Parallele Echtzeitsimulation mechatronischer Systeme

D. Clemens

Forschungsbericht Nr. 6/93

Meß-, Steuer- und Regelungstechnik

Übersicht: In diesem Bericht wird ein neues Integrationsverfahren zur Echtzeitsimulation vorgestellt. Das Verfahren gestattet die Verwendung von großen Schrittweiten auch bei dynamisch schnellen Systemen. Es liefert speziell bei steifen Systemen, d.h. Systemen mit schnellen und langsamen Dynamiken, kürzere Rechenzeiten und ermöglicht somit erst die Echtzeitsimulation. Durch eine Implementierung auf einem Parallelrechner kann die Simulation nocheinmal beschleunigt werden. Dazu muß das System in schnelle und langsame Teilsysteme zerlegt werden. Es werden eine Stabilitätsbetrachtung und eine Fehlerabschätzung für das Integrationsverfahren vorgenommen. Simulationsergebnisse und Rechenzeitvergleiche für verschiedene Systeme auf einem Transputer zeigen die Anwendbarkeit und die hohe Beschleunigung des Verfahrens.

Gerhard-Mercator-Universität - GH Duisburg
Meß-, Steuer- und Regelungstechnik
Prof. Dr.-Ing. H. Schwarz

Inhaltsverzeichnis

Formelzeichen und Bezeichnungen

$\boldsymbol{A}$	Systemmatrix
$\boldsymbol{A}_D$	Verfahrensmatrix
a_j	Koeffizienten eines Mehrschrittverfahrens
$\boldsymbol{B}$	Eingangsmatrix
$B_{1,2,3}$	Drosselbeiwerte
b_j	Koeffizienten eines Mehrschrittverfahrens
c	Konstante
$c_{1,\ldots,6}$	Koeffizienten der Synchronmaschine
E_{Oel}	Kompressionsmodul
$\boldsymbol{f}$	allgemeine nichtlineare Systemfunktion
$\boldsymbol{g}$	allgemeine nichtlineare Ausgangsfunktion
h	Schrittweite
$\boldsymbol{I}$	Einheitsmatrix
$k_{1,\ldots,5}$	Koeffizienten der Turbine
L	Lipschitz–Konstante
l	Anzahl der Eingänge
n	Systemdimension
p	Anzahl der Teilsysteme
p	Anzahl der parallelen Prozessoren
$p_{1,2}$	Drücke
r	Anzahl der Ausgänge
St	Steifigkeit
t	Zeit
t_0	Anfangszeitpunkt
$t(1)$	Ausführungszeit eines sequentiellen Algorithmus
$t(p)$	Ausführungszeit eines parallelen Algorithmus mit p Prozessoren
T	Zeitkonstante
T_R	Rechenzeit
$\boldsymbol{u}$	Eingangsvektor
$v_{1,2}$	Volumen
$\boldsymbol{x}$	Zustandsvektor
$\boldsymbol{x}_i$	diskreter Zustandswert
$\boldsymbol{y}$	Ausgangsvektor
$1(t)$	Einheitssprungfunktion
ϵ	Fehlertoleranz
η	Effizienz eines parallelen Algorithmus
λ	Eigenwert

μ	Normierungsfaktor
$\boldsymbol{\Phi}$	Fundamentalmatrix
$\boldsymbol{\phi}$	Verfahrensfunktion
σ	Beschleunigung eines parallelen Algorithmus
$\mathbf{0}$	Nullvektor / Nullmatrix

Operatoren

$\in$	Element von
δ	Abweichung vom optimalen Wert
$\max(\cdot)$	Funktion zur Bestimmung des Maximums
$\min(\cdot)$	Funktion zur Bestimmung des Minimums
$\mathbb{R}$	Menge der reellen Zahlen
$\mathbb{N}$	Menge der natürlichen Zahlen
$(\cdot)^{-1}$	Inverse Matrix
$(\cdot)!$	Fakultät
$O(\cdot)$	Komplexität eines Algorithmus (Landau'sches Symbol)
$\lvert\cdot\rvert$	Betragsbildung
$\lvert\lvert\cdot\rvert\rvert$	Normierung
dim	Dimension, Ordnung
$\sum(\cdot)$	Summation
$\dot{(\cdot)}$	Differentiation nach der Zeit
$\tilde{(\cdot)}$	kleine Abweichung

Indizierung

max, min	maximaler, minimaler Wert
i, j, k	Laufindizes
0	Arbeitspunkt

Abkürzungen

AB3	Adam–Bashworth–Integrationsverfahren dritter Ordnung
DD	Druckdynamik
DM	Dieselmotor und Hydropumpe
DRE	discrete response equivalent Integrationsverfahren
DRE0	DRE–Verfahren mit Abtast–Halte–Glied nullter Ordnung
DRE1	DRE–Verfahren mit Abtast–Halte–Glied erster Ordnung
EUL	Euler–Integrationsverfahren
HMA	Hydraulischer Mobilantrieb
HPV	Verstelleinrichtung der Hydropumpe

MDM	Motor(DM)–Druckdynamik(DD)–Motor(MGL)
MGL	Hydromotor, Schaltgetriebe und Last
ODE23	Integrationsprogramm aus Matlab (1989)
pDRE	paralles DRE–Verfahren
RK3	Runge–Kutta–Integrationsverfahren dritter Ordnung

1 Einleitende Übersicht

Echtzeitsimulationen werden als Ersatz für reale Systeme immer dann eingesetzt, wenn ihre Anwendung Vorteile in bezug auf Anwendungssicherheit, Platzbedarf und vor allem Kosten hat. Eine Echtzeitsimulation steht immer in Verbindung mit einem realen System (simulation–in–the–loop). Dies kann zum einen der Mensch sein (z.B. Flugsimulator), zum anderen technische Teilsysteme (z.B. eine Kraftfahrzeugfeder mit simulierter Fahrzeugdynamik). Die Simulation muß die Reaktionen des zu simulierenden physikalischen Systems auf eine Anregung in Echtzeit zur Verfügung stellen. Alle Operationen, wie z.B. das Ein- und Auslesen von Daten sowie die numerische Integration, müssen in einer begrenzten Zeit erfolgt sein. Diese Zeit ist abhängig von dem dynamischen Verhalten des realen, interaktiven Systems und von der kleinsten Zeitkonstante des zu simulierenden Systems. Die Echtzeit muß mit der Simulationszeit parallel laufen. Daher ist bei einer Echtzeitsimulation eine konstante Schrittweite zu verwenden.

Um alle notwendigen Rechenoperationen innerhalb dieser Schrittweite auszuführen, können zur Simulation großer Systeme Parallelrechner eingesetzt werden (Hwang u.a. 1988, Thompson 1992 und Jelali 1992). Die Kriterien für eine Aufteilung des Systems gibt Clemens (1992a). Für eine hohe Beschleunigung der Simulation hat eine möglichst gleiche Lastverteilung höchste Priorität (Grote 1993). Desweiteren darf die Schrittweite nicht zu klein werden, um die Echtzeitfähigkeit zu gewährleisten. Probleme bei der numerischen Integration bereiten Systeme mit sehr unterschiedlichen Zeitkonstanten T. Ein entsprechendes Differentialgleichungs (DGL)–System wird als steif bezeichnet. Es ist durch die Steifheit

$$St = \frac{|\lambda_{max}|}{|\lambda_{min}|} = \frac{1/T_{min}}{1/T_{max}} \tag{1.1}$$

gekennzeichnet. λ sind die Eigenwerte des Systems.

Mechatronische Systeme zeichnen sich durch die Kombination von mechanischen und elektronischen Komponenten aus, deren Zeitkonstanten sehr unterschiedlich sein können (Ebner 1992). Um eine stabile Simulation zu erhalten, muß entweder eine sehr kleine Schrittweite gewählt oder spezielle rechenaufwendigere Integrationsverfahren (Gear 1971) verwendet werden. Diese Verfahren sind jedoch alle implizit und somit für eine Echtzeitsimulation nicht geeignet (Gear 1977). Zur Echtzeitsimulation können nur explizite Integrationsverfahren benutzt werden, da sie zur Ermittlung eines neuen Zustandswertes nur zurückliegende Werte benötigen (Clemens 1992b). Explizite Mehrschrittverfahren bieten gegenüber Einschrittverfahren Rechenzeitvorteile. Sie können jedoch bei veränderlichen Eingangssignalen innerhalb des Integrationsintervalls instabil werden. Daher eignen sie sich nur beschränkt zur Echtzeitsimulation.

Zur Beschleunigung der Simulation können unter bestimmten Voraussetzungen Parallelrechner eingesetzt werden. Die mögliche Verkürzung der Rechenzeit wird bei diesen

Rechnern durch die Beschleunigung

$$\sigma = \frac{t(1)}{t(p)} \tag{1.2}$$

beschrieben, mit $t(1)$, als der Rechenzeit eines einzelnen Prozessors mit dem schnellsten sequentiellen Algorithmus, sowie $t(p)$, als der Zeit für p Prozessoren mit einem parallelen Algorithmus. Mit dem Quotienten σ durch p ergibt sich die Effizienz

$$\eta = \frac{\sigma}{p}\,100\% \tag{1.3}$$

des parallelen Algorithmus. Das Ziel der Parallelisierung ist eine hohe Effizienz, gleichbedeutend mit einer hohen Beschleunigung, damit ein gutes Kosten/Nutzen–Verhältnis erreicht wird. Bei Simulationen läßt sich dies vor allem durch den Einsatz eines parallelen Multirate–Verfahrens erreichen (Clemens 1992a), d.h. schnelle Teilsysteme werden mit einer kleineren Schrittweite integriert als langsamere. Die Größe der Schrittweite ist durch die numerische Stabilität und die gewünschte Genauigkeit bestimmt. Zum Erreichen des gleichen Zeitschrittes muß die Integration mit der kleineren Schrittweite mehrmals ausgeführt werden. Das Gantt Chart (Quinn 1988) in Bild 1.1 verdeutlicht den parallelen Programmablauf für ein Verhältnis der Schrittweiten von 1 zu 3 für zwei Prozessoren. Da gleiche Rechenzeiten der Prozessoren meistens nicht genau erreicht werden, beinhaltet

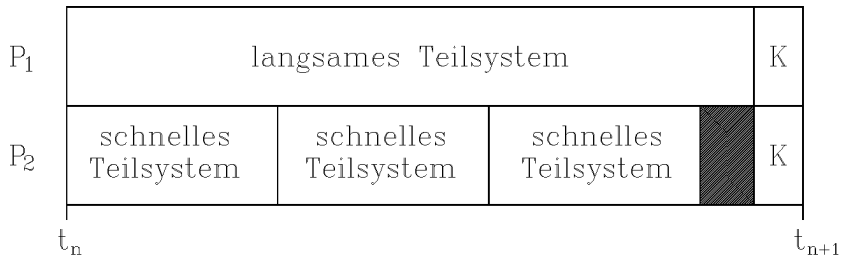

Bild 1.1: Gantt Chart der Integration mit einem Multirate–Verfahren.

das Gantt Chart die Stillstandszeit (schwarzer Bereich) eines Prozessors. Diese Zeit ist abhängig von dem zu simulierenden System und dessen Aufteilung in Teilsysteme.

In diesem Bericht wird ein neues Integrationsverfahren vorgestellt. Es vereinigt beide Aspekte für eine schnelle und damit echtzeitfähige Simulation, den Einsatz eines Parallelrechners und die Wahl einer möglichst großen Schrittweite. In Abschnitt 2 werden das neue Integrationsverfahren und die Implementierung auf einem Parallelrechner dargestellt. Es werden eine Stabilitätsbetrachtung und eine Fehlerabschätzung für das Integrationsverfahren vorgenommen. Simulationsergebnisse und Rechenzeitvergleiche für verschiedene Systeme sind der Inhalt des 3. Abschnitts. Diese zeigen die Anwendbarkeit des neuen Verfahrens zur Echtzeitsimulation dynamisch schneller Systeme. Die Simulationen lassen sich in ihrer Rechenzeit erheblich beschleunigen. Eine Zusammenfassung mit anschließendem Ausblick beendet den Bericht.

2 Ein neues Verfahren zur Echtzeitsimulation

Von Metzger (1992) wird ein DRE (discrete response equivalent)–Integrationsverfahren für spezielle Differentialgleichungen angegeben. Es beruht auf der linearen skalaren Systembeschreibung

$$\dot{x}(t) = \lambda x(t) + cu(t) \tag{2.1}$$

mit dem Eigenwert $\lambda < 0$ des Systems und dem Eingang u. Mit Hilfe von Abtast–Halte–Gliedern nullter und erster Ordnung sowie dem allgemeinen expliziten Mehrschrittverfahren

$$\boldsymbol{x}_{i+1} = \sum_{j=0}^{m} a_j \boldsymbol{x}_{i-j} + h \sum_{j=0}^{m} b_j \dot{\boldsymbol{x}}_{i-j} \tag{2.2}$$

wurden zwei Integrationsverfahren entwickelt, die bei Systemen der Form (2.1) für jede Schrittweite h eine stabile Simulation liefern. Liegt jedoch ein System von gekoppelten Differentialgleichungen vor, eventuell mit konjugiert komplexen Eigenwerten oder einer nichtlinearen Systembeschreibung mit vom Arbeitspunkt abhängigen Eigenwerten, ist das Integrationsverfahren nicht anwendbar.

2.1 Erweiterung auf Systeme von Differentialgleichungen

Hier soll nun zunächst die Erweiterung des DRE–Verfahrens auf Systeme mit gekoppelten nichtlinearen Differentialgleichungen durchgeführt werden. Die zu simulierenden Systeme sind durch die gewöhnlichen Differentialgleichungen der Form

$$\dot{\boldsymbol{x}}(t) = \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}, t) \quad , \boldsymbol{x}(t_0) = \boldsymbol{x}_0 \tag{2.3a}$$

$$\boldsymbol{y}(t) = \boldsymbol{g}(\boldsymbol{x}, t) \tag{2.3b}$$

mit $\boldsymbol{x} \in \mathbb{R}^n$, $\boldsymbol{u} \in \mathbb{R}^l$ und $\boldsymbol{y} \in \mathbb{R}^r$ beschrieben. Der hier entwickelte Algorithmus basiert auf der linearen Beschreibung der Systemdynamik

$$\dot{\boldsymbol{x}}(t) = \boldsymbol{A}\boldsymbol{x}(t) + \boldsymbol{B}\boldsymbol{u}(t) \ . \tag{2.4}$$

Das System sei stabil, d.h. $\boldsymbol{A}$ habe stabile Eigenwerte. Die Systemdynamik (2.3a) kann für kleine Abweichungen von dem Arbeitspunkt $\boldsymbol{x}_0, \boldsymbol{u}_0$ durch Gl. (2.4) approximiert werden. Die Matrizen sind durch

$$\boldsymbol{A} = \left.\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{x}}\right|_{\boldsymbol{x}_0\,,\,\boldsymbol{u}_0} \quad \text{und} \quad \boldsymbol{B} = \left.\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{u}}\right|_{\boldsymbol{x}_0\,,\,\boldsymbol{u}_0} \tag{2.5}$$

gegeben. Werden die Eingangssignale über Abtast–Halte–Glieder nullter Ordnung eingelesen, läßt sich die zu (2.4) äquivalente zeitdiskrete Systembeschreibung (Schwarz 1979) durch

$$\boldsymbol{x}_{i+1} = \boldsymbol{\Phi}\boldsymbol{x}_i + \boldsymbol{A}^{-1}(\boldsymbol{\Phi} - \boldsymbol{I})\boldsymbol{B}\boldsymbol{u}_i \quad , \quad \boldsymbol{\Phi} = e^{h\boldsymbol{A}} \tag{2.6}$$

mit $\boldsymbol{I}$, eine $n \times n$ Einheitsmatrix, ausdrücken. Benutzt man die explizite Mehrschrittformel (2.2) zur Integration des Systems (2.4), so ergibt sich für $m = 0$

$$\boldsymbol{x}_{i+1} = a_0 \boldsymbol{x}_i + h b_0 \dot{\boldsymbol{x}}_i \tag{2.7}$$

und mit (2.4)

$$\boldsymbol{x}_{i+1} = (a_0 + h b_0 \boldsymbol{A}) \boldsymbol{x}_i + h b_0 \boldsymbol{B} \boldsymbol{u}_i \, . \tag{2.8}$$

Durch einen Vergleich der Gln. (2.6) und (2.8) bestimmt man die Koeffizienten zu

$$h b_0 \boldsymbol{B} = \boldsymbol{A}^{-1}(\boldsymbol{\Phi} - \boldsymbol{I})\boldsymbol{B} \quad \Rightarrow \quad b_0 = \frac{1}{h}\boldsymbol{A}^{-1}(\boldsymbol{\Phi} - \boldsymbol{I}) \quad \text{und} \tag{2.9a}$$

$$a_0 + h b_0 \boldsymbol{A} = \boldsymbol{\Phi} \quad \Rightarrow \quad a_0 = \boldsymbol{I}. \tag{2.9b}$$

Damit läßt sich das DRE0–Verfahren[1] für gekoppelte Differentialgleichungen angeben zu

$$\boldsymbol{x}_{i+1} = \boldsymbol{x}_i + \boldsymbol{A}_D \dot{\boldsymbol{x}}(\boldsymbol{x}_i, \boldsymbol{u}_i, t_i) \tag{2.10}$$

mit der Verfahrensmatrix

$$\begin{aligned} \boldsymbol{A}_D &= \boldsymbol{A}^{-1}(\boldsymbol{\Phi} - \boldsymbol{I}) \\ &= \sum_{j=1}^{\infty} \frac{h^j}{j!} \boldsymbol{A}^{j-1} \quad . \end{aligned} \tag{2.11}$$

Die Matrix $\boldsymbol{A}_D$ kann für lineare Systeme off–line berechnet werden. Damit sind die Lösungen des linearen Systems (2.4), bei konstanten Eingangssignalen während eines Integrationsschrittes, zu den diskreten Zeitpunkten $t = nh$ $(n = 1, 2, \ldots)$ sehr genau approximierbar. Das System kann unabhängig von den Zeitkonstanten und der gewählten Schrittweite stabil simuliert werden.

Wird das nichtlineare System (2.3a) mit diesem Verfahren integriert, muß die Matrix $\boldsymbol{A}$ nach Gl. (2.5) und somit $\boldsymbol{A}_D$ aus Gl. (2.11) bestimmt werden. Dies ist bei großen Systemen sehr rechenaufwendig und kann auch numerisch ungenau sein (Svaricek 1990). Eine spezielle Betrachtung erfordert daher die Kondition der Matrix $\boldsymbol{A}$ (Golub und Van Loan (1989) sowie Abschnitt 2.4). Ein einfaches Beispiel soll den Aufwand verdeutlichen: Die nichtlineare Systembeschreibung (2.3a) habe die Dimension $n = 10$, d.h. zur Bestimmung der Systemmatrix $\boldsymbol{A}$ aus Gl. (2.5) müssen $n^2 = 100$ Gleichungen bestimmt werden. Zusätzlich erfordert die Verfahrensmatrix $\boldsymbol{A}_D$ das Multiplizieren von Matrizen. Jede Multiplikation hat die Komplexität $O(n^3)$. Rechenzeitvorteile wird die Simulation eines nichtlinearen Systems hoher Ordnung mit dem DRE0–Verfahren somit nicht liefern.

[1] Die Null steht für ein Abtast–Halte–Glied nullter Ordnung.

2.2 Parallelisierung

Bei der problemstrukturellen parallelen Simulation (Keller 1987) eines komplexen dynamischen Systems (2.3a) werden die Differentialgleichungen auf p Prozessoren verteilt. Die Beschleunigung der Simulation ist von der gleichmäßigen Lastverteilung, dem notwendigen Datenaustausch der Prozessoren und damit von dem verwendeten Integrationsverfahren abhängig (Clemens 1992b). Nur einen Datenaustausch pro Integrationsintervall erfordern die expliziten Mehrschrittverfahren und das Euler–Verfahren. Das Euler–Verfahren besitzt jedoch nur ein kleines Stabilitätsgebiet, so daß die Schrittweite meist sehr klein gewählt werden muß. Die Mehrschrittverfahren besitzen ein größeres Stabilitätsgebiet, haben jedoch für veränderliche Eingangssignale im Integrationsintervall eine geringere Fehlerordnung. Dies liegt an der Interpolation über mehrere Stützstellen. Dagegen benötigt das hier vorgestellte Verfahren ebenfalls nur einen Datenaustausch im Integrationsintervall und erlaubt wesentlich größere Schrittweiten als das Euler–Verfahren. Ungenauigkeiten, hervorgerufen durch veränderliche diskrete Eingangssignale, treten nicht auf.

Komplexe Systeme zeichnen sich dadurch aus, daß sie aus vielen Teilsystemen bestehen. Diese sind untereinander aufgrund ihrer physikalischen Funktionen durch einzelne Zustände miteinander gekoppelt. Die Systemmatrix $\boldsymbol{A}$, welche nach Gl. (2.5) aus der nichtlinearen Systembeschreibung (2.3a) bestimmt wird, hat bei komplexen Systemen die Struktur einer Bandmatrix. Engeln–Müllges und Reutter (1988) definieren dies als stark diagonal dominant, d.h. die Matrix hat auf der Hauptdiagonalen Blockmatrizen, die die Dynamik der Teilsysteme angeben, und einzelne Elemente am Rand, welche die Kopplungen der Teilsysteme beschreiben. Im einfachsten Fall, es sind zwei Teilsysteme vorhanden, ergibt sich die lineare Systembeschreibung zu

$$\dot{\boldsymbol{x}}(t) = \boldsymbol{A}\boldsymbol{x}(t) + \boldsymbol{B}\boldsymbol{u}(t) \tag{2.12a}$$

mit

$$\boldsymbol{A} = \begin{pmatrix} \boldsymbol{A}_1 & \boldsymbol{A}_{12} \\ \boldsymbol{A}_{21} & \boldsymbol{A}_2 \end{pmatrix}, \quad \boldsymbol{B} = \begin{pmatrix} \boldsymbol{B}_1 \\ \boldsymbol{B}_2 \end{pmatrix}, \tag{2.12b}$$

$$\boldsymbol{x}(t) = \begin{pmatrix} \boldsymbol{x}_1(t) \\ \boldsymbol{x}_2(t) \end{pmatrix}, \quad \boldsymbol{u}(t) = \begin{pmatrix} \boldsymbol{u}_1(t) \\ \boldsymbol{u}_2(t) \end{pmatrix} \tag{2.12c}$$

und $\boldsymbol{x}_1 \in \mathbb{R}^{n_1}$, $\boldsymbol{x}_2 \in \mathbb{R}^{n_2}$, $\boldsymbol{u}_1 \in \mathbb{R}^{l_1}$, $\boldsymbol{u}_2 \in \mathbb{R}^{l_2}$ sowie $n = n_1 + n_2$, $l = l_1 + l_2$. Für jedes Teilsystem wird ein der Dynamik angepaßtes Integrationsverfahren entwickelt. Das zu simulierende nichtlineare System (2.3a) kann mit

$$\boldsymbol{x}(t) = (\boldsymbol{x}_1(t), \boldsymbol{x}_2(t), \ldots, \boldsymbol{x}_p(t))^T \quad \text{und} \tag{2.13}$$

$$\boldsymbol{f}(t) = (\boldsymbol{f}_1(t), \boldsymbol{f}_2(t), \ldots, \boldsymbol{f}_p(t))^T \tag{2.14}$$

in p Teilsysteme zerlegt werden. Die p Integrationsverfahren (pDRE0)[2] ergeben sich mit Gl. (2.5) bis (2.10) zu

$$\boldsymbol{x}_{k\,i+1} = \boldsymbol{x}_{k\,i} + \boldsymbol{A}_{Dk}\,\dot{\boldsymbol{x}}_k(\boldsymbol{x}, \boldsymbol{u}, t_i) \tag{2.15a}$$

[2] parallele DRE0–Verfahren

$$= \quad \boldsymbol{x}_{k\,i} + \boldsymbol{A}_{Dk}\, \boldsymbol{f}_k(\boldsymbol{x}, \boldsymbol{u}, t_i) \tag{2.15b}$$

mit $k = 1, 2, \ldots, p$. Dabei berechnen sich die Verfahrensmatrizen $\boldsymbol{A}_{Dk}$ nur aus den Blockmatrizen $\boldsymbol{A}_k$ der einzelnen Teilsysteme. Die Kopplungen werden in der Teilsystembeschreibung

$$\dot{\boldsymbol{x}}_k(t) = \boldsymbol{f}_k(\boldsymbol{x}, \boldsymbol{u}, t) \tag{2.16}$$

berücksichtigt. Das pDRE0–Verfahren (2.15) kann auf p Prozessoren parallel ausgeführt werden. Nach jedem Integrationsschritt muß ein Austausch der Kopplungen zwischen den jeweiligen Teilsystemen erfolgen. Die Berechnung der Systemmatrizen

$$\boldsymbol{A}_k = \left.\frac{\partial \boldsymbol{f}_k}{\partial \boldsymbol{x}_k}\right|_{\boldsymbol{x}_e} \qquad k = 1, \ldots, p \tag{2.17}$$

kann auf den p Prozessoren parallel ausgeführt werden. Ist das Gesamtsystem mit $n = 10$ in fünf Teilsysteme mit $n_i = 2$ zerlegbar, hat jeder Prozessor nur 4 Gleichungen auszuwerten, im Vergleich zu 100 bei dem DRE0–Verfahren.

Da die Berechnung der Verfahrensmatrix die Komplexität $O(n^3)$ besitzt, kann durch die Reduzierung der Systemordnung die Berechnung wesentlich beschleunigt werden. Die Verringerung des Rechenaufwandes läßt sich zu

$$\frac{O(n^3)}{O(n_i^3)} \approx \frac{n^3}{n_i^3} = \left(\frac{10}{2}\right)^3 = 125 \tag{2.18}$$

abschätzen.

2.3 Stabilitätsbetrachtung

Das DRE0–Verfahren gehört zu der Klasse der Einschritt–Verfahren. Diese sind durch die folgende Definition bestimmt:

Definition 2.1
Ein Einschritt–Verfahren zur näherungsweisen Berechnung der Lösung von Differentialgleichungen ist durch die Form

$$\boldsymbol{x}_{i+1} = \boldsymbol{x}_i + h\boldsymbol{\phi}(\boldsymbol{x}_i, \boldsymbol{u}_i, t_i, h) \tag{2.19}$$

mit der Verfahrensfunktion $\boldsymbol{\phi}$ beschrieben (Gear 1971). □

Das einfachste Einschritt–Verfahren ist das von Euler

$$\boldsymbol{x}_{i+1} = \boldsymbol{x}_i + h\boldsymbol{f}(\boldsymbol{x}_i, \boldsymbol{u}_i, t_i)\ . \tag{2.20}$$

Genügt die Verfahrensfunktion $\boldsymbol{\phi}$ der Lipschitzbedingung

$$\|\boldsymbol{\phi}(\boldsymbol{x}, \boldsymbol{u}, t, h) - \boldsymbol{\phi}(\tilde{\boldsymbol{x}}, \boldsymbol{u}, t, h)\| \leq L\, \|\boldsymbol{x} - \tilde{\boldsymbol{x}}\|\ , \tag{2.21}$$

so ist das Einschritt–Verfahren (2.19) für $0 \leq h \leq h_0$ stabil. Dies ist jedoch nur dann der Fall, wenn die Funktion $\boldsymbol{f}$ ebenfalls die Lipschitzbedingung (2.21) erfüllt. Gilt für das DRE0–Verfahren (2.10)

$$\boldsymbol{x}_{i+1} = \boldsymbol{x}_i + \boldsymbol{A}_D \dot{\boldsymbol{x}}(\boldsymbol{x}_i, \boldsymbol{u}_i, t_i) \tag{2.22}$$

die Beschränkung

$$\|\boldsymbol{A}_D\| \leq L \,, \tag{2.23}$$

mit einer positiven endlichen Konstanten L, so ist das DRE0–Verfahren stabil. Für stabile Systeme, wie vorausgesetzt, und einer beschränkten Schrittweite ist die Ungleichung erfüllt.

2.4 Fehlerabschätzung

Die exakte Lösung der Differentialgleichung (2.3a) bei bekanntem Zustand $\boldsymbol{x}_n$ und beliebig oft differenzierbarer Funktion $\boldsymbol{f}$ ist durch die Taylor–Reihe (die Argumente werden aus Gründen der Übersichtlichkeit weggelassen)

$$\boldsymbol{x}_{i+1} = \boldsymbol{x}_i + \sum_{j=1}^{\infty} \frac{h^j}{j!} \boldsymbol{x}_i^{(j)} \tag{2.24a}$$

$$= \boldsymbol{x}_i + h\boldsymbol{f}_i + \frac{h^2}{2}\dot{\boldsymbol{f}} + \frac{h^3}{6}\ddot{\boldsymbol{f}}_i + \cdots \tag{2.24b}$$

gegeben. Eine Approximation mit dem DRE0–Verfahren beruht auf Gl (2.10)

$$\boldsymbol{x}_{i+1} = \boldsymbol{x}_i + \boldsymbol{A}_D \boldsymbol{f}_i \tag{2.25a}$$

$$= \boldsymbol{x}_i + \sum_{i=1}^{\infty} \frac{h^i}{i!} \boldsymbol{A}^{i-1} \boldsymbol{f}_i \tag{2.25b}$$

$$= \boldsymbol{x}_i + h\boldsymbol{f}_i + \frac{h^2}{2}\boldsymbol{A}\boldsymbol{f}_i + \frac{h^3}{6}\boldsymbol{A}^2\boldsymbol{f}_i + \cdots \quad . \tag{2.25c}$$

Für ein lineares System (2.4) mit diskreten Eingangssignalen ergeben die zeitlichen Ableitungen von $\boldsymbol{f}$

$$\dot{\boldsymbol{f}} = \boldsymbol{A}\dot{\boldsymbol{x}} + \boldsymbol{B}\dot{\boldsymbol{u}} = \boldsymbol{A}\boldsymbol{f} \quad ; \quad \dot{\boldsymbol{u}} = \boldsymbol{0} \tag{2.26}$$

$$\ddot{\boldsymbol{f}} = \boldsymbol{A}\dot{\boldsymbol{f}} = \boldsymbol{A}^2\boldsymbol{f} \tag{2.27}$$

$$\vdots \tag{2.28}$$

$$\boldsymbol{f}^{(j)} = \boldsymbol{A}^j \boldsymbol{f} \,. \tag{2.29}$$

Diese stimmen mit den Koeffizienten von Gl. (2.24) überein. Eine Simulation mit dem DRE0–Verfahren ist daher für jede Schrittweite h exakt.

Die Ermittlung der unendlichen Reihe (2.11) muß aus praktischen Gründen nach einem endlichen Term abgebrochen werden. Für eine zu erzielende Genauigkeit von ϵ kann eine Fehlerabschätzung bei dem Abbruch nach dem j–ten Summanden zu

$$\frac{h^{j+1}}{(j+1)!}\left\|\boldsymbol{A}^{j}\right\| \leq \epsilon \tag{2.30}$$

vorgenommen werden. Dabei ist $\|\cdot\|$ die Zeilensummennorm

$$\|\boldsymbol{A}\| = \max \sum_{k=1}^{n} |a_{ik}| \quad , \quad 1 \leq i \leq n \tag{2.31}$$

einer quadratischen Matrix der Dimension $\mathbb{R}^{n\times n}$. Um diese Genauigkeit mit einem kleinen Index j zu erfüllen, muß die Matrix $h\boldsymbol{A}$ normiert werden. Moler und Van Loan (1978) zeigen, daß dies effektiv durch $h\boldsymbol{A}/\mu$ gewährleistet wird. Die Rücknahme der Normierung geschieht, aufgrund der Identität

$$e^{h\boldsymbol{A}} = \left(e^{\frac{h\boldsymbol{A}}{\mu}}\right)^{\mu} \quad , \quad \mu = 2^{k} \quad k \in \mathbb{N}\,, \tag{2.32}$$

mittels mehrmaligen Quadrierens von $e^{\frac{h\boldsymbol{A}}{\mu}}$. Die Bestimmung der Anzahl von Termen kann vor der Simulation erfolgen. Bei einer stark vom Arbeitspunkt abhängenden Systemmatrix müssen mehrere Rechnungen durchgeführt werden. Einen großen Einfluß auf die Genauigkeit hat die Schrittweite h. Damit die Bedingung (2.30) mit einem kleinem Index j erfüllt wird, sollte die Schrittweite klein sein. Hier muß ein Vergleich mit der Echtzeitbedingung, für die ein großes h gefordert ist, durchgeführt werden.

Bei dem pDRE0–Verfahren ist die Genauigkeit zusätzlich noch von der Aufteilung des Systems abhängig. Je schwächer die Teilsysteme miteinander gekoppelt sind, desto genauer ist die Simulation und die Schrittweite kann größer gewählt werden. An dem einfachen System (2.12a) wird dies deutlich: Die Zustände berechnen sich mit Gl. (2.12a) und (2.15b) zu

$$\boldsymbol{x}_{1\,n+1} = \boldsymbol{x}_{1\,n} + h\boldsymbol{f}_{1\,n} + \frac{h^2}{2}\boldsymbol{A}_1\boldsymbol{f}_{1\,n} + \cdots + \frac{h^i}{i!}\boldsymbol{A}_1^{i-1}\boldsymbol{f}_{1\,n} \tag{2.33a}$$

$$\boldsymbol{x}_{2\,n+1} = \boldsymbol{x}_{2\,n} + h\boldsymbol{f}_{2\,n} + \frac{h^2}{2}\boldsymbol{A}_2\boldsymbol{f}_{2\,n} + \cdots + \frac{h^i}{i!}\boldsymbol{A}_2^{i-1}\boldsymbol{f}_{2\,n} \quad . \tag{2.33b}$$

Verglichen mit der exakten Lösung (2.25a) ergibt sich eine Differenz ab dem quadratischen Term. Dieser ist bei der exakten Lösung durch

$$\boldsymbol{A}\boldsymbol{f}_n = \begin{pmatrix} \boldsymbol{A}_1^2 + \boldsymbol{A}_{12}\boldsymbol{A}_{21} & \boldsymbol{A}_1\boldsymbol{A}_{12} + \boldsymbol{A}_{12}\boldsymbol{A}_2 \\ \boldsymbol{A}_2\boldsymbol{A}_{21} + \boldsymbol{A}_{21}\boldsymbol{A}_1 & \boldsymbol{A}_2^2 + \boldsymbol{A}_{21}\boldsymbol{A}_{12} \end{pmatrix}\boldsymbol{x} + \begin{pmatrix} \boldsymbol{A}_1\boldsymbol{B}_1 & \boldsymbol{A}_{12}\boldsymbol{B}_2 \\ \boldsymbol{A}_{21}\boldsymbol{B}_1 & \boldsymbol{A}_2\boldsymbol{B}_2 \end{pmatrix}\boldsymbol{u} \tag{2.34}$$

und bei dem pDRE0–Verfahren (2.33a) durch

$$\begin{pmatrix} \boldsymbol{A}_1\boldsymbol{f}_{1\,n} \\ \boldsymbol{A}_2\boldsymbol{f}_{2\,n} \end{pmatrix} = \begin{pmatrix} \boldsymbol{A}_1^2 & \boldsymbol{A}_1\boldsymbol{A}_{12} \\ \boldsymbol{A}_2\boldsymbol{A}_{21} & \boldsymbol{A}_2^2 \end{pmatrix}\boldsymbol{x} + \begin{pmatrix} \boldsymbol{A}_1\boldsymbol{B}_1 & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{A}_2\boldsymbol{B}_2 \end{pmatrix}\boldsymbol{u} \tag{2.35}$$

gegeben. Damit der Fehler möglichst klein ist, sollten die Kopplungsterme gegenüber der Systemmatrix einen geringen Einfluß auf die Lösung haben. Dies wird zum einen durch eine kleine Schrittweite erreicht. Zum anderen sollten die Koppelgrößen die Bedingung

$$\frac{\sum_{j=1}^{p} \|\boldsymbol{A}_{ij}\boldsymbol{x}_j\|}{\|\boldsymbol{A}_i\boldsymbol{x}_i\|} \ll 1 \quad , i = 1, \ldots, p \quad , \quad i \neq j \tag{2.36}$$

erfüllen, da die Echtzeitbedingung eine möglichst große Schrittweite fordert. Damit läßt sich die Güte der Aufteilung des Systems in Teilsysteme quantitativ bewerten. Clemens (1992a) gibt als Kriterium gleiche physikalische Funktionen an. Durch diese Aufteilung wird die Bandstruktur der Systemmatrix berücksichtigt und die Bedingung (2.36) nimmt ein Minimum an.

Die Eingangssignale wirken sich, genauso wie bei einem expliziten Einschrittverfahren erster Ordnung und allen expliziten Mehrschrittverfahren, erst nach einem Integrationsschritt auf die anderen Teilsysteme aus. Aus dem Vergleich der exakten (2.34) mit der des pDRE0–Verfahrens (2.35) wird dies deutlich. Eine Anregung des Systems durch $\boldsymbol{u}_{1\,i}$ beeinflußt im ersten Integrationsschritt nur $\boldsymbol{x}_{1\,i+1}$ und nicht $\boldsymbol{x}_{2\,i+1}$, da ab dem quadratischen Term (2.35) die Kopplungsmatrizen der Eingangssignale null sind. Nach einem Integrationsschritt wird $\boldsymbol{x}_2$ dann durch die Kopplung der Systemmatrix angeregt. Um die Abweichung von der exakten Lösung gering zu halten, sollte die Schrittweite kleiner sein als die größte Zeitkonstante des Systems. Sie bestimmt hauptsächlich das Übergangsverhalten des Systems und dadurch wird die Simulation der dominierenden Dynamik des Systems gewährleistet. Gleichzeitig dürfen jedoch die Kopplungen nicht zu stark sein, d.h. die Bedingung (2.36) ist zu erfüllen.

3 Simulationsergebnisse

Die Anwendbarkeit des DRE0– und pDRE0–Verfahrens wurde anhand von Testgleichungen untersucht. Alle dargestellten Rechenzeiten wurden auf einem Transputer–System (Inmos 1988a) mit Prozessoren vom Typ T 800/25 MHz ermittelt. Die Algorithmen sind in der Programmiersprache OCCAM (Inmos 1988b) geschrieben. Da zur einfacheren Programmierung Unterprogramme verwendet wurden, lassen sich die absoluten Rechenzeiten noch verringern. Im Vergleich zu dem Euler–Verfahren (EUL), einem Runge–Kutta–Verfahren dritter Ordnung (RK3) und dem Adam–Bashworth–Verfahren dritter Ordnung (AB3), erlaubt das hier vorgestellte Verfahren eine wesentlich größere Schrittweite. Speziell für steife Systeme ergibt sich damit eine deutlich kürzere Rechenzeit. Dies soll an einem einfachen Beispiel erläutert werden.

3.1 Steife Systeme

Betrachtet wird die Reihenschaltung von drei Drosseln innerhalb eines hydraulischen Systems (Bild 3.1).

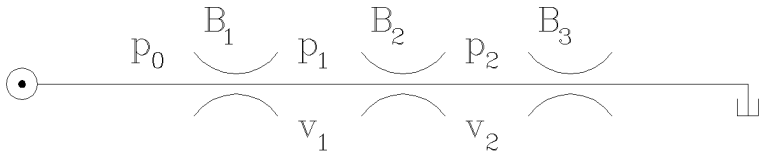

Bild 3.1: Reihenschaltung von drei Drosseln

Die Drücke in der Leitung lassen sich durch die Differentialgleichungen

$$\dot{p}_1 = \frac{E_{Oel}}{v_1}\left(B_1\left(p_0 - p_1\right) - B_2\left(p_1 - p_2\right)\right) \tag{3.1}$$

$$\dot{p}_2 = \frac{E_{Oel}}{v_2}\left(B_2\left(p_1 - p_2\right) - B_3\, p_2\right) \tag{3.2}$$

mit den Eigenwerten und der Steifheit

$$\lambda_1 = -40 \quad , \quad \lambda_2 = -50010 \quad , \quad St = 1250$$

beschreiben (Zhang und Ulrich 1989).

Die Berechnung der Verfahrensmatrix $\boldsymbol{A}_D$ kann ohne eine Normierung nicht durchgeführt werden. Mit dem Normierungsfaktor μ kann die Abbruchbedingung beeinflußt werden, so daß zur Berechnung der Verfahrensmatrix $\boldsymbol{A}_D$ nur sehr wenige Terme notwendig sind. Dies soll an dem Beispiel der Reihenschaltung von drei Drosseln gezeigt werden. Die Systemmatrix ergibt sich zu

$$\boldsymbol{A} = \begin{pmatrix} 40 & -20 \\ -10 & 50010 \end{pmatrix} \tag{3.3}$$

mit der Spaltennorm

$$\|\boldsymbol{A}\| = 50030 \; . \tag{3.4}$$

Für eine Schrittweite von $h = 1$ ms und einer Genauigkeit von $\epsilon = 1$ % wird die Bedingung (2.30) in den ersten 50 Termen nicht erfüllt. Mit einer Normierung von $\mu = 16$ (Gl. (2.32)) kann die Berechnung der Verfahrensmatrix nach dem zweiten Term abgebrochen werden. Die Normierung wird durch Potenzieren der Matrix $\boldsymbol{A}_D$ zurückgenommen.

Damit die Abweichung des pDRE0–Verfahrens bei dem zweiten Zustand p_2 nicht zu groß wird, sollte die Schrittweite nicht größer als $h = 1$ ms sein. Die Bedingung (2.36) ist für diesen Zustand mit

$$\frac{\|\boldsymbol{A}_{21} x_1\|}{\|\boldsymbol{A}_2 x_2\|} \leq \frac{\|\boldsymbol{A}_{21}\|}{\|\boldsymbol{A}_2\|} \frac{\|\boldsymbol{x}_1\|}{\|\boldsymbol{x}_2\|} = \frac{10}{50010} \frac{50}{0,01} \approx 1 \tag{3.5}$$

nicht erfüllt. Durch das Verhältnis der großen Zeitkonstante zu der kleinen Schrittweite von

$$\frac{T_{max}}{h} = \frac{1}{\lambda_{min} h} \approx \frac{1}{40} \frac{1}{10^{-3}} = 25 \tag{3.6}$$

liegt jedoch die Abweichung unter 1 %. Wird nur der erste Zustand innerhalb der Echtzeitsimulation benötigt, kann die Schrittweite nochmals erhöht werden. Dies ist möglich, da der zweite Zustand nur einen geringen Einfluß auf den ersten hat. Anhand von (2.36) mit

$$\frac{\|\boldsymbol{A}_{12} x_2\|}{\|\boldsymbol{A}_1 x_1\|} \leq \frac{\|\boldsymbol{A}_{12}\|}{\|\boldsymbol{A}_1\|} \frac{\|\boldsymbol{x}_2\|}{\|\boldsymbol{x}_1\|} = \frac{20}{40} \frac{0,01}{50} = 10^{-4} \ll 1 \tag{3.7}$$

wird dies deutlich.

Bild 3.2 stellt den Druckverlauf p_2 dar. Aufgetragen ist für einen Eingangsdruck von $p_0 = 100$ bar $\cdot 1(t)$ der Zeitraum von 300 ms. Die Lösungen mit dem DRE0–Verfahren sind für jede Schrittweite exakt und stabil, da das Verfahren von der Differenzengleichung abgeleitet wurde. Um veränderliche Eingangssignale während des Integrationsschrittes besser berücksichtigen zu können, kann das Abtast–Halte–Glied nullter Ordnung durch ein Glied erster Ordnung ersetzt werden. Damit läßt sich ein DRE1–Verfahren entwickeln. Eine Simulation mit Schrittweitensteuerung (ODE 23 aus Matlab (1989)) zeigt ein stark oszillierendes Verhalten und ist deshalb nur eingeschränkt verwendbar (Punkte in Bild 3.2). Mit dem pDRE0–Verfahren, die Differentialgleichungen werden auf je einem Prozessor integriert, ist das System ebenfalls mit sehr großen Schrittweiten noch stabil zu simulieren. In Bild 3.2 ist der Druckverlauf für eine Schrittweite von $h = 10$ ms aufgetragen. Die Verfahren Euler und Adam–Bashworth sind aufgrund ihrer Stabilitätsgebiete (Gear 1971) für eine Schrittweite von

$$h > \frac{2}{|\lambda|} = \frac{2}{50010} \approx 4\,10^{-5}\text{s} \tag{3.8}$$

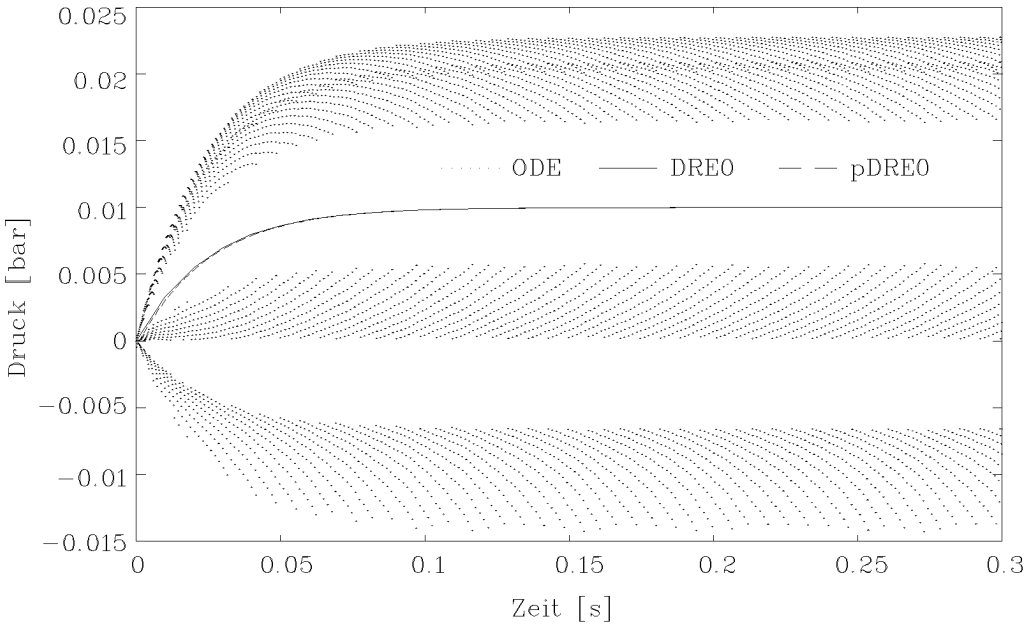

Bild 3.2: Druck p_2 nach der zweiten Drossel

instabil. Die leichte Abweichung des pDRE0–Verfahrens von der exakten Lösung liegt an der Vernachlässigung der Kopplungen während des Integrationsschrittes. Der Fehler ist abhängig von der Schrittweite und der Stärke der Kopplungen.

Rechenzeitvergleich

In Tabelle 3.1 sind die Rechenzeiten T_R für verschiedene Integrationsverfahren aufgetragen. Auffallend ist die lange Simulationszeit von $T_R = 3768$ ms des ODE–Verfahrens,

Verfahren	h[s]	T_R[ms]	p	σ
ODE	–	3768	1	–
EUL	10^{-5}	667	1	–
DRE0	10^{-3}	16	1	41,7
pDRE0	10^{-3}	15	2	44,5

Tabelle 3.1: Rechenzeiten für das Drosselmodell

welche aus einer sehr kleinen Schrittweite resultiert. Mit dem EUL–Verfahren muß eine maximale Schrittweite von 10^{-5} s eingehalten werden. Die Rechenzeit beträgt damit $T_R = 667$ ms. Eine Echtzeitsimulation ist mit dem EUL–Verfahren somit nicht möglich. Das AB3–Verfahren ist bei dieser Schrittweite und dem Eingangssignal von $p_0 = 100\,1(t)$ schon instabil und deshalb in der Tabelle 3.1 nicht enthalten. Durch die Verwendung des DRE0–Verfahrens kann die Simulation um den Faktor $\sigma = 44.5$ auf $T_R = 15$ ms beschleunigt werden. Eine Echtzeitsimulation des Drosselmodells wird damit möglich.

3.2 Nichtlineare Systeme

Als Beispiel eines nichtlinearen Systems sollen die Ergebnisse der Simulation eines Turbogenerators vorgestellt werden. Es handelt sich um ein nichtlineares System fünfter Ordnung mit den zwei Teilsystemen, Turbine und Synchronmaschine (Clemens 1993). Die Systembeschreibung nach (2.3a) lautet

$$\dot{x}_1 = x_2 \tag{3.9a}$$
$$\dot{x}_2 = -c_1 x_2 - c_2 \sin(x_1) x_3 - 0,5 c_3 \sin(2x_1) + \frac{1}{M} x_5 \tag{3.9b}$$
$$\dot{x}_3 = c_5 \cos(x_1) - c_4 x_3 + c_6 u_2 \tag{3.9c}$$
$$\dot{x}_4 = k_1 u_1 - k_2 x_2 - k_3 x_4 \tag{3.9d}$$
$$\dot{x}_5 = k_4 x_4 - k_5 x_5 \; . \tag{3.9e}$$

Die ersten 3 Zustände beschreiben die Synchronmaschine, die Zustände 4 und 5 die Turbine. Simuliert wurde eine Laständerung von 80% auf 20% über eine Zeit von 3 s. durch

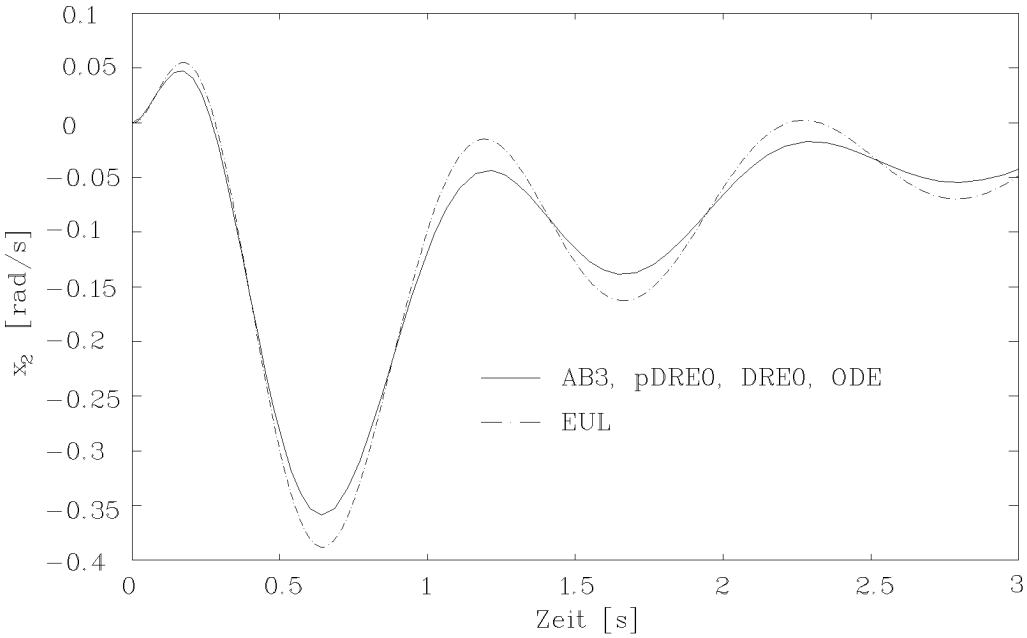

Bild 3.3: Geschwindigkeit des Rotorwinkels simuliert mit $h = 10$ ms

ein Runge–Kutta–Verfahren (ODE) mit Schrittweitensteuerung und einer Genauigkeit von 10^{-6}. Dargestellt ist der zweite Zustand, die Geschwindigkeit des Rotorwinkels der Synchronmaschine. Dies ist der Zustand mit den stärksten Nichtlinearitäten und die Differentialgleichung enthält die Kopplung mit dem zweiten Teilsystem, der Turbine. In Bild 3.3 sind zusätzlich die Simulationsergebnisse für die verschiedenen Integrationsverfahren mit einer konstanten Schrittweite von $h = 10$ ms aufgetragen. Für diese Bild 3.3 zeigt

die Referenzsimulation Schrittweite liefert das EUL–Verfahren noch akzeptable Werte.

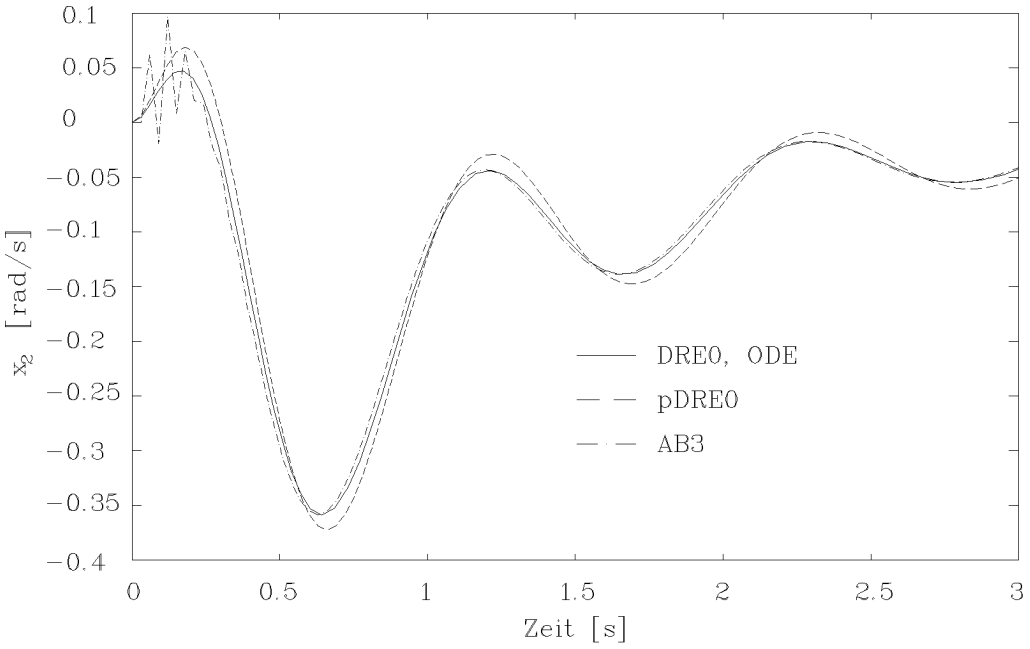

Bild 3.4: Geschwindigkeit des Rotorwinkels simuliert mit $h = 30$ ms

Die Ergebnisse mit einer Schrittweite von $h = 30$ ms sind in Bild 3.4 dargestellt. Das EUL–Verfahren ist für diese Schrittweite instabil. Bei einer Änderung des Eingangssignals zeigt das Mehrschrittverfahren (AB3) eine deutliche Schwingneigung. Eine Simulation mit veränderlichen diskreten Eingangssignalen wäre somit in der Genauigkeit stark reduziert.

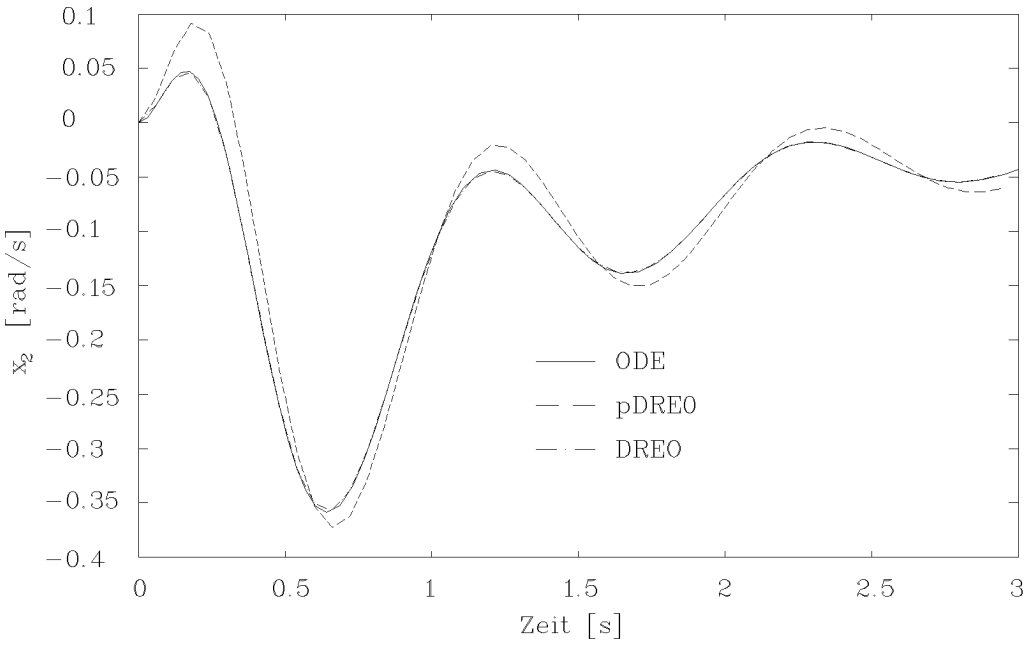

Bild 3.5: Geschwindigkeit des Rotorwinkels simuliert mit $h = 60$ ms

Die Verfahren DRE0 und pDRE0 liefern, im Gegensatz zum AB3–Verfahren, auch für noch größere Schrittweiten eine stabile Simulation. Das DRE0–Verfahren hat im Vergleich zum pDRE0 eine höhere Genauigkeit, da die Änderungen der Kopplungen im Integrationsschritt mit berücksichtigt werden (Bild 3.5).

Rechenzeitvergleich

Die ermittelten Rechenzeiten für das nichtlineare System Turbogenerator sind in der Tabelle 3.2 enthalten. Auffallend ist die lange Rechenzeit des DRE0–Verfahrens. Das Ver-

Verfahren	T_R[s]			p	σ
	h =10 ms	h =20 ms	h =30 ms		
AB3	72	24	instabil	1	–
EUL	40	instabil	instabil	1	–
DRE0	468	162	81	1	–
pDRE0	78	26	13	2	6,2

Tabelle 3.2: Rechenzeiten für das nichtlineare System Turbogenerator

fahren benötigt, im Vergleich zu dem EUL–Verfahren, etwa die 12 fache Zeit, um das System fünfter Ordnung zu simulieren. Die Berechnung der Verfahrensmatrix $\boldsymbol{A}_D$ konnte für einen Fehler von $\epsilon = 1\%$ mit der Bedingung (2.36) nach dem 3. Glied abgebrochen werden. Das pDRE0–Verfahren erzielt eine Beschleunigung von

$$\sigma = \frac{T_R(DRE0)}{T_R(pDRE0)} = \frac{486}{78} \approx 6,2 \tag{3.10}$$

im Vergleich zu dem DRE0–Verfahren. Dabei muß berücksichtigt werden, daß die Prozessoren nicht gleichmäßig ausgelastet sind. So ist die Integration der Synchronmaschine, welche sämtliche Nichtlinearitäten enthält und eine Systemordnung größer ist, rechenaufwendiger als die der Turbine. Für Systeme mit einer gleichmäßigen Zerlegung läßt sich der Beschleunigungswert noch steigern.

Auch mit diesen ungünstigen Voraussetzungen und der Verwendung von nur zwei Prozessoren läßt sich die Simulation, verglichen mit dem EUL–Verfahren, um

$$\sigma = \frac{T_R(EUL)}{T_R(pDRE0)} = \frac{40}{13} = 3,1 \tag{3.11}$$

beschleunigen, bei vergleichbarer Genauigkeit. Gegenüber dem AB3–Verfahren ist sogar eine Beschleunigung um

$$\sigma = \frac{T_R(AB3)}{T_R(pDRE0)} = \frac{72}{13} = 5,5 \tag{3.12}$$

möglich, jedoch mit einer geringeren Genauigkeit.

Alle Verfahren sind echtzeitfähig. Je nach zusätzlichen Anforderungen, wie z.B. Sicherungs-, Steuer- und Regelungsalgorithmen, kann es jedoch notwendig sein, die Rechenzeit der Simulation zu minimieren.

3.3 Komplexe Systeme mit schnellen Teilsystemen

Besonders geeignet ist das pDRE0–Verfahren bei komplexen Systemen mit separierbaren schnellen Teilsystemen. Unter der Bedingung (Clemens 1992b)

$$n_1 = \dim \boldsymbol{z} < \dim \boldsymbol{v} = n_2 \tag{3.13}$$

mit n_1, als der Ordnung der schnellen Teilsystemzustände $\boldsymbol{z}$, und n_2, als der Ordnung der langsamen Restsystemzustände $\boldsymbol{v}$, verlängern die notwendigen zusätzlichen Rechenoperationen des pDRE0–Verfahrens nicht die Simulation. Als Beispiel eines komplexen Systems mit einem schnellen Teilsystem wird hier die Simulation eines hydraulischen Mobilantriebes (HMA) (Ebner 1992) vorgestellt. Es handelt sich um ein stark nichtlineares System siebter Ordnung.

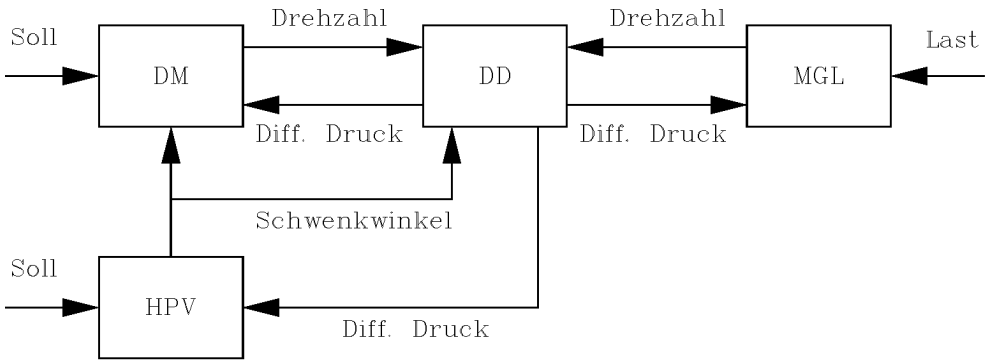

Bild 3.6: Struktur des hydraulischen Mobilantriebes (HMA)

Die Struktur des HMA ist in Bild 3.6 dargestellt. Es besteht aus den Teilsystemen:

- Dieselmotor und Hydropumpe (DM), ein System 2. Ordnung,
- Verstelleinrichtung der Hydropumpe (HPV), ein System 2. Ordnung,
- Druckdynamik (DD), ein System 2. Ordnung,
- Hydromotor, Schaltgetriebe und Last (MGL), ein System 1. Ordnung.

Das Bild 3.7 zeigt die Verläufe der Zustände Drehzahl des Dieselmotors (DM), Differenzdruck, Schwenkwinkel und Drehzahl des Hydromotors (HM) für einen Drehzahlsollwertsprung von 100 %, einem Schwenkwinkelsollwertsprung von 50 % und einem Lastmoment von

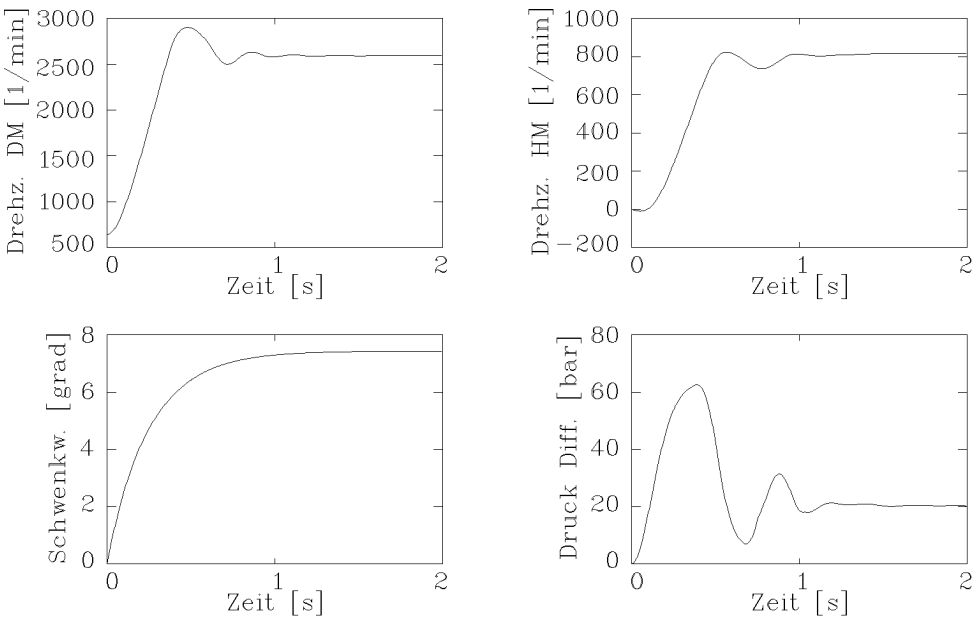

Bild 3.7: Verlauf der Zustände des hydraulischen Mobilantriebes

10 Nm. Die Verläufe wurden mit einem Einschrittverfahren fünfter Ordnung und Schrittweitensteuerung bei einer Genauigkeit von $\varepsilon = 10^{-6}$ ermittelt. Sie dienen für die folgenden Simulationen als Referenz.

Um eine gleichmäßige Lastverteilung zu erreichen, wurden die Teilsysteme unabhängig

Teilsystem	Schrittweite h [ms]		
	EUL	RK3	pDRE0
Dieselmotor und Hydropumpe (DM)	1	10	–
Verstelleinrichtung der Hydropumpe (HPV)	0,1	0,1	10
Druckdynamik (DD)	1	10	–
Hydromotor, Schaltgetriebe und Last (MGL)	1	10	–

Tabelle 3.3: Mögliche Schrittweiten der verschiedenen Verfahren

voneinander simuliert. Dies dient zur Ermittlung der maximal möglichen Schrittweite und der benötigten Rechenzeit. Zur Integration wurden das EUL– und das RK3–Verfahren verwendet. Das EUL–Vefahren benötigt nur eine Bestimmung der Systemfunktion, im Vergleich zu drei Auswertungen bei dem RK3–Verfahren. Jedoch ermöglicht letzteres die Verwendung einer größeren Schrittweite. Beide Verfahren erfüllen die Echtzeitbedingung der Integration (Clemens 1992a), d.h. zur Ermittlung eines neuen Zustandswertes werden nur zurückliegende Werte benötigt. In der Tabelle 3.3 sind die maximal möglichen Schrittweiten zur Integration der Teilsysteme für einen Simulationsfehler von $\varepsilon < 1$ % enthalten. Bei dieser Genauigkeit lassen sich ohne Ausschnittvergrößerungen keine Un-

terschiede zu der Referenzsimulation in Bild 3.7 erkennen. Deshalb werden im folgenden keine Zustandsverläufe dargestellt. Das schnellste Teilsystem, die Verstelleinrichtung der Hydropumpe, läßt sich mit den beiden Verfahren EUL und RK3 nur mit einer Schrittweite von $h = 0,1$ ms integrieren. Die anderen Teilsysteme erlauben für das EUL–Verfahren $h = 1$ ms und für das RK3–Verfahren $h = 10$ ms. Durch die Verwendung des pDRE0–Verfahrens kann die Schrittweite für das Teilsystem HPV auf $h = 10$ ms erhöht werden.

Bei Matrizen mit sehr großen Normen wird die Abbruchbedingung (2.30) erst für einen hohen Index j erfüllt. Für die HPV ergab sich bei einer Schrittweite von $h = 10$ ms und einer Genauigkeit von $\epsilon = 1$ %, daß die Bedingung (2.30) in den ersten 50 Termen nicht erfüllt wird. Mit einer Normierung von $\mu = 64$ kann die Berechnung der Verfahrensmatrix nach dem zweiten Term abgebrochen werden. Die notwendigen Rechenzeiten T_R

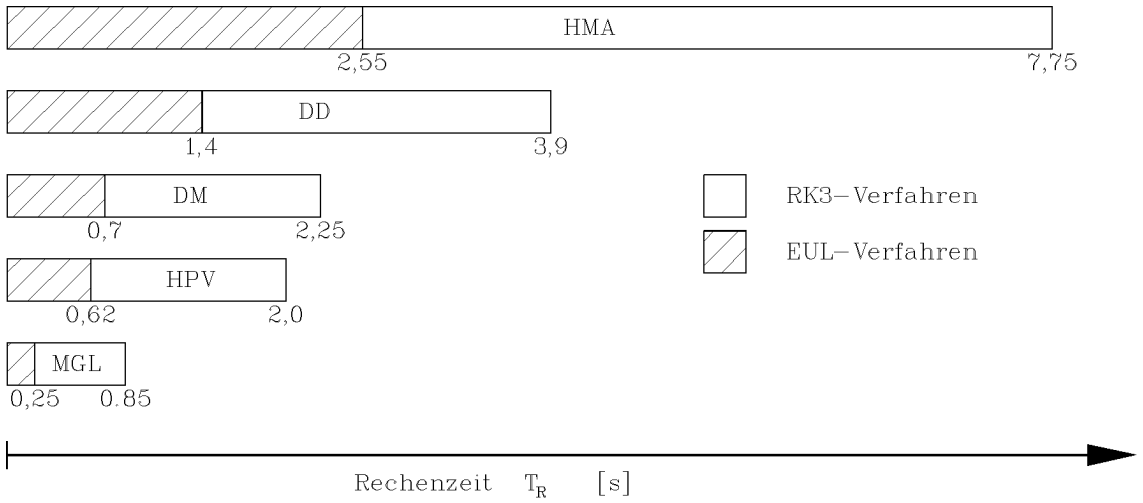

Bild 3.8: Rechenzeit der Teilsysteme für eine Echtzeitsekunde

der Teilsysteme für eine Simulationszeit von 1 s sind in Bild 3.8 aufgetragen. Es wurde eine einheitliche Schrittweite von $h = 0,1$ ms verwendet, da nur für diese Schrittweite das Gesamtsystem (HMA) stabil zu simulieren ist. Eine Echtzeitsimulation ist mit diesen Verfahren und nur einem Prozessor nicht möglich. Die benötigten Rechenzeiten betragen $T_{R,HMA}(EUL) = 2,55$ s und $T_{R,HMA}(RK3) = 7,75$ s. Die Aufteilung des Systems auf vier Prozessoren würde die Simulation etwa auf die Rechenzeit $T_{R,DD}(EUL) = 1,4$ s oder $T_{R,DD}(RK3) = 3,9$ s der Druckdynamik beschleunigen, jedoch sind die Stillstandszeiten und die notwendige Kommunikation der Prozessoren sehr groß.

Eine effizientere Systemaufteilung besteht in der Trennung von schnellen und langsamen Teilsystemen. Unter der Bezeichnung Motor–Druckdynamik–Motor (MDM) sind die Teilsysteme DM, DD und MGL zusammengefaßt. Bild 3.9 zeigt die notwendigen Rechenzeiten. Der Vergleich der Rechenzeiten der Teilsysteme MDM und HPV für das EUL– und das RK3–Verfahren zeigt, daß die Bedingung (3.13) erfüllt ist. Somit kann ein Multirate–

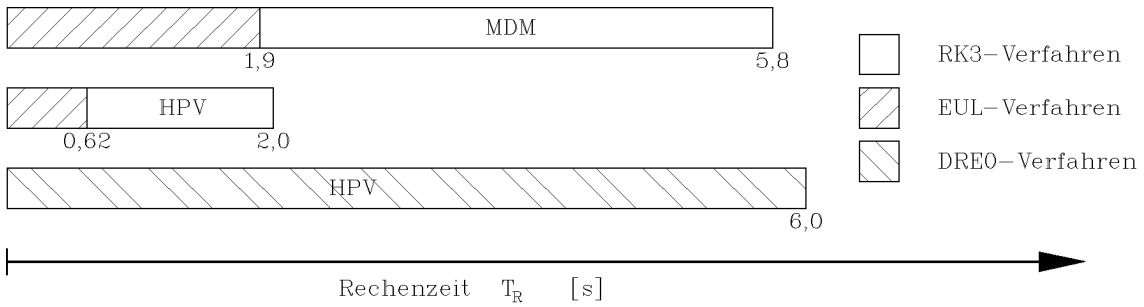

Bild 3.9: Rechenzeit der Teilsysteme MDM und HPV für eine Echtzeitsekunde

Verfahren (Clemens 1992b) zur Beschleunigung der Simulation verwendet werden. Dabei wird das Teilsystem MDM mit einer größeren Schrittweite als die schnelle HPV simuliert, welche, zum Erreichen des gleichen Zeitschrittes, mehrmals integriert wird. Beide Teilsysteme können auf je einem Prozessor implementiert werden. Das Gantt Chart in Bild 3.10

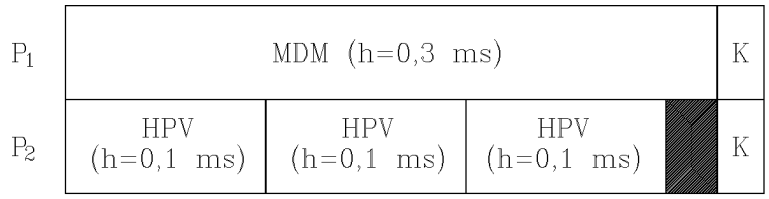

Bild 3.10: Gantt Chart der Integration mit dem Multirate–Verfahren.

verdeutlicht den parallelen Programmablauf. Prozessor P_1 integriert das Teilsystem MDM mit $h_{MDM} = 0,3$ ms und Prozessor P_2 die HPV mit $h_{HPV} = 0,1$ ms. Die Stillstandszeit (schwarzer Bereich) ist sehr klein. Der Buchstabe K kennzeichnet die Kommunikation der Prozessoren. In der Tabelle 3.4 sind die erzielten Rechenzeiten für diese parallele Simula-

Verfahren	T_R [s]	p	σ	η [%]
RK3 sequentiell	7,75	1	–	–
EUL sequentiell	2,55	1	–	–
RK3 parallel	2,15	2	3,6	180
EUL parallel	0,67	2	3,8	190

Tabelle 3.4: Rechenzeiten der parallelen Simulation mit dem Multirate–Verfahren

tion enthalten. Mit dem EUL–Multirate–Verfahren läßt sich die Simulation um $\sigma = 3,8$ auf eine Rechenzeit von $T_R = 670$ ms beschleunigen. Der Vergleich mit der sequentiellen Rechenzeit ($h_{MDM} = h_{HPV} = 0,1$ ms) ergibt eine Effizienz von $\eta = 190$ %. Eine Echtzeitsimulation ist mit dieser Anordnung möglich.

Durch die Verwendung des pDRE0–Verfahrens zur Integration der HPV kann die Rechenzeit nochmals erheblich reduziert werden, da sich dann beide Teilsysteme mit $h = 10$ ms simulieren lassen. Für das Motor–Druckdynamik–Motor (MDM)–System muß dabei aus Gründen der Genauigkeit das RK3–Verfahren benutzt werden. Ein Vergleich der Rechenzeiten in Bild 3.9 liefert nahezu gleiche Werte. Dies ist für eine gleichmäßige Auslastung der Prozesoren notwendig, um eine möglichst hohe Beschleunigung zu erzielen. Tabelle 3.5 zeigt die ermittelten Rechenzeiten. Die Simulation einer Echtzeitsekunde erfordert

Verfahren	T_R [s]	p	σ	η [%]
RK3 sequentiell	7,75	1	–	–
MDM (RK3) HPV (pDRE0)	0,0625	2	124	6200

Tabelle 3.5: Rechenzeiten der parallelen Simulation mit dem pDRE0–Verfahren

mit diesem Verfahren nur $62,5$ ms, was einer Beschleunigung von $\sigma = 124$ gegenüber dem sequentiellen Algorithmus mit dem RK3–Verfahren entspricht. Damit ist es möglich noch weitere Funktionen, z.B. Überwachungs–, Steuer– und Regelungsaufgaben, die on–line ablaufen, zusätzlich zur Echtzeitsimulation auf den Prozessoren zu implementieren.

Die Integration der HPV mit dem pDRE0–Verfahren kann mit dem Teilsystem MDM auch sequentiell ausgeführt werden. In der Tabelle 3.6 ist die Rechenzeit des sequentiellen mit

Verfahren	T_R [ms]	p	σ	η [%]
MDM (RK3) HPV (pDRE0) sequentiell	120	1	–	–
MDM (RK3) HPV (pDRE0) parallel	62,5	2	1,92	96

Tabelle 3.6: Rechenzeit der sequentiellen und parallelen Simulation mit dem pDRE0–Verfahren

der des parallelen Algorithmus verglichen. Die Beschleunigung reduziert sich bei diesem Vergleich auf $\sigma = 1,92$. Für größere Systeme, welche die Bedingung (3.13) erfüllen, lassen sich durch Hinzunahme weiterer Prozessoren höhere Beschleunigungen erzielen.

4 Zusammenfassung und Ausblick

Die Realisierung von Echtzeitsimulationen ermöglicht die Verbindung von realen und simulierten Systemen (simulation–in–the–loop). Bei der Modellbildung treten häufig steife Differentialgleichungen auf, welche die Einhaltung der Echtzeitbedingung erschweren. Speziell bei mechatronischen Systemen, eine Verbindung von mechanischen und elektronischen Komponenten, kann dies eine Echtzeitsimulation verhindern. In diesem Bericht wurde ein neues Integrationsverfahren (DRE0 = discrete response equivalent mit Abtastglied nullter Ordnung) vorgestellt, welches sich zur Echtzeitsimulation eignet. Die guten Stabilitätseigenschaften erlauben die Verwendung großer Schrittweiten, da das Integrationsverfahren der jeweiligen Systemdynamik angepaßt wird. Dies ermöglicht speziell bei steifen Systemen eine sehr schnelle Simulation im Vergleich zu einfachen expliziten Integrationsverfahren, wie sie zur Echtzeitsimulation eingesetzt werden.

Das Integrationsverfahren eignet sich zur Implementierung auf einem Parallelrechner. Damit kann die Rechenzeit nocheinmal deutlich verringert werden, da jeder Prozessor nur ein Teilsystem integrieren muß. Eine weitere Beschleunigung kann durch die unterschiedliche Behandlung von schnellen und langsamen Systemteilen erzielt werden. Sie lassen sich mit unterschiedlichen Schrittweiten und verschiedenen Verfahren simulieren. Das neue parallele DRE0–Verfahren gestattet bei steifen Systemen eine einheitlich große Schrittweite. Dabei werden die schnellen Teilsysteme mit dem der Teilsystemdynamik angepaßten parallelen DRE0–Verfahren integriert und die langsamen Teilsysteme mit einem einfachen expliziten Verfahren. Es werden eine Stabilitätsbetrachtung und eine Fehlerabschätzung für das Integrationsverfahren vorgenommen. Für eine möglichst gleichmäßige Lastverteilung auf den Prozessoren und somit eine gute Effizienz müssen die Rechenzeiten der Teilsysteme ermittelt werden. Anhand der Simulation eines hydraulischen Mobilantriebes wird die Anwendung des pDRE0–Verfahrens zur parallelen Simulation erläutert. Die Rechenzeit konnte dabei gegenüber dem sequentiellen RK3–Verfahren um den Faktor $\sigma = 124$ beschleunigt werden.

Bei einer simulation–in–the–loop ist die Echtzeitsimulation mit einem realen System über Ein– und Ausgangsgrößen gekoppelt. Die Zustände des realen Systems, welche als Eingangsgrößen auf die Simulation einwirken, ändern sich kontinuierlich. Um eine genauere Simulation zu erzielen, kann das Abtast–Halte–Glied nullter Ordnung durch ein Glied erster Ordnung ersetzt werden. Das resultierende Verfahren sowie die Anwendung bei verschiedenen Systemen ist Gegenstand derzeitiger Untersuchungen.

5 Literaturverzeichnis

Clemens, D. 1992a. *Parallele Simulation mit Transputer–Systemen.* Forschungsbericht 5/92. MSRT Universität –GH– Duisburg.

Clemens, D. 1992b. Parallele Simulation mit Transputern. *Tagungsband des 4. Transputer Anwender Treffen.* Berlin: Springer.

Clemens, D. 1993. Zur hierarchisch optimalen Regelung nichtlinearer Systeme. Forschungsbericht 2/93. MSRT Universität –GH– Duisburg.

Ebner, R. 1992. *Zur Modellbildung und Simulation hydraulischer Mobilantriebe mit Dieselmotor.* Zwischenbericht I/92 MSRT (unveröffentlicht). Universität –GH– Duisburg.

Engeln–Müllges, G. und F. Reutter. 1988. *Formelsammlung zur Numerischen Mathematik mit Standard–FORTRAN 77–Programmen.* Mannheim: BI Wissenschaftsverlag.

Gear, C. W. 1971. *Numerical Initial Value Problems in Ordinary Differential Equations.* Englewood Cliffs: Prentice–Hall.

Gear, C. W. 1977. Simulation: Conflicts between Real–Time and Software. *Mathematical Software III,* ed. J. R. Rice. New York: Academic Press.

Golub, G. H. und C. F. Van Loan. 1989. *Matrix Computations.* Baltimore: Johns Hopkins.

Grote, D. 1993. *Erweiterung einer blockorientierten Simulationsumgebung auf einem Transputer–System.* Studienarbeit MSRT (unveröffentlicht). Universität –GH– Duisburg.

Hwang, R. S. u.a. 1988. Parallel Processing for Real–Time Dynamic System Simulation. *Proc. Conf. Advances in Design Automation.* 509–518.

Inmos Ltd. 1988a. *Transputer reference manual.* London: Prentice Hall.

Inmos Ltd. 1988b. *OCCAM 2: Reference Manual.* London: Prentice Hall.

Jelali, M. 1992. *Verteilte Simulation eines hydraulischen Mobilantriebes auf einem Transputer–System.* Studienarbeit MSRT (unveröffentlicht). Universität –GH– Duisburg.

Keller, H. B. 1987. Algorithmische und problemstrukturelle Parallelität–Ansätze zur verteilten Simulation komplexer dynamischer Systeme. *4. Symposium Simulationstechnik.* Berlin: Springer.

Matlab 1989. *MATLAB for 80386-based MS-DOS Personal Computers.* The Math Works Inc., 21 Eliot Street, South Natick.

Metzger, M. 1992. DRE Integration Algorithms for Real–Time Simulation. *Computational Systems Analysis.* 399–404.

Moler, C. B. und C. F. Van Loan. 1978. Nineteen Dubious Ways to Compute the Exponential of a Matrix. *SIAM Review 20.* 801–836.

Quinn, M. J. 1988. *Algorithmenbau und Parallelcomputer.* Hamburg: McGraw–Hill.

Schwarz, H. 1979. *Zeitdiskrete Regelungssysteme.* Braunschweig: Vieweg und Sohn.

Svaricek, F. 1990. Zur numerischen Zuverlässigkeit regelungstechnischer Analyse– und Syntheseprogramme. *Automatisierungstechnik 12.* 447–453.

Thompson, H. A. 1992. *Parallel Processing for Jet Engine Control.* Berlin: Springer.

Zhang, H. und H. Ulrich. 1989. Leistungsfähiges Integrationsverfahren zur dynamischen Simulation hydraulischer Systeme. *Ölhydraulik und Pneumatik 33.* 743–747.