand betrachtet werden, durch Simulation kann der Effekt der EM-Identifikation bewertet werden. Die Umgebung und Anfangsbedingungen werden vor dem Identifikation angemessen eingestellt. Bei Simulation erscheint es, dass die EM-Identifikation bei der Simulation gut funktioniert, danach wird die EM-Identifikation bei einer Testfahrt angewendet. Die Ergebnisse der Identifikation wird wieder in die Simulation eingebracht und durch den Vergleich zwischen der Ausgabe der Simulation und der Messung kann man die Qualität der Identifikation bewerten. Die Identifikation hat die Erwartung erfüllt.

4 Schlussbemerkungen und Ausblick

Dieses Arbeit erklärt die Theorie über den EM-Algorithmus bei der Systemidentifikation und demonstriert die Parameter-Identifikation für ein Fahrzeugmodell. Der EM-Algorithmus bestimmt die wahrscheinlichste Funktion, die der Likelihood Funktion des Systems unter den Bedingungen der ungenauen Parameter ähnelt. Durch Bestimmung der Maxima der Likelihood Funktion lassen sich die Parameter den realen Wert annähern. Zur Abbildung der Likelihood Funktion ist die Verteilungsfunktion zwischen den angrenzenden Zuständen $p(x_t|x_{t-1})$ und $p(y_t|x_t)$ einzusetzen. Mit Hilfe der Partikel-Methode stellt diese Verteilungsfunktion eine diskrete Form(mit Partikeln) dar, dies ermöglicht die EM-Identifikation. Diese Identifikationsmethode ist geeignet für die "Offline"Identifikation nichtlinearer Systeme.
Für ein Fahrzeugmodell sind einige Parameter schwer zu bestimmen, wie die Reifensteifigkeit, Trägheitsmoment oder die Position der Schwerpunkt, sie verändern sich durch unterschiedliche Situationen. Dafür ist die EM-Identifikation für die Bestimmung solcher Parameter eine gute Lösung. Bei der Modellierung des Fahrzeugs wird das Fahrzeug hauptsächlich in Querrichtung betrachtet. Viele Effekte wie Neigung in Längsrichtung und die Änderung der Geschwindigkeit sind berechnet, um das Fahrzeugmodell genauer zu beschreiben. Das lässt das Fahrzeugmodell ein nichtlineares System sein.
Vor dem Einsatz der Identifikationsmethode für Fahrzeuge ist die Simulation notwendig damit die Effekt überprüft werden können. Durch die Lösung der DGL des Fahrzeugmodells können virtuelle Messdaten hergestellt werden. Die Bestimmung der Varianzmatrix des Prozessrauschen Ξ ist sehr wichtig, da es über die Genauigkeit der Identifikation entscheidet. Eine falsche Einstellung für Ξ kann die Identifikation unsinnig machen. Hierbei ist die Einstellung von Ξ bei der Simulation unbedingt nötig.
Eine andere Lösung zur Bestimmung der Varianzmatrix Ξ ist die Erweiterung des Parametervektors. Die Varianzmatrix Ξ wird als Parameter zur Identifikation angenommen. Durch den EM-Algorithmus wird damit Ξ optimal bestimmt. Aber das bedeutet: Die Dimension des Zustands wird stark vergrößert und die Zeitdauer der Identifikation wird erhöht.
Die Testfahrt bietet die realen Messdaten an. Die Parameter werden identifiziert und in die Überprüfung eingebracht. Die Ergebnisse zeigen, dass die Identifikation die Parameter gut identifiziert hat.
Die EM-Identifikation ist eine ´´Offline´´ Methode, die Identifikation dauert sehr lang, der

Rechenaufwand muss also berücksichtigt werden. Die Dimension des Zustands, Anzahl der Partikel und die Zeitspanne der Messreihe muss kontrolliert werden, wenn die Qualität der Identifikation garantiert werden soll. Zusätzlich können als Gegenmaßnahme auch unnötige Rechnungen in der Programmierung vermieden werden um den Rechenaufwand gering zu halten.

A Anhang

A.1 Herleitung der Umsetzung physikalischer Größe von Referenz-Punkt zu Schwerpunkt für Fahrzeugmodell

Das Referenz-Punkt des Fahrzeugmodells ist als dem Mittelpunkt der hinterer Achse definiert. Es gibt irgendeinen Punkt, welches das Ursprung stationäres Koordinatensystems ist. Der Winkel zwischen stationärem Koordinatensystem und fahrzeugfestem Koordinatensystem wird als φ bezeichnet. Die Darstellung siehe folgende Abbildung:

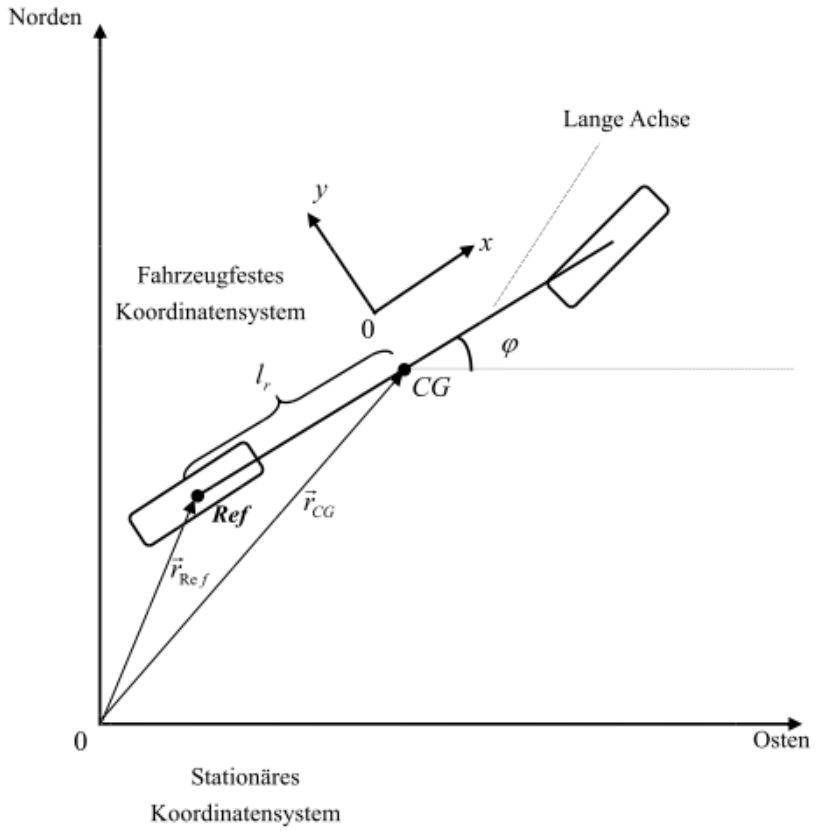


Abbildung A.1: Umsetzung von Referenz-Punkt zu Schwerpunkt am Fahrzeugmodell

Hier sind die Vektoren: $\vec{r}_{Ref} = \begin{bmatrix} x_{north}^{Ref} & y_{east}^{Ref} \end{bmatrix}^T$ und $\vec{r}_{CG} = \begin{bmatrix} x_{north}^{CG} & y_{east}^{CG} \end{bmatrix}^T$. Die Beziehung dazwischen ist:

$$\vec{r}_{CG} = \vec{r}_{Ref} + R(\varphi) \begin{pmatrix} l_r \\ 0 \end{pmatrix} \tag{A.1}$$

Hier ist die Matrix $R(\varphi) = \begin{pmatrix} \cos\varphi & -\sin\varphi \\ \sin\varphi & \cos\varphi \end{pmatrix}$ die rotatorische Matrix. Seine Ableitung lautet:

$$\dot{R}(\varphi) = \begin{pmatrix} -\omega\sin\varphi & -\omega\cos\varphi \\ \omega\cos\varphi & -\omega\sin\varphi \end{pmatrix} \tag{A.2}$$

$$= \underbrace{\begin{pmatrix} 0 & -\omega \\ \omega & 0 \end{pmatrix}}_{S(\omega)} \underbrace{\begin{pmatrix} \cos\varphi & -\sin\varphi \\ \sin\varphi & \cos\varphi \end{pmatrix}}_{R(\varphi)} \tag{A.3}$$

Dann die Ableitung oberer Formel wird ermittelt:

$$\dot{\vec{r}}_{CG} = \dot{\vec{r}}_{Ref} + \dot{R}(\varphi) \begin{pmatrix} l_r \\ 0 \end{pmatrix} \tag{A.4}$$

$$= R(\varphi)\vec{V}^{Ref} + R(\varphi)S(\omega) \begin{pmatrix} l_r \\ 0 \end{pmatrix} \tag{A.5}$$

$$\tag{A.6}$$

Der Vektor $\vec{V}^{Ref}$ ist der Geschwindigkeit im Fahrzeugkoordinatensystem. Es ist $\begin{pmatrix} V_x^{Ref} \\ V_y^{Ref} \end{pmatrix}$.
Weiter zur Abteilung:

$$\ddot{\vec{r}}_{CG} = R(\varphi) \left[\dot{\vec{V}}^{Ref} + S(\omega)\vec{V}^{Ref} + S(\dot{\omega}) \begin{pmatrix} l_r \\ 0 \end{pmatrix} + S(\omega)^2 \begin{pmatrix} l_r \\ 0 \end{pmatrix} \right] \tag{A.7}$$

$$= R(\varphi) \left[\dot{\vec{V}}^{Ref} + \omega \begin{pmatrix} -V_y^{Ref} \\ V_x^{Ref} \end{pmatrix} + l_r \begin{pmatrix} -\omega^2 \\ \dot{\omega} \end{pmatrix} \right] \tag{A.8}$$

Dann es ergibt sich:
Die Beschleunigung am Schwerpunkt im Fahrzeugkoordinatensystem:

$$\begin{pmatrix} \dot{V}_x^{CG} \\ \dot{V}_y^{CG} \end{pmatrix} = \dot{\vec{V}}^{Ref} + \omega \begin{pmatrix} -V_y^{Ref} \\ V_x^{Ref} \end{pmatrix} + l_r \begin{pmatrix} -\omega^2 \\ \dot{\omega} \end{pmatrix} \tag{A.9}$$

A.2 Markov-Prozess

Es gibt einen zufälligen Prozess, wenn die Wahrscheinlichkeit von sein aktuellem Zustand nur bezüglich auf dem Zustand an letztem Zeitpunkt ist, ist dieser Prozess als Markov-Prozess definiert(siehe folgende Abbildung). Ein Systemmodell mit Formel (2.24) und (2.25) beschrieben wird, ist einer typischer Markov-Prozess.

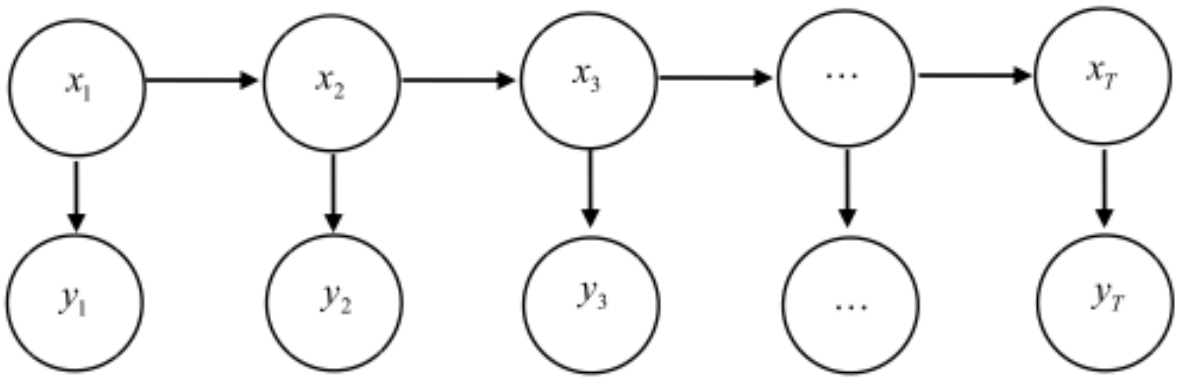


Abbildung A.2: Markov-Kette

Die obere Abbildung zeigt, dass die Zustand an jeweiligem Zeitpunkt nur vom Zustand an letztem Zeitpunkt abhängig ist. Die Wahrscheinlichkeitsfunktion zwischen zwei angrenzenden Zustande lautet: $p(x_t|x_{t-1})$, damit der gesamte Wahrscheinlichkeitsfunktion einer Messreihe unter der Beeinflussung von Parameter θ wird als unterem Formel bezeichnet:

$$p_\theta(y_{1:T}) = \prod_{t=2}^{T} p_\theta(y_t|y_{t-1}) = p_\theta(x_1) \prod_{t=2}^{T} p_\theta(x_t|x_{t-1}) \prod_{t=1}^{T} p_\theta(y_t|x_t) \tag{A.10}$$

Dann die Linkelihood Funktion weist in Logarithmus auf:

$$L_\theta(y_{1:T}) = \log(p_\theta(y_{1:T})) = \log(x_1) + \sum_{t=2}^{T} \log(p_\theta(x_t|x_{t-1})) + \sum_{t=1}^{T} \log(p_\theta(y_t|x_t)) \tag{A.11}$$

A.3 Approximation der Verteilungsfunktion in diskretem Form

Die direkte Ermittelung der Verteilungsfunktion ist bei Anwendung unmöglich, stattdessen mit vielen Partikel die Verteilung in diskretem Form zu approximieren. Die Verteilung wird als gaußsche Verteilung angenommen. Wie z.B: es gibt die Funktion $y = h(x)$ mit der Steuerung σ der Unsicherheit von y, dafür lautet der Verteilungsfunktion: $p(y|x) \sim N(y; h(x), \sigma)$, seine Darstellung siehe folgendes:

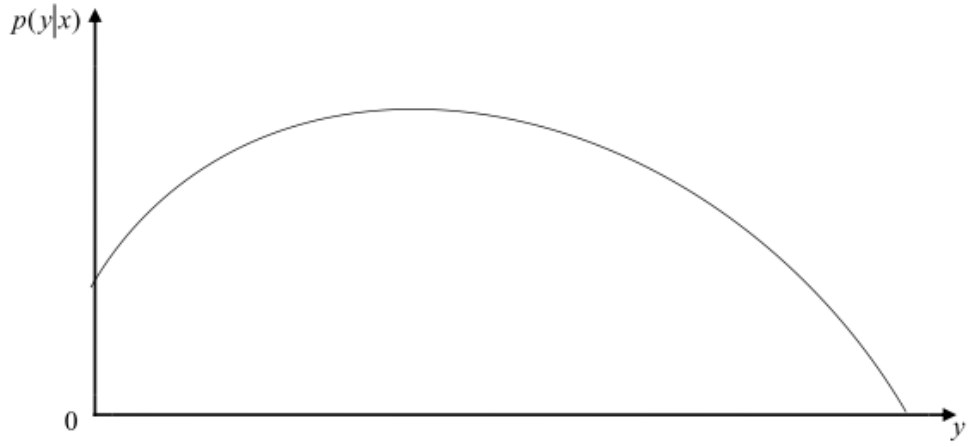


Abbildung A.3: Verteilungsfunktion

Jetzt gibt es N Partikel, solche Partikel verteilen gaußsche um x , jeder Partikel ist als x^i bezeichnet, der einzelner Partikel wird in die Funktion $h(x)$ eingebracht und gewichtet: $w^i = N(y; h(x^i), \sigma)$, dann normiert den Gewicht $\tilde{w}^i = \sum_{i=1}^{N} w^i$. Der Verteilungsfunktion wird diskretisiert:

$$p(y|x) \approx \sum_{i=1}^{N} \tilde{w}^i \delta(y - y^i) \tag{A.12}$$

Diese Approximation ist in folgender Abbildung dargestellt:

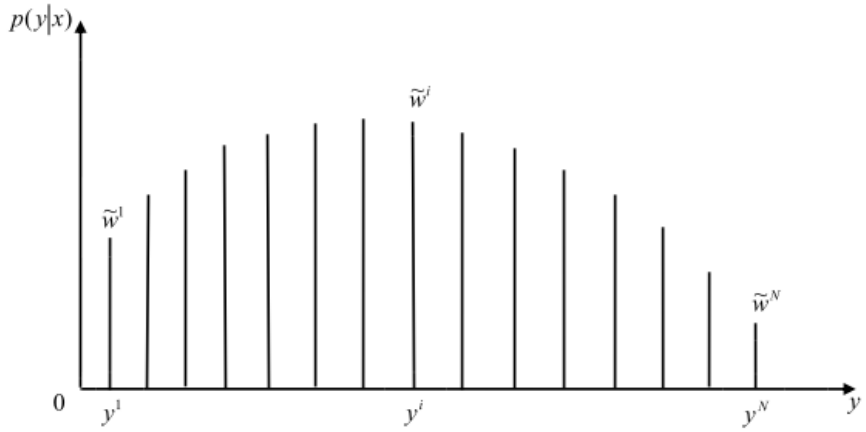


Abbildung A.4: Approximation der Verteilungsfunktion

A.4 Resampling Methode

In Partikel-Filter ist es wichtig den Partikel mit geringem Gewicht aufzugeben. Deswegen ist des Resampling nötig, die allgemeine Vorgehensweise ist, von alten Partikel sucht den einige Partikel nach den Gewicht aus. Dann jeder ausgesuchter Partikel erzeugt seine "Sohn-Partikel". Der Zahl von "Sohn-Partikel" hängt von dem Gewicht ihren "Vater-Partikel" ab. Der gesamter Zahl allen "Sohn-Partikel" ist gleich den gesamten Zahl von "Vater-Partikel". Alle "Sohn-Partikel" wird den gleichen Gewichtig $\frac{1}{Zahl_{gesamt}}$ verteilt. Für Resampling stellt viele Methoden zur Verfügung, hier bei werden drei Reference Methode vorgestellt.

Multinomial Resampling

Das ist die allgemein Methode für Resampling, einer zufälliger Zahl inzwischen Null und Eins ist generiert für die Auswahl in "Vater-Partikel". Die Vorgehensweise siehe folgendes(Sileshi u. a., 2013):

- generiert N unabhängig zufälligen Zahl $u \in [0 \quad 1]$, berechnet die Summe von normiertem Gewicht: $s_i = \sum_{j=1}^{i} \tilde{w}_t^j$.
- bestimmt s_i, damit $u \in [s_{i-1} \quad s_i]$, der ite Partikel ist ausgesucht.
- geben $\{x_t^i \quad \tilde{w}_t^i\}$ für $j = 1, ..., N$, generiert neue Partikel x_t^j durch Vermehrung von Partikel x_t^i nach dem Gewicht $\tilde{w}_t^i$.
- $\tilde{w}_t^i = \frac{1}{N}$.

Systematic Resampling

Systematic Resampling ist auch eine beliebte Methode, es erstellt einen zufälligen Zahl, mit dem den Resampling auszuführen, die Vorgehensweise siehe folgendes(Sileshi u. a., 2013):

- Erstellt einen Zufälligen Zahl u, welcher $u \in \left[0 \quad \frac{1}{N}\right]$ erfüllt und definiert $u_i = u_1 + \frac{i-1}{N}$ für $i = 2, ..., N$.
- Mit u_i bestimmt den Partikel x_i wie bei Multinomial Resampling.

Stratified Resampling

Bei Stratified Resampling erstellt N zufälligen Zahl u_i und damit für Resampling eingebracht. Die Vorgehensweise siehe folgendes(Sileshi u. a., 2013):

- Generiert N zufälligen Zahl: $u_i = \frac{(i-1)+\tilde{u}_i}{N}$ mit $\tilde{u}_i \in [0 \quad 1]$
- Mit u_i bestimmt den Partikel x_i wie bei Multinomial Resampling.

A.5 Runge-Kutta-Verfahren

Die Runge-Kutta-Verfahren ist das Einschrittverfahren zur Lösung der DGL in der numerischen Mathematik. Dieses Verfahren ist nach Carl Runge und Martin Wilhelm Kutta benannten(siehe Hermann, 2004, S. 185). Für die DGL $\dot{x} = f(t, x)$ lautet die Lösung für nächsten Zeitpunkt:

$$x_{t+1} = x_t + \int_t^{t+1} f(t,\ x)dx \tag{A.13}$$

Mit Einschrittverfahren wird oberer Formel so gelöst:

$$x_{t+1} = x_t + \Delta t \cdot \varphi(t,\ x_t) \tag{A.14}$$

Diese Formel heißt explizites, m-stufiges Runge-Kutta-Verfahren, wenn die Verfahrensfunktion $\varphi(t,\ x_t)$ eine Linearkombintion von Funktionswerten $f(t,\ x)$ an verschiedenen Stellen (t, x_t) ist. Die Verfahrensfunktion $\varphi(t,\ x_t)$ lautet:

$$\varphi(t,\ x_t) = a_1 k_1(t,\ x_t) + a_2 k_2(t,\ x_t) + \ldots + a_m k_m(t,\ x_t) \tag{A.15}$$

Mit:

$$k_1(t,\ x_t) = f(t,\ x_t) \tag{A.16}$$
$$k_2(t,\ x_t) = f(t + b_2\Delta t,\ x_t + \Delta t \cdot c_{21} \cdot k_1(t, x_t) \tag{A.17}$$
$$k_3(t,\ x_t) = f(t + b_3\Delta t,\ x_t + \Delta t\left[c_{31}k_1(t, x_t + c_{32}k_2(t, x_t\right]) \tag{A.18}$$
$$\vdots \tag{A.19}$$
$$k_m(t,\ x_t) = f(t + b_m\Delta t,\ x_t + \Delta t \sum_{j=1}^{m-1} c_{mj}k_j(t,\ x_t)) \tag{A.20}$$

Davon geht aus, die Verfahrensfunktion lautet bei 2-stufigem explizitem Runge-Kutta-Verfahren:

$$a_1 = \frac{1}{2}; a_2 = \frac{1}{2}; \tag{A.21}$$
$$b_2 = 1; \tag{A.22}$$
$$c_{21} = 1 \tag{A.23}$$
$$x_{t+1} = x_t + \frac{\Delta t}{2}\left[f(t,\ x_t) + \Delta t \cdot f(t,\ x_t)\right] \tag{A.24}$$

Für klassisches 4-stufiges Runge-Kutta-Verfahren:

$$a_1 = \frac{1}{6}; a_2 = \frac{1}{3}; a_3 = \frac{1}{3}; a_4 = \frac{1}{6} \tag{A.25}$$

$$b_2 = \frac{1}{2}; b_3 = \frac{1}{2}; b_4 = 1 \tag{A.26}$$

$$c_{21} = \frac{1}{2}; c_{31} = 0; c_{32} = \frac{1}{2}; c_{41} = 0; c_{42} = 0; c_{43} = 1 \tag{A.27}$$

$$k_1 = f(t,\ x_t) \tag{A.28}$$

$$k_2 = f(t + \frac{\Delta t}{2},\ x_t + \frac{\Delta t}{2} \cdot k_1) \tag{A.29}$$

$$k_3 = f(t + \frac{\Delta t}{2},\ x_t + \frac{\Delta t}{2} \cdot k_2) \tag{A.30}$$

$$k_4 = f(t + \Delta t,\ x_t + \Delta t \cdot k_3) \tag{A.31}$$

$$x_{t+1} = x_t + \frac{\Delta t}{6} [k_1 + 2k_2 + 2k_3 + k_4] \tag{A.32}$$

Literaturverzeichnis

Hermann, M. (2004). *Numerik gewöhnlicher Differentialgleichungen: Anfangs-und Randwertprobleme*. Walter de Gruyter.

Isermann, R. (2008). *Mechatronische Systeme*. Schriftenreihe / Institut für Mess- und Regelungstechnik, Karlsruher Institut für Technologie. Springer-Verlag Berlin Heidelberg.

Krengel, U. (1988). „Markoffsche Ketten“. In: *Einführung in die Wahrscheinlichkeitstheorie und Statistik*. Springer, S. 187–226.

Ljung, L. (1998). *System identification*. Springer.

Pacejka, H. (2005). *Tire and vehicle dynamics*. Elsevier.

Rajamani, R. (2011). *Vehicle dynamics and control*. Springer Science & Business Media.

Särkkä, S. (2013). *Bayesian filtering and smoothing*. 3. Cambridge University Press.

Schön, T. (2009). „An explanation of the expectation maximization algorithm“. In:

Sileshi, B., Ferrer, C & Oliver, J (2013). „Particle filters and resampling techniques: Importance in computational complexity analysis“. In: *Design and Architectures for Signal and Image Processing (DASIP), 2013 Conference on*. IEEE, S. 319–325.

Van Der Merwe, R. (2004). „Sigma-point Kalman filters for probabilistic inference in dynamic state-space models“. Diss. Oregon Health & Science University.

Wan, E., Van Der Merwe, R. u.a. (2000). „The unscented Kalman filter for nonlinear estimation“. In: *Adaptive Systems for Signal Processing, Communications, and Control Symposium 2000. AS-SPCC. The IEEE 2000*. IEEE, S. 153–158.

Wenzel, T. A., Burnham, K., Blundell, M. & Williams, R. (2006). „Dual extended Kalman filter for vehicle state and parameter estimation“. In: *Vehicle System Dynamics* 44.2, S. 153–171.

Wills, A., Schön, T. B., Ninness, B. u.a. (2008). „Parameter estimation for discrete-time nonlinear systems using EM“. In: S. 1–6.