Echtzeit-Implementierung eines algebraischen Ableitungsschätzverfahrens

Realtime Implementation of an Algebraic Derivative Estimation Scheme

Josef Zehetner, Johann Reger und Martin Horn

In der vorliegenden Arbeit wird die Echtzeitimplementierung eines algebraischen Verfahrens zur Schätzung von Zeitableitungen gemessener Signale vorgestellt. Ausgehend von einer Taylorreihendarstellung des betrachteten Signals werden mit Methoden der Operatorenrechnung Berechnungsvorschriften für Zeitableitungen beliebiger Ordnung hergeleitet. Diese können effizient in Echtzeit ausgewertet werden. Im vorliegenden Fall wird Matlab/Simulink als Werkzeug eingesetzt. Mit Hilfe eines Codegenerators und eines Target-Compilers können die Echtzeitroutinen mühelos auf verschiedene Zielplattformen übertragen werden. An Hand eines praktischen Anwendungsbeispiels wird die Leistungsfähigkeit des Ansatzes demonstriert.

In this article, a realtime implementation of an algebraic method for estimating the time derivatives of measured time signals is presented. Based on a Taylor series expansion of the time signal, elementary techniques from operational calculus are used for deriving estimator formulas for time derivatives of arbitrary order. These formulas may be evaluated very efficiently, in realtime. This task is solved by Matlab/Simulink by means of which resorting to code generation libraries and target link compilers realtime routines can easily be transferred to different target architectures. A laboratory setup demonstrates the applicability and impressing performance in practice.

Schlagwörter: Operatorenrechnung, Echtzeitsystem, Matlab, Simulink, S-Function

Keywords: Operational calculus, realtime system, Matlab, Simulink, S-Function

1 Einleitung

Die Lösung einer Vielzahl von Problemen in der Regelungstechnik, Fehlerdiagnose, Signalverarbeitung und Signalübertragung erfordert die zuverlässige Schätzung von zeitlichen Ableitungen gemessener Signale, welche in der Praxis oft starkem Rauschen unterliegen. Häufig greift man dabei auf numerische Methoden der Differentiation zurück [4]. Beispiele hierfür sind das Savitzky-Golay-Differentiationsschema [9], das Wavelet-Differentiationsschema [2] und das Gemittelte-finite-Differenzen-Verfahren [1]. Einige der genannten Verfahren wurden bereits erfolgreich zur Zustandsbeobachtung in nichtlinearen Systemen eingesetzt [3; 5; 13].

Eine neuere Methode, die im Wesentlichen auf Operatorenrechnung beruht, wurde kürzlich auf internationalen Sommerschulen vorgestellt [17–19]. Erste Ergebnisse wurden in [6; 15] publiziert. Die vorliegende Arbeit zeigt die Echtzeit-Implementierung dieser Methode und bietet somit die Grundlage für den praktischen Einsatz des gezeigten Ableitungsschätzverfahrens. Dabei wird bereits in der Simulation besonderes Augenmerk auf die Überprüfbarkeit der Implementation des Verfahrens gelegt.

Die mathematischen Grundlagen des Verfahrens sind in Abschnitt 2 beschrieben. Die notwendige Adaption des Verfahrens für den praktischen Einsatz mit abgetasteten Zeitsignalen wird in Abschnitt 3 gezeigt. Ein detaillierter Überblick über die eigentliche Implementierung sowie mögliche Erweiterungen wird in Abschnitt 4 gegeben. Ergebnisse, die die Leistungsfähigkeit der Methode unterstreichen, werden in Abschnitt 5 gezeigt. Eine Zusammenfassung sowie einen Ausblick auf weitere Anwendungen und Einsatzmöglichkeiten enthält abschließend Abschnitt 6.

2 Algebraische Ableitungsschätzung

Im Folgenden soll ein algebraisches Verfahren zur Schätzung zeitlicher Ableitungen von Signalen vorgestellt werden. Die Bezeichnung *Algebraische Ableitungsschätzung* orientiert sich begrifflich an den Arbeiten von Fliess, Sira Ramírez et al. Grundlegende Vorarbeiten finden sich in [6; 7], Erweiterungen wurden in [10; 14; 15] veröffentlicht.

Gegeben sei eine (reellwertige) Funktion $y(t)$, von der eine zeitliche Ableitung beliebiger Ordnung ermittelt werden soll. Es wird vorausgesetzt, dass $y(t)$ für $t \geq 0$ mit Hilfe einer Taylorreihe

$$y(t) \approx y_N(t) = \sum_{i=0}^{N} \alpha_i \frac{t^i}{i!} \tag{1}$$

der Ordnung N angenähert werden kann. Die Koeffizienten

$$\alpha_i = \left.\frac{d^i y}{dt^i}\right|_{t=0} =: y^{(i)}(t=0) \tag{2}$$

entsprechen dabei den gesuchten zeitlichen Ableitungen der Funktion $y(t)$ zum Zeitpunkt $t = 0$. Das Ziel der folgenden Ausführungen besteht darin, Berechnungsvorschriften für diese Koeffizienten anzugeben. Deren Ermittlung gestaltet sich im Bildbereich besonders einfach. Dazu wird die Funktion $y_N(t)$ im Bildbereich betrachtet, d. h.

$$y_N(t) = \sum_{i=0}^{N} \alpha_i \frac{t^i}{i!} \quad \circ\!\!-\!\!\bullet \quad Y_N(s) = \sum_{i=0}^{N} \frac{\alpha_i}{s^{i+1}} \,. \tag{3}$$

Die Idee besteht nun darin, ausgehend von (3) für einen beliebigen Koeffizienten α_j, $j = 0, 1, \ldots, N$, einen expliziten Ausdruck anzugeben, d. h. die j-te Ableitung $y^{(j)}(0)$ zu bestimmen. Dies gelingt in einfacher Weise, denn in Gleichung (3) treten die Koeffizienten linear auf; sie sind mit Hilfe von $y_N(t)$ „linear identifizierbar“ [8].

Nach einigen geschickten Manipulationen von Gleichung (3), die in Anhang A angeführt sind, und einer anschließenden Rücktransformation in den Zeitbereich erhält man folgenden Ausdruck für den gesuchten Koeffizienten:

$$\alpha_j = \int_0^t \Pi_j(t, \tau)\, y_N(\tau)\, d\tau \,. \tag{4}$$

Die Funktion $\Pi_j(t, \tau)$ hängt unter anderem von der Ordnung N der Taylorreihe (1) und einer weiteren, vorgebbaren Konstante ν ab. Diese ist ganzzahlig sowie nichtnegativ. Sie dient der Reduktion der Empfindlichkeit des Verfahrens gegenüber hochfrequentem Messrauschen. Aus Gründen der Übersichtlichkeit ist $\Pi_j(t, \tau)$ im Anhang A zu finden (13).

Bei der praktischen Anwendung von (4) ist $y_N(t)$ durch den gemessenen Wert $y(t)$ zu ersetzen. Als Zeitfensterbreite, d. h. als obere Integrationsgrenze in (4), kann theoretisch ein beliebig kleiner Wert $t = T > 0$ gewählt werden. Bei der Wahl der Fensterbreite T ist ein Kompromiss zwischen Rechenaufwand und Güte der Schätzung zu finden. T sollte zwar einerseits so klein gewählt werden, dass der zeitliche Aufwand zur Auswertung von (4) vertretbar bleibt, andererseits erhöht eine Vergrößerung von T die Qualität der Schätzung bei verrauschten Messsignalen $y(t)$. Dies ist darauf zurückzuführen, dass hier eine Tiefpassfilterung wirksam wird (siehe auch Anhang A).

Man beachte, dass Gleichung (4) lediglich einen Schätzwert für die Ableitungen von $y(t)$ zum Zeitpunkt $t = 0$ liefert. Schätzwerte zu beliebigen Zeitpunkten $t > 0$ kann man aber auf sehr einfache Weise bestimmen: Anstatt die Messwerte von $y(t)$ im Intervall $[t, t+T]$ zur Ermittlung der Ableitung $y^{(j)}(t)$ zu verwenden, werden vergangene Werte von $y(t)$ auf dem Intervall $[t, t-T]$, d. h. *gespiegelt*, eingesetzt. Somit kann erneut die Taylorreihe (1) herangezogen werden, wenn man wegen der Zeitumkehr ein negatives Vorzeichen bei ungeradzahligen Ableitungen einbezieht. Damit erhält man letztendlich für die j-te Zeitableitung, $j = 0, 1, \ldots, N$, die Abschätzung

$$y^{(j)}(t) = (-1)^j \int_0^T \Pi_j(T, \tau)\, y(t-\tau)\, d\tau \,. \tag{5}$$

3 Schätzung bei abgetasteten Signalen

In diesem Abschnitt wird der Tatsache Rechnung getragen, dass bei praktischen Anwendungen ein Signal $y(t)$ nicht kontinuierlich erfasst wird, sondern typischerweise nur zu äquidistanten Zeitpunkten $t = i\,T_s$. Hierbei ist T_s die sogenannte Abtastzeit (*sampling time*). Das Signal, dessen zeitliche Ableitungen ermittelt werden sollen, liegt somit in Form einer Zahlenfolge

$$(y_i) = (y_0, y_1, y_2, \ldots) \text{ mit } y_i = y(i\,T_s),\ i = 0, 1, 2, \ldots$$

vor. Zwischen zwei aufeinander folgenden Abtastzeitpunkten wird $y(t)$ als konstant angenommen, d. h.

$$y(t) = \text{konst.} \quad \text{für} \quad i\,T_s \leq t < (i+1)T_s,\ i = 0, 1, 2, \ldots$$

In Gleichung (5) wird die Fensterbreite T als ganzzahliges Vielfaches der Abtastzeit T_s gewählt, d. h. es gilt

$$T = M\,T_s \quad \text{wobei} \quad M = 1, 2, 3, \ldots$$

Diese Wahl ermöglicht es, das Integral in Gleichung (5) durch entsprechende Summen zu ersetzen und somit eine Approximation $y_i^{(j)} \approx y^{(j)}(t)$ der gesuchten Ableitung zu erhalten. Eine ausführliche Darstellung der beschriebenen Vorgangsweise ist in Anhang B zu finden. Für die gesuchte j-te zeitliche Ableitung zum Zeitpunkt $t = i\,T_s$ erhält man somit

$$y_i^{(j)} = \sum_{k=1}^{M} \Pi_{j,k}\, y_{i-k+1} \tag{6}$$

für $j = 0, 1, 2, \ldots, N$, wobei $\Pi_{j,k}$ analog zum zeitkontinuierlichen Fall von N und ν sowie von der Konstanten M abhängt. Der Ausdruck für $\Pi_{j,k}$ ist, wiederum aus Gründen der Übersichtlichkeit, als (15) im Anhang B zu finden. Interpretiert man (6) als ein zeitdiskretes System mit der Eingangsfolge (y_i) und der Ausgangsfolge $(y_i^{(j)})$, so erkennt man, dass es sich hierbei um ein sogenanntes

nichtrekursives Filter handelt. Das Element $y_i^{(j)}$ errechnet sich nämlich *ausschließlich* aus Elementen der Eingangsfolge (y_i). Man beachte auch, dass die in (6) benötigten Größen $\Pi_{j,1}, \Pi_{j,2}, \ldots, \Pi_{j,M}$ offline aus den gewählten Parametern N, ν, M und j berechnet werden können. Ihre Ermittlung kostet somit in einer Echtzeitanwendung keine kostbare Rechenzeit.

4 Implementierung

4.1 Motivation

Die Ermittlung von Zeitableitungen von Signalen in Echtzeit-Anwendungen gestaltet sich oft recht schwierig, da – im Gegensatz zur offline-Schätzung – die zur Verfügung stehende Rechenzeit begrenzt ist. In den folgenden Abschnitten wird gezeigt, dass das in Abschnitt 3 vorgestellte Verfahren auch in Echtzeit-Anwendungen einsetzbar ist.

Ziel ist es, den vorgestellten Algorithmus möglichst allgemein und effizient zu implementieren. Darunter verstehen wir, dass die zur Ausführung notwendigen Parameter vom Anwender vorgegeben werden können und diese auch *während der Laufzeit* variiert werden können. Die Variation der Parameter ist besonders für Echtzeitsysteme relevant, da hier die optimale Kombination der Parameter im Versuch ermittelt werden kann, ohne das System neu initialisieren zu müssen. Die zur Schätzung der Ableitungen notwendige Rechenzeit muss innerhalb vorgegebener Grenzen bleiben.

Darüber hinaus müssen die erstellten Funktionen sowohl auf Simulations- als auch auf Echtzeitplattformen eingesetzt werden können. Dies gewährleistet, dass die korrekte Funktionalität der Software bereits in der Simulation überprüft werden kann.

4.2 Zielplattform

Als Zielplattform sind prinzipiell zwei Arten von Systemen denkbar, einerseits Simulationen, die nicht zwingend in Echtzeit ablaufen müssen (*offline*) und andererseits in Echtzeit ablaufende Systeme (*online*), sogenannte Realtime-Systeme.

Als Simulationsumgebung setzen wir Matlab/Simulink ein. Als Echtzeit-Anwendungen setzen wir Systeme ein, für die von Matlab/Simulink aus entsprechende Anwendungen generiert werden können. Dabei werden Simulink-Modelle mittels eines Code-Generators in C-Sourcecode übersetzt. Der generierte Sourcecode wird mittels eines Target-Compilers in eine auf der Zielplattform lauffähige Anwendung übersetzt. Dies geschieht üblicherweise vollautomatisch.

4.3 Simulink S-Function

Die *unangenehme* Eigenschaft der vorgestellten Berechnung ist die faltungsähnliche Struktur der Integrale (5) bzw. Summen (6). Diese Struktur macht den Einsatz von Schleifen zur Berechnung der Summen unumgänglich (siehe Abschnitt 3). Da die Anwendung auf den in Abschnitt 4.2 gewählten Plattformen ablaufen soll, bietet sich der Einsatz einer Simulink *S-Function* an.

Eine S-Function ist die Beschreibung eines Matlab/Simulink-Blocks mit Hilfe einer Programmiersprache [11]. Es stehen eine Reihe von möglichen Programmiersprachen zur Verfügung, etwa die Matlab-eigene Skriptsprache MATLAB, oder auch höhere Programmiersprachen wie z. B. C, C++, Ada und Fortran. Im technischen Bereich hat sich, wie auch in weiten Bereichen der Informatik, die Sprache C bzw. C++ etabliert. Die von uns gewählten Realtime-Umgebungen setzen zwingend die Sprache C voraus, weshalb im Weiteren lediglich auf eine Implementierung von S-Functions in C eingegangen wird.

Der C-Code kann, wie in Abschnitt 4.1 gefordert, *unabhängig* von der eingesetzten Zielplattform erstellt werden. Die semantische Verifikation der Implementierung des Verfahrens kann somit bereits mit der Simulationsumgebung vorgenommen werden, da ein und derselbe Sourcecode für alle Zielplattformen eingesetzt wird.

Durch Einsatz von Compiler-Direktiven können spezielle Eigenschaften der Systeme verwendet werden, z. B. Warn-/Infofenster für die Simulationsumgebung, die für andere Systeme nicht zweckmäßig sind.

4.4 Datenstruktur

Der gewählte Algorithmus benötigt die im Abschnitt 3 genannten Parameter. Die im Zeitfenster konstanter Breite $T = [\, t - (M-1)\, T_s,\ t\,]$ liegenden, vergangenen Messwerte werden zur Berechnung des aktuellen Schätzwerts $y_i^{(j)}$ eingesetzt, wobei $t = i\, T_s,\ i = 0, 1, 2, \ldots,$ gilt (siehe Abschnitt 3). Das Fenster *gleitet* zu jedem Abtastschritt um ein Zeitintervall weiter, wobei der aktuelle Messwert y_i angefügt und der Messwert y_{i-M} entfernt wird.

Die gespeicherten Werte können äußerst effizient in einem *Ringbuffer* abgelegt werden. Es handelt sich hierbei um einen Speicherbereich (Array) der Größe $M = T/T_s$. Schreib- und Lesezugriffe erfolgen immer über *modulo*-Operationen[1]. Beim Einfügen eines neuen Messwerts in den Ringbuffer wird der jeweils letzte Wert[2] überschrieben.

4.5 Softwarestruktur

In Bild 1 ist das Flussdiagramm der realisierten Funktion dargestellt. Es soll darauf hingewiesen werden, dass diese Struktur für *jeden* Abtastzeitschritt durchlaufen werden muss.

Die notwendige Schleife zur Berechnung der Summe, ist der kritische Bereich (Felder 5 und 6). In der Variable `RB` ist der *Ringbuffer* abgelegt, `BPos` ist die aktuelle Position im Ringbuffer, d. h. der Messwert zum Zeitpunkt t wird an dieser Stelle mittels einer modulo-Operation abgelegt

[1] Eine *modulo*-Division ergibt den Rest der Division zweier Ganzzahlen, z. B. $p = 96\ modulo\ 80 = 16$.
[2] Messwert zum Zeitpunkt $(t - T) = (t - M\, T_s)$.

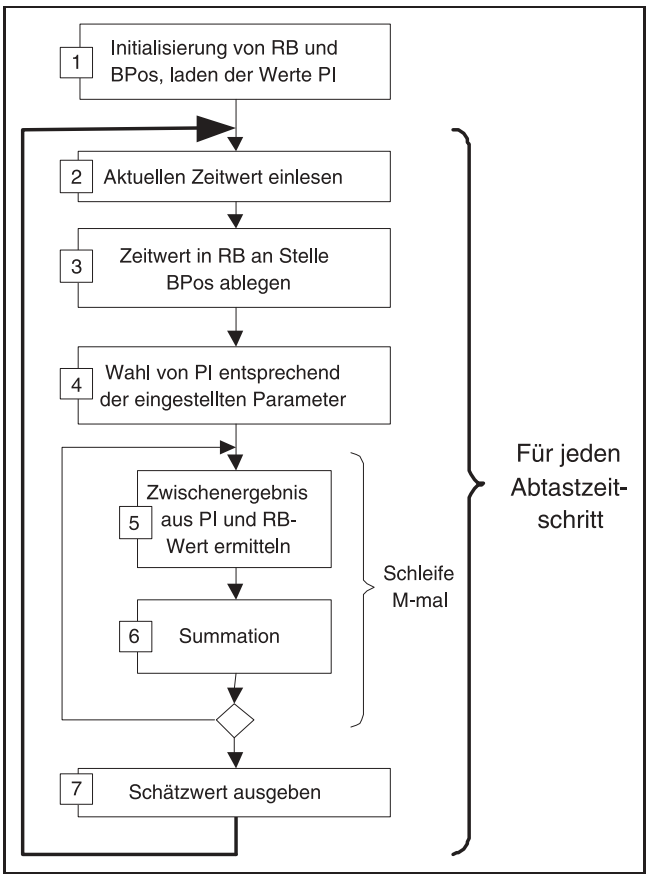

Bild 1: Flussdiagramm (siehe Text).

(Felder 2 und 3). In Feld 4 wird $\Pi_{j,k}$ entsprechend der vom Benutzer eingestellten Parameter ausgewählt (siehe Abschnitt 3). Diese Auswahl geschieht zu jedem Zeitschritt, d. h. nicht nur während der Initialisierung (siehe Abschnitt 4.1). Für die Versuche standen etwa 50 offline berechnete Datensätze für $\Pi_{j,k}$ für unterschiedliche Kombinationen von j, N, ν und M zur Verfügung. In Feld 5 wird ein aus dem Ringbuffer mittels modulo-Operation ausgelesener Wert mit dem passenden Wert von $\Pi_{j,k}$ multipliziert. In Feld 6 werden die Zwischenergebnisse aufsummiert.

`RB` und `BPos` sind statische Variablen, d. h. der Inhalt der Variablen muss auch beim Aufruf der Funktion zum Zeitpunkt $t = (i+1)\,T_s$ zur Verfügung stehen (beim neuerlichen Aufruf einer Funktion werden im Allgemeinen alle Variablen neu initialisiert!). In Matlab/Simulink stehen zu diesem Zweck sogenannte *Work Vectors* [11] zur Verfügung. Diese werden innerhalb der zentralen *SimStruct*-Variable abgelegt und stehen bei jedem Simulationsschritt unverändert zur Verfügung, erfüllen also die von uns gewünschten Anforderungen. Es steht jeweils ein Array frei wählbarer Größe für verschiedene Datentypen zur Verfügung, unter anderem für die Typen *integer* und *double*, wobei der Zugriff auf diese Speicherbereiche mittels spezieller C-Konstrukte erfolgt.

4.6 Anmerkungen

Eine Besonderheit des präsentierten Algorithmus ist, dass Ableitungen unterschiedlichen Grades zeitgleich mittels *einer* S-Function berechnet werden können. Der zusätzliche Rechenaufwand ist hierbei sehr gering, da der gesamte Zusatzaufwand, welcher durch Einsatz einer Schleife entsteht, nur einmal für alle Berechnungen notwendig ist. Die Parametrierung erfolgt dann über Vektoren anstatt über Skalare, entsprechend ergibt sich ein mehrdimensionaler Ausgang.

Weiters können auch Ableitungen für jede Komponente einer vektoriellen Eingangsgröße berechnet werden, z. B. analog zum in Abschnitt 5.2 gezeigten Beispiel, die jeweils erste Ableitung von vier Radgeschwindigkeiten eines Fahrzeugs.

Bei Echtzeit-Systemen mit hohen Abtastraten steht nur eine äußerst beschränkte Anzahl an Schleifendurchläufen pro Zeitschritt zur Verfügung (z. B. bei der im Abschnitt 5.1 verwendeten MicroAutoBox: bei einer Abtastzeit von $T_s = 0{,}001\,\text{sec}$ stehen maximal $M_{max} \approx 150$ Schleifendurchläufe zur Verfügung was einer maximalen Fensterbreite $T_{max} \approx 0{,}15\,\text{sec}$ entspricht).

Um auch unter diesen Bedingungen eine ausreichende Rauschunterdrückung zu gewährleisten, kann etwa lediglich jeder L-te Messwert ($L \geq 2$) zur Berechnung herangezogen werden. Durch diese Maßnahme wird bei gleichbleibender Anzahl an Schleifendurchläufen die Fensterbreite um den Faktor L vergrößert und gleichzeitig eine Vorfilterung vorgenommen. Weiters kann nur zu jedem P-ten Zeitschritt ein neuer Ausgabewert berechnet werden (dies entspricht einer Nachabtastung). Die Schleifendurchläufe M werden somit auf P Zeitschritte aufgeteilt, wobei pro Zeitschritt somit $\tilde{M} = M/P$ Berechnungen notwendig sind.

Die Zahlen $\Pi_{j,k}$ werden offline für bestimmte Kombinationen der Parameter j, N, ν und M berechnet. Eine Ermittlung der Werte in Echtzeit wäre wünschenswert, um die Verwendung beliebiger Parameterkombinationen zu ermöglichen. Bei zeitkritischen Echtzeitsystemen reicht hierfür möglicherweise die zur Verfügung stehende Rechenzeit nicht aus. Dieses Problem kann umgangen werden, indem, bei einer Änderung der Parameter zum Zeitpunkt $t = i\,T_s$, kein neuer Schätzwert $y_i^{(j)}$ berechnet wird, sondern stattdessen die Berechnung der M Zahlen für $\Pi_{j,k}$ vorgenommen wird. Für diesen Abtastzeitschritt kann z. B. $y_i^{(j)} = y_{i-1}^{(j)}$ gesetzt werden. Zum Zeitpunkt $t = (i+1)\,T_s$ wird der Schätzwert $y_{i+1}^{(j)}$ dann mit den neu ermittelten Zahlen $\Pi_{j,k}$, d. h. mit den neuen Parametern, berechnet.

5 Ergebnisse

In diesem Abschnitt wird die Anwendung der entworfenen Toolbox in einer Echtzeitanwendung beschrieben.

5.1 dSpace MicroAutoBox

Die MicroAutoBox (MABX) der Firma dSpace ist ein Prototypensteuergerät, das speziell auf den Einsatz im automotiven Bereich zugeschnitten ist. Es wird ein Target-Compiler für Simulink angeboten. Zum Einsatz kommt das

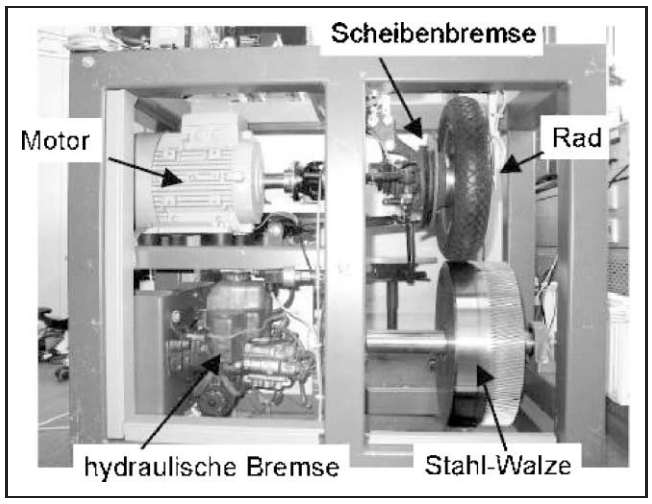

Bild 2: Prüfstand am Institut für Regelungs- und Automatisierungstechnik.

System *MicroAutoBox DS1401/1501*. Die genauen Spezifikationen sind in [12] zu finden. Die Implementierung der S-Function auf der MABX verlief problemlos.

Zur Erprobung der Funktionalität wurde die beschriebene Toolbox an einem Prüfstand am *Institut für Regelungs- und Automatisierungstechnik* der *Technischen Universität Graz* eingesetzt (siehe Bild 2). Dieser dient der Implementierung und Erprobung von Antischlupfregelungen (ASR) und Antiblockiersystemen (ABS) für Kraftfahrzeuge. Die Anlage besteht aus einem Reifen, der mit Hilfe eines Elektromotors angetrieben und mittels einer hydraulischen Bremse gebremst werden kann sowie einer drehbar gelagerten Walze. Durch die Kontaktkraft zwischen Rad und Walze wird beim Beschleunigen bzw. Bremsen des Rades auch die Walze beschleunigt bzw. gebremst. In Bild 3 ist der Aufbau der Anlage schematisch dargestellt.

Größen, die sich auf die Motorwelle beziehen, werden im Folgenden mit dem Index 1 versehen, Größen an der Walzenwelle mit dem Index 2. Die Drehbewegungen von Rad und Walze können mit Hilfe des Drallsatzes beschrieben werden.

Mit J_1 wird das Massenträgheitsmoment von Motor, Rad, Bremse und Welle bezüglich der Rotationsachse bezeichnet, ω_1 ist die Winkelgeschwindigkeit der Radwelle. Das

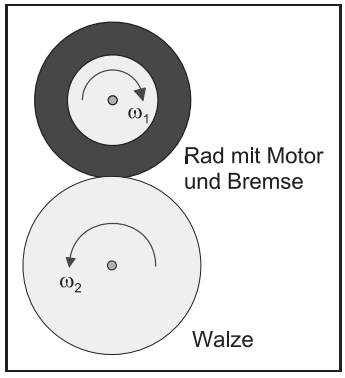

Bild 3: Prinzipieller Aufbau der Versuchsanlage.

vom Motor gelieferte Moment ist M_a, M_b ist das Bremsmoment und $M_{R,1}$ repräsentiert alle wirkenden Reibmomente. Das Moment $F_x r_1$ ist auf den Kontakt zwischen Rad und Walze zurückzuführen, wobei F_x die übertragene Längskraft ist und r_1 den Radius des Rades repräsentiert. Die Differentialgleichung zur Beschreibung der Radrotation lautet damit:

$$J_1 \frac{d\omega_1}{dt} = M_a - M_b - M_{R,1} + F_x r_1 \,. \tag{7}$$

Analog dazu gilt für die Drehbewegung der Walze:

$$J_2 \frac{d\omega_2}{dt} = -M_{R,2} - F_x r_2 \,. \tag{8}$$

Hierbei ist J_2 das Massenträgheitsmoment bezüglich der Drehachse, $M_{R,2}$ ist das Reibungsmoment und $F_x r_2$ ist das Drehmoment durch den Reifenkontakt, wobei r_2 der Walzenradius ist.

5.2 Messergebnisse

In Bild 4 sind die Rad- und Walzengeschwindigkeit anstelle der gemessenen Winkelgeschwindigkeiten dargestellt, wobei $v_1 = \omega_1 r_1$ und $v_2 = \omega_2 r_2$ gilt. Das Steuersignal *Antrieb* stellt den Bereich der Volllastbeschleunigung des Systems dar, wobei keine Antriebsschlupfregelung vorgesehen ist. Das Steuersignal *Bremse* zeigt den Bereich der Bremsung an, wobei hier ein Antiblockiersystem eingesetzt wird. Details zur Funktionsweise des ABS sind in [16] zu finden.

Die Rad- bzw. Walzenbeschleunigung,

$$a_1 = \frac{dv_1}{dt} \quad \text{bzw.} \quad a_2 = \frac{dv_2}{dt} \,,$$

wurde mit dem vorgestellten Verfahren geschätzt. Zum Vergleich wurden *gefilterte* Ableitungen mittels des Filters

$$P(s) = \frac{s}{0{,}05s + 1}$$

berechnet. Dabei wurde die Zeitkonstante der spektralen Zusammensetzung des zu filternden Signals angepasst.

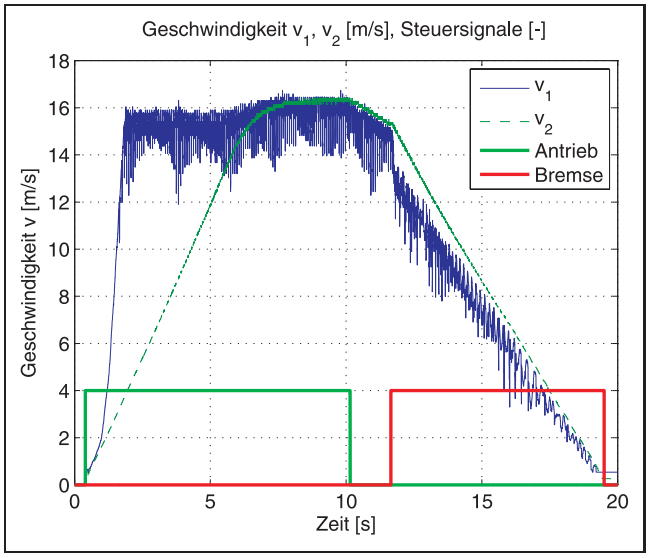

Bild 4: Gemessene Geschwindigkeiten und Steuersignale.

In Bild 5 ist der Verlauf der Radbeschleunigung a_1 für den ABS-Bremsvorgang dargestellt, wobei die geschätzte Beschleunigung für den ABS-Regelvorgang, d. h. zur Ansteuerung der ABS-Ventile, verwendet wurde. Die gefilterte Beschleunigung ist für diesen Zweck auf Grund des hohen Rauschpegels nicht geeignet. Die zur Schätzung verwendeten Parameter sind $j = 1$, $N = 2$, $\nu = 0$ und $T = 0{,}4$ sec.

Bild 6 zeigt den Verlauf für die Walzenbeschleunigung a_2 über den gesamten Versuchszeitraum. Im Bereich I ist die Beschleunigung des Rades auf die Sollgeschwindigkeit zu sehen, wobei bei $t \approx 5{,}5$ s die maximale Beschleunigung erreicht wird (optimaler Schlupf). Für den Bereich II ist $M_a = 0$, die Verzögerung der Walze resultiert aus den Reibungsmomenten. Während der ABS-Bremsung im Bereich III stellt sich eine annähernd konstante Verzögerung von $a_2 \approx -2\,\mathrm{m/s^2}$ ein. Wie im Vergleich mit der gefilterten Ableitung zu erkennen ist, konnte das Rauschen bei der in Echtzeit geschätzten Beschleunigung praktisch vollständig eliminiert werden. Die eingesetzten Parameter sind $j = 1$, $N = 1$, $\nu = 0$ und $T = 0{,}2$ sec.

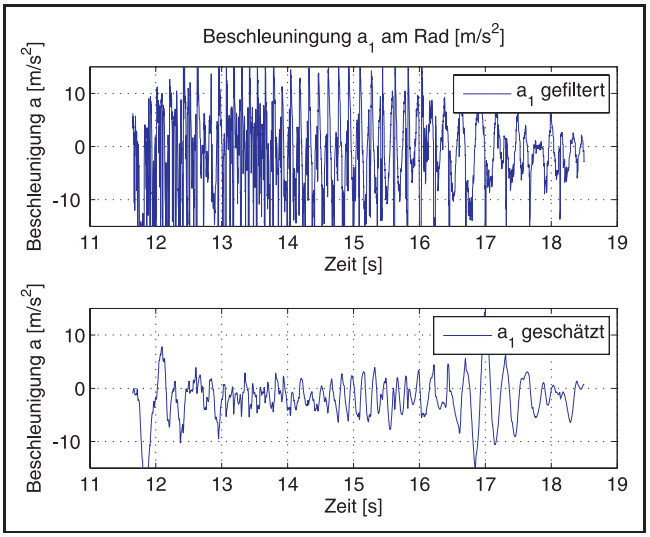

Bild 5: Vergleich der Beschleunigung am Rad.

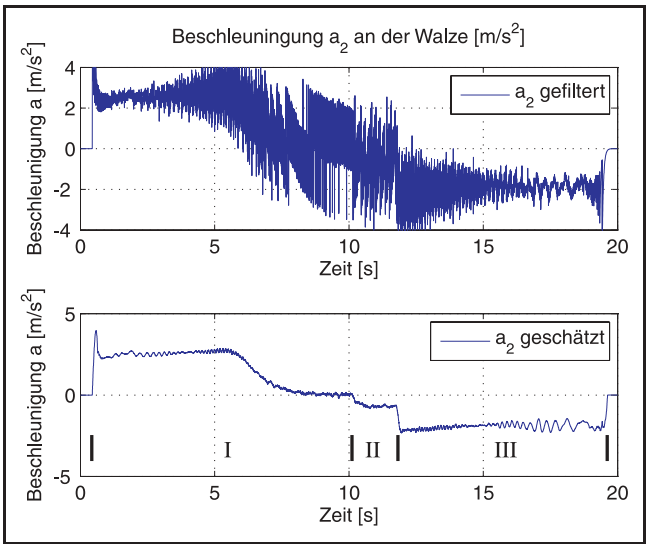

Bild 6: Vergleich der Beschleunigung an der Walze.

6 Zusammenfassung und Ausblick

In dieser Arbeit wird die Implementierung eines *algebraischen Ableitungsschätzverfahrens* für Echzeitanwendungen gezeigt. Dazu wird das Verfahren hergeleitet und gezeigt, wie bei abgetasteten Systemen eine Vereinfachung der Schätzgleichung in Form einer gewichteten Summe erzielt werden kann. Die besondere Form der Berechnungen erfordert den Einsatz von komplexen Werkzeugen. Die erstellte Implementierung eignet sich sowohl für Simulationen als auch für Echtzeitsysteme. Die Ergebnisse aus Abschnitt 5 zeigen, dass die Algorithmen bei Anwendung mit stark verrauschten Messsignalen ausgezeichnete Ergebnisse liefern.

Mit den in dieser Arbeit gezeigten Versuchen soll lediglich die korrekte Funktionsweise der Implementierung verdeutlicht werden. Ein nächstes Ziel ist die Verwendung der mittels der *Derivative Estimation Toolbox* ermittelten Ableitungen zur Ermittlung von Parametern und Zustandsgrößen, *online* und in Echtzeit. Die in Abschnitt 4.6 beschriebene Möglichkeit einer online-Berechnung der Werte für $\Pi_{j,k}$ ist derzeit in Entwicklung.

Anhang A

Es soll ein expliziter Ausdruck für den Koeffizienten α_j der Taylorentwicklung (1) hergeleitet werden. Dazu multipliziert man Gleichung (3) im Bildbereich mit dem Operator s^{N+1}. Damit ergibt sich zunächst

$$s^{N+1}\, Y_N(s) = \sum_{i=0}^{N} \alpha_i\, s^{N-i}\,,$$

was nun anschließend $N - j$ mal bezüglich s differenziert wird. Auf diese Weise eliminiert man alle Koeffizienten $\alpha_{j+1}, \ldots, \alpha_N$, d. h. es ergibt sich

$$\frac{d^{N-j}}{ds^{N-j}}\left(s^{N+1} Y_N(s)\right) = \sum_{i=0}^{j} \alpha_i\, \frac{(N-i)!}{(j-i)!}\, s^{j-i}\,.$$

Im nächsten Schritt werden die Koeffizienten $\alpha_0, \ldots, \alpha_{j-1}$ zum Verschwinden gebracht. Zu diesem Zweck multipliziert man die letzte Beziehung von links mit dem Operator $1/s$

$$\frac{1}{s}\frac{d^{N-j}}{ds^{N-j}}\left(s^{N+1} Y_N(s)\right) = \frac{(N-j)!}{s}\alpha_j + \sum_{i=0}^{j-1} \alpha_i \frac{(N-i)!}{(j-i)!} s^{j-i-1}$$

und differenziert j mal nach s, um die Summe auf der rechten Seite zu eliminieren. Diese Schritte ergeben

$$\frac{d^j}{ds^j}\left(\frac{1}{s}\frac{d^{N-j}}{ds^{N-j}}\left(s^{N+1} Y_N(s)\right)\right) = \frac{(-1)^j\, j!\,(N-j)!}{s^{j+1}}\alpha_j\,. \quad (9)$$

Der Ausdruck auf der linken Seite von (9) enthält s^N als Operatormonom höchsten Grades, was im Zeitbereich einer

N-fachen Zeitableitung gleichkommt. Um diese Zeitableitungen der Signale zu vermeiden, multipliziert man (9) mit $1/s^{N+\nu+1}$, $\nu \geq 0$. Folglich wird jedes Zeitsignal mindestens einmal integriert[3]. Dieses Vorgehen führt auf

$$\frac{1}{s^{N+\nu+1}} \frac{d^j}{ds^j} \left(\frac{1}{s} \frac{d^{N-j}}{ds^{N-j}} \left(s^{N+1} Y_N(s) \right) \right) = \frac{(-1)^j \, j! \, (N-j)!}{s^{N+j+\nu+2}} \alpha_j \, . \tag{10}$$

Gleichung (10) lässt sich mit Hilfe der Formel von Leibniz zur mehrfachen Differentiation von Produkten auswerten. Dazu führt man bezüglich der linken Seite folgende Rechenschritte durch:

$$\frac{1}{s^{N+\nu+1}} \left(\frac{d^j}{ds^j} \left(\frac{1}{s} \frac{d^{N-j}}{ds^{N-j}} \left(s^{N+1} \, Y_N(s) \right) \right) \right)$$

$$= \cdots \left(\frac{d^j}{ds^j} \sum_{\kappa_1=0}^{N-j} \binom{N-j}{\kappa_1} \frac{(N+1)!}{(N-\kappa_1+1)!} s^{N-\kappa_1} \frac{d^{N-j-\kappa_1} Y_N(s)}{ds^{N-j-\kappa_1}} \right)$$

$$= \frac{1}{s^{N+\nu+1}} \sum_{\kappa_1=0}^{N-j} \sum_{\kappa_2=0}^{j} \binom{N-j}{\kappa_1} \binom{j}{\kappa_2} \frac{(N+1)! \, (N-\kappa_1)!}{(N-\kappa_1+1)! \, (N-\kappa_1-\kappa_2)!} \times$$

$$s^{N-\kappa_1-\kappa_2} \frac{d^{N-\kappa_1-\kappa_2} Y_N(s)}{ds^{N-\kappa_1-\kappa_2}} = \sum_{\kappa_1=0}^{N-j} \sum_{\kappa_2=0}^{j} \binom{N-j}{\kappa_1} \binom{j}{\kappa_2} \times$$

$$\frac{(N+1)!}{(N-\kappa_1-\kappa_2)! \, (N-\kappa_1+1)} \frac{1}{s^{\nu+\kappa_1+\kappa_2+1}} \frac{d^{N-\kappa_1-\kappa_2} Y_N(s)}{ds^{N-\kappa_1-\kappa_2}} \, .$$

Damit ist Gleichung (10) äquivalent zu

$$\frac{1}{s^{N+j+\nu+2}} \alpha_j = \frac{(-1)^j}{j! \, (N-j)!} \sum_{\kappa_1=0}^{N-j} \sum_{\kappa_2=0}^{j} \binom{N-j}{\kappa_1} \binom{j}{\kappa_2} \times$$

$$\frac{(N+1)!}{(N-\kappa_1-\kappa_2)! \, (N-\kappa_1+1)} \frac{1}{s^{\nu+\kappa_1+\kappa_2+1}} \frac{d^{N-\kappa_1-\kappa_2} Y_N(s)}{ds^{N-\kappa_1-\kappa_2}} \, . \tag{11}$$

Gleichung (11) kann nun „ohne Schwierigkeiten“ in den Zeitbereich zurücktransformiert werden: Der Ausdruck auf der linken Seite entspricht einem Monom der Zeit, Ausdrücke in s auf der rechten Seite entsprechen Faltungsintegralen. Es ergibt sich

$$\alpha_j = \int_0^t \Pi_j(t, \tau) \, y_N(\tau) \, d\tau \tag{12}$$

mit der Funktion

$$\Pi_j(t, \tau) = \frac{(N+j+\nu+1)! \, (N+1)! \, (-1)^j}{t^{N+j+\nu+1}} \sum_{\kappa_1=0}^{N-j} \sum_{\kappa_2=0}^{j}$$

$$\frac{(t-\tau)^{\nu+\kappa_1+\kappa_2} \, (-\tau)^{N-\kappa_1-\kappa_2}}{\kappa_1! \kappa_2! (N-j-\kappa_1)! (j-\kappa_2)! (N-\kappa_1-\kappa_2)! (\nu+\kappa_1+\kappa_2)! (N-\kappa_1+1)} . \tag{13}$$

[3] Anstelle einer Mehrfachintegration ist die Verwendung anderer Filter passender Ordnung natürlich ebenso möglich.

Anhang B

Im Folgenden soll aus Gleichung (5) eine integralfreie Schätzung für die Ableitungen $y^{(j)}(t)$ hergeleitet werden. Dazu setzt man $\tau = \mu \, T$ und bestimmt zunächst

$$\int_0^T (T-\tau)^{\nu+\kappa_1+\kappa_2} \, (-\tau)^{N-\kappa_1-\kappa_2} \, y(t-\tau) \, d\tau =$$

$$= (-1)^{N-\kappa_1-\kappa_2} \int_0^1 (T-\mu T)^{\nu+\kappa_1+\kappa_2} (\mu T)^{N-\kappa_1-\kappa_2} y(t-\mu T) \, T \, d\mu$$

$$= (-1)^{N-\kappa_1-\kappa_2} T^{N+\nu+1} \int_0^1 (1-\mu)^{\nu+\kappa_1+\kappa_2} \mu^{N-\kappa_1-\kappa_2} y(t-\mu T) \, d\mu$$

$$= \cdots \int_0^1 \sum_{i=0}^{\nu+\kappa_1+\kappa_2} \binom{\nu+\kappa_1+\kappa_2}{i} (-1)^i \mu^i \, \mu^{N-\kappa_1-\kappa_2} y(t-\mu \, T) \, d\mu$$

$$= \cdots \sum_{i=0}^{\nu+\kappa_1+\kappa_2} \binom{\nu+\kappa_1+\kappa_2}{i} (-1)^i \int_0^1 \mu^{N+i-\kappa_1-\kappa_2} y(t-\mu \, T) \, d\mu$$

Kann nun das Intervall $[t-T, t]$ gemäß einer periodischen Abtastung in *M äquidistante* Teilintervalle mit konstanten Werten von y unterteilt werden, d. h. $y(t-\frac{k-1}{M} T) = \text{konst.}$, $k = 1, \ldots, M$, so gilt

$$\int_0^T (T-\tau)^{\nu+\kappa_1+\kappa_2} \, (-\tau)^{N-\kappa_1-\kappa_2} \, y(t-\tau) \, d\tau$$

$$= \cdots \sum_{i=0}^{\nu+\kappa_1+\kappa_2} \binom{\nu+\kappa_1+\kappa_2}{i} (-1)^i \sum_{k=1}^{M} \int_{\frac{k-1}{M}}^{\frac{k}{M}} \mu^{N+i-\kappa_1-\kappa_2} y\left(t-\tfrac{k-1}{M} T\right) d\mu$$

$$= \cdots \sum_{i=0}^{\nu+\kappa_1+\kappa_2} \binom{\nu+\kappa_1+\kappa_2}{i} (-1)^i \sum_{k=1}^{M} y\left(t-\tfrac{k-1}{M} T\right) \int_{\frac{k-1}{M}}^{\frac{k}{M}} \mu^{N+i-\kappa_1-\kappa_2} \, d\mu$$

$$= \cdots \sum_{i=0}^{\nu+\kappa_1+\kappa_2} \binom{\nu+\kappa_1+\kappa_2}{i} (-1)^i \sum_{k=1}^{M} y\left(t-\tfrac{k-1}{M} T\right) \left[\frac{\mu^{N+i-\kappa_1-\kappa_2+1}}{N+i-\kappa_1-\kappa_2+1} \right]_{\frac{k-1}{M}}^{\frac{k}{M}}$$

$$= (-1)^{N-\kappa_1-\kappa_2} \, T^{N+\nu+1} \sum_{i=0}^{\nu+\kappa_1+\kappa_2} \binom{\nu+\kappa_1+\kappa_2}{i} \frac{(-1)^i}{N+i-\kappa_1-\kappa_2+1} \times$$

$$\sum_{k=1}^{M} y\left(t-\tfrac{k-1}{M} T\right) \left(\left(\tfrac{k}{M}\right)^{N+i-\kappa_1-\kappa_2+1} - \left(\tfrac{k-1}{M}\right)^{N+i-\kappa_1-\kappa_2+1} \right)$$

Dieses Ergebnis führt nun zusammen mit den Gleichungen (5) und (13) auf das wichtige Resultat: Die j-te Zeitableitung eines Zeitsignals y bestimmt sich gemäß

$$y^{(j)}(t) = \sum_{k=1}^{M} \Pi_{j,k} \, y(t-(k-1) T_s) \tag{14}$$

für $j = 0, 1, 2, \ldots, N$ und der Abkürzung

$$\Pi_{j,k} = \frac{(N+j+\nu+1)! \, (N+1)!}{(MT_s)^j} \sum_{\kappa_1=0}^{N-j} \sum_{\kappa_2=0}^{j} \sum_{i=0}^{\nu+\kappa_1+\kappa_2}$$

$$\frac{(-1)^{N+i-\kappa_1-\kappa_2} \left(\left(\frac{k}{M}\right)^{N+i-\kappa_1-\kappa_2+1} - \left(\frac{k-1}{M}\right)^{N+i-\kappa_1-\kappa_2+1} \right)}{\kappa_1! \kappa_2! i! (N-j-\kappa_1)! (j-\kappa_2)! (N-\kappa_1-\kappa_2)! (\nu-i+\kappa_1+\kappa_2)! (N-\kappa_1+1)(N+i-\kappa_1-\kappa_2+1)} \tag{15}$$

Literatur

[1] R. S. Anderssen, F. de Hoog, M. Hegland, „A stable finite difference Ansatz for higher order differentiation of non-exact data“, *Bull. Austral. Math. Soc., Volume 58, ff. 223–232 (1998)*.

[2] C. S. Burrus, R. A. Gopinath, H. Guo, „Introduction to Wavelets and Wavelet Transforms. A Primer“, *Prentice-Hall, New Jersey, 1998*.

[3] S. Diop, J. W. Grizzle, P. E. Moraal, A. Stefanopoulou, „Interpolation and numerical differentiation for observer design“, *American Control Conference, Volume 2, ff. 1329–1333 (1994)*.

[4] S. Diop, J. W. Grizzle, F. Chaplais, „On numerical differentiation algorithms for nonlinear estimation“, *Proceedings of the 39th IEEE Conference on Decision and Control, Volume 2, ff. 1133–1138 (2000)*.

[5] S. Diop, V. Fromion, J. W. Grizzle, „A global exponential observer based on numerical differentiation“, *Proceedings of the 40th IEEE Conference on Decision and Control, Volume 4, ff. 3344–3349 (2001)*.

[6] M. Fliess, C. Join, M. Mboup and H. Sira-Ramírez, „Analyse et représentation de signaux transitoires: application à la compression, au débruitage, et à la détection de ruptures“, *GRETSI 2005, Louvain-la-Neuve, Belgium, 2005*.

[7] M. Fliess and H. J. Sira-Ramírez, „Control via state-estimation of some nonlinear systems“, *IFAC Symp. on Nonlinear Control Systems (NOLCOS), Stuttgart, Germany, 2004*.

[8] M. Fliess, H. Sira-Ramírez, „An algebraic framework for linear identification“, *ESAIM Control Optim. Calculus Variations, 9, 151–168, 2003*.

[9] W. Gander, U. von Matt, „Smoothing filters“, In: W. Gander, J. Hrebicek (eds.), „Solving Problems in Scientific Computating Using Maple and Matlab“, *3rd Edition, Springer, Berlin, ff. 121–139 (1997)*.

[10] J. Jouffroy, J. Reger, „An algebraic perspective to single-transponder underwater navigation“, *IEEE Conf. on Control Applications (CCA), Munich, Germany, 2006*.

[11] The MathWorks, Inc., „Simulink Help: Writing S-Functions“, *2004–2006*.

[12] dSpace GmbH, „MicroAutoBox Features, Release 5.1“, *May 2006*.

[13] F. Plestan, J. W. Grizzle, „Synthesis Of Nonlinear Observers Via Structural Analysis And Numerical Differentiation“, *Proceedings of the 5th European Control Conference, CD-ROM, September 1999*.

[14] J. Reger, M. Mai, and H. J. Sira-Ramírez, „Robust algebraic state estimation of chaotic systems“, *IEEE Conf. on Control Applications (CCA), Munich, Germany, 2006*.

[15] J. Reger, H. J. Sira-Ramírez, and M. Fliess, „On non-asymptotic observation of nonlinear systems“, *IEEE Int. Conf. on Decision and Control (CDC), Seville, Spain, 2005*.

[16] Robert Bosch GmbH; H. Bauer (Hrsg.): „Kraftfahrtechnisches Taschenbuch“, *25. Auflage, Vieweg, Wiesbaden, 2003*.

[17] Sommerschule: *Workshop on Identification, State Reconstruction, and Generalized PI-Control*, 4.–8. Juli 2005, München, `http://www.unibw-muenchen.de/campus/ET8/et81/Mitarbeiter/reger/workshop/index.htm`

[18] Sommerschule: *Fast Estimation Methods in Automatic Control and Signal Processing*, 18.–20. Juli 2005, Paris, `http://www-futurs.inria.fr/activites/colloques/mne05.htm`

[19] Sommerschule: *Fast Estimation and Identification Methods in Control and Signal*, 11.–15. September 2006, Grenoble, `http://www.lag.ensieg.inpg.fr/ecole-ete-auto/session/s_27.php`

Manuskripteingang: 10. April 2007.

Dipl.-Ing. Josef Zehetner ist am Institut für Regelungs- und Automatisierungstechnik der Technischen Universität Graz tätig. Hauptarbeitsgebiete: Fahrdynamikregelung, Sliding Mode Control, algebraische Methoden der Zustands- und Parameterschätzung.

Adresse: Technische Universität Graz, Institut für Regelungs- und Automatisierungstechnik, Kopernikusgasse 24/II, 8010 Graz, ÖSTERREICH,
E-Mail: josef.zehetner@gmail.com

Dr.-Ing. Johann Reger ist Mitarbeiter am Control Systems Laboratory der University of Michigan in Ann Arbor, USA. Hauptarbeitsgebiete: Passivitäts- und flachheitsbasierte Regelung nichtlinearer Systeme, algebraische Methoden der Zustands- und Parameterschätzung, Fehlerdiagnose im Anwendungsfeld der Mechatronik.

Adresse: 1301 Beal Ave. (4123 EECS Bldg.), University of Michigan, Ann Arbor, MI 48109-2122 USA, E-Mail: reger@ieee.org

Ao.Univ.-Prof. Dr.techn. Martin Horn ist am Institut für Regelungs- und Automatisierungstechnik der Technischen Universität Graz tätig. Hauptarbeitsgebiete: Anwendung von regelungstechnischen Methoden in der Fahrzeugtechnik, Entwurf von Werkzeugen zur Synthese linearer und nichtlinearer Regelkreise.

Adresse: Technische Universität Graz, Institut für Regelungs- und Automatisierungstechnik, Kopernikusgasse 24/II, 8010 Graz, ÖSTERREICH,
E-Mail: martin.horn@tugraz.at