

# Autoskalierung dynamischer Systeme zur Codegenerierung für Festkommaprozessoren

Dipl.-Ing. Roman Frank Starbek, Stuttgart

Prof. Dr.-Ing. Hermann Henrichfreise, Köln

Dr.-Ing. Ulrich Kiffmeier, Paderborn

## Kurzfassung

Der Einsatz von Codegeneratoren zur automatischen Erzeugung von Produktionscode für Seriensteuergeräte führt zu einer deutlichen Reduktion der Entwicklungszeiten. Die für die Festkomma-Arithmetik notwendige Skalierung von Variablen dynamischer Systeme erfolgt allerdings häufig noch manuell. Sie ist sehr zeitaufwändig und muss zur Vermeidung von Überläufen und unerwünschten Sättigungen mit großer Sorgfalt erfolgen. Durch die Verwendung von Skalierungswerkzeugen lässt sich die Entwicklungszeit weiter verkürzen und zuverlässiger Steuergerätecode erzeugen.

In diesem Beitrag wird solch ein Skalierungswerkzeug vorgestellt und dessen Integration in die Umgebung des Codegenerators TargetLink<sup>1</sup> beschrieben. Am Beispiel einer Zustandslageregelung für ein elektromechanisches Positioniersystem wird der optimale Prototyp des Reglers nach einer geeigneten systematischen Aufbereitung in Produktionscode für Seriensteuergeräte überführt. Die Realisierung der Regelung mit Festkomma-Arithmetik zeigt das gleiche sehr gute Führungs- und Störverhalten wie der Gleitkomma-Prototyp.

## Abstract

The use of code generators to create production code for electronic control units (ECU) results in a significant reduction of development time. However, in most cases the necessary scaling of variables for fixed-point arithmetics is still performed manually. This time-consuming process must be conducted very carefully to avoid integer overflows and undesired saturations. The use of appropriate scaling tools can reduce the development time even further and generates reliable ECU code.

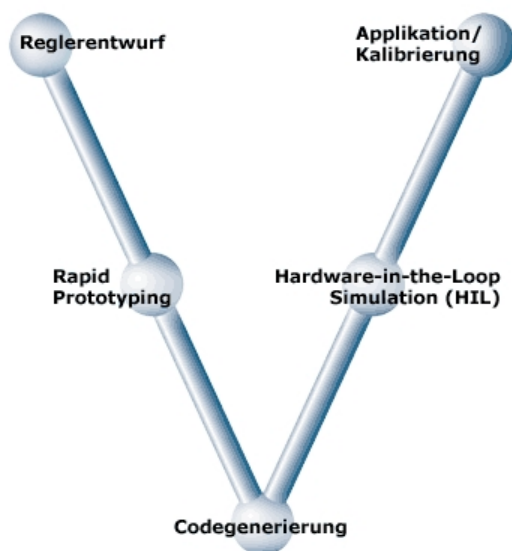
This paper presents such a scaling tool and its integration into the TargetLink<sup>1</sup> environment. A state-space position controller of an electro-mechanical positioning-system is used to demonstrate how an optimal controller prototype is systematically prepared and converted into production code for an ECU. The implementation of the controller in fixed-point arithmetics yields the same tracking performance and disturbance rejection as the floating-point prototype.

---

<sup>1</sup> Fa. dSPACE GmbH, Paderborn

## 1 Einleitung

Die automatische Generierung von Produktionscode für Seriensteuergeräte gewinnt vor allem in der Automobilindustrie zunehmend an Bedeutung. Durch den steigenden Wettbewerbsdruck wird eine Reduktion der Entwicklungszeiten für Steuergeräte und entsprechender Software bis zur Serienreife zwingend notwendig. Ein etablierter Prozess zur zeit- und kosteneffizienten Steuergeräteentwicklung wird durch das V-Modell in Bild 1 beschrieben [1].



**Bild 1:** *Entwicklungsprozess nach dem V-Modell*

Es zeigt die Entwicklungsstufen, die mit Hilfe von Softwarewerkzeugen automatisiert durchlaufen werden. Voraussetzung für die Vorgehensweise nach dem V-Modell ist eine ausführbare Spezifikation der zu implementierenden Steuerung und Regelung in Form eines Simulationsmodells. Damit werden Lösungsansätze bereits in der Offline-Simulation und anschließend mit der Methodik des Rapid Control Prototyping (RCP) auf leistungsfähiger Echtzeithardware überprüft [2]. Auf der Basis dieses Simulationsmodells erfolgt die automatische Generierung des Steuergerätescodes. Nach der Implementierung der Software steht das Seriensteuergerät dann für die Einbindung in eine Hardware-in-the-Loop-Umgebung, für weitere Tests am realen System und anschließend für die Kalibrierung von Parametern zur Verfügung.

Die beim RCP von Steuerungs- und Regelungsalgorithmen verwendeten Prozessoren arbeiten mit Gleitkomma-Arithmetik. Dabei werden die Aussteuerbereiche der Variablen im Normalfall nicht verlassen. Für die Implementierung der Algorithmen auf einem

Seriensteuergerät mit Festkomma-Arithmetik muss das Simulationsmodell mittels geeigneter Skalierung so aufbereitet werden, dass alle Variablen auf dem begrenzten Aussteuerbereich der Festkommahardware dargestellt werden können. Dabei soll die Quantisierung klein gehalten und sollen Wertebereichsüberläufe vermieden werden. Ohne Werkzeugunterstützung ist die Skalierung jeder einzelnen Variablen eine mühselige Arbeit. Besonders die Skalierung von Variablen dynamischer Systemanteile ist sehr zeitaufwändig. Die Aussteuerbereiche der Zustands- und Ausgangsgrößen sind in der Regel nicht bekannt, so dass diese nur mit Hilfe von Simulationen abgeschätzt werden können. Wertebereichsüberläufe lassen sich mit dieser Methode allerdings nicht ganz ausschließen und können deshalb nur durch zusätzliche Sättigung der Variablen vermieden werden.

Die Lösung für eine schnelle und zuverlässige Skalierung der Variablen dynamischer Systemanteile ist die Verwendung eines Autoskalierungswerkzeuges. Dieses berechnet die maximal möglichen physikalischen Aussteuerbereiche aller Variablen und führt die optimale Skalierung automatisch durch.

In diesem Beitrag wird ein im Labor für Mechatronik der Fachhochschule Köln (Cologne Laboratory of Mechatronics, CLM) entwickeltes Werkzeug vorgestellt, mit dem die Skalierung der Zustands- und Ausgangsvariablen diskreter Zustandsmodelle und Übertragungsfunktionen auf Basis der maximal möglichen physikalischen Aussteuerbereiche automatisch durchgeführt werden kann. Mit der Integration des Werkzeuges in die Umgebung des Codegenerators TargetLink wird dem Anwender die mühselige Skalierungsarbeit abgenommen und so der Entwicklungsprozess erheblich beschleunigt. Zudem können Wertebereichsüberläufe entsprechender Variablen auf der Festkommahardware ausgeschlossen werden und kann deshalb zusätzlicher Sättigungscode entfallen.

Die Funktionalität des Skalierungswerkzeuges wird am Beispiel einer Zustandslageregelung für ein elektromechanisches Positioniersystem nachgewiesen. In dieser Regelung treten erschwerend absolute Lagesignale mit großen Aussteuerbereichen und hohen Genauigkeitsanforderungen auf – Eigenschaften, die mit Festkomma-Arithmetik bei gegebener Wortbreite nicht gleichzeitig darstellbar sind. Dieses Problem wird durch eine spezielle Aufbereitung der Zustandsgleichungen des Reglers gelöst. Durch die Transformation der Gleichungen auf die so genannte Relativform enthalten die neuen Variablen nur noch relative Lageinformation mit entsprechend kleinen Aussteuerbereichen und hoher Auflösung. Anschließend erfolgen die automatische Skalierung der Zustandsgleichungen und die Generierung des Steuergerätescodes mit TargetLink. Nach der

Implementierung des Reglers auf einem Festkommaprozessor zeigt der Vergleich der Ergebnisse mit Gleit- und Festkomma-Arithmetik eine sehr gute Übereinstimmung.

## 2 Skalierung der Variablen diskreter Zustandsmodelle

Gegeben sei das diskrete Zustandsmodell

$$\begin{aligned}\underline{x}(k+1) &= \underline{A} \cdot \underline{x}(k) + \underline{B} \cdot \underline{u}(k) \\ \underline{y}(k) &= \underline{C} \cdot \underline{x}(k) + \underline{D} \cdot \underline{u}(k) \ ,\end{aligned}\tag{1}$$

dessen m Eingangs-, n Zustands- und r Ausgangsvariablen in den Vektoren  $\underline{u}$ ,  $\underline{x}$  und  $\underline{y}$  für die rekursive Lösung der Gleichungen mit Festkomma-Arithmetik skaliert werden sollen.

Die Darstellung einer Variablen v, die den Wert einer physikalischen Größe repräsentiert, ist im einfachsten Fall mit Hilfe ihres skalierten Festkomma-Äquivalents v' durch die Beziehung

$$v = S_v \cdot v' \tag{2}$$

gegeben. Dabei stellt der Skalierungsfaktor  $S_v$  die Quantisierungsstufe der Variablen dar. Entsprechend Gleichung (2) gilt für die Variablen des Zustandsmodells

$$\begin{aligned}\underline{u} &= \underline{S}_u \cdot \underline{u}' \ , \quad \underline{S}_u = \text{diag}(S_{u,p}) \ , \quad p = 1, \dots, m \ , \\ \underline{x} &= \underline{S}_x \cdot \underline{x}' \ , \quad \underline{S}_x = \text{diag}(S_{x,i}) \ , \quad i = 1, \dots, n \ , \\ \underline{y} &= \underline{S}_y \cdot \underline{y}' \ , \quad \underline{S}_y = \text{diag}(S_{y,q}) \ , \quad q = 1, \dots, r \ .\end{aligned}\tag{3}$$

Nach Einsetzen dieser Gleichungen in die Gleichung (1) folgt daraus mit

$$\underline{A}' = \underline{S}_x^{-1} \underline{A} \cdot \underline{S}_x \ , \quad \underline{B}' = \underline{S}_x^{-1} \underline{B} \cdot \underline{S}_u \ , \quad \underline{C}' = \underline{S}_y^{-1} \underline{C} \cdot \underline{S}_x \ , \quad \underline{D}' = \underline{S}_y^{-1} \underline{D} \cdot \underline{S}_u \tag{4}$$

das skalierte Zustandsmodell

$$\begin{aligned}\underline{x}'(k+1) &= \underline{A}' \underline{x}'(k) + \underline{B}' \underline{u}'(k) \\ \underline{y}'(k) &= \underline{C}' \underline{x}'(k) + \underline{D}' \underline{u}'(k) \ .\end{aligned}\tag{5}$$

Die Skalierungsfaktoren in den Matrizen  $\underline{S}_u$ ,  $\underline{S}_x$  und  $\underline{S}_y$  lassen sich mit Hilfe der Wortbreite b der Festkommahardware und der physikalischen Aussteuerbereiche  $[u_{p,\min} , u_{p,\max}]$  der Eingangsvariablen,  $[x_{i,\min} , x_{i,\max}]$  der Zustandsvariablen und  $[y_{q,\min} , y_{q,\max}]$  der Ausgangsvariablen angeben. So erhält man beispielsweise bei Verwendung des Datentyps Signed-Integer für die Festkomma-Variablen die Skalierungsfaktoren

$$\begin{aligned}
S_{u,p} &= \frac{\max(|u_{p,\min}|, |u_{p,\max}|)}{2^{b-1} - 1}, \\
S_{x,i} &= \frac{\max(|x_{i,\min}|, |x_{i,\max}|)}{2^{b-1} - 1}, \\
S_{y,q} &= \frac{\max(|y_{q,\min}|, |y_{q,\max}|)}{2^{b-1} - 1}.
\end{aligned} \tag{6}$$

Sind die physikalischen Aussteuerbereiche aller Variablen bekannt, bereitet die Berechnung der Skalierungsfaktoren  $S_{u,p}$ ,  $S_{x,i}$  und  $S_{y,q}$  nach Gleichung (6) keine Schwierigkeiten. Bei dynamischen Systemen sind in der Regel die Aussteuerbereiche der Eingangsvariablen bekannt, nicht aber die der Zustands- und der Ausgangsvariablen. Deshalb müssen zuvor die maximal möglichen Aussteuerbereiche der Zustands- und Ausgangsvariablen berechnet werden. Die Vorgehensweise zur Ermittlung der maximalen Aussteuerbereiche wird im folgenden Kapitel vorgestellt.

### 3 Berechnung der maximalen Aussteuerbereiche von Zustands- und Ausgangsgrößen

Die Zeitantworten diskreter Zustandsmodelle können durch die rekursive Auswertung der Zustandsdifferenzen- und der Ausgangsgleichung aus Gleichung (1) berechnet werden. Für den Zeitschritt  $k$  erhält man die Lösungsformeln

$$\begin{aligned}
\underline{x}(k) &= \underline{A}^k \underline{x}(0) + \sum_{j=0}^{k-1} \underline{A}^j \underline{B} \underline{u}(k-1-j) \\
\underline{y}(k) &= \underline{C} \underline{A}^k \underline{x}(0) + \sum_{j=0}^{k-1} \underline{C} \underline{A}^j \underline{B} \underline{u}(k-1-j) + \underline{D} \cdot \underline{u}(k)
\end{aligned} \tag{7}$$

zur Berechnung der Werte der Zustandszahlenfolgen  $\{\underline{x}(k)\}$  und der Ausgangszahlenfolgen  $\{\underline{y}(k)\}$ . Darin stellen die Terme  $\underline{A}^k \underline{x}(0)$  und  $\underline{C} \underline{A}^k \underline{x}(0)$  die homogenen Lösungen und die Faltungssummen die partikulären Lösungen der Zustandsdifferenzen- und der Ausgangsgleichung dar.

Für stabile Systeme klingen die homogenen Anteile der Zustands- und der Ausgangszahlenfolgen nach Gleichung (7) mit steigendem Zeitschritt  $k$  ab. Zusätzlich streben die Werte  $\underline{A}^j \underline{B}$  der Impulsantwortzahlenfolgen der Zustände und die Werte  $\underline{C} \underline{A}^j \underline{B}$  der Impulsantwortzahlenfolgen der Ausgänge in den Faltungssummen mit steigendem  $j$  gegen Null. Für beschränkte Eingangszahlenfolgen  $\underline{u} \in [\underline{u}_{\min}, \underline{u}_{\max}]$  sind daher auch die

Zustandszahlenfolgen mit  $\underline{x} \in [\underline{x}_{\min}, \underline{x}_{\max}]$  und die Ausgangszahlenfolgen mit  $\underline{y} \in [\underline{y}_{\min}, \underline{y}_{\max}]$  beschränkt. Ein solches System wird als BIBO<sup>2</sup>-stabil bezeichnet.

Die gesuchten Aussteuerbereiche zur Berechnung der Skalierungsfaktoren mit Hilfe von Gleichung (6) sind durch die minimal und maximal möglichen Werte der Zustands- und Ausgangszahlenfolgen gegeben. Im Folgenden werden diese Werte für gegebene beschränkte Eingangssignale berechnet. Dazu sei der Fall betrachtet, dass ein Signal auf den p-ten Eingang aufgeschaltet wird und alle übrigen Eingangssignale den Wert Null annehmen.

Für den i-ten Zustand des Systems erhält man dafür nach Gleichung (7)

$$x_{i,p}(k) = \text{row}_i(\underline{A}^k) \underline{x}(0) + \sum_{j=0}^{k-1} g_{x,ip}(j) u_p(k-1-j) . \quad (8)$$

Dabei wird zur Berechnung der Werte

$$g_{x,ip}(j) = \text{row}_i(\underline{A}^j) \cdot \text{col}_p(\underline{B}) \quad (9)$$

der Impulsantwortzahlenfolge  $\{g_{x,ip}(j)\}$  des Zustandes bei Anregung am entsprechenden Eingang die i-te Zeile von  $\underline{A}^j$  und die p-te Spalte von  $\underline{B}$  herangezogen.

Für die Berechnung der minimalen bzw. maximalen Werte der Zustandszahlenfolge  $\{x_{i,p}(k)\}$  wird nun die Gleichung (8) für jeden Zeitschritt  $k = 1, 2, 3, \dots$  ausgewertet, wodurch man für die ersten Schritte

$$\begin{aligned} x_{i,p}(1) &= \text{row}_i(\underline{A}^1) \underline{x}(0) + g_{x,ip}(0) u_p(0) \\ x_{i,p}(2) &= \text{row}_i(\underline{A}^2) \underline{x}(0) + g_{x,ip}(0) u_p(1) + g_{x,ip}(1) u_p(0) \\ x_{i,p}(3) &= \text{row}_i(\underline{A}^3) \underline{x}(0) + g_{x,ip}(0) u_p(2) + g_{x,ip}(1) u_p(1) + g_{x,ip}(2) u_p(0) \\ &\dots \end{aligned}$$

erhält.

In der Faltungssumme wird dabei für  $j = k-1$  jeweils ein neuer Wert  $u_p(0)$  der Eingangszahlenfolge  $\{u_p(k)\}$  so bestimmt, dass der Wert  $x_{i,p}(k)$  der Zustandszahlenfolge im betrachteten Zeitschritt mit der Faltungssumme ein Minimum bzw. Maximum annimmt. Die Wahl erfolgt in Abhängigkeit des Vorzeichens des Wertes  $g_{x,ip}(k-1)$  der Impulsantwortzahlenfolge gemäß Tabelle 1. Durch das Hinzufügen eines neuen Wertes  $u_p(0)$

---

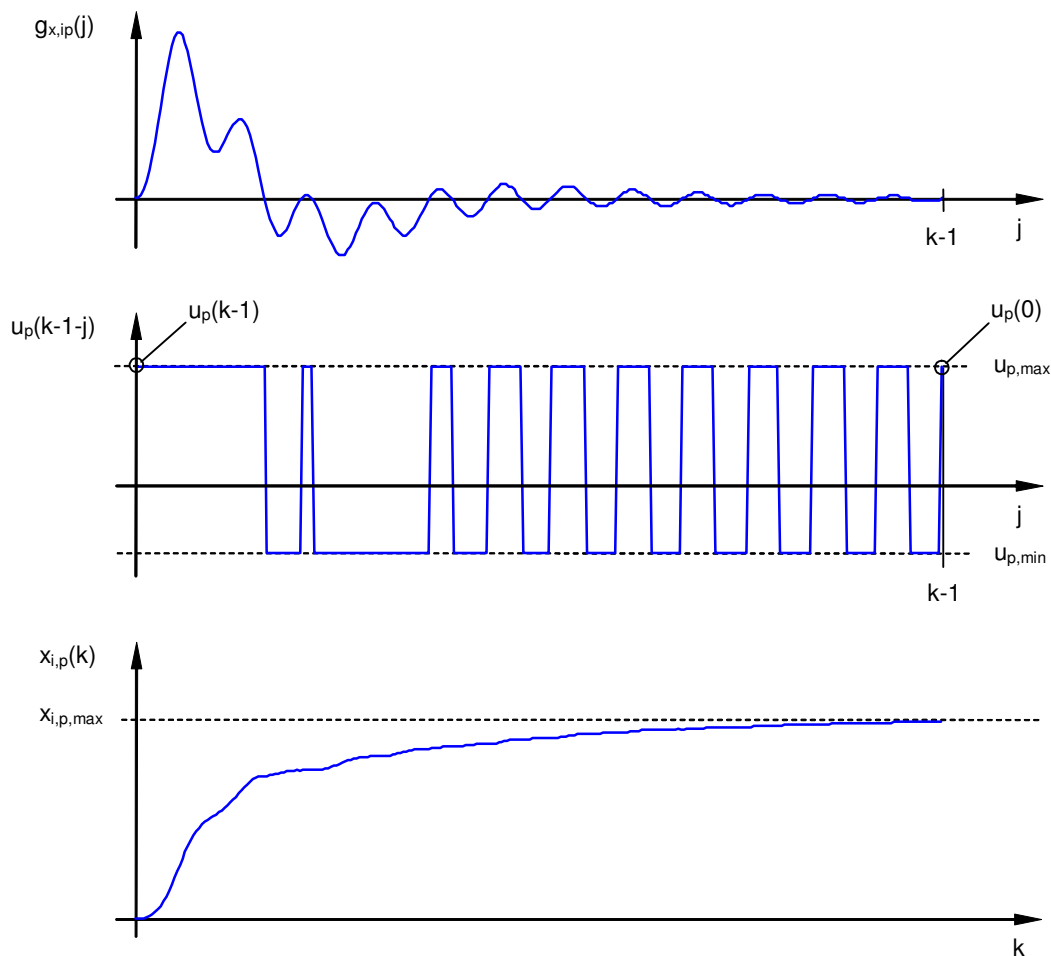
<sup>2</sup> Bounded Input Bounded Output

verschieben sich die bereits berechneten Werte der Eingangszahlenfolge jeweils um einen Zeitschritt.

**Tabelle 1:** Wahl eines neuen Wertes  $u_p(0)$  der Eingangszahlenfolge im Zeitschritt  $k$  zur Minimierung bzw. Maximierung des Wertes  $x_{i,p}(k)$

Vorzeichen von $g_{x_{i,p}}(k-1)$	Minimierung von $x_{i,p}(k)$	Maximierung von $x_{i,p}(k)$
positiv	$u_p(0) = u_{p,\min}$	$u_p(0) = u_{p,\max}$
negativ	$u_p(0) = u_{p,\max}$	$u_p(0) = u_{p,\min}$

Mit der beschriebenen Vorgehensweise ergeben sich für die  $p = 1, \dots, m$  Eingänge und  $i = 1, \dots, n$  Zustände des Systems insgesamt  $2 \cdot m \cdot n$  worst-case Eingangszahlenfolgen, die jeweils alle Werte  $x_{i,p}(k)$  der zugehörigen Zustandszahlenfolgen in ihr Minimum bzw. Maximum treiben und deren Werte sich an den Grenzen  $u_{p,\min}$  und  $u_{p,\max}$  befinden.



**Bild 2:** Impulsantwortzahlenfolge  $\{g_{x_{i,p}}(j)\}$  (oben) und Eingangszahlenfolge  $\{u_p(k-1-j)\}$  (mitte) zur Maximierung der Werte der Zustandszahlenfolge  $\{x_{i,p}(k)\}$  (unten)

Bild 2 veranschaulicht diesen Sachverhalt für die Maximierung der Werte der i-ten Zustandszahlenfolge über die p-te Eingangszahlenfolge mit der Anfangsbedingung  $\underline{x}(0) = \underline{0}$  für den Zustandsvektor.

Analog zur Gleichung (8) erfolgt die Berechnung der q-ten Ausgangszahlenfolge des Systems mit der Gleichung

$$y_{q,p}(k) = \text{row}_q(\underline{CA}^k)\underline{x}(0) + \sum_{j=0}^{k-1} g_{y,qp}(j) u_p(k-1-j) + d_{qp}u_p(k) . \quad (10)$$

Darin ist

$$g_{y,qp}(j) = \text{row}_q(\underline{CA}^j) \cdot \text{col}_p(\underline{B}) \quad (11)$$

der j-te Wert der Impulsantwortzahlenfolge  $\{g_{y,qp}(j)\}$  des Ausgangs bei Anregung am Eingang p. Der Koeffizient  $d_{qp}$  ist das Element in der q-ten Zeile und p-ten Spalte der Durchgriffsmatrix  $\underline{D}$  aus Gleichung (7).

Mit Gleichung (10) erfolgt, ähnlich wie oben für die Zustandszahlenfolgen, für  $k = 0, 1, 2, 3, \dots$  die Wahl jeweils eines neuen Wertes  $u_p(0)$  der Eingangszahlenfolge  $\{u_p(k)\}$ , mit dem im Zeitschritt k der Wert  $y_{q,p}(k)$  der Ausgangszahlenfolge minimal bzw. maximal wird. Dieser Wert  $u_p(0)$  wird im Zeitschritt  $k = 0$  abhängig vom Vorzeichen des Elementes  $d_{qp}$  im Durchgriff bestimmt. Für die Zeitschritte  $k = 1, 2, 3, \dots$  wird der neue Wert  $u_p(0)$  mit Hilfe des Vorzeichens des Wertes  $g_{y,qp}(k-1)$  der Impulsantwortzahlenfolge gewählt. Durch das Hinzufügen eines neuen Wertes  $u_p(0)$  verschieben sich die bereits berechneten Werte der Eingangszahlenfolge, wie bereits oben beschrieben, jeweils um einen Zeitschritt. Die Vorgehensweise zur Minimierung bzw. Maximierung der Werte der Ausgangszahlenfolge ist in Tabelle 2 zusammengefasst. Sie wird, wie oben für Gleichung (8), durch die Auswertung von Gleichung (10) für die Zeitschritte  $k = 0, 1, 2, 3, \dots$

$$y_{q,p}(0) = \text{row}_q(\underline{CA}^0)\underline{x}(0) + d_{qp}u_p(0)$$

$$y_{q,p}(1) = \text{row}_q(\underline{CA}^1)\underline{x}(0) + d_{qp}u_p(1) + g_{y,qp}(0) u_p(0)$$

$$y_{q,p}(2) = \text{row}_q(\underline{CA}^2)\underline{x}(0) + d_{qp}u_p(2) + g_{y,qp}(0) u_p(1) + g_{y,qp}(1) u_p(0)$$

$$y_{q,p}(3) = \text{row}_q(\underline{CA}^3)\underline{x}(0) + d_{qp}u_p(3) + g_{y,qp}(0) u_p(2) + g_{y,qp}(1) u_p(1) + g_{y,qp}(2) u_p(0)$$

...

deutlich.

**Tabelle 2:** Wahl eines neuen Wertes  $u_p(0)$  der Eingangszahlenfolge im Zeitschritt  $k$  zur Minimierung bzw. Maximierung des Wertes  $y_{q,p}(k)$

Vorzeichen von $d_{q,p}$ im Zeitschritt $k = 0$	Minimierung von $y_{q,p}(k)$	Maximierung von $y_{q,p}(k)$
positiv	$u_p(0) = u_{p,\min}$	$u_p(0) = u_{p,\max}$
negativ	$u_p(0) = u_{p,\max}$	$u_p(0) = u_{p,\min}$
Vorzeichen von $g_{y,q,p}(k-1)$ in den Zeitschritten $k = 1, 2, 3, \dots$	Minimierung von $y_{q,p}(k)$	Maximierung von $y_{q,p}(k)$
positiv	$u_p(0) = u_{p,\min}$	$u_p(0) = u_{p,\max}$
negativ	$u_p(0) = u_{p,\max}$	$u_p(0) = u_{p,\min}$

Für die  $p = 1, \dots, m$  Eingänge und  $q = 1, \dots, r$  Ausgänge des Systems resultieren  $2 \cdot m \cdot r$  worst-case Eingangszahlenfolgen, die jeweils alle Werte  $y_{q,p}(k)$  der Ausgangszahlenfolgen in ihr Minimum bzw. Maximum treiben und deren Werte sich an den Grenzen  $u_{p,\min}$  und  $u_{p,\max}$  befinden.

Wird der  $i$ -te Zustand gleichzeitig von allen zugehörigen worst-case Eingangszahlenfolgen für die Minimierung von  $x_{i,p}(k)$  angeregt, erhält man mit Hilfe des Überlagerungsprinzips als absolutes Minimum der resultierenden Zustandszahlenfolge

$$x_{i,\min} = \min \left\{ \sum_{p=1}^m x_{i,p}(k) \right\} ; k = 0, 1, 2, \dots \quad (12)$$

Entsprechend gilt bei Anregung des Systems mit den Eingangszahlenfolgen für die Maximierung von  $x_{i,p}(k)$  für das absolute Maximum des  $i$ -ten Zustandes

$$x_{i,\max} = \max \left\{ \sum_{p=1}^m x_{i,p}(k) \right\} ; k = 0, 1, 2, \dots \quad (13)$$

Für den  $q$ -ten Ausgang erhält man in gleicher Weise mit den zugehörigen Eingangszahlenfolgen

$$y_{q,\min} = \min \left\{ \sum_{p=1}^m y_{q,p}(k) \right\} ; k = 0, 1, 2, \dots \quad (14)$$

und

$$y_{q,\max} = \max \left\{ \sum_{p=1}^m y_{q,p}(k) \right\} ; k = 0, 1, 2, \dots \quad (15)$$

Die Ermittlung der maximalen Aussteuerbereiche der Zustands- und Ausgangszahlenfolgen gemäß Gleichungen (8) bis (15) wurde mit Hilfe von MATLAB M-Funktionen realisiert. Dabei musste für die numerische Auswertung der Gleichungen (8) und (10) ein Abbruchkriterium eingeführt werden. Dieses führt zu einem Abbruch der Auswertung zu einem Zeitschritt  $k_{\max}$ , wenn die Änderung der Absolutwerte aller Zustände und Ausgänge gleichzeitig kleiner als eine bestimmte Konvergenztoleranz ist [3]. Die dadurch bedingten Abbruchfehler für die Aussteuerbereiche werden durch geeignete Restgliedabschätzungen korrigiert.

Die so berechneten Aussteuerbereiche gehen schließlich in Gl. (6) aus Kapitel 2 zur Berechnung der Skalierungsfaktoren  $S_{x,i}$  und  $S_{y,q}$  ein.

Eine detaillierte Beschreibung zur Ermittlung der maximalen Aussteuerbereiche der Variablen diskreter Zustandsmodelle, Übertragungsfunktionen und Filter ist in [3] nachzulesen.

#### **4 State-Space Scaling Tool**

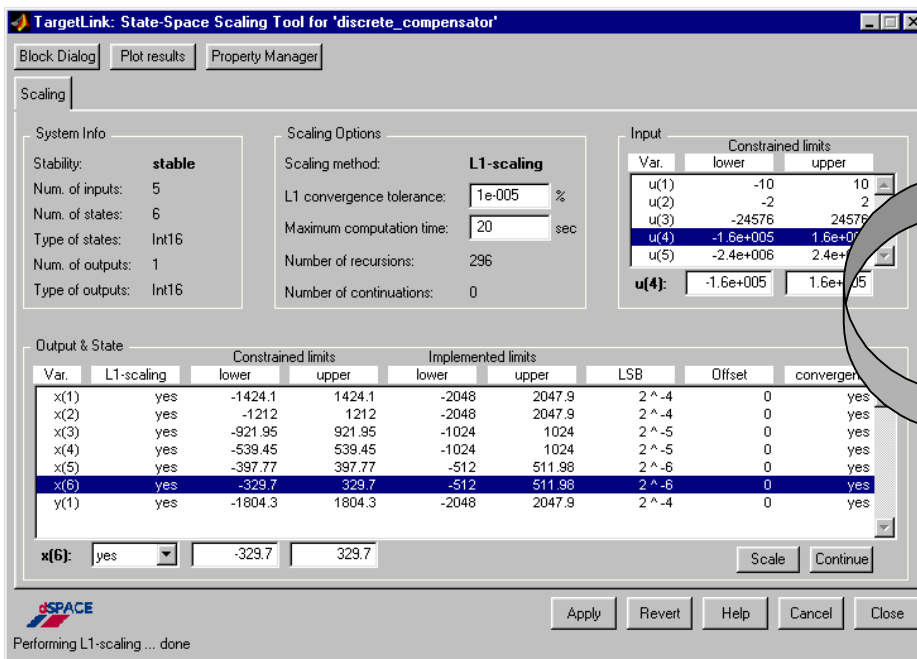
Die Skalierung der Variablen diskreter Zustandsmodelle nach Kapitel 2 und die zugehörige Berechnung der maximalen Aussteuerbereiche nach Kapitel 3 wurden in die Entwicklungsumgebung TargetLink als State-Space Scaling Tool integriert. Die Implementierung erfolgte in Form so genannter API<sup>3</sup>-Tools in MATLAB M-Sprache. Diese API-Tools bieten Funktionsschnittstellen für die Verwendung durch weitere Werkzeuge, wie z.B. Funktionen und grafische Bedienoberflächen blockübergreifender Gesamtwerkzeuge.

Bild 3 zeigt die Dialogbox des State-Space Scaling Tools für das nachfolgende Anwendungsbeispiel. Sie wird nach Doppelklick auf den betrachteten TargetLink-Block in einem Untermenü geöffnet und ermöglicht die interaktive Steuerung des Skalierungsprozesses. Nach Vorgabe der bekannten Aussteuerbereiche der Eingangsgrößen ermittelt das Werkzeug automatisch die worst-case Aussteuerbereiche der Zustands- und Ausgangsgrößen. Mit den Grenzen der Aussteuerbereiche werden die für die gewählten Datentypen erforderlichen Skalierungsfaktoren berechnet und die Blockgleichungen automatisch skaliert.

Zur Analyse der Ergebnisse können die ermittelten worst-case Eingangszahlenfolgen auf das Zustandsmodell aufgeschaltet werden. Die resultierenden Zeitantworten der Zustands- und Ausgangsgrößen werden nach der Simulation in einem Plot-Fenster visualisiert.

---

<sup>3</sup> Application Programming Interface



Vorgabe der Aussteuerbereiche der Eingangsgrößen

worst-case Aussteuerbereiche und Skalierungsparameter für die Zustands- und Ausgangsgrößen

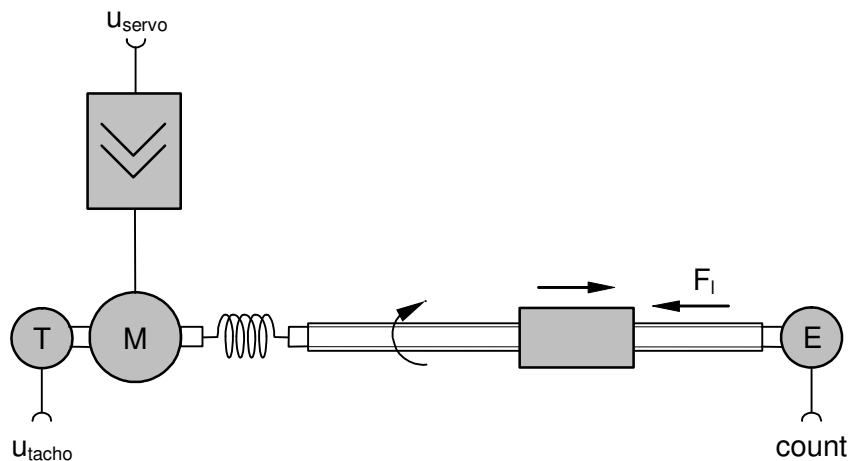
**Bild 3:** State-Space Scaling Tool

Eine detaillierte Beschreibung der Handhabung des State-Space Scaling Tools ist in [3] zu finden.

## 5 Anwendungsbeispiel

Die Leistungsfähigkeit des Skalierungswerkzeuges wird im Folgenden am Beispiel einer LQG-Lageregelung für ein elektromechanisches Positioniersystem (EMPS) nachgewiesen. Ausgangspunkt ist der Gleitkomma-Prototyp der Regelung in Form eines Simulink-Modells, der die Referenz für die Entwicklung des Produktionscodes für ein Festkomma-Steuergerät darstellt. Nach einer geeigneten Aufbereitung des Prototyps erfolgt die automatische Skalierung der Variablen und anschließend die Codegenerierung mit TargetLink. Für die effiziente und erfolgreiche Erzeugung des Produktionscodes sind eine systematische Vorgehensweise und der Werkzeugeinsatz unverzichtbar.

Bild 4 zeigt das EMPS. Es besteht aus einem stromgeregelten Gleichstrommotor auf der Antriebsseite und einer Lineareinheit mit Kugelgewindetrieb und Schlitten auf der Abtriebsseite. Der Kugelgewindetrieb ist spielfrei, er bringt aber als Nachteil eine erhebliche Reibung in das System ein. Durch die elastische Kopplung von An- und Abtriebsseite kommt erschwerend eine schwach gedämpfte elastische Schwingung mit einer Frequenz von etwa 80 Hz hinzu. Diese Frequenz liegt im Bereich der gewünschten Regelungsbandbreite.

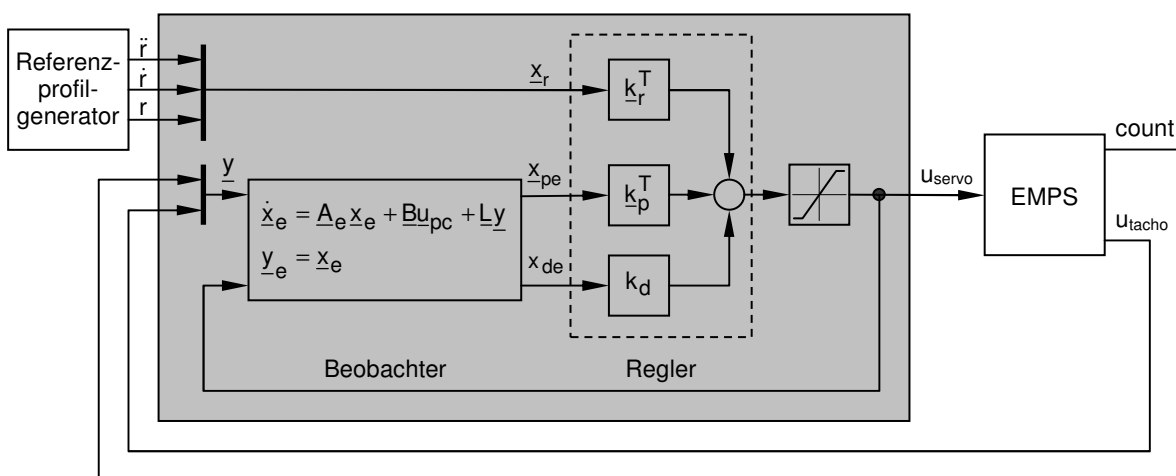


**Bild 4:** Elektromechanisches Positioniersystem

Stellgröße in das System ist der Motorstromsollwert  $u_{servo}$ , als Messgrößen stehen die Tachospaltung  $u_{tacho}$  für die antriebsseitige Winkelgeschwindigkeit und der Zählerstand  $count$  eines inkrementellen Lagegebers für den lastseitigen Winkel zur Verfügung. Die genaue Positionierung des Schlittens wird durch eine externe Störkraft  $F_1$  erschwert.

Das beschriebene EMPS stellt eine Musteranwendung im CLM dar, die von grundlegender Bedeutung für unterschiedlichste Anwendungsbereiche ist. Viele mit diesem System erzielte Ergebnisse konnten bereits auf entsprechende Systeme aus der Fahrzeugtechnik, dem Flugzeugbau und der Antriebstechnik übertragen werden.

Der geschlossene Regelkreis mit LQG-Kompensator ist in Bild 5 dargestellt.

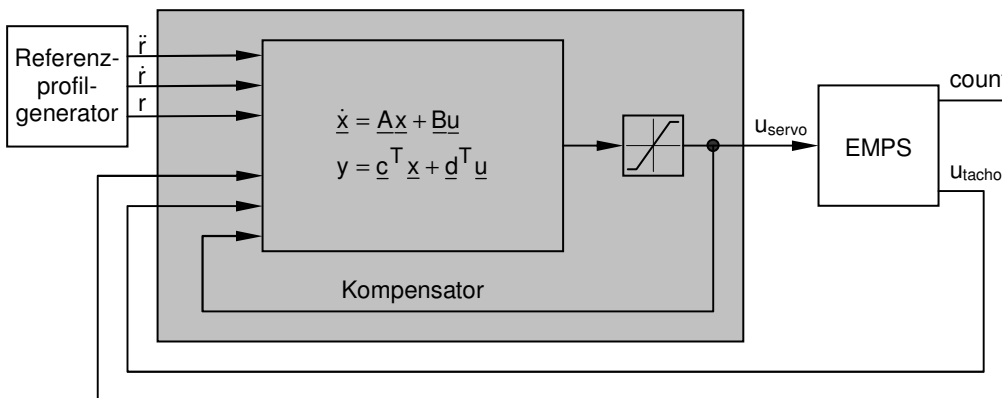


**Bild 5:** Regelkreis mit Kompensator

Der dynamische Kompensator entsteht durch die Verkopplung eines statischen Zustandsreglers mit den Reglerverstärkungen  $k_r^T$ ,  $k_p^T$  und  $k_d$  und eines linearen dynamischen Beobachters. Die Führungsgrößen werden von einem Referenzprofilgenerator bereitgestellt und über die Verstärkungsmatrix  $k_r^T$  im Regler auf die Stellgröße geschaltet. Ferner enthält der Regler die Rückführung des Zustandsvektors der Regelstrecke über die Verstärkungsmatrix  $k_p^T$  und die Aufschaltung lastseitiger Störmomente (infolge Reibung und externer Störkraft  $F_l$  am Schlitten) über die Verstärkung  $k_d$ . Die nicht messbaren Zustands- und Störgrößen werden vom Beobachter aus den Messgrößen und der Stellgröße rekonstruiert. Der Entwurf der Kompensatorsubsysteme Zustandsregler und Beobachter ist in [4] ausführlich beschrieben.

Ausgangspunkt für die Implementierung der Regelung auf einem Festkomma-Steuergerät ist der kontinuierliche Kompensator aus Bild 5. Die erforderlichen Schritte zur Erzeugung des Festkommacodes sind eine spezielle Aufbereitung der Kompensatorgleichungen, ihre Diskretisierung sowie die automatische Skalierung und Codegenerierung.

Die Aufbereitung beginnt mit der Zusammenfassung der Kompensatorsubsysteme Regler und Beobachter zu einem Zustandsmodell. Das Ergebnis ist in Bild 6 dargestellt.



**Bild 6:** Regelkreis mit zusammengefasstem Kompensator

Diese Maßnahme verringert die Anzahl der in Echtzeit durchzuführenden Rechen- und Speicheroperationen und ist für die weitere Aufbereitung des Kompensators für Festkomma-Arithmetik günstig.

Für den folgenden Schritt sind die Anforderungen an die betrachtete Lageregelung verantwortlich. Die Regelgröße enthält absolute Lageinformation mit einem großen Aussteuerbereich. Gleichzeitig wird eine hohe Genauigkeit der Lage verlangt. Bei einem

Aussteuerbereich von  $\pm 0.15$  m und einer Auflösung der Lagemessung von  $1.25 \mu\text{m}$  beträgt der Unterschied mehr als 5 Größenordnungen. Das gleiche gilt auch für andere Größen in den Kompensatorgleichungen, die absolute Lageinformation enthalten. Die Realisierung der Aussteuerbereiche und die Anforderungen an die Auflösung der Lagegrößen sind mit Festkomma-Arithmetik und einer Wortlänge von 16 Bit gleichzeitig nicht darstellbar.

Zur Erfüllung der Anforderung an die Genauigkeit der Regelung dient die folgende Transformation der Kompensatorgleichungen auf die so genannte Relativform [5]. Ausgangspunkt dafür ist die allgemeine Zustandsdarstellung eines kontinuierlichen Lagereglers

$$\begin{aligned}\dot{\underline{x}} &= \underline{A} \underline{x} + \underline{b}_m m + \underline{b}_r r + \underline{b}_r \dot{r} + \underline{b}_r \ddot{r} + \sum_i \underline{b}_i u_{m,i} \\ y &= \underline{c}^T \underline{x} + d_m m + d_r r + d_r \dot{r} + d_r \ddot{r} + \sum_i d_i u_{m,i}\end{aligned}\tag{16}$$

mit der Solllage  $r$  zur Regelung der gemessenen Istlage  $m$ . Für den betrachteten Kompensator aus Bild 6 ist  $m$  der Zählerstand count. Die Soll- und Istlage weisen jeweils große Aussteuerbereiche auf und haben eine hohe Auflösung. Weiterhin werden dem Lageregler die Sollgeschwindigkeit  $\dot{r}$ , die Sollbeschleunigung  $\ddot{r}$  sowie weitere gemessene Hilfsgrößen  $u_{m,i}$  am Eingang zugeführt. Für den Kompensator aus Bild 6 sind letztere die Messgröße  $u_{m,1} = u_{\text{tacho}}$  und die Stellgröße  $u_{m,2} = u_{\text{servo}}$ . Die abgeleiteten Größen und Hilfsgrößen nehmen für konstante stationäre Lagewerte den Wert Null an, der gleichzeitig ihren Mittelwert darstellt. Sie stellen keine so hohen Anforderungen an die Auflösung und sind deshalb mit der gegebenen begrenzten Wortlänge gut darstellbar.

Die Transformation überführt den Regler in die Relativform

$$\begin{aligned}\dot{\underline{\xi}} &= \underline{A} \underline{\xi} + \tilde{\underline{b}}_m e_r + (\underline{b}_r - \underline{\beta}) \dot{r} + \underline{b}_r \ddot{r} + \sum_i \underline{b}_i u_{m,i} \\ y &= \underline{c}^T \underline{\xi} + \tilde{d}_m e_r + d_r \dot{r} + d_r \ddot{r} + \sum_i d_i u_{m,i},\end{aligned}\tag{17}$$

für die als Eingang in den Kompensator anstelle der absoluten Soll- und Istlage der Lagefehler  $e_r$  tritt. Der Maximalwert des Lagefehlers ist zusätzlich zur Erhöhung der Rechengenauigkeit mit

$$e_r = f_s (m - r)\tag{18}$$

auf den Zahlenbereich der Zielhardware hochskaliert.

Entsprechend verkleinern sich die zugehörigen Reglerverstärkungen

$$\tilde{\mathbf{b}}_m = \frac{1}{f_s} \mathbf{b}_m \quad (19)$$

und

$$\tilde{\mathbf{d}}_m = \frac{1}{f_s} \mathbf{d}_m \quad (20)$$

An die Stelle des Zustandsvektors  $\underline{x}$  mit absoluter Lageinformation aus Gleichung (16) tritt der relative Zustandsvektor

$$\underline{\xi} = \underline{x} - \underline{x}_s \quad (21)$$

der sich um einen gleitenden Mittelwert

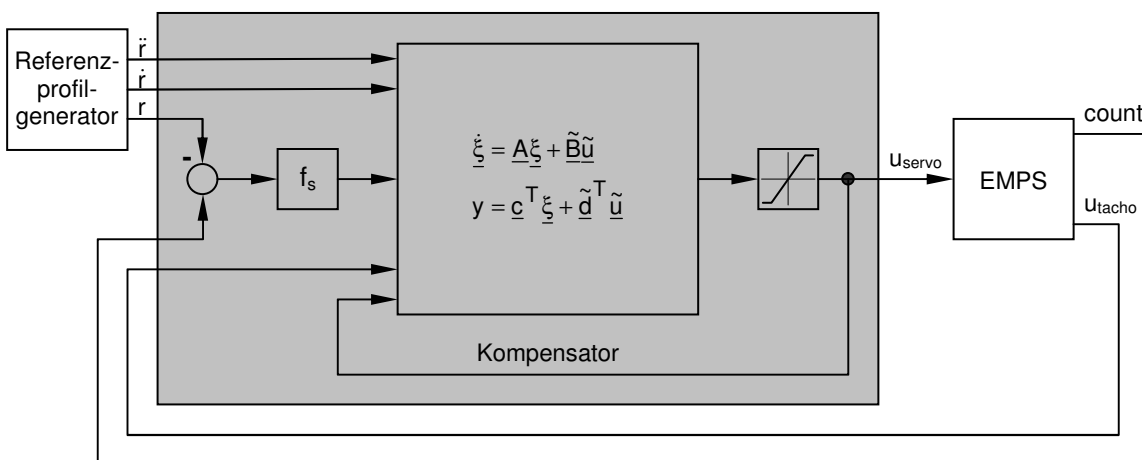
$$\underline{x}_s = \underline{\beta} \cdot r \quad (22)$$

mit

$$\underline{\beta} = (\underline{M}^T \underline{M})^{-1} \underline{M}^T \begin{bmatrix} \underline{b}_m + \underline{b}_r \\ \underline{d}_m + \underline{d}_r \end{bmatrix}, \quad \underline{M} = \begin{bmatrix} -\underline{A} \\ -\underline{c}^T \end{bmatrix} \quad (23)$$

bewegt.

In Gleichung (17) tritt weder in den Eingangs- noch in den Zustandsgrößen absolute Lageinformation auf. Die entsprechenden Größen sind durch relative Größen mit kleinen Aussteuerbereichen ersetzt. Die geforderte Auflösung wird durch die Skalierung dieser Größen auf den Zahlenbereich der verwendeten Festkommahardware sichergestellt.



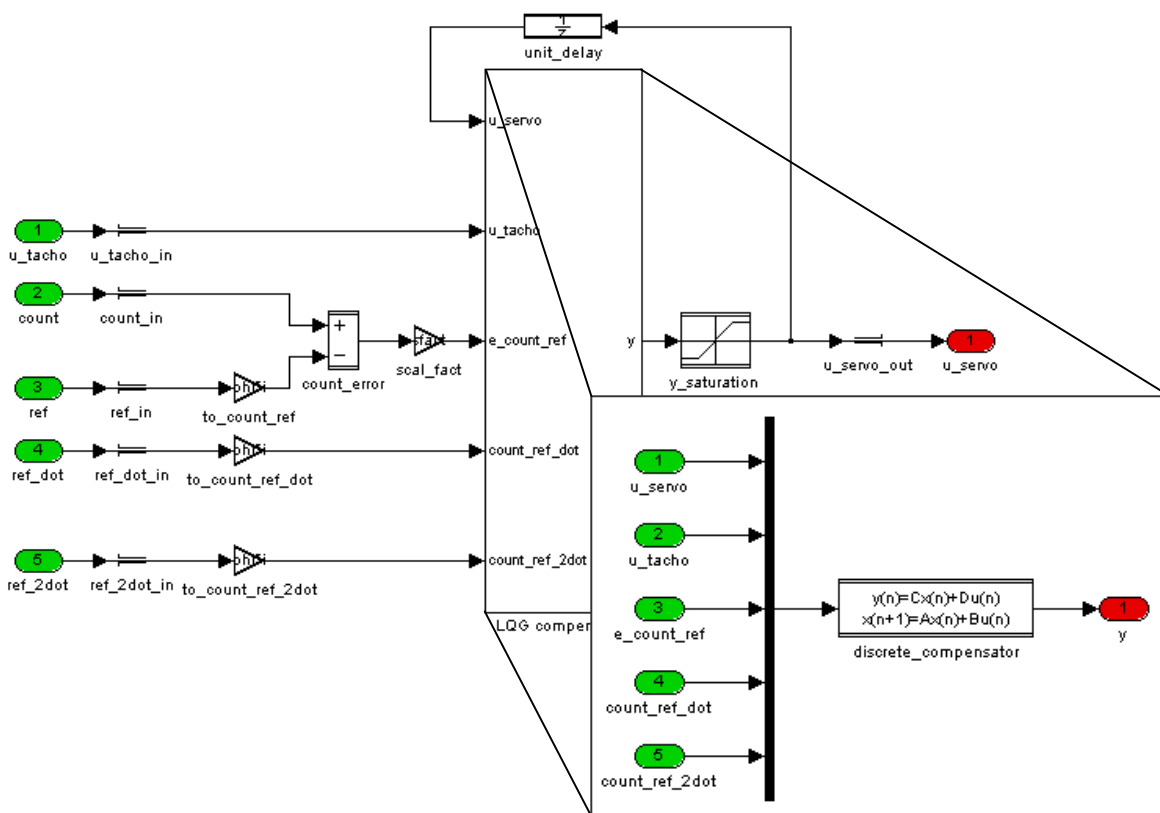
**Bild 7:** Regelkreis mit Kompensator in Relativform

Die Transformation der Kompensatorgleichungen auf Relativform erfolgt in MATLAB mit Hilfe einer Funktion aus der CLM-Funktionsbibliothek.

Bild 7 zeigt das Ergebnis der Transformation für das Blockdiagramm. Der Kompensatorblock enthält nun die Zustandsgleichungen des Kompensators in Relativform mit neuem Eingangs- und Zustandsvektor. Zusätzlich wird der im Eingangsvektor enthaltene hochskalierte Lagefehler bereitgestellt.

Weitere Schritte für eine gute Implementierung des Kompensators mit Festkomma-Arithmetik sind die Transformation der Kompensatorgleichungen auf die reelle Modalform [6] und die Diskretisierung.

Der diskrete Kompensator wird danach in die Entwicklungsumgebung TargetLink [3] für die Festkomma-Codegenerierung importiert. Die Blöcke für den Gleitkomma-Prototyp werden dabei per Knopfdruck durch entsprechende Festkommablöcke ersetzt. Bild 8 zeigt das Resultat dieser automatischen Konvertierung für das Kompensatorsubsystem. Der Zustandsblock enthält die diskreten Zustandsgleichungen des Kompensators in Relativform.



**Bild 8:** TargetLink-Subsystem mit Festkommasspezifikation des Kompensators

Die Auswertung der Kompensatorgleichungen kann vollständig mit 16-Bit-Festkomma-Arithmetik erfolgen. Allein der hochskalierte Lagefehler wird durch Subtraktion zweier 32-Bit-Variablen gebildet und als 16-Bit-Variable gespeichert.

In dieser Festkommaspezifikation des Kompensators sind für die einzelnen Festkommablöcke weitere Angaben über die Datentypen (Standard ist Signed-Integer mit einer Wortlänge von 16 Bit) und Skalierungsfaktoren für die Zustands- und Ausgangsgrößen erforderlich. Für Eingangsgrößen gelten die Angaben zu den Ausgangsgrößen der vorgeschalteten Blöcke. Die entsprechenden Skalierungen der Größen und Koeffizienten werden automatisch durchgeführt.

Für einfache Blöcke wie z.B. Eingangs-, Verstärkungs-, Summations-, Verzögerungs- und Sättigungs-Blöcke erhält man den Skalierungsfaktor im einfachsten Fall aus der Beziehung

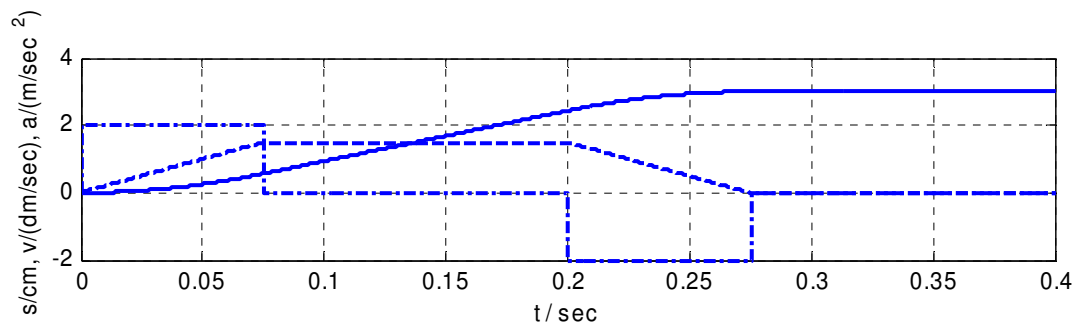
$$y = S_y y'$$

nach Gleichung (2) aus Kapitel 2. Darin ist  $y$  die ursprüngliche physikalische Ausgangsgröße (z.B. eine Spannung mit einem Aussteuerbereich von  $\pm 10$  V) und  $y'$  das zugehörige Integer-Äquivalent (z.B. mit einem Zahlenbereich von  $-32768 \dots 32767$  für den Datentyp Signed-Integer mit einer Wortlänge von 16 Bit). Der Skalierungsfaktor  $S_y$  entspricht dem physikalischen Wert des LSB (Least Significant Bit) des Integer-Äquivalents. Er wird so gewählt, dass der Zahlenbereich des Integer-Äquivalents den Aussteuerbereich der physikalischen Größe vollständig abdeckt. Dabei sind wegen einer effizienteren Rechnung für den Skalierungsfaktor Zweierpotenzen vorzuziehen. Für das oben genannte Zahlenbeispiel würde der Skalierungsfaktor  $S_y = 2^{-11}$  V gewählt, der für die physikalische Größe  $y$  einen Aussteuerbereich von  $-16$  V ...  $15.999512$  V erlaubt.

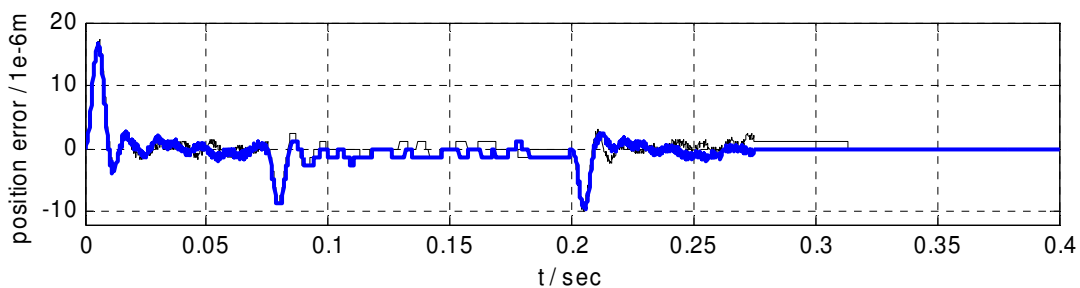
Für dynamische Subsysteme, wie den Zustandsblock für den Kompensator, gestaltet sich die Berechnung der Skalierungsfaktoren nicht so einfach. Alle Zustands- und Ausgangsgrößen sind auf den Zahlenbereich des gewählten Festkomma-Datentyps zu skalieren. Deren Aussteuerbereiche sind aufgrund der Dynamik des Subsystems zunächst nicht bekannt. Hier kommt das in Kapitel 4 vorgestellte Skalierungswerkzeug zum Einsatz. Durch Doppelklick auf den Zustandsblock wird die in Bild 3 dargestellte Dialogbox des Werkzeuges geöffnet. Nach Vorgabe der bekannten Aussteuerbereiche der Eingangsgrößen ermittelt das Werkzeug, wie in Kapitel 4 beschrieben, automatisch die worst-case Aussteuerbereiche der Zustands- und Ausgangsgrößen. Mit den Grenzen der Aussteuerbereiche werden anschließend für die gewählten Datentypen die Skalierungsfaktoren berechnet und die Blockgleichungen skaliert.

Nun erfolgt wie beim Rapid Control Prototyping die Inbetriebnahme des Kompensators durch die automatische Codegenerierung, jetzt allerdings für die Zielhardware mit Festkomma-Arithmetik. Anschließend wird die Regelung zunächst in der Simulation und dann im Experiment getestet. Die Bilder 10 und 11 zeigen gemessene Verläufe des Lagefehlers für die Realisierung der Regelung mit Festkomma- und Gleitkomma-Arithmetik aus einer Processor-in-the-Loop-Simulation [7]. Processor-in-the-Loop heißt, dass der Code auf einem Evaluierungsboard für den Zielprozessor im geschlossenen Kreis mit der Simulation der nichtlinearen Regelstrecke betrieben wird.

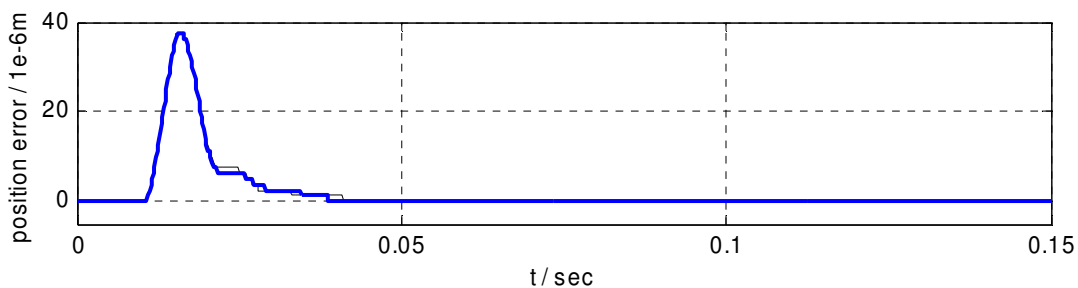
Die Führungsantwort wurde für den parabelförmigen Solllageverlauf und die zugehörigen Geschwindigkeits- und Beschleunigungsverläufe für den Schlitten aus Bild 9 ermittelt. Die Störantwort gilt für eine sprungförmige Störkraft am Schlitten des EMPS.



**Bild 9:** Führungsgrößen



**Bild 10:** Führungsantwort mit Gleitkomma-Arithmetik (dünn) und Festkomma-Arithmetik (fett)



**Bild 11:** Störantwort mit Gleitkomma-Arithmetik (dünn) und Festkomma-Arithmetik (fett)

Die Ergebnisse für den Seriencode mit Festkomma-Arithmetik unterscheiden sich kaum von denen für den Gleitkomma-Prototyp der Regelung. Solche Ergebnisse mit Festkomma-Arithmetik sind nicht selbstverständlich. Sie sind allein auf die systematische Vorgehensweise von der Aufbereitung des Kompensators bis zur Codegenerierung zurückzuführen. Dabei spielt der Einsatz des vorgestellten Skalierungswerkzeuges eine entscheidende Rolle.

## **6 Zusammenfassung und Ausblick**

In diesem Beitrag wurden die theoretischen Grundlagen für die Skalierung diskreter Zustandsmodelle zur Codegenerierung für Festkommaprozessoren und die Implementierung eines Werkzeuges für die automatische Skalierung vorgestellt. Das Werkzeug wurde in eine Entwicklungsumgebung für die Generierung von Produktionscode für Seriensteuergeräte eingebettet. Durch die automatische Skalierung werden Überläufe vermieden, daher ist keine zusätzliche Sättigung der Variablen erforderlich.

Die Entwicklungszeit zur Erzeugung von Steuergerätecode wird durch den Einsatz von Skalierungswerkzeugen drastisch reduziert. Zudem sind die Code-Zuverlässigkeit und Reproduzierbarkeit der erzielten Ergebnisse sichergestellt.

Die Funktionalität des Werkzeuges wurde am Beispiel einer LQG-Lageregelung für ein elektromechanisches Positioniersystem nachgewiesen. Es wurde gezeigt, dass durch die systematische Aufbereitung der Reglergleichungen und eine entsprechende Werkzeugunterstützung die Erzeugung von Festkommacode für Steuergeräte genauso beherrschbar ist, wie das Rapid Control Prototyping mit leistungsfähiger Gleitkomma-Hardware. Die experimentellen Ergebnisse mit Gleit- und Festkomma-Arithmetik sind praktisch identisch.

Mit Hilfe der API-Funktionalität des Werkzeuges werden Schnittstellen für den Zugriff durch weitere Werkzeuge bereitgestellt. Ein Beispiel ist das im CLM entwickelte Skalierungswerkzeug für diskrete Übertragungsfunktionen und Filter, das sich dieser Schnittstellen bedient. Gegenwärtig wird für die Entwicklungsumgebung TargetLink ein übergeordnetes Skalierungswerkzeug entwickelt, mit dem alle Variablen komplexer Blockschaltbilder automatisch skaliert werden können. Auch dieses Werkzeug greift auf die API-Tools zurück.

## Literaturverzeichnis

- [1] U. Kiffmeier, L. Köster, M. Meyer und C. Witte: *Automatische Generierung von Produktionscode für Seriensteuergeräte*. Automatisierungstechnik 47 (1999), Heft 7, S. 295-304.
- [2] H. Hanselmann: *DSP in Control: The Total Development Environment*. ICSPAT, Intl. Conference On Signal Processing & Technology, Aug. 1995, MA, USA.
- [3] dSPACE: *TargetLink Production Code Generation Guide*. dSPACE GmbH, Paderborn, 2001.
- [4] H. Henrichfreise: *Prototyping of a LQG Compensator for a Compliant Positioning System with Friction*. 1. Workshop TransMechatronik - Entwicklung und Transfer von Entwicklungssystemen der Mechatronik, HNI-Verlagsschriftenreihe, Band 23, 1. Edition, Hrsg: Jürgen Gausemeier, Paderborn 1997. Aufsatz verfügbar über die CLM-Homepage <http://www.clm-online.de/>.
- [5] H. Hanselmann: *Low-Resolution Implementation of High Resolution Position Control*. IEEE Transaction on Automatic Control, Vol. 33, No. 11, S. 1074-1078, 1988.
- [6] H. Hanselmann: *Implementation of Digital Controllers – A Survey*. Automatica, Vol. 23, No. 1, S. 7-32, 1987.
- [7] H. Henrichfreise, J. Jusseit und M. Weller: *Der mechatronische Entwicklungsprozess am Beispiel der Regelung eines elektromechanischen Positioniersystems*. 9. IIR F&E-Zukunftsforum, Rüsselsheim, 10. - 11. April 2002. Aufsatz verfügbar über die CLM-Homepage <http://www.clm-online.de/>.