Forschungszentrum Karlsruhe
in der Helmholtz-Gemeinschaft

Wissenschaftliche Berichte
FZKA 6900

Proceedings
13. Workshop Fuzzy Systeme
Dortmund, 19. - 21. November 2003

R. Mikut, M. Reischl (Hrsg.)

Institut für Angewandte Informatik

November 2003

Forschungszentrum Karlsruhe

in der Helmholtz-Gemeinschaft

Wissenschaftliche Berichte

FZKA 6900

Proceedings 13. Workshop Fuzzy Systeme

Ralf Mikut, Markus Reischl (Hrsg.)

Institut für Angewandte Informatik

Dortmund, 19.-21. November 2003

Forschungszentrum Karlsruhe GmbH, Karlsruhe

2003

Forschungszentrum Karlsruhe GmbH
Postfach 3640, 76021 Karlsruhe

Mitglied der Hermann von Helmholtz-Gemeinschaft
Deutscher Forschungszentren (HGF)

ISSN 0947-8620

VORWORT

Dieser Tagungsband enthält die Beiträge des 13. Workshops „Fuzzy Systeme“ des Fachausschusses 5.22 „Fuzzy Control“ der VDI/VDE-Gesellschaft für Mess- und Automatisierungstechnik (GMA) und der Fachgruppe „Fuzzy-Systeme und Soft-Computing“ der Gesellschaft für Informatik (GI), der vom 19.-21. November 2003 im Haus Bommerholz, Dortmund, stattfindet.

Der jährliche Workshop unseres Fachausschusses bietet ein Forum zur Diskussion neuer methodischer Ansätze und industrieller Anwendungen auf dem Gebiet der Fuzzy-Logik und in angrenzenden Gebieten wie Künstlichen Neuronalen Netzen und Evolutionären Algorithmen. Besondere Schwerpunkte sind automatisierungstechnische Anwendungen, z.B. in der Verfahrenstechnik, Energietechnik, Kfz-Technik, Robotik und Medizintechnik, aber auch Lösungen in anderen Problemgebieten (z.B. Data Mining für technische und nichttechnische Anwendungen) sind von Interesse.

Die Ergebnisse werden von den Mitgliedern und Gästen aus Hochschulen, Forschungseinrichtungen und der Industrie präsentiert und in Klausuratmosphäre intensiv diskutiert. Dabei ist es gute Tradition, auch neue Ansätze und Ideen bereits in einem frühen Entwicklungsstadium vorzustellen, in dem sie noch nicht vollständig ausgereift sind.

Nähere Informationen zum GMA-Fachausschuss bzw. zur GI-Fachgruppe erhalten Sie unter

http://www.iai.fzk.de/medtech/biosignal/gma/index.html

bzw.

http://public.rz.fh-wolfenbuettel.de/~klawonn/GI/gi_deutsch.html.

Die Herausgeber bedanken sich an dieser Stelle bei allen Autoren und Rednern sowie bei Prof. Dr. Klawonn (Fachhochschule Braunschweig/Wolfenbüttel), Herrn Dr. Runkler (Siemens AG), Herrn Dr. Jäkel (Forschungszentrum Karlsruhe) und Herrn Dr. Kroll (ABB Heidelberg), die maßgeblich an der Vorbereitung des Workshops beteiligt waren.

Ralf Mikut und Markus Reischl

INHALTSVERZEICHNIS

Ein verbesserter Fuzzy-Clustering-Ansatz mit Anwendungen auf z-Kurven

Frank Klawonn

Fachbereich Informatik
Fachhochschule Braunschweig/Wolfenbüttel
Salzdahlumer Str. 46-48
38302 Wolfenbüttel
Tel.: (05331) 9396111
Fax: (05331) 9396002
E-Mail: f.klawonn@fh-wolfenbuettel.de

1 Einleitung

Die meisten Fuzzy-Clusteranalyseverfahren verwenden als einen wesentlichen Parameter den so genannten Fuzzifier, mit dem gesteuert werden kann, wie fließend die Grenzen zwischen (überlappenden) Clustern sein sollten. Der Wert des Fuzzifiers kann zwischen 1 (scharfe Clustereinteilung) und unendlich (alle Daten gehören mit dem gleichen Zugehörigkeitsgrad zu allen Cluster n) gewählt werden. Eine unangenehme Eigenschaft des Fuzzifiers besteht darin, dass alle Daten auf alle Cluster Einfluss nehmen, sowie der Fuzzifier gröër als 1 gewählt wird, d.h., wenn nicht scharfes Clustering angewendet werden soll. Dies kann zu Problemen führen, insbesondere dann, wenn der Datensatz viele Daten enthält oder die Clusterprototypen möglichst genau bestimmt werden sollten.

In diesem Beitrag analysieren wir die Eigenschaften des Fuzzifiers genauer und leiten Vorschläge für verbesserte Ansätze ab. Angewendet wurden die Verfahren unter Anderem auf biologische Datensätze, die dreidimensionale Repräentationen von DNA-Sequenzen (so genannte z-Kurven) enthalten.

2 Zielfunktionsbasiertes Fuzzy-Clustering

Fuzzy-Clustering zielt auf das Auffinden von Strukturen in Daten ab. Ein Datensatz wird in eine endliche Anzahl von Clustern aufgeteilt, die die Strukturen in den Daten repräsentieren. Beim Fuzzy-Clustering wird – im Gegensatz zum herkömmlichen partitionierenden (harten) Clustering – ein einzelner Datenvektor nicht eindeutig einem Cluster zugeordnet, sondern es wird ein Zugehörigkeitsgrad des Datenvektors zu jedem Cluster bestimmt, um verrauschte und nicht eindeutig zuzuordnende Daten handhaben zu können. Die meisten Fuzzy-Clustering-Verfahren basieren auf der Minimierung einer Zielfunktion unter Nebenbedingungen. Der am häufigsten verwendete Ansatz ist das probabilistische Clustering mit der zu minimierenden Zielfunktion

$$f = \sum_{i=1}^{c} \sum_{j=1}^{n} u_{ij}^{m} d_{ij} \tag{1}$$

unter den Nebenbedingungen

$$\sum_{i=1}^{c} u_{ij} \;=\; 1 \qquad \text{für alle } j = 1, \ldots, n. \tag{2}$$

Dabei wird die Anzahl der Cluster zunächst fest gewählt. Für den zu clusternden Datensatz $\{x_1, \ldots, x_n\} \subset \mathbb{R}^p$ müssen die Zugehörigkeitsgrade u_{ij} des jeweiligen Datenvektors x_j zum $i$ten Cluster bestimmt werden. d_{ij} ist ein Distanzmaß, das den Abstand oder die Unähnlichkeit des Datenvektors x_j zum Cluster i spezifiziert, zum Beispiel der (quadratierte) euklidische Abstand von x_j zum $i$ten Clusterzentrum. Der Parameter $m > 1$ – Fuzzifier gennant – steuert, wie stark Cluster bezogen auf die Zugehörigkeitsgrade überlappen dürfen. Die Nebenbedingungen (2) begründen den Namen probabilistisches Clustering, da die Zugehörigkeitsgrade u_{ij} in diesem Fall auch als die Wahrscheinlichkeit, dass x_j zum Cluster i gehört, interpretierbar sind.

Die in der Zielfunktion zu optimierenden Parameter sind die Zugehörigkeitsgrade u_{ij} und die Clusterparameter, die hier nicht explizit angegeben sind. Sie sind in den Distanzen d_{ij} verborgen. Der gebräuchliche Weg, dieses nicht-lineare Optimierungsproblem – die Minimierung der Zielfunktion (1) – zu lösen, besteht aus einem alternierenden Schema, das jeweils abwechselnd einen der Parametersätze Zugehörigkeitsgrade und Clusterparameter festhält und den anderen Parametersatz bezüglich des festgehaltenen optimiert. In dieser Arbeit geht es nicht um die verschiedenartigen Clusterformen (Kugeln, Ellipsoide, lineare Mannigfaltigkeiten, Quadriken,...), die durch eine geeignete Wahl der Distanzfunktion darstellbar sind. (Einen Überblick über derartige Verfahren findet man z.B. in [2, 4].)

Die hier vorgestellten Techniken lassen sich ebenfalls auf das Noise Clustering [3] anwenden, bei dem der probabilistische Grundgedanke beibehalten wird. Neben den zu bestimmenden Clustern wird jedoch noch ein zusätzliches Cluster für Rauschdaten eingeführt. Alle Datenvektoren besitzen einen festen (großen) Abstand zum Rauschcluster. Auf diese Weise behalten Datenvektoren, die an der Grenze zwischen zwei Clustern liegen, weiterhin einen hohen Zugehörigkeitsgrad zu beiden Clustern. Datenvektoren, die Rauschen darstellen und von allen Clustern weiter entfernt sind, werden dem Rauschcluster mit einem hohen Zugehörigkeitsgrad zugeordnet und beeinflussen die anderen Cluster weniger.

Possibilistisches Clustering [7], bei dem die probabilistische Nebenbedingung unter Einführung eines Strafterms für kleine Zugehörigkeitsgrade vollständig aufgegeben wird, um die triviale Lösung $u_{ij} = 0$ zu vermeiden, weist ähnliche Probleme auf wie das probabilistische Clustering, was den Fuzzifier betrifft. Possibilistisches Clustering ist ohnehin formal nicht sauber begründbar, da das Ziel der Clusteranalyse explizit in der Vermeidung des globalen Minimums der Zielfunktion und in der Suche nach einem lokalen Minimum besteht [8].

3 Die Rolle des Fuzzifiers

Wie bereits im vorhergehenden Abschnitt erwähnt, wird die Zielfunktion (1) unter den Nebenbedingungen (2) optimiert, indem man zunächst die Clusterprototypen

zufällig initialisiert und dann abwechselnd die Zugehörigkeitsgrade und die Clusterprototypen optimiert, während der jeweils andere Parametersatz als fix betrachtet wird.

Um die Zugehörigkeitsgrade zu berechnen, werden für die Nebenbedingungen (2) Lagrange Multiplikatoren eingeführt, so dass die zu minimierende Zielfunktion folgendermaßen aussieht:

$$L \;=\; \sum_{i=1}^{c}\sum_{j=1}^{n} g(u_{ij})d_{ij} \;+\; \sum_{j=1}^{n}\lambda_j\left(1-\sum_{i=1}^{c}u_{ij}\right),$$

wobei $g(u) = u^m$ gilt.

Leitet man diese erweiterte Zielfunktion nach den Zugehörigkeitsgraden partiell ab, ergibt sich

$$\frac{\partial L}{\partial u_{ij}} \;=\; g'(u_{ij})d_{ij} - \lambda_j. \tag{3}$$

Setzt man die partiellen Ableitungen null und berücksichtigt die Nebenbedingungen (2), erhält man

$$u_{ij} \;=\; \frac{1}{\sum_{k=1}^{c}\left(\frac{d_{ij}}{d_{kj}}\right)^{\frac{1}{m-1}}} \tag{4}$$

als Berechnungsformel für die Zugehörigkeitsgrade. Man sieht an dieser Formel bereits, dass die Zugehörigkeitsgrade niemals null oder eins werden. (Außer in dem seltenen Fall, dass der Abstand eines Datums zu einem Clusterprototypen exakt null beträgt.

Um diesen Effekt besser zu verstehen, betrachten wir noch einmal die zu minimierende Zielfunktion (3). An einem Minimum der Zielfunktion müssen die partiellen Ableitung (3) null sein, d.h., es muss $\lambda_j \;=\; g'(u_{ij})d_{ij}$ gelten. Da λ_j unabhängig von i ist, muss $g'(u_{ij})d_{ij} = g'(u_{kj})d_{kj}$ für alle i, k erfüllt sein. Das bedeutet, dass die Zugehörigkeitsgrade so gewählt werden müssen, dass die die Werte $g'(u_{ij})d_{ij}$ ausbalanciert (alle gleich) sind. Da für $g(u) = u^m$ folgt, dass $g'(0) = 0$ gilt, treten Zugehörigkeitsgrade mit dem Wert Null nicht auf.

Die Aufgabe der Funktion g besteht darin, eine Transformation der Zugehörigkeitsgrade durchzuführen. Für einfachste Form $g(u) = u$ ergibt sich das scharfe Clustering, da ein Ausbalancieren der $g'(u_{ij})d_{ij}$-Werte wegen $g'(u) = 1$ nicht möglich ist.

Welche Minimalforderungen sollte die Transformation g erfüllen? g sollte sicherlich monoton wachsend sein und $g(0) = 0$ and $g(1) = 1$ erfüllen. Um den Effekt des Ausbalancierens zu erreichen, sollte außerdem g' monoton wachsend sein. Außerdem sollte $g'(0) > 0$ gelten, um den Effekt zu vermeiden, dass Zugehörigkeitsgrade mit dem Wert Null vermieden werden.

Eine pragmatische Bedingung an g besteht darin, dass sich die Zugehörigkeitsgrade aus der zu minimierenden Zielfunktion (3) analytisch berechnen lassen. In [5] und [6] wurden Vorschläge für geeignete Transformtaionen g gemacht:

$$\begin{aligned} g(u) &= \frac{1}{e^{\alpha}-1}\left(e^{\alpha u}-1\right) \\ g(u) &= \alpha u^2 + (1-\alpha)u \end{aligned}$$

Verzichtet man auf explizite Gleichungen zur direkten Bestimmung der Zugehörigkeitsgrade, so kann man allgemeinere Transformationen g betrachten und die Zugehörigkeitsgrade iterativ bestimmen, indem man versucht, die Werte $g'(u_{ij})d_{ij}$ auszugleichen.

4 Schlussbemerkungen

Die beschriebenen Verfahren wurden im Rahmen eines Projektes über Fuzzy-Clustering implementiert und erfolgreich auf verschiedene Datensätze angewendet, unter Anderem auf die Clusterung von z-Kurven. z-Kurven sind Kurven im Raum, die aus DNA-Sequenzen abgeleitet werden. Da z-Kurven unterschiedlich lang sind und die Ähnlichkeit zwischen ihnen auf Teilsequenzen basiert, wurde ein modifizierter Fuzzy-c-Means-Algorithmus angewendet, dessen Prototypen Teilsequenzen von z-Kurven sind.

Literatur

[1] Bezdek, J. C.: *Pattern Recognition with Fuzzy Objective Function Algorithms*. Plenum Press, New York. 1981.

[2] J.C. Bezdek, J. C.; Keller, J.; Krishnapuram, R.; Pal, N. R.: *Fuzzy Models and Algorithms for Pattern Recognition and Image Processing*. Kluwer, Boston. 1999.

[3] Davé, R. N.: Characteriztion and Detection of Noise in Clustering. *Pattern Recognition Letters* 12 (1991), S. 657–664.

[4] Höppner, F.; Klawonn, F.; Kruse, R.; Runkler, T.: *Fuzzy Cluster Analysis*. Wiley, Chichester. 1999.

[5] Klawonn, F., Höppner, F: What is Fuzzy About Fuzzy Clustering? Understanding and Improving the Concept of the Fuzzifier In: Berthold, M.R., Lenz, H.-J., Bradley, E., Kruse, R., Borgelt C. (eds.): Advances in Intelligent Data Analysis V. Springer, Berlin 2003, S. 254-264.

[6] Klawonn, F., Höppner, F: An Alternative Approach to the Fuzzifier in Fuzzy Clustering to Obtain Better Clustering Results Proc. 3rd Eusflat Conference, Zittau 2003, S. 730-734.

[7] Krishnapuram, R.; Keller, J.: A Possibilistic Approach to Clustering. *IEEE Trans. on Fuzzy Systems* 1 (1993), S. 98–110.

[8] Timm, H.; Borgelt, C.; Kruse, R.: A Modification to Improve Possibilistic Cluster Analysis. *IEEE Intern. Conf. on Fuzzy Systems* Honululu (2002).

Fuzzy-Clustering über simultan aufgezeichnete Ganganalyse-Zeitreihen

Tobias Loose, Ralf Mikut, Georg Bretthauer

Forschungszentrum Karlsruhe GmbH, Institut für Angewandte Informatik,
D-76021 Karlsruhe, Postfach 3640,
Telefon: (07247) 82-5757, Fax: (07247) 82-5786, E-Mail: tobias.loose@iai.fzk.de

Abstract
Die computerbasierte Analyse großer, komplexer Datenmengen erweist sich oftmals als iterativer Prozess, bei dem für ein Anwendungsgebiet die Auswahl, Modifikation und Anpassung existierender Verfahren mit der Interpretation bzw. Plausibilitätsprüfung der Ergebnisse einher gehen. In diesem Beitrag werden Strategieelemente aus diesem iterativen Prozess für die Anwendung von Fuzzy-Cluster-Verfahren bei der Zeitreihenanalyse vorgestellt. Fuzzy-Cluster-Verfahren eignen sich prinzipiell für die Erkennung von zunächst unbekannten Gruppierungen in Daten (unüberwachte Verfahren). Als Anwendungsbeispiel dient die Instrumentelle Ganganalyse, die ein etabliertes Verfahren zur Quantifizierung von Bewegungen ist. Für den Einsatz der Verfahren wird ebenfalls die Eignung von Merkmalen zum Clustern untersucht. Als leistungsfähig erweist sich bei der Merkmalsextraktion eine vorgelagerte Dimensionsreduktion mit einer Hauptkomponentenanalyse, um Effekte durch starke Korrelationen in Zeitreihen zu unterdrücken. Durch Validierungsmaße können die Ergebnisse der Cluster-Verfahren beurteilt werden. Dadurch kann die Merkmalsauswahl und Bestimmung einer Cluster-Anzahl mit einem interaktiven Vorgehen zumindest teilautomatisiert werden.

1 Einleitung

Die Analyse komplexer technischer oder nichttechnischer Systeme erfolgt oftmals mit Messmethoden, die durch die simultane Aufzeichnung mehrerer Messgrößen als Zeitreihen große Datenmengen erzeugen, z.B. für System-Diagnosen, Schadensanalysen oder System-Optimierungen. Eine wichtige Aufgabe ist das Finden von Subgruppen, die beispielsweise die Fehlerfreiheit oder verschiedene Fehlerarten für technische Prüfstände oder unterschiedliche Pathologien im medizinischen Bereich beschreiben. Allerdings ist die Analyse dieser Zeitreihen anwendungsspezifisch und meistens iterativ mit empirisch gesammeltem Wissen zu entwickeln. Bei der Analyse großer Datenmengen ist es prinzipiell ein Problem, relevante Information zu finden. „Relevante Information“ bezieht sich auf eine gegebene Problemformulierung, wie z.B. die Erkennung wesentlicher (zunächst unbekannter) Subgruppen. Oftmals werden viele Merkmale pauschal aufgezeichnet. Dabei sind Zusammenhänge mehr oder minder vorhanden, Heterogenitäten einzelner Objekte existieren und Interpretationen sind aufgrund der Komplexität schwierig. Weiterhin sind Informationen meistens redundant vorhanden (z.B. hohe Korrelationen benachbarter Abtastzeitpunkte, Messung ähnlicher Mechanismen durch unterschiedliche Merkmale). Merkmale sind verrauscht bzw. tragen keine Information zur Lösung des Problems, zeigen mitunter gegensätzliche Lösungen auf und sind unterschiedlich zuverlässig.

Ein Beispiel ist die klinische Instrumentelle Ganganalyse [1–4] zur Quantifizierung neurogener Bewegungsstörungen. Insbesondere werden hierbei Kinematikverläufe der unteren Extremitäten von Patienten bzw. Referenzpersonen als Zeitreihen simultan gemessen, siehe Bild 1. Die Datenanalyse für einzelne Patienten (Diagnosestellung sowie die Auswahl und Validierung einer geeigneten Therapie) erfolgt derzeit rein subjektiv durch hochspezialisierte Ärzte. Nachteile der subjektiven Analyse sind z.B. die schwere Formalisierbarkeit und Übertragbarkeit von Fachkenntnissen, die Limitierung auf relativ einfache Zusammenhänge oder unvollständiges Wissen. Methoden der computerbasierten, quantitativen Datenanalyse zur Behebung der Nachteile sind auf diesem Gebiet wenig etabliert. Gründe sind die mangelnde Interpretierbarkeit derartiger Ergebnisse und die beschränkte Anpassung von Analysemethoden an die jeweilige Problemstellung des Anwendungsgebiets [5, 6].

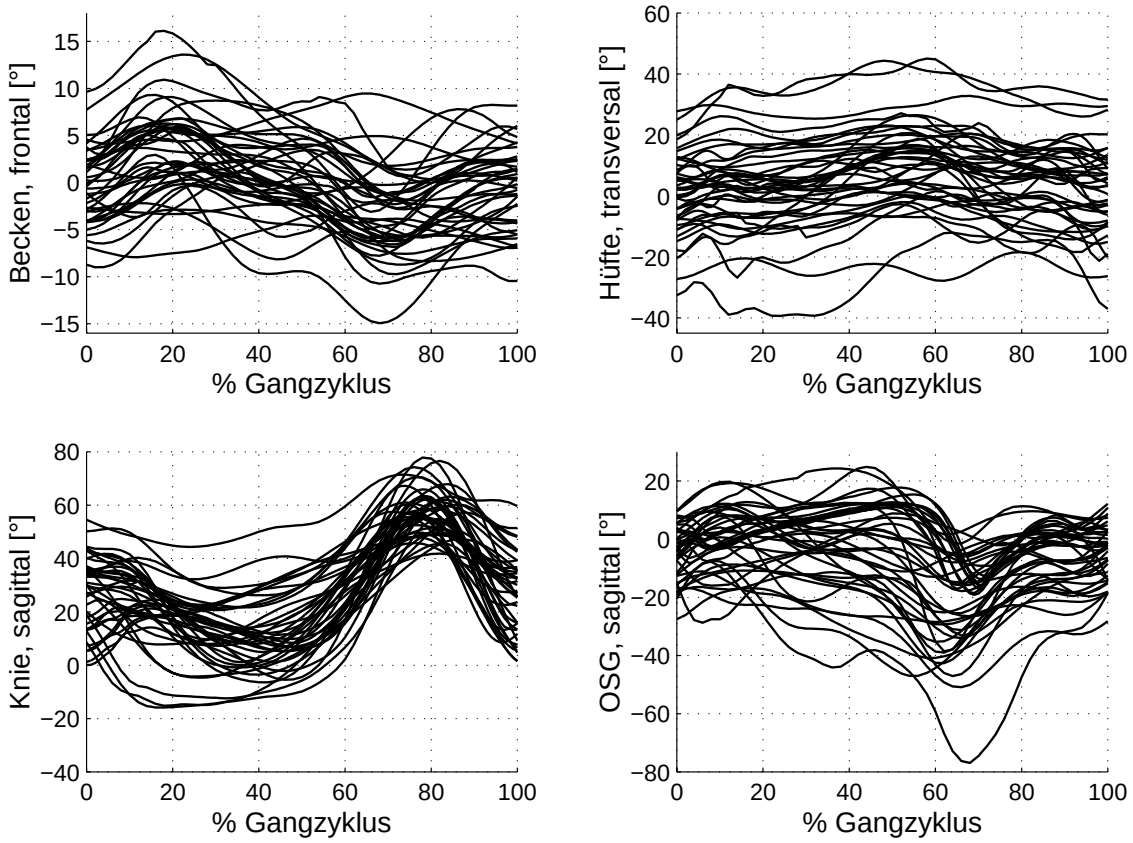


Bild 1: Einblick in Komplexität der Daten: Beispiel gemessener Zeitreihen aus der Instrumentellen Ganganalyse von 15 Patienten und 5 Probanden, jeweils linke und rechte Körperseite. OSG bedeutet Oberes Sprunggelenk, die körperbezogenen Betrachtungsebenen sind frontal (von vorn), transversal (von oben) und sagittal (von der Seite).

Die Erkennung von Subgruppen dient einem besseren Verständnis (z.B. Arten von Kompensationsmechanismen), einer besseren Diagnosestellung und damit u.U. gezielteren Therapieplanung bis hin zur Prognose. Der Begriff „Klassen“ bzw. „Gruppen“ wird in diesem Beitrag für semantische Einteilungen verwendet [7], z.B. Referenzpersonen und Patienten. Entsprechend werden weitere *existierende* Unterteilungen als Subgruppen bezeichnet, wie z.B. unterschiedliche pathologische (krankhafte) Ausfallerscheinungen innerhalb einer Patientengruppe. Als Cluster werden hier die *berechneten* Dateneinteilungen bezeichnet. Die Cluster dienen dem Ziel, die

unterschiedlichen realen Gruppierungen zu finden. In diesem Beitrag werden Fuzzy-Cluster-Methoden vorgestellt und miteinander verglichen, um sie auf die Zeitreihenanalyse in der Ganganalyse anzupassen. Fuzzy-Cluster-Methoden bestimmen graduelle Zugehörigkeiten (ZGH) der Datensätze zu Cluster-Zentren. Die Positionen der Cluster-Zentren ergeben sich aus den Datensätzen entsprechend der ZGH. Die Anwendung der hier verwendeten Methoden soll die Merkmalsextraktion, die Bewertung der Existenz von Subgruppen (als Cluster beschrieben) in einer oder mehreren Zeitreihen und die Festlegung der Cluster-Anzahl automatisieren.

Um die Methoden erfolgreich anzuwenden, muss eine Bewertung definiert werden, Abschnitt 2. Die Bewertung ist empirisch gesammeltes Wissen aus Datenanalyse- und Anwendungs-Sicht. Diese Sammlung basiert auf einem iterativen Prozess. Da die Datenanalyse (hier das Clustern) in hochdimensionalen Räumen erfolgt, ist eine subjektive Analyse nahezu aussichtslos und das Ergebnis, das das Clustern liefert, daher völlig unbekannt. Dazu existieren einige Cluster-Methoden mit unterschiedlichen Eigenschaften. Es gibt aber keine Methode, die für alle Anwendungen gleichermaßen funktioniert und gute Ergebnisse liefert [8]. Somit werden die Methoden anfangs pauschal angewandt (zunächst auf einem reduzierten Suchraum, mit einfacher Fragestellung). Das Ergebnis wird anschließend interpretiert und auf Plausibilität geprüft. Zur Verbesserung der Leistungsfähigkeit und Ausdehnung des Suchraums werden nun die Methoden variiert und angepasst – unter Einbezug der vorangegangenen Interpretation. Daraus begründet sich ein iterativer Prozess, bei dem Elemente und Vorgehensweisen mitunter auf andere Anwendungsgebiete übertragbar ist. Dazu gibt Abschnitt 3 einen methodischen Überblick. Darin ist eine kompakte Einführung zu Fuzzy-Cluster-Methoden und Möglichkeiten zur Validierung der Ergebnisse gegeben. Zudem sind Strategieelemente zur Extraktion potenzieller Information aus Zeitreihen gezeigt. Vergleiche und Interpretationen der Verfahren sind in Abschnitt 4 dargestellt.

2 Empirische Bewertung von Subgruppen

Um ein berechnetes Analyseergebnis (Einschätzung von Cluster-Güten zum Erkennen guter Cluster-Ergebnissen) bewerten zu können, müssen Kriterien festgelegt werden, die an intuitiven Ergebnisinterpretationen und teilweise vorhandenem empirischen Wissen angelehnt sind, wie z.B.

1. Bevorzugung überwiegend eindeutiger Zugehörigkeiten der Datensätze zu den Clustern (Gewinnung von Information, Gegenbeispiel: Alle Datensätze sind gleichermaßen zu allen Cluster-Zentren zugeordnet),
2. Cluster-Zentren sollen weitgehend getrennt und nicht eng benachbart sein (Ermittlung wesentlicher, unterschiedlicher Subgruppen, z.B. hier: orthopädisch unterschiedliche Gangbilder, Gegenbeispiel: Alle Cluster-Zentren liegen übereinander),
3. Bestimmung einer geeigneten Cluster-Anzahl,
4. Vermeidung des Anlernens einzelner Datensätze, d.h. Einhaltung einer „gewissen Abstraktion“ (Generalisierung) in einem Cluster-Zentrum als Zusammenfassung von mehreren Datensätzen,

5. Finden bekannter Klassen als Cluster (z.B. bereits ermittelte Cluster im eingeschränkten Suchraum oder empirisch bekannte Gruppen, wie Referenzgruppe oder Beschreibung von Datensätzen mit unterschiedlichen, semantischen Eigenschaften durch separates Cluster-Zentrum) und

6. plausible Zuordnung einzelner Datensätze zu Clustern.

Das Ziel besteht darin, Zeitreihen mit ausgeprägten Gruppen zu finden und diese durch Cluster zu beschreiben. Dabei sind die Anzahl der Gruppen sowie Zusammenhänge zwischen den Zeitreihen meist unbekannt. Zudem lassen sich oftmals derartige Fragestellungen nicht eindeutig lösen, so dass mit einer gewissen Unschärfe zu rechnen ist, wie z.B. in der Ganganalyse, siehe Abschnitt 4.

3 Verfahren zum automatisierten Finden von Subgruppen

3.1 Cluster-Algorithmen

Viele Fuzzy-Cluster-Techniken [8, 9] finden Cluster-Zentren $\mathbf{V}_{s\times C} = (\mathbf{v}_1 \cdots \mathbf{v}_C)$ (s = Anzahl Merkmale, C = Cluster-Anzahl) durch Lösen des Optimierungsproblems

$$J_q(\mathbf{U},\mathbf{V}) = \sum_{j=1}^{N}\sum_{c=1}^{C}(u_{j,c})^q \cdot d_c^2(\mathbf{x}_j,\mathbf{v}_c) \rightarrow \min_{\mathbf{U},\mathbf{V}}, \tag{1a}$$

$$\sum_{c=1}^{C} u_{j,c} = 1, \forall j = 1\ldots N, \text{ mit } u_{j,c} \geq 0, \tag{1b}$$

$$\sum_{j=1}^{N} u_{j,c} > 0, \forall c = 1\ldots C. \tag{1c}$$

Die Nebenbedingungen erzwingen eine probabilistische Cluster-Einteilung Gl. (1b) und fordern, dass kein Cluster leer ist Gl. (1c) [8]. Für eine große Zugehörigkeit $u_{j,c}$ eines Datensatzes $j = 1\ldots N$ zu einem Cluster-Zentrum $c = 1\ldots C$ ist eine kleine quadratische Distanz d_c^2 optimal und umgekehrt. Durch den Fuzzifizierungsgrad q können „harte“ ($q = 1$) oder einheitlich „weiche“ Zugehörigkeiten $q \rightarrow \infty$ eingestellt werden. Mit der Nebenbedingung Gl. (1b), Nullsetzen der Ableitung von Gl. (1a) für $q > 1$ nach $\mathbf{U}$ bzw. $\mathbf{V}$ ergibt sich die Zugehörigkeit (ZGH)

$$u_{j,c} = \frac{[d_c^2(\mathbf{x}_j,\mathbf{v}_c)]^{\frac{1}{1-}}}{\sum_{i=1}^{C}[d_i^2(\mathbf{x}_j,\mathbf{v}_i)]^{\frac{1}{1-}}} \tag{2}$$

bzw. die Lage der Cluster-Zentren, z.B. für Euklidische Distanzen

$$\mathbf{v}_c = \frac{\sum_{j=1}^{N}(u_{j,c})^q\mathbf{x}_j}{\sum_{j=1}^{N}(u_{j,c})^q}. \tag{3}$$

Datensätze mit einer großen Zugehörigkeit ziehen das Cluster-Zentrum stärker in ihre Richtung als Datensätze mit kleinen Zugehörigkeiten. Da Gl. (1) nicht geschlossen lösbar ist, werden nach einer Wahl von Start-Clustern (z.B. zufällig oder je

nach Datendichte verteilt) Gl. (2) und Gl. (3) iterativ berechnet. Die Iteration wird beendet, wenn die Änderung aller Zugehörigkeiten in aufeinanderfolgenden Iterationsschritten $u^*_{j,c}$ nach $u_{j,c}$ eine bestimmte Schranke ε mit $\Sigma_{j=1}^{N}\Sigma_{c=1}^{C}\left(u^*_{j,c}-u_{j,c}\right)^2<\varepsilon$ unterschreitet. Ein Wert von $\varepsilon=10^{-3}$ hat sich hier bewährt, wobei sich die Cluster-Zentren meist schon nach wenigen Iterationsschritten an die endgültigen Positionen annähern. Freiheitsgrade für eine problemspezifische Anpassung der hier vorgestellten automatisierten Verfahren bestehen in der Wahl

- der Distanz d_c (Abschnitt 3.2),
- der Merkmale $\mathbf{X}$ (Abschnitt 3.3),
- des Algorithmus zur Festlegung der Cluster-Anzahl C (Abschnitt 3.4) und
- dem Fuzzifizierungsgrad q.

Die Literatur empfiehlt oftmals einen Fuzzifizierungsgrad von $q=2$ [2, 3, 8]. Dieser wird in dem vorliegenden Beitrag auch einheitlich auf $q=2$ eingestellt, Arbeiten zur Optimierung finden sich z.B. unter [10].

3.2 Wahl der Distanz

Der Fuzzy-C-Means-Algorithmus (FCM) verwendet eine einheitliche Distanz $d(\mathbf{x}_j,\mathbf{v}_c)$ mit

$$d_c^2(\mathbf{x}_j,\mathbf{v}_c)=d^2(\mathbf{x}_j,\mathbf{v}_c)=(\mathbf{x}_j-\mathbf{v}_c)^T\cdot\mathbf{M}\cdot(\mathbf{x}_j-\mathbf{v}_c). \tag{4}$$

Beispiele sind die Euklidische (EU, $\mathbf{M}=\mathbf{I}_s$: Einheitsmatrix) oder Mahalanobis-Distanz (MA, $\mathbf{M}=\mathbf{S}^{-1}$: inverse Kovarianzmatrix über alle Datensätze). Der Gustafson-Kessel-Algorithmus (GK) basiert auf cluster-spezifischen Distanzen $d_c(\mathbf{x}_j,\mathbf{v}_c)$ mit

$$d_c^2(\mathbf{x}_j,\mathbf{v}_c)=(\mathbf{x}_j-\mathbf{v}_c)^T\cdot\mathbf{M}_c\cdot(\mathbf{x}_j-\mathbf{v}_c)\text{ mit }\mathbf{M}_c=\sqrt{\det(\mathbf{S}_c)}\mathbf{S}_c^{-1} \tag{5}$$

mit konstantem Cluster-Volumen M_C, aber variabler Cluster-Form als „Fuzzy-Kovarianzmatrix“ [8]

$$\mathbf{S}_c=\frac{\sum_{j=1}^{N}u_{j,i}^q(\mathbf{x}_j-\mathbf{v}_i)(\mathbf{x}_j-\mathbf{v}_i)^T}{\sum_{j=1}^{N}u_{j,i}^q}. \tag{6}$$

Sie modifiziert die Berechnung der Kovarianzmatrix [11] für jedes Cluster durch Verwendung der Zugehörigkeiten $u_{j,i}^q$ (mit Werten aus dem Intervall $[0,1]$). Der Gath-Geva-Algorithmus (GG) nimmt eine mehrdimensionale Normalverteilung der Datensätze in einem Cluster an. Die Distanz d_c ergibt sich aus der geschätzten reziproken Wahrscheinlichkeit, dass ein Datensatz zu einem Cluster gehört:

$$\begin{aligned} d_c^2(\mathbf{x}_j,\mathbf{v}_c) &= \frac{(2\pi)^{\frac{s}{2}}\sqrt{\det(\mathbf{S}_c)}}{\widehat{p}_c}\exp\left(\frac{1}{2}(\mathbf{x}_j-\mathbf{v}_c)^T\mathbf{S}_c^{-1}(\mathbf{x}_j-\mathbf{v}_c)\right) \\ \text{mit }\widehat{p}_c &= \frac{\sum_{j=1}^{N}u_{j,c}^q}{\sum_{j=1}^{N}\sum_{l=1}^{C}u_{j,l}^q}. \end{aligned} \tag{7}$$

Die geschätzte Wahrscheinlichkeit für ein Auftreten eines Datensatzes in einem Cluster $\widehat{p}_c$ ergibt sich aus den relativen Zugehörigkeiten zum Cluster c. Beim GK- und GG-Algorithmus wird in jedem Iterationsschritt für jedes Cluster die Kovarianzmatrix aller Datensätze im Cluster neu geschätzt [8].

3.3 Merkmale

Die Cluster-Algorithmen verwenden entweder alle K Abtastzeitpunkte einer oder mehrerer Zeitreihen $Z_{ij}[k]$ (i Zeitreihe, z.B. sagittale Knie-Zeitreihe, $j = 1 \ldots N$ Datensatz, $k = 1 \ldots K$ Abtastzeitpunkt) oder daraus extrahierte Einzelmerkmale [4, 12] als Merkmale $\mathbf{X} = ((x_{lj}))$ mit $l = 1..s$. Einzelmerkmale entstehen z.B. durch eine dimensionsreduzierende Lineartransformation der Zeitreihe i (nach Normierung auf ein Intervall $[-1, +1]$) mit

$$\begin{aligned} \mathbf{x}_{L,ij}^T &= \mathbf{z}_{ij}^T \cdot \mathbf{\Phi} \text{ mit } z_{ijk} = Z_{ij}[k], \\ \mathbf{x}_{L,ij} &= [x_{L,i1j}, \ldots x_{L,isj}]^T, \mathbf{\Phi} = [\boldsymbol{\phi}_1, \ldots \boldsymbol{\phi}_l, \ldots \boldsymbol{\phi}_s]. \end{aligned} \tag{8}$$

Die (K, s)-dimensionale Transformationsmatrix $\mathbf{\Phi}$ wird so entworfen, dass die jeweils interessierende Information möglichst gut erhalten und komprimiert wird [11], z.B. durch Lösen eines Eigenwertproblems

$$(\mathbf{A} - \lambda_l \mathbf{I}_K)\boldsymbol{\phi}_l = \mathbf{0}, \quad \lambda_1 \geq \ldots \geq \lambda_s \geq \ldots \geq \lambda_K. \tag{9}$$

($\mathbf{I}_K$: Einheitsmatrix, λ_l: Eigenwerte, $\boldsymbol{\phi}_l$: Eigenvektoren). Dabei maximiert die Hauptkomponentenanalyse (HKA) mit $\mathbf{A} = \mathbf{T}$ ($\mathbf{T} = N \cdot \mathbf{S}$, Total Variance = Gesamtstreuung aller Datensätze) die Kovarianzinformation für einen s-dimensionalen Unterraum. Die Kovarianzmatrix $\mathbf{S}$ berechnet sich aus allen N Datensätzen.

Im Folgenden gilt für die i-te Zeitreihe entweder $\mathbf{x}_j = (Z_{ij}[1], \ldots Z_{ij}[K])^T$ (ZR: Abtastzeitpunkte der Zeitreihe, $s = K$ Merkmale) bzw. $\mathbf{x}_j = (x_{L,i1j}, x_{L,i2j})^T$ (HKA: lineartransformierte Merkmale aus einer Hauptkomponentenanalyse, $s = 2$ Merkmale) oder für zwei Zeitreihen i und r $\mathbf{x}_j = (Z_{ij}[1], \ldots Z_{ij}[K], Z_{rj}[1], \ldots Z_{rj}[K])^T$ (ZR, $s = 2 \cdot K$ Merkmale) bzw. $\mathbf{x}_j = (x_{L,i1j}, x_{L,i2j}, x_{L,r1j}, x_{L,r2j})^T$ (HKA, $s = 4$) usw.

3.4 Validierungen von Cluster-Ergebnissen und Festlegung Cluster-Anzahl

Die Cluster-Anzahl C sowie die Merkmalsselektion lassen sich allerdings nicht aus dem Wert von Gl. (1a) bestimmen, da $J_q(\mathbf{U}, \mathbf{V})$ für steigende Cluster-Anzahl monoton fällt. Für eine gleiche Anzahl von Clustern und Datensätzen ergibt die Funktion Null. Daher werden anwendungsspezifisch Validierungsmaße verwendet, die bei wiederholter Cluster-Berechnung mit verschiedener Cluster-Anzahl bestimmt werden [8, 13], wie z.B. der Partitionskoeffizient (partition coefficient), die Partitionsentropie oder der Verhältnis-Repräsentant (proportion exponent). Der im Folgenden verwendete Trennungsgrad (separation)

$$S(\mathbf{U}, C) = \frac{J_q(\mathbf{U}, \mathbf{V})}{C \cdot \min\limits_{c,i=1\ldots C, c \neq i} (d^2(\mathbf{v}_c, \mathbf{v}_i))} \tag{10}$$

erweitert Gl. (1a) um die Cluster-Anzahl C und die minimale quadratische Distanz zwischen zwei benachbarten Clustern. Dadurch werden eng benachbarte Cluster-Zentren bestraft. Eine gute Anzahl von Clustern wird durch das erste lokale Minimum angegeben [8]. Der Trennungsgrad S kann bei unterschiedlichen Merkmalsräumen und Distanzen nur bedingt verglichen werden.

3.5 Typische Probleme

Die bisher diskutierten Algorithmen liefern gute Ergebnisse, wenn alle Merkmale relevante Subgruppen enthalten und untereinander nicht korreliert sind. Bild 2a verdeutlicht diesen Fall anhand eines illustrativen Beispiels mit zwei Merkmalen und zwei klar ausgeprägten Clustern, deren Cluster-Zentren sowohl separat für jedes Merkmal (Darstellung auf den jeweiligen Achsen) als auch für die Kombination beider Merkmale (2D-Darstellung inkl. Achsenprojektion) ermittelt wurden (kompatible Cluster).

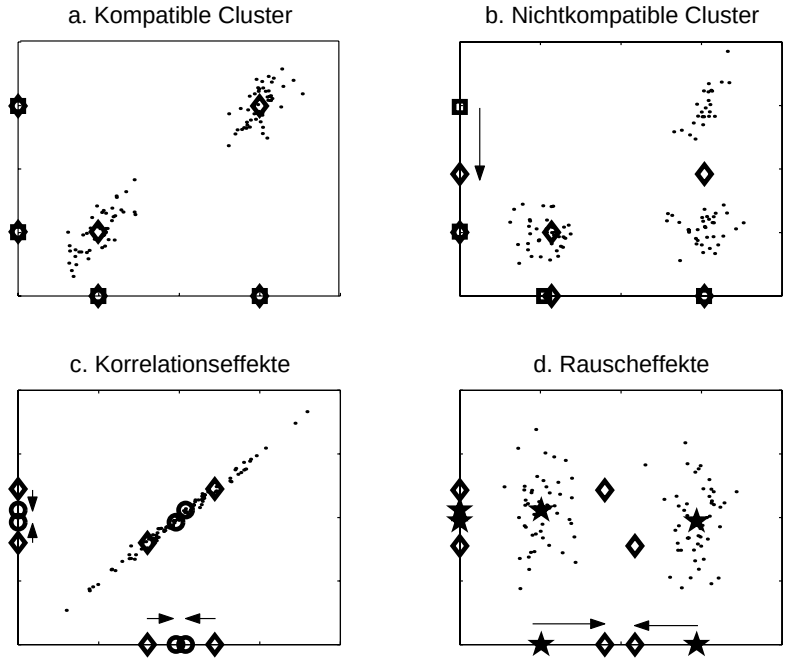


Bild 2: Typische Effekte bei der Cluster-Berechnung: a. Kompatible Cluster, b. Nichtkompatible Cluster, c. Korrelationseffekte, d. Rauscheffekte

Komplizierter ist der Fall, wenn zwar für jedes Merkmal wenige Subgruppen existieren, aber im höherdimensionalen Raum eine höhere Cluster-Anzahl erforderlich ist. Diesen Fall zeigt Bild 2b, bei dem jedes Merkmal je zwei, deren Kombination aber drei Cluster aufweist. Zu niedrige Cluster-Anzahlen führen dann zu irreführenden Ergebnissen bei der Projektion der Cluster-Zentren ($\diamond$). Deshalb werden die zwei Cluster für diese Merkmale als nichtkompatible Cluster bezeichnet. Bei einer Vielzahl solcher unterschiedlicher Zusammenhänge in hochdimensionalen Datenräumen führt das u.U. zu sehr hohen Cluster-Anzahlen, die mit einer limitierten Anzahl von Datensätzen N nicht valide ermittelbar sind und dem Ziel einer Abstraktion widersprechen.

Einige Metriken verursachen Probleme bei korrelierten und z.T. informationslosen Merkmalen. Bild 2c illustriert das für zwei korrelierte Merkmale, bei denen die Mahalanobis-Distanz ($\circ$) im Gegensatz zur Euklidischen Distanz ($\diamond$) die Cluster-Zentren „zusammenzieht“. Ähnlich wirken zusätzliche informationslose Merkmale (Bild 2d), die für größere Distanzen zum jeweiligen Cluster-Zentrum sorgen. Das führt beim FCM-Algorithmus zu weniger ausgeprägten Zugehörigkeiten, weshalb auch entfernte Datensätze die Cluster-Zentren beeinflussen und diese somit in Richtung eines gemeinsamen Schwerpunktes verschieben ($\diamond$). Bei Metriken wie beim GK- und GG-Algorithmus ($\star$) oder separatem Clustern über einzelne Merkmale tritt dieser Effekt kaum auf.

4 Vergleiche der Verfahren

4.1 Datensatz und Validierungskennzahlen

Die in Abschnitt 3 vorgestellten Verfahren werden nun anhand realer Daten verglichen. Dazu ist zunächst jeweils nur *ein* Parameter variabel, wie die Distanz, der Merkmalsraum oder die Cluster-Anzahl. Später werden dann mehrere Parameter bei vergrößertem Suchraum (mehrere Zeitreihen) variiert. Dabei werden hier beispielhaft die Kniebeugung (sagittale Knie-Zeitreihe), die Beugung des oberen Sprunggelenks (sagittale OSG-Zeitreihe) und deren normierte Winkelgeschwindigkeit (sagittale OSG-Geschwindigkeitszeitreihe) sowie die seitliche Neigung des Oberkörpers (frontale Becken-Zeitreihe) analysiert.

Die $N = 106$ Datensätze stammen von einer Referenzgruppe (20 Datensätze) und einer ICP-Patientengruppe (ICP=Infantile Zerebralparese, 86 Datensätze). Dieses Krankheitsbild ist durch eine enorme Komplexität und ausgeprägte pathologische Ausfälle über mehrere Zeitreihen charakterisiert [1, 2, 5, 6]. Aus klinischer Sicht existieren wesentliche Unterschiede zwischen Referenzgruppe und Patientengruppe. Die Patienten weisen wiederum teilweise bekannte Subgruppen auf. Wenn die hier eingesetzten *unüberwachten* Verfahren ebenfalls diese Unterscheidungen finden, deutet das auf valide Cluster hin. Ausgewählte Forderungen in Abschnitt 2 werden durch die folgenden Kennzahlen ansatzweise formalisiert:

- die mittlere maximale Zugehörigkeit zu einem Cluster-Zentrum (MW_{maxZGH}, maximal für eindeutige Zugehörigkeiten der Datensätze),
- die minimale quadratische Euklidische Distanz zwischen zwei Cluster-Zentren ($d^2_{min,ZR} = \min\left(d^2(\mathbf{v}_c, \mathbf{v}_i)\right)$ mit $c, i = 1 \ldots C, c \neq i$, maximal für gut getrennte Cluster-Zentren),
- der Trennungsgrad S (minimal für eine geeignete Cluster-Anzahl und valide Cluster),
- die Anzahl zugeordneter Datensätze zu den Clustern (Anzahl DS, nicht zu klein zum Erzielen eines angemessenen Abstraktionsniveaus) und
- die maximale Zugehörigkeit des Mittelwerts der Referenzgruppe zu einem Cluster (ZGH_R, groß zum Finden dieser bekannten Gruppe).

4.2 Distanzen bei K Abtastzeitpunkten als Merkmale

Anhand der sagittalen Knie-Zeitreihe mit $K = 101$ Abtastzeitpunkten als Merkmalsraum und vier Cluster-Zentren trennt lediglich der FCM-Algorithmus mit Euklidischer Distanz die Datensätze in klinisch plausible Cluster auf (Bild 3a).

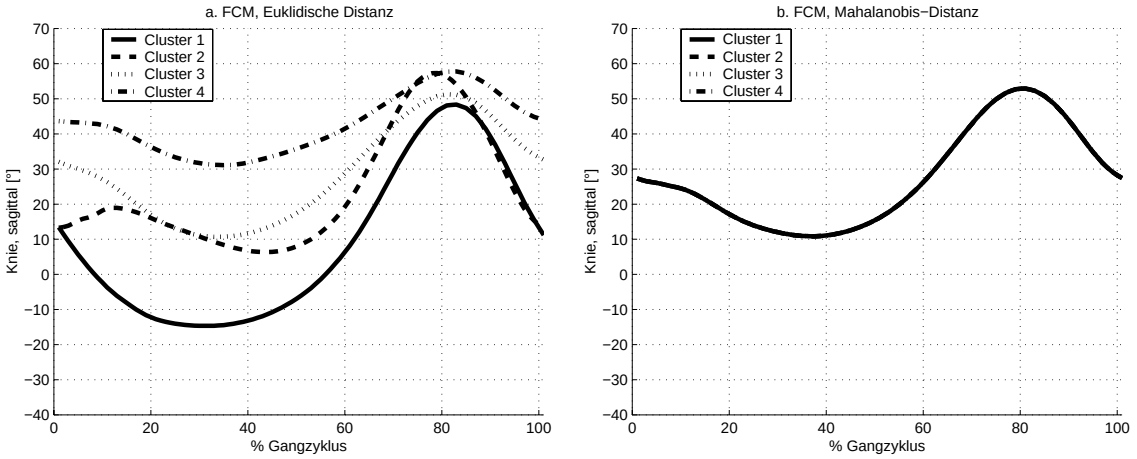


Bild 3: Cluster-Ergebnis durch a. FCM-Algorithmus mit Euklidischer Distanz und b. FCM-Algorithmus mit Mahalanobis-Distanz für die sagittale Knie-Zeitreihe

Probleme beim FCM-Algorithmus mit Mahalanobis-Distanz, GK- und GG-Algorithmus resultieren aus Problemen bei der Inversion von Kovarianzmatrizen zur Berechnung der Distanzen. Die vergleichsweise geringe Anzahl von Datensätzen führt hier zu einem Rangabfall der Kovarianzmatrizen, der beim Invertieren nur durch Berechnung von Pseudoinversen zu behandeln ist. Dadurch werden alle Datensätze zu den Clustern nahezu gleichverteilt zugeordnet (Bild 3b und Tabelle 1). Die Zentren liegen somit beim gemeinsamen Mittelwert über alle Datensätze.[1]

d_c	**X**	C	S	ZGH_R	MW_{maxZGH}	Anzahl DS	$d^2_{min,ZR}$	Bild
EU	= 101 ZR	4	15.21	0.92	0.69 0.67 0.59 0.63	11 26 45 24	7588.22	a.
MA	= 101 ZR	4	$5 * 10^8$	0.25	0.25 0.25 0.25 0.25	27 27 26 26	$8 * 10^{-7}$	b.
GK	= 101 ZR	4	6.62	0.25	0.25 0.25 0.25 0.25	27 27 26 26	10^{-10}	c.
GG	= 101 ZR	4	6.62	0.25	0.25 0.25 0.25 0.25	27 27 26 26	10^{-10}	d.
EU	= 2 HKA	4	9.24	0.97	0.770.73 0.72 0.71	11 37 35 23	8319.63	–

Tabelle 1: Vergleich verschiedener Distanzen d_c für die sagittale Knie-Zeitreihe, vgl. Bild 3 ($s = K$ Abtastzeitpunkte bzw. dessen Transformation mit $s = 2$ Merkmalen, vgl. Abschnitt 4.3). Die numerischen Probleme auf Grund nahezu gleicher ZGH bei Mahalanobis-Distanz, GK- und GG-Algorithmus wurden geeignet behandelt.

Die mit dem FCM-Algorithmus und Euklidischer Distanz gefundenen Cluster stimmen mit etablierten klinischen Subgruppen überein [1]. Dabei entspricht „Clu-

[1]Darüberhinaus gibt es bereits beim Clustern von Zeitreihenabschnitten mit wenigen Abtastzeitpunkten (ab $K = 3..5$) Probleme durch die Korrelationen benachbarter Abtastzeitpunkte. Zwar können (abschnittsweise) die Cluster durch unterschiedliche Steigungen in den Zeitreihen charakterisiert sein, allerdings dominiert der Rauschanteil. Der FCM-Algorithmus mit Mahalanobis-Distanz, GK- und GG-Algorithmus verstärken diese (Ko-)Varianzen und so den Rauschanteil beim Clustern. Die Auswirkungen ähneln Bild 3b und Tabelle 1 mit Clustern beim gemeinsamen Mittelwert über alle Datensätze.

ster 2“ nahezu dem Mittelwert der Referenzgruppe, „Cluster 1“ dem Recurvatum (Knieüberstreckung), „Cluster 3 und 4“ einem starken und milden Kauergang (crouch gait) [1, 5].

4.3 Vergleich Merkmalsraum (K Abtastzeitpunkte gegen zwei HKA-Merkmale)

Da nur der FCM-Algorithmus mit Euklidischer Distanz bei Zeitreihen gute Ergebnisse liefert, wird dieser beim Vergleich von Zeitreihen mit $s = K = 101$ Abtastzeitpunkten und deren dimensionsreduzierende Transformation durch die HKA gemäß Abschnitt 3.3 mit $s = 2$ Einzelmerkmalen verwendet. Beide Ansätze liefern beim Clustern des Beispiels aus Abschnitt 4.2 vergleichbar gute Ergebnisse, siehe HKA, EU in Tabelle 1.

Ein komplexeres Beispiel ist bei Vorgabe von $C = 4$ Cluster-Zentren die normierte Geschwindigkeits-Zeitreihe des oberen Sprunggelenks (OSG, sagittal GZR), die „versteckte“ Subgruppen enthält. Versteckte Subgruppen sind dadurch charakterisiert, dass sie sich über große Bereiche in der Zeitreihe kaum voneinander unterscheiden, z.B. „Cluster 1, 3 und 4“ zwischen 20% bis 50% des Gangzyklus in Bild 4. Die vorhandenen Unterschiede zwischen Datensätzen in diesem Bereich wirken folglich wie Rauscheffekte, vgl. Bild 2.

Das Cluster-Ergebnis mit K Abtastzeitpunkten im Zeitreihen-Merkmalsraum trennt lediglich die Referenzgruppe von allen Patienten, siehe Bild 4a und $ZGH_R = 0.98$ zu „Cluster 2“ in Tabelle 2. Weiterhin sind die maximalen ZGH zu „Cluster 2“ relativ hoch (Bild 4d, Mittelwert $MW_{maxZGH} = 0.68$), was für eine gute Trennung dieses Clusters steht. Das Cluster enthält 19 von 20 Datensätzen der Referenzpersonen in den 23 zugeordneten Datensätzen. Die Patientendaten werden allerdings zu den restlichen drei Clustern nahezu gleichverteilt zugeordnet ($MW_{maxZGH} \approx \frac{1}{3}$ in Tabelle 2). Dadurch sind diese drei Cluster-Zentren eng benachbart, was zu einer kleinen minimalen quadratischen Euklidischen Distanz in Zeitreihen $d^2_{min,ZR} = 0.07$ führt.

d_c	**X**	C	S	ZGH_R	MW_{maxZGH}	Anzahl DS	$d^2_{min,ZR}$	Bild
EU	= 101 ZR	4	6401.65	0.98	0.34 0.68 0.34 0.34	42 23 34 7	0.07	a.
EU	= 2 HKA	4	8.24	0.99	0.85 0.72 0.66 0.68	22 31 25 28	23.41	b.

Tabelle 2: Vergleich von Cluster-Ergebnissen in Zeitreihen (ZR) und in deren transformiertem Raum durch die HKA für die sagittale OSG-Geschwindigkeitszeitreihe, siehe auch Bild 4

Die Trennung der Patienten in Subgruppen ist durch Verwendung der HKA mit zwei Merkmalen besser. Deren Interpretation ist möglich, wenn mit den Zugehörigkeiten nach abgeschlossener Optimierung die Cluster-Zentren mit $Z_{ij}[k]$ für die Zeitreihen berechnet werden, siehe Bild 4b. Hier sind die mittleren ZGH zu den einzelnen Cluster-Zentren relativ groß ($MW_{maxZGH} \gg \frac{1}{C}$, Tabelle 2 und Bild 4d). Die HKA bietet somit Vorteile bei „versteckten“ Gruppierungen, weil Bereiche in ZR, in denen sich die Gruppen nicht unterscheiden (und eher wenig streuen), durch die HKA unterdrückt werden.

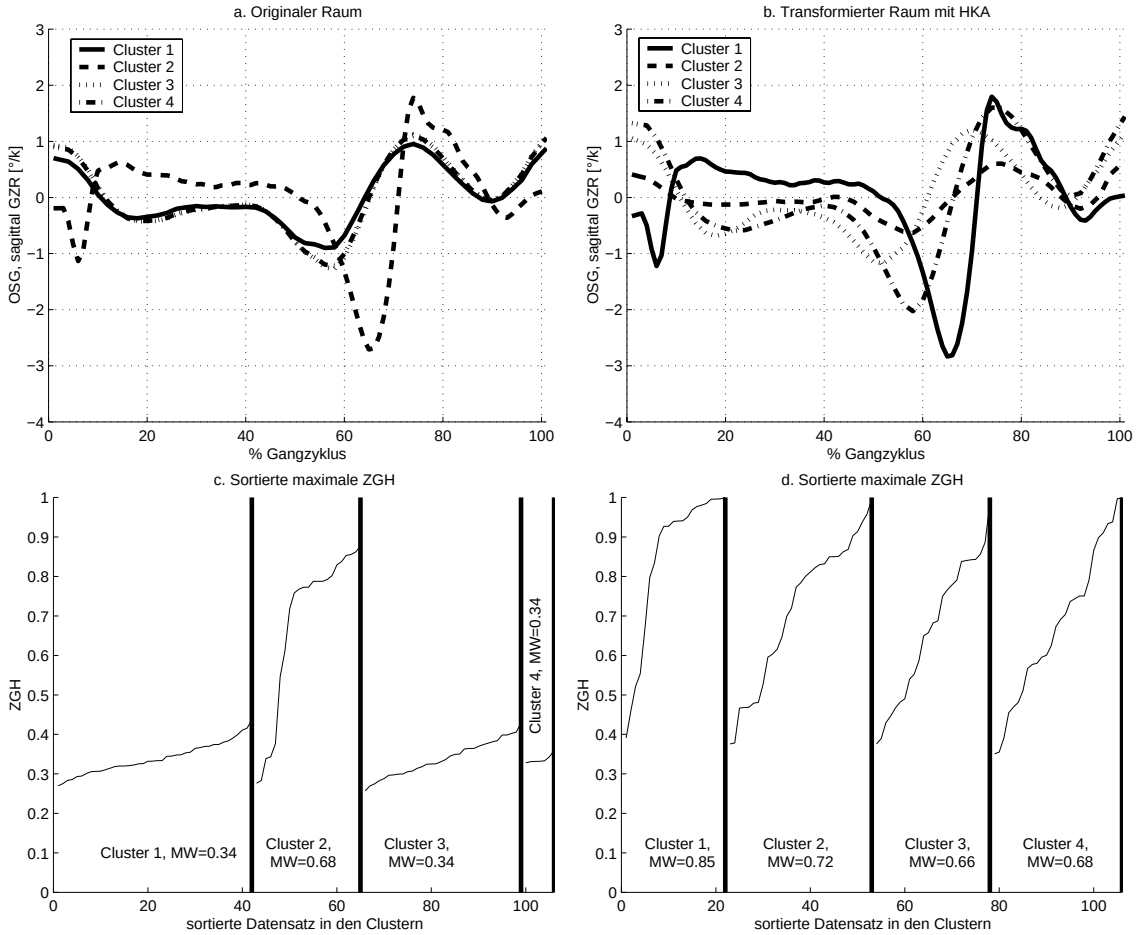


Bild 4: Cluster-Ergebnisse für die sagittale OSG-Geschwindigkeitszeitreihe in a. und c. ZR-Merkmalsraum ($s = K = 101$) bzw. in b. und d. im HKA-Merkmalsraum ($s = 2$), wobei die Darstellung der Cluster-Zentren als Zeitreihen durch die ZGH erfolgt.

4.4 Bestimmung Cluster-Anzahl bei zwei HKA-Merkmalen

In den bisherigen Abschnitten wurde die Cluster-Anzahl fest gesetzt. Im Folgenden soll nun versucht werden, mit den eingeführten Kennzahlen eine optimale Cluster-Anzahl C zu finden. Hierbei wird zunächst die Euklidische Distanz mit zwei HKA-Merkmalen verwendet und anschließend mit anderen Distanzen verglichen.

Die Ergebnisse für die sagittale Knie-Zeitreihe zeigt Tabelle 3. Die Literaturempfehlung, das erste lokale Minimum des Trennungsgrades $S(C)$ (hier $C = 3$) zu verwenden, zeigt bezüglich der mittleren maximalen Zugehörigkeiten und der Anzahl der zugeordneten Datensätze sowie der Trennung der Cluster-Zentren befriedigende Ergebnisse. Allerdings findet diese Zuordnung die Referenzgruppe nur unzureichend (0.85), was sich auch an einer deutlichen Abweichung des entsprechenden „Clusters 2“ vom Referenzverlauf äußert (Bild 5a).

Der Grund für eine derart „falsche“ Einschätzung durch den Trennungsgrad liegt darin, dass mitunter das Ergebnis durch sehr eindeutige Gruppen (wie die der Referenz) verfälscht wird. Diese eindeutige Gruppe verursacht hohe ZGH. In der Nähe liegende, aber abweichende Datensätze von Patienten beeinflussen die Lage des Cluster-Zentrums insbesondere bei einer niedrigen Cluster-Anzahl. Der Trennungsgrad bevorzugt nun hohe ZGH bei niedriger Cluster-Anzahl mit der Folge, dass

d_c	$\mathbf{x}$	C	S	ZGH_R	MW_{maxZGH}	Anzahl DS	$d^2_{min,ZR}$	Bild
EU	= 2 HKA	2	12.79	0.92	0.81 0.83	63 43	29633.67	-
EU	= 2 HKA	3	5.65	0.85	0.80 0.79 0.80	13 64 29	25306.88	-
EU	= 2 HKA	4	9.24	0.97	0.77 0.73 0.72 0.71	11 37 35 23	8319.63	-
EU	= 2 HKA	5	5.53	0.96	0.73 0.74 0.66 0.72 0.81	11 29 35 21 10	8391.59	a.
EU	= 2 HKA	6	6.42	0.71	0.67 0.80 0.68 0.58 0.71 0.77	11 17 24 25 19 10	4644.69	-
EU	= 2 HKA	7	5.92	0.92	0.61 0.71 0.70 0.66 0.54 0.69 0.74	10 7 16 20 24 19 10	4136.41	-
MA	= 2 HKA	2	19.88	0.97	0.82 0.77	54 52	8808.20	-
MA	= 2 HKA	3	7.53	0.99	0.70 0.78 0.69	31 43 32	8734.90	-
MA	= 2 HKA	4	6.91	0.99	0.66 0.74 0.64 0.67	16 36 24 30	8950.80	-
MA	= 2 HKA	5	6.31	0.93	0.64 0.75 0.61 0.66 0.61	8 24 30 26 18	4915.86	-
MA	= 2 HKA	6	2.84	0.96	0.68 0.70 0.64 0.64 0.62 0.64	7 28 18 27 13 13	6191.34	b.
MA	= 2 HKA	7	4.25	0.70	0.65 0.69 0.71 0.67 0.64 0.58 0.58	7 20 18 14 23 11 13	3372.57	-
GK	= 2 HKA	2	10.71	0.99	0.85 0.82	60 46	7874.67	-
GK	= 2 HKA	3	7.43	1.00	0.85 0.71 0.72	41 41 24	4644.46	-
GK	= 2 HKA	4	5.08	0.98	0.79 0.69 0.76 0.67	33 27 25 21	2832.97	c.
GK	= 2 HKA	5	14.60	0.75	0.91 0.64 0.77 0.70 0.70	5 33 25 26 17	1644.25	-
GK	= 2 HKA	6	8.35	0.86	0.90 0.69 0.71 0.76 0.65 0.67	5 18 28 19 20 16	1788.14	-
GK	= 2 HKA	7	7.43	0.82	0.85 0.76 0.64 0.66 0.58 0.72 0.73	5 18 25 12 22 13 11	3850.51	-
GG	= 2 HKA	2	789.05	1.00	0.99 0.97	1 105	72248.62	-
GG	= 2 HKA	3	89.21	1.00	0.87 0.93 0.90	4 48 54	6309.23	-
GG	= 2 HKA	4	7.63	1.00	0.94 0.86 0.95 0.84	3 20 74 9	9024.21	d.
GG	= 2 HKA	5	6.81	0.53	0.89 0.80 0.73 0.74 0.81	3 12 41 38 12	6584.36	-
GG	= 2 HKA	6	0.07	1.00	0.81 0.71 0.00 0.00 0.99 0.99	2 2 0 0 2 100	8176.74	-
GG	= 2 HKA	7	0.07	1.00	1.00 0.68 0.00 0.00 0.00 0.99 1.00	2 2 0 0 0 2 100	5330.36	-

Tabelle 3: Ergebnisse der Algorithmen anhand von zwei transformierten Merkmalen durch die HKA für die sagittale Knie-Zeitreihe, siehe Bild 6

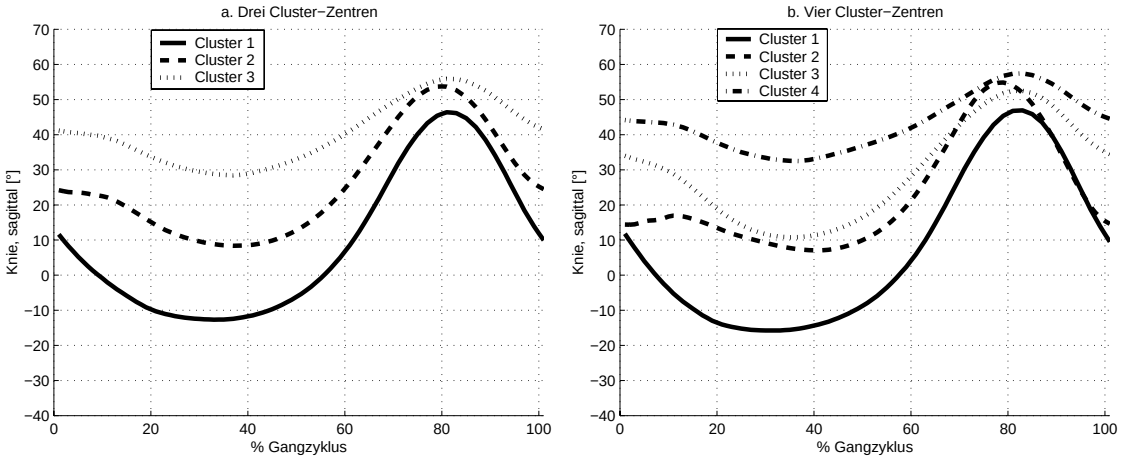


Bild 5: Vergleich von drei und vier Cluster-Zentren für die sagittale Knie-Zeitreihe

einige Patienten und die Referenz in einem Cluster (hier „Cluster 2“) beschrieben werden und das eigentliche Gangverhalten der beiden Gruppen vermischt wird. Werden bei diesem Beispiel *nur* die Patienten-Datensätze verwendet, so gibt der Trennungsgrad ein $C = 3$ vor, wodurch die drei beschriebenen Patienten-Subgruppen gut gefunden werden, vgl. Abschnitt 4.2. Für dieses Beispiel ist davon auszugehen, dass für den gesamten Datensatz mindestens $C = 4$ Cluster-Zentren vorhanden sind (mit dem klinischen Hintergrund, dass die Patienten abweichend vom Referenzverhalten laufen). Eine Erhöhung der Cluster-Anzahl auf $C = 4$ bzw. $C = 5$ verbessert die Kennzahl mit der Ähnlichkeit zur Referenz (Bild 5b bzw. 6a), aber reduziert den Abstand der Cluster-Zentren. Bei $C = 5$ beschreiben zwei Cluster-Zentren ein ähnliches orthopädisches Gangbild, siehe „Cluster 3 und 4“ in Bild 6. Ähnliche Effekte treten mitunter auch bei anderen Zeitreihen auf. Deshalb fällt es wegen der unterschiedlichen Zielstellungen in einem multikriteriellen Optimierungsproblem schwer, eine richtige Cluster-Anzahl zu empfehlen. Eine finale Entscheidung wird deshalb interaktiv getroffen.

Im Gegensatz zu den in Abschnitt 4.2 gezeigten Ergebnissen, liefern FCM-Algorithmus mit Mahalanobis-Distanz, GK- bzw. GG-Algorithmus im transformier-

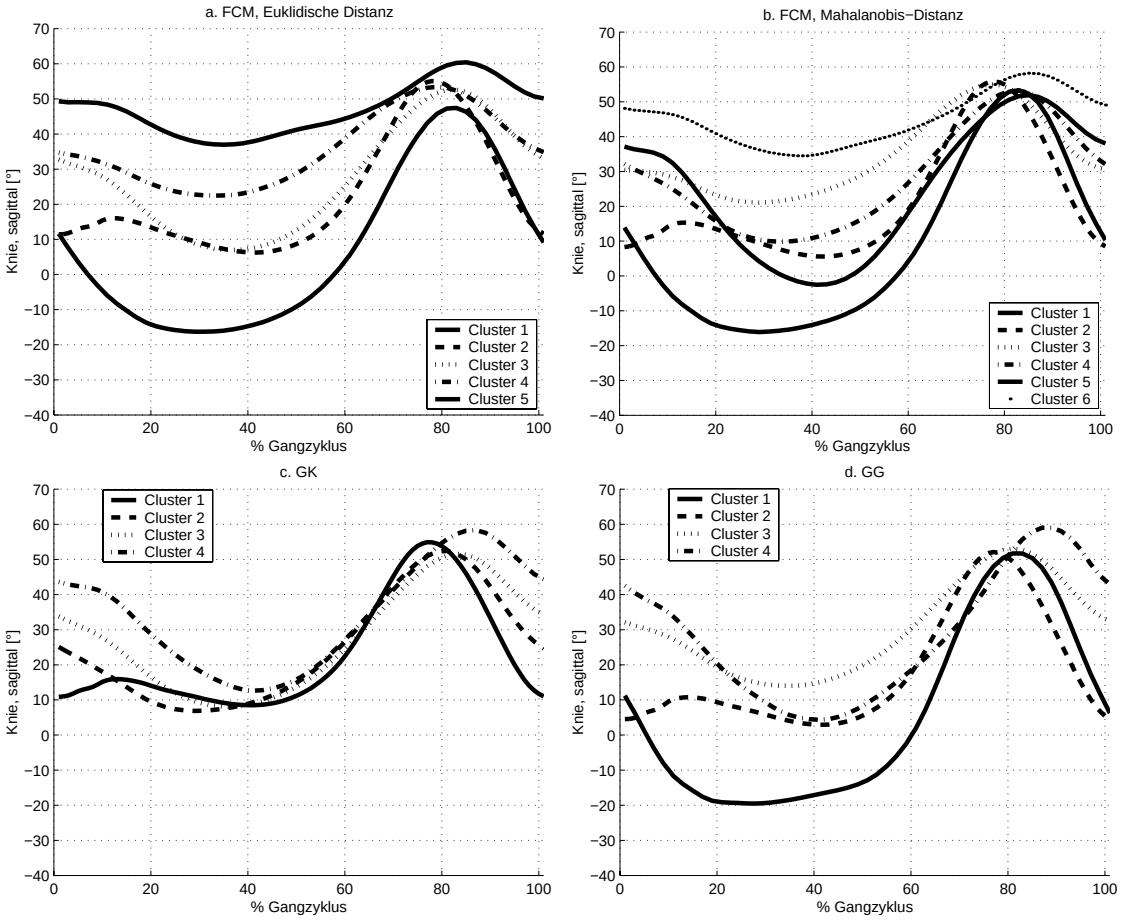


Bild 6: Visualisierung von Cluster-Zentren bei der sagittalen Knie-Zeitreihe

ten Merkmalsraum durch die HKA befriedigende Ergebnisse, siehe Tabelle 3 und Bild 6. Beim FCM-Algorithmus mit Mahalanobis-Distanz weist der Trennungsgrad auf eine Cluster-Anzahl von $C = 6$ hin, siehe Bild 6b. Allerdings beschreiben die Cluster-Zentren ähnliche Gangbilder. Eng benachbart sind insbesondere „Cluster 3 und 4“. Eine Reduzierung auf $C = 5$ verbessert die Situation der eng benachbarten Cluster-Zentren nicht. Eine weitere Reduzierung hat zur Folge, dass die bereits ermittelten Subgruppen nur unbefriedigend gefunden werden.

Der GK-Algorithmus findet bei $C = 4$ (anhand des Trennungsgrades) nicht die Patienten-Subgruppen (Bild 6c). Bei Erhöhung auf $C = 5$ wird die bekannte Gruppe „Referenz“ verfälscht. Zudem treten sehr eng benachbarte Cluster-Zentren auf. Der GG-Algorithmus liefert zwar auf den ersten Blick mit $C = 4$ „gute Clusterzentren“ („Cluster 2“ als Referenzverhalten, „Cluster 1, 3 und 4“ als bereits beschriebene Patienten-Subgruppen, vgl. Abschnitt 4.2). Allerdings ist die Verteilung der Datensätze ungleichmäßig. Für die Ganganalysedaten treten bei diesem Kriterium insbesondere beim GG-Algorithmus oftmals Probleme auf. Der Grund ist, dass der GG-Algorithmus versucht, sehr hohe ZGH zu erreichen. Somit bezieht er mitunter nur wenige Datensätze zu einem Cluster ein.

Das Ziel der Hauptkomponentenanalyse ist, anhand der maximalen Streuung so viel Information wie möglich aus den Zeitreihen zu extrahieren. Der Informationsgehalt fällt somit bei den transformierten Merkmalen mit kleineren Eigenwerten ab. Die transformierten Merkmale haben einen unterschiedlichen Wertebereich entsprechend

ihrem Informationsgehalt (entlang der größten, zweitgrößten, usw. Streuung innerhalb der *normierten* Zeitreihen auf ein [-1,+1] Intervall). Die Mahalanobis-Distanz bewirkt eine einheitliche Skalierung der transformierten Merkmale und zerstört somit die Ziele der Transformation, indem sie weniger wichtige Merkmale verstärkt. Ähnliche Probleme resultieren durch die Fuzzy-Kovarianzmatrizen beim GK- und GG-Algorithmus. Die Ergebnisse hängen stark von den Start-Clustern ab, sind auf die Lerndatensätze angepasst und widersprechen somit der geforderten Generalisierung. Letzteres wird durch Berechnung der „Fuzzy-Kovarianzmatrizen“ entsprechend der Datenverteilung verursacht. Eine starke Abhängigkeit der Start-Cluster ist zudem ein Indiz, dass sehr viele Rausch-Effekte existieren (es können keine klaren Gruppierungen gefunden werden). Wie in Abschnitt 4.2 zu sehen, wächst der Rausch-Anteil mit der Anzahl der Hauptkomponenten, der mit diesen Algorithmen zudem (durch die einheitliche Skalierung) verstärkt wird. Insbesondere bei Zeitreihen-Kombinationen liefern diese Algorithmen nur spärliche Ergebnisse.

4.5 Erweiterung des Suchraums auf mehrere Zeitreihen: Simultanes Clustern

Das Problem der multikriteriellen Optimierung erweitert sich, wenn neben der Wahl der Cluster-Zentren auch besonders geeignete Kombinationen von Zeitreihen automatisiert gesucht werden. In der Ganganalyse ist es derzeit völlig unbekannt, welche Zeitreihen sich kombinieren lassen. Dieses Problem ist hier aus klinischer Sicht besonders von Interesse, weil unterschiedliche Gangmechanismen, weitläufigere Zusammenhänge oder Kompensationseffekte über einen erweiterten Suchraum bislang ungeklärt sind [5, 6]. Die Lösung dient einem besseren Verständnis von komplexen Pathologien bei der Diagnose und Therapieplanung.

Ein methodischer Vergleich zur Lösung des Problems ist anhand Kombination A (sagittale Knie-Zeitreihe und sagittale OSG-Zeitreihe) gegen Kombination B (sagittale Knie-Zeitreihe und frontale Becken-Zeitreihe) mit dem FCM-Algorithmus und Euklidischer Distanz gezeigt, siehe Bild 7 und Tabelle 4. Das Finden unterschiedlicher, unbekannter Cluster-Formen (typische Eigenschaften des GK- und GG-Algorithmus) wurde hier teilweise durch die Datenvorverarbeitung mit der Berechnung zusätzlicher Zeitreihen erreicht, siehe Abschnitt 3.3. Die Anwendung dieser methodischen Vorgehensweise mit einer klinischen Diskussion über alle Kombinationen ist in [14, 15] gegeben.

d_c	**X**	C	S	ZGH_R	MW_{maxZGH}	Anzahl DS	$d^2_{min,ZR}$	Bild
EU	A: = 4 HKA	2	16.07	0.95	0.76 0.81	41 65	57025.79	-
EU	A: = 4 HKA	3	11.67	0.96	0.66 0.65 0.77	22 38 46	35249.59	-
EU	A: = 4 HKA	4	10.45	0.97	0.58 0.58 0.59 0.73	20 23 25 38	23945.66	-
EU	A: = 4 HKA	5	6.52	0.98	0.51 0.61 0.57 0.71 0.59	13 17 24 35 17	23633.81	a.
EU	A: = 4 HKA	6	11.45	0.99	0.57 0.55 0.50 0.53 0.65 0.52	9 17 19 14 30 17	9888.43	-
EU	A: = 4 HKA	7	11.81	0.89	0.52 0.51 0.55 0.45 0.68 0.49 0.51	9 12 9 21 20 21 14	7836.88	-
EU	B: = 4 HKA	2	32.28	0.54	0.72 0.73	54 52	2278.89	-
EU	B: = 4 HKA	3	19.93	0.94	0.62 0.64 0.63	33 40 33	4676.30	b.
EU	B: = 4 HKA	4	25.26	0.52	0.49 0.54 0.57 0.56	27 28 25 26	4446.53	-
EU	B: = 4 HKA	5	23.79	0.44	0.45 0.49 0.45 0.51 0.47	21 24 21 16 24	1068.98	-
EU	B: = 4 HKA	6	16.71	0.61	0.45 0.46 0.45 0.50 0.43 0.45	19 13 19 17 18 20	1821.21	-
EU	B: = 4 HKA	7	20.47	0.53	0.53 0.46 0.44 0.47 0.42 0.40 0.49	5 12 18 20 20 19 12	2137.42	-

Tabelle 4: Ergebnisse beim simultanen Clustern mit Kombination A bzw. B

Die Gründe für die schlechten Ergebnisse bei Kombination B verdeutlicht Bild 8. Hierbei wurden zunächst für die sagittale Knie-Zeitreihe vier Cluster gesucht (Zentren: □ in Bild 8a). Beim Einzeichnen dieser Zugehörigkeiten in die Merkmale

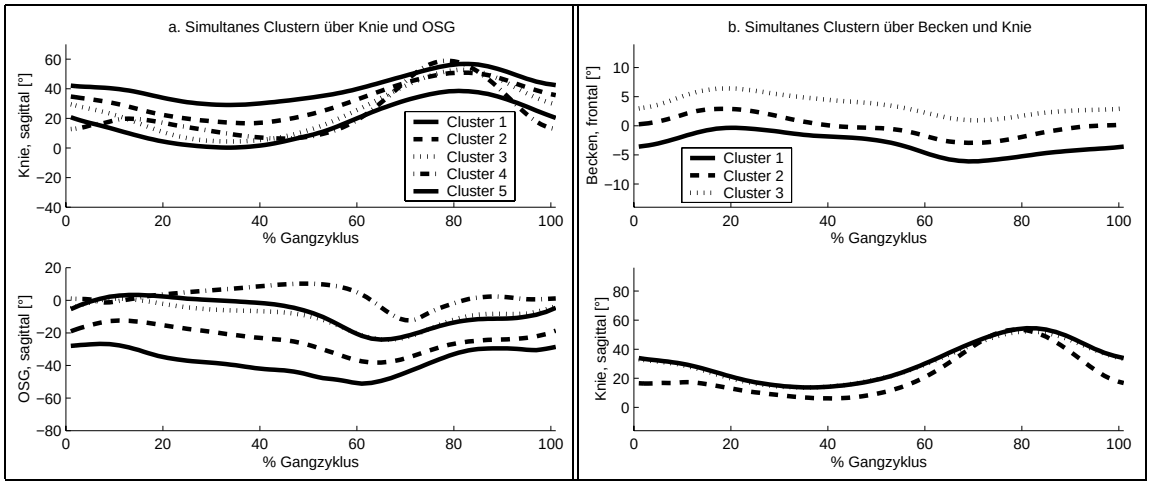


Bild 7: Simultanes Clustern über für a. die Kombination A (sagittale Knie- und OSG-Zeitreihe) und b. die Kombination B (sagittale Knie- und frontale Becken-Zeitreihe)

der frontalen Beckenzeitreihe ($\cdot, \circ, \times, +$ in Bild 8b) zeigt sich keine Struktur (etwaige Gleichverteilung), obwohl auch hier bei einem separaten Clustern ausgeprägte Cluster gefunden werden ($\triangleright$). Das deutet auf inkompatible Cluster entsprechend Abschnitt 3.5 hin. Ein simultanes Clustern verschiebt alle Cluster-Zentren in Richtung der Mittelwerte, ohne ausgeprägte Cluster zu finden ($\star$). Somit sinken auch die Zugehörigkeiten zu den simultanen Clustern im Vergleich zu den separaten Clustern, z.B. Datensatz DS 80: $u_{80,1} = 0.96$ (Knie), $u_{80,4} = 0.84$ (Becken), $u_{80,4} = 0.39$ (simultan Knie + Becken).

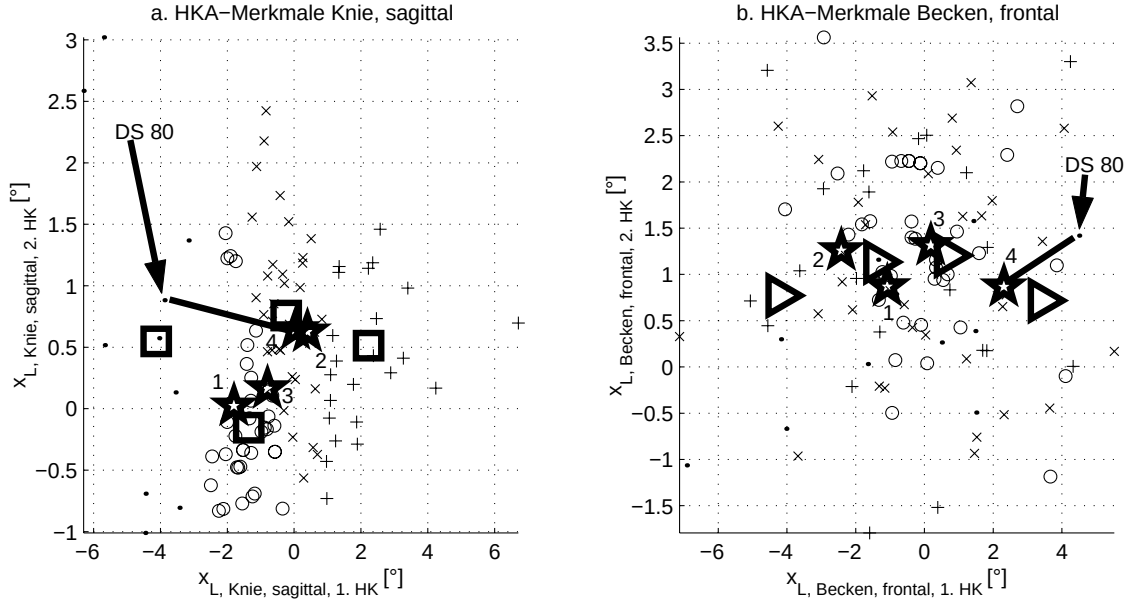


Bild 8: HKA-Merkmale für sagittale Knie-Zeitreihe (links) und frontale Becken-Zeitreihe (rechts) mit eingezeichneten Cluster-Zentren ($\square$: separat Knie, $\triangleright$: separat Becken, $\star$: Kombination B mit simultanem Clustern für Knie und Becken; $\cdot, \circ, \times, +$: Datensätze entsprechend maximaler Zugehörigkeit für separate Knie-Cluster)

Bei Darstellung der Cluster-Zentren als Zeitreihen (Bild 7b) wird dieser Effekt verdeutlicht. Bei $C = 3$ werden die bekannten Patienten-Subgruppen im Knie nicht

gefunden, d.h. die Cluster-Zentren liegen in der Nähe des Mittelwerts aller Patientendaten (ausgenommen von den Referenzdaten, da diese im Wesentlichen durch das „Cluster 2“ abgedeckt sind). Bei $C = 4..6$ werden entweder bekannte Subgruppen oder die Referenz nicht gefunden. Erst bei $C = 7$ werden sowohl die Patienten-Subgruppen als auch die Referenz durch die Form der Cluster-Zentren beschrieben. Allerdings sind die Cluster-Zentren eng benachbart und die Referenz wird durch zwei Cluster beschrieben (das erklärt den niedrigen Wert von $ZGH_R = 0.53$).

Besser sind die Ergebnisse bei Kombination A. Der Trennungsgrad weist auf eine Cluster-Anzahl von $C = 4$ hin. Mit dieser Anzahl sind die Patienten-Subgruppen im Knie einigermaßen gut abgebildet und Zusammenhänge zum sagittalen OSG beschrieben. Der Grund, dass das „Recurvatum“ nicht vollständig abgebildet ist, liegt an der starken Inhomogenität und Komplexität der Daten. Es existieren keine eindeutigen, klaren Zusammenhänge innerhalb der Daten, lediglich Tendenzen, die mit diesem Vorgehen aufgedeckt werden können. Eine Variation von C verbessert die Ergebnisse nicht (keine vollständige Erkennung der Patienten-Subgruppen im Knie bzw. eng benachbarte Cluster-Zentren).

5 Zusammenfassung und Ausblick

Das Ziel dieses Beitrags besteht in dem automatisierten Finden von zunächst unbekannten Gruppierungen in Zeitreihen am Beispiel der Instrumentellen Ganganalyse. Ähnliche Probleme (mühsame subjektive Datenanalyse, wenig etablierte, standardisierte Computermethoden) treten auch in anderen Bereichen auf, z.B. bei der Analyse von Verbrennungsvorgängen [16]. Auf derartigen komplexen Gebieten sind nur wenige empirische Gruppierungen ohne übergreifende Zusammenhänge bekannt. Zum Finden von Gruppierungen wurden Fuzzy-Cluster-Techniken erfolgreich angewandt. Hierzu wurden verschiedene Techniken betrachtet und der Fuzzy-C-Means-Algorithmus mit Euklidischer Distanz über lineartransformierte Zeitreihen mittels Hauptkomponentenanalyse (HKA) als vielversprechender Ansatz weiter verwendet. Mit den lineartransformierten Merkmalen konnten Vorteile erreicht werden, insbesondere wenn Zeitbereiche keine wesentlichen Unterschiede zwischen Subklassen aufweisen oder wenn sich bei mehreren Zeitreihen Subklassen überlagern. Dadurch lassen sich tendenziell auch weniger offensichtliche Zusammenhänge finden, der Rechenaufwand reduzieren und die numerische Robustheit verbessern. Deren Interpretation ist möglich, wenn mit den Zugehörigkeiten (ZGH) nach abgeschlossener Optimierung in den HKA-Merkmalen die Cluster-Zentren im Zeitreihen-Merkmalsraum berechnet werden.

Alle beschriebenen Untersuchungen entstanden als interaktiver Prozess zum Auffinden wesentlicher Zusammenhänge und nicht als vollautomatische Analyse. Dabei wird zunächst geprüft, mit welchen Parametrierungen Cluster-Algorithmen bekannte Zusammenhänge selbstständig finden, bevor sie für die Suche nach unbekannten Zusammenhängen eingesetzt werden. Da bisher keine vollständig befriedigende quantitative Bewertung der Cluster-Qualität existiert, ist ein interaktives Vorgehen aus automatischem Clustern, empirischem Validieren der Ergebnisse und eventuellem Modifizieren der Parameter bei der Suche nach unbekannten Zusammenhängen auch zukünftig erforderlich. Dabei sind sowohl Ärzte als auch

Datenanalyse-Experten einzubeziehen. Teilweise offene Fragen bestehen bei der Wahl der Cluster-Anzahl, wobei hilfreiche Elemente zur teilweisen Automatisierung vorgestellt wurden.

Methodische Ergänzungen sind z.B. bei

- der Merkmalsvorgabe (über mehrere Zeitreihen verbundene HKA-Merkmale, durch Diskriminanz-Analyse erzeugte Merkmale für ermittelte Subgruppen, Fourier-transformierte Merkmale [17]),
- der Merkmalsauswahl (automatisch gewählte Anzahl der HKA-Merkmale oder bestimmte Bereiche einer Zeitreihe),
- der Optimierung des Fuzzifizierungsgrades q,
- der erweiterten Darstellung von Cluster-Zentren (Einzugsbereich entsprechend der ZGH)und
- der Strategie zum simultanen Clustern (separates Clustern mit bereits gefundene Subgruppen in anderen Zeitreihen für größere Datensätze)

möglich. Gefundene Cluster lassen sich mit überwachten Verfahren charakterisieren. Ein Ansatz mit einer automatischen Generierung von Erklärungstexten ist in [4, 15] zu finden.

Die Autoren bedanken sich bei der DFG für die finanzielle Unterstützung der Forschungsarbeiten im Rahmen des Projektes „Diagnoseunterstützung in der Ganganalyse“ (GE 1139-1, BR 1303-6) und bei unseren Kollegen Rüdiger Rupp und Matthias Schablowski (Orthopädische Universitätsklinik Heidelberg) sowie Jens Jäkel und Lutz Gröll für zahlreiche Diskussionen.

Literatur

[1] Winters, T.; Gage, J.; Hicks, R.: Gait Patterns in Spastic Hemiplegia in Children and Young Adults. *Journal of Bone and Joint Surgery* 69 (1987), S. 437–441.

[2] O'Malley, M.; Abel, M.; Damiano, D.; Vaughan, C.: Fuzzy Clustering of Children with Cerebral Palsy Based on Temporal-Distance Gait Parameters. *IEEE Transactions on Rehabilitation Engineering* Vol. 5, No. 4 (1997), S. 300–309.

[3] Su, F.; Wu, W.; Cheng, Y.; Chou, Y.: Fuzzy Clustering of Gait Patterns of Patients After Ankle Arthrodesis Based on Kinematic Parameters. *Medical Engineering and Physics* 23 (2001), S. 83–90.

[4] Loose, T.; Jäkel, J.; Mikut, R.: Datenbasierte Generierung natürlichsprachlicher Erklärungstexte am Beispiel der Instrumentellen Ganganalyse. In: *Proc. 12. Workshop Fuzzy Systeme, Dortmund*, S. 43–57. Forschungszentrum Karlsruhe, FZKA 6767. 2002.

[5] Sutherland, D.; Kaufman, K.; Wyatt, M.; Chambers, H.; Mubarak, S.: Double-Blind Study of Botulinum A Toxin Injections Into the Gastrocnemius Muscle in Patients with Cerebral Palsy. *Gait and Posture* 10 (1999), S. 1–9.

[6] Chau, T.: A Review of Analytical Techniques for Gait Data. *Gait and Posture* 13 (2001), S. 49–66 (Part 1); 102–120 (Part 2).

[7] Bocklisch, S.: *Prozeßanalyse mit unscharfen Verfahren*. VEB Verlag Technik, Berlin. 1987.

[8] Höppner, F.; Klawonn, F.; Kruse, R.; Runkler, T.: *Fuzzy Cluster Analysis - Methods for Classification, Data Analysis and Image Recognition*. Wiley. 1999.

[9] Bezdek, J. C.: *Pattern Recognition with Fuzzy Objective Function Algorithms*. New York: Plenum Press. 1981.

[10] F. Klawonn, F. Höppner: An Alternative Approach to the Fuzzifier in Fuzzy Clustering to Obtain Better Clustering Results. In: *3rd Eusflat. EUSFLAT*, S. 730–734. Hochschule Zittau. 2003.

[11] Tatsuoka, M. M.: *Multivariate Analysis*. New York: Macmillan. 1988.

[12] Loose, T.; Malberg, H.; Mikut, R.; Dieterle, J.; Schablowski, M.; Wolf, S.; Abel, R.; Döderlein, L.; Rupp, R.: Ein modulares Verfahren zur automatisierten Auswertung von Ganganalysedaten. *Biomedizinische Technik* 47, Ergänzungsband 1 (2002), S. 700–703.

[13] Klawonn, F.: Visualisierung von Fuzzy-Clusteranalyse-Ergebnissen. In: *Proc. 12. Workshop Fuzzy Systeme, Dortmund*, S. 1–12. Forschungszentrum Karlsruhe, FZKA 6767. 2002.

[14] Loose, T.; Mikut, R.; Dieterle, J.; Rupp, R.; Abel, R.; Schablowski, M.; Gerner, H. J.; Bretthauer, G.: Automatisierte Bewegungscharakterisierung für Patienten mit inkompletter Querschnittlähmung. In: *AUTOMED 2003, 4. Workshop*, S. 20–21. Forschungszentrum Karlsruhe, FZKA 6875, http://bibliothek.fzk.de/zb/berichte/FZKA6875.pdf. 2003.

[15] Loose, T.; Dieterle, J.; Mikut, R.; Rupp, R.; Abel, R.; Schablowski, M.; Bretthauer, G.; Gerner, H. J.: Automatisierte Interpretation von Zeitreihen am Beispiel von klinischen Bewegungsanalysen. *Automatisierungstechnik (at)* (eingereichter Beitrag in 2003).

[16] Fick, A.; Keller, H.: Modellierung des Verhaltens Dynamischer Systeme mit erweiterten Fuzzyregeln. In: *Proc. 10. Workshop Fuzzy Control des GMA-FA 5.22, Dortmund*, S. 126–139. Forschungszentrum Karlsruhe, FZKA 6509. 2000.

[17] Beringer, J.; Hüllermeier, E.: Online Clustering of Data Streams. *Interner Bericht: Fachbereich Mathematik und Informatik AG Intelligente Systeme, Universität Marburg* (2003).

Sequence Processing with Recurrent Self-Organizing Maps Based Models

Volker Baier and Wilfried Brauer

Lehrstuhl für Theoretische Informatik und Künstlicher Intelligenz
Boltzmann Str. 3
D-85748 Garching/Munich
Tel.: (089) 28917213
Fax: (089) 28917207
E-Mail: volker.baier@cs.tum.edu

1 Introduction

Representation and processing of spatio-temporal information in different levels of granularity, or in other words, on several time scales, is the key topic of our work. We argue for using a multi-layered model consisting mainly of Recurrent Self-Organizing Maps and a feed-forward network for prediction. This paper introduces the core processing structure of our biologically inspired spatio-temporal motion processing model. The model is self contained and can be used for motion planning and prediction as well as all kind of context-sensitive information processing with prediction.

2 Recurrent Self-Organizing Maps

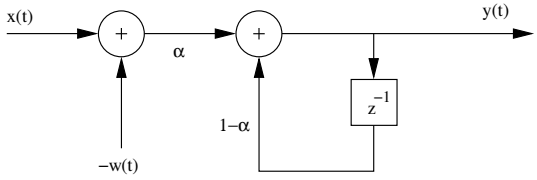
The original Self-Organizing Map (SOM) algorithm of Kohonen [4] was altered and extended in many ways. One rather interesting extension is the introduction of time. Most of the present work regarding temporal information is related to time-series processing and prediction. There are several ways to introduce time into a SOM-like network. For example, one can add a special structure like a shift-register to form a temporal memory for each neuron [12], or the activation of each neuron can be fed back to its input [5]. The most biological way, at least in our opinion, is the feedback solution as illustrated in Figure (1). We focus on the recurrent Self-Organizing Map (RSOM) because of the similarity to the organization in the visual cortex of the human brain.

Revisiting the original algorithm of Kohonen we will now describe the basic elements of a Self-Organizing Map and its learning mechanism. Later on, temporal properties will be added to the formalism, and we will introduce the multi-layer structure of our model.

A SOM can be defined by a two-dimensional lattice with neurons placed on its junction points. The neurons are defined by their two-dimensional position vector $r_i \in A$, and their weight vector $w_i(n)$.

Training a SOM is based on an unsupervised scheme. Randomly chosen input vectors $x(n)$ are taken from the p-dimensional input space $\mathbf{X}$ and presented to the net. The adaptation of the neurons' weight vectors $w_i(n)$ at training step n to the

Figure 1: Signal flow diagram of a recurrent SOM neuron



input depends on their euclidian distance $||\cdot||_p$ to the current input vector $x(n)$. The output of a neuron $y_i(n)$ is:

$$y_i(n) \quad = \quad ||x(n) - w_i(n)||_p \tag{1}$$

One of the neurons having the smallest distance to $x(n)$ at iteration step n is called best matching neuron, its distance is denoted by $y_b(n)$.

$$y_b(n) = ||x(n) - w_b(n)||_p = \min_{i \in M}\{||x(n) - w_i(n)||_p\} \tag{2}$$

To generate a topology preserving representation of the input vectors, similar vectors have to be represented by neurons with a similar weight vector. Therefore a neighborhood function is introduced. The adaptation of the weights $w_i(n)$ of the neurons addressed by the position vector r_i in the neighborhood of the best matching neuron at position $r_b(n)$ decreases according to the neighborhood function:

$$N_{ib}(n) \quad = \quad exp(-||r_i - r_b(n)||_2^2/\sigma(n)^2) \tag{3}$$

$\sigma(n)$ is used to reduce the region around r_b in which the adaptation is strong. $||\cdot||_2$ is the distance defined on the two-dimensional SOM lattice. The weights of all neurons are adapted for the learning step $(n+1)$ by:

$$w_i(n+1) = w_i(n)\gamma(n)N_{ib}(n)((x(n) - w_i(n)) \tag{4}$$

By variating $\gamma(n)$ from $0 \leq \gamma(n) \leq 1$ one may regulate how far the weight vectors are moved in one step towards the current input vector. We want to emphasize, that this adaption phase is not dependent on an error measure - the system is unsupervised. Usually adaption is done for a predefined number of iterations.

To add time to the SOM model we consider the input values $x(t)$ as dependent on time t and add a feedback loop to each neuron which feeds, after a time-step, the activation of each neuron back to its input, but damped by a factor α, $0 < \alpha \leq 1$. Figure (1) shows the signal flow diagram of a single RSOM neuron.

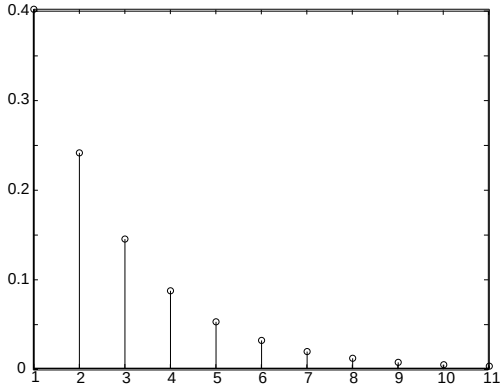
The output of a recurrent neuron is a combination of the damped output of the last time step, with the current output:

$$y_i(t) \quad = \quad (1-\alpha)y_i(t-1) + \alpha(x(t) - w_i(t)) \tag{5}$$

Thus we do not calculate the distance between input vector and weight vector, the best matching neuron b at time step t is the output vector with the smallest norm:

$$y_b(t) = \min_i\{||y_i(t)||_p\} \tag{6}$$

Figure 2: System response $\uparrow \alpha = 0.4 : \rightarrow t$



The temporal behavior of a neuron is determined by its response $h(k)$ for all $k \geq 0$ to one single input of maximal size, at time 0[1]. As only the recurrent feedback (see Figure 1) is relevant for the temporal behavior of the neuron, $w_i(t)$ is neglected. A normalized input is applied by setting the input value at time step $t = 0$ $x(0) = 1$. At every other time steps $t \neq 0$ the input is $x(t) = 0$. The resulting response $h(k)$ is the so called system response

$$\begin{aligned}
t = 0 : h(0) = &\quad (1-\alpha) \cdot 0 + \alpha \cdot 1 = &\quad \alpha \\
t = 1 : h(1) = &\quad (1-\alpha) \cdot \alpha + \alpha \cdot 0 = &\quad (1-\alpha) \cdot \alpha \\
t = 2 : h(2) = &\quad (1-\alpha)^2 \cdot \alpha + \alpha \cdot 0 = &\quad (1-\alpha)^2 \cdot \alpha \\
t = 3 : h(3) = &\quad (1-\alpha)^3 \cdot \alpha + \alpha \cdot 0 = &\quad (1-\alpha)^3 \cdot \alpha \\
&\quad \vdots \qquad\qquad \vdots &\quad \vdots \\
t = k : h(k) = &\quad &\quad \alpha(1-\alpha)^k
\end{aligned} \tag{7}$$

As one can see, a recurrent neuron reacts with a discretized exponential decrease of its output: Figure (2). The decreasing output values of all neurons are molding a temporally ordered output pattern, as sketched in Figure (4). Through this pattern one can access the spatial and the temporal properties simultaneously what we call an orthogonal input access. The demand of an orthogonal memory access was one key topic of the visual perception model of K. Schill and C. Zetzsche [10] we used as a foundation to develop our model.

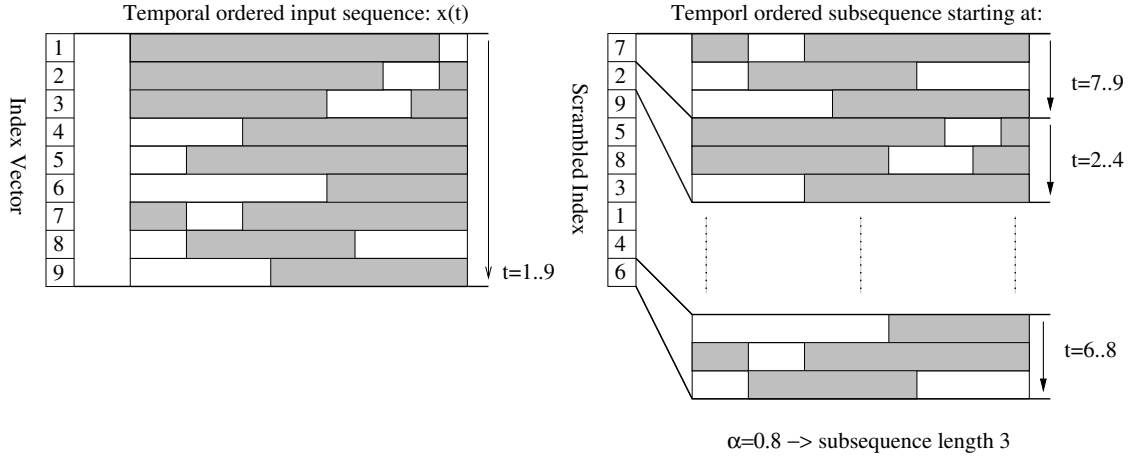
The weight adaptation is extended by replacing the simple distance measure by the recurrent output calculation in (5)

$$w_i(t+1) \quad = \quad w_i(t) + \gamma(t) N_{ib}(t)(y_i(t)) \tag{8}$$

To make the initial SOM network suitable for its task, the weights of the neurons have to be adapted to generate a mapping of the input vectors to the proper neurons. Training a non recurrent SOM is done by randomly choosing an input vector out of the training set, and moving the best matching neuron with its neighbors in direction to the input vector. Drawing the input vectors randomly out of the training set prevents the system from poor adaptation, e.g. a twisted net would

[1]For signal theory background see [8].

Figure 3:



lead to low generalization ability.
Using the RSOM model one can process temporal information. To do so, the ordering of the input values is of crucial importance. A simple random drawing of input values to prevent poor adaptation is therefore not suitable any more. By subdividing the input data in subsequences it is possible to realize good training results and also preserving the temporal structure of the input data. The system response is used, to determine the length of a subsequence. Due to the exponential decrease of the neurons output value cannot reach 0, thereby one has to set a threshold, i.e. five percent of the original output value. Within a subsequence the input vectors' temporal ordering is assured. By generating a permuted index vector it is possible to access the subsequences in a pseudo random fashion: Figure (3). This scheme of drawing the input vectors prevents the system from bad adaptation and also preserves the temporal relationship of the input.
The RSOM algorithm described in [5] was originally introduced for the purpose of time series prediction. Due to the decayed feedback of the neurons, one gets a temporally ordered pattern of past winning neurons. In this application a neuron represents a characteristic pattern of the given time series. The next section introduces our multi-layer model based on RSOM networks.

3 Spatio-Temporal RSOM

One key topic we are investigating with our neural network model is the representation, categorization and prediction of spatio-temporal sequences on different time scales. There are several multi-layer networks known which are capable of storing information for a certain period of time. They were often linguistically inspired and used for natural language processing (e.g. DSOM [12] or [11]).
The Adaptive Behavior Cognition (ABC) model of G. Briscoe [2] is a very general attempt of a cognition model that is able to simulate a part of the cognition apparatus of the human brain. ABC is more a proof of concept than a applicable model. We focused on the functional core of ABC, Learning and Processing Sequences (LAPS). LAPS is a multi-layer model, consisting of two standard SOMs,

and an additional feed-forward network for prediction purpose. Based on LAPS and the RSOM algorithm we developed a motion-perception model that is able to classify and predict spatio-temporal sequence information. The boundary conditions of the model and several design issues were constrained by our main aim, the simulation of the behavior of the visual-motion processing area (area MT or V5) of the human brain.

The over all structure of our model is shown in Figure (4). The RSOM layers are handled by the standard algorithm described above, apart the output value of the neurons. Since we try to base our model on biological facts, we calculate the maximum activation normalized by the dimension d of the weight vectors to simulate a biological neurons output:

$$y_i(t) = (1-\alpha)y_i(t-1) + \alpha \cdot \exp\left(-\frac{|x(t)-w_i(t)|}{\sigma^2 \cdot d}\right) \quad (9)$$

Thus the best matching neuron y_b is determined by the maximum activation:

$$y_b = \max(y_i) \quad (10)$$

The input for the next network layer $z(t)$ is generated by combining all calculated activations of the first map to one long vector.

$$z(t) = y_1(t) \oplus y_2(t) \oplus y_3(t) \oplus \ldots \oplus y_m(t) \quad (11)$$

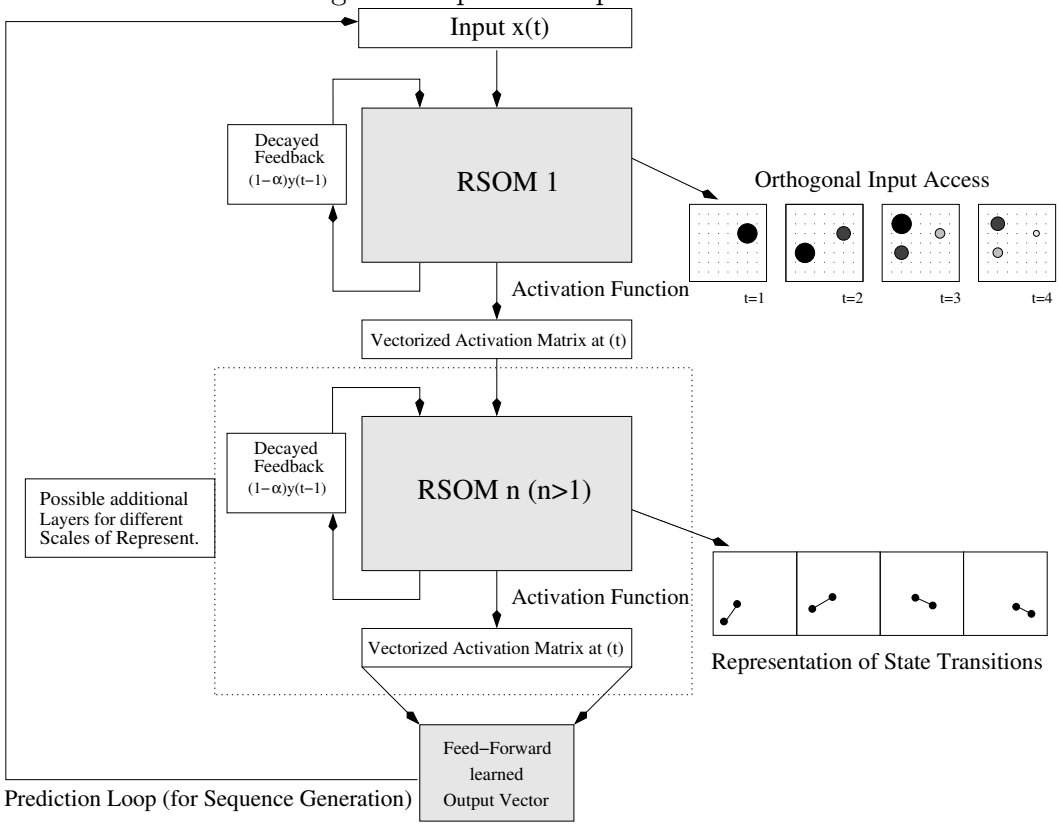
This vectorized activation matrix is generated dependent on the system response of the neurons in the RSOM layer. By waiting several time steps the past winning neurons generate a activation pattern. We propose, that If the activation value of a former winning neuron falls below five percent of the original activation value, the neuron has no temporal influence to future active neurons. The time span between the activation of a neuron and its activation falling below five percent delimits the number of input vectors composing a subsequence and is altered by α (9).

The time span between two generated input vectors for the auxiliary processing layer is determined indirectly by the system response (7). As we want to preserve the temporal structure of the input data, we have to choose the time span shorter than the subsequence length. This ensures an overlapping of the subsequences.

Prediction is realized by introducing an alternate network structure connected to the second RSOM layer. The structure is realized by a feed-forward net. It provides us a look ahead memory for the next most likely input vector. This prediction network is trained with standard back-propagation [3] to associate the current activation pattern of the second RSOM layer $z(t)$ (11) with the next upcoming input $x(t+1)$ the first RSOM layer will be presented.
The RSOM is a unsupervised trained network while the feed-forward net is self trained through the system. By the introduction of additional RSOM layers, it is possible to generate representations for coarser time scales. So you can represent motion primitives, like “left-turn”, “u-turn”, “right-turn” or whole motion plans by one neuron.

Several demands on a multi-layer model arose out of the properties of the task of visual motion-perception. First, the representation in different scales. Second a

Figure 4: Spatio-Temporal RSOM



memory structure that allows an orthogonal access of spatial and temporal information. Third, prediction capability. The first demand is fulfilled by the introduction of a multi-layer model. One key issue for the multi-layer model is the system response, introduced earlier. Before propagating the vectorized activation vector of the first layer to the second one, we wait for several time steps. The number of time steps is dependent on the feedback weighted by α in (5) and therefore on the system response (7). By connecting the prediction network to an other layer than the second one allows to alter the time scale of the prediction.

4 Application Domains

4.1 Motion-Perception

Imagine a visual stimulus represented by a single black dot, moving on a whiteboard. The RSOM would be able to predict the next position of the black dot, only if the direction keeps the same. Or the movement is within a contextual entity i.e. a movement in a bow. The first RSOM represents single states of the stimulus for example the direction and speed of the movement. To simulate a part of the visual motion-perception ability of the Brain, the model has to be able to extrapolate movement and to deal with problems like occlusions. An occlusion translated to the real world domain could be represented by a child running over the sidewalk, passing through two parked cars and reappearing on the street while the spectator sits in a

car traveling down the road. The saccadic eye movement would be to slow to search for the reappearing child in the whole scene. So a prediction mechanism is moving the eye to the point of reappearance of the child. This ability is also important to avoid approaching obstacles like a ball thrown in your direction. These examples give a good connection to robotics.

4.2 Robot Navigation

The robot navigation domain is essentially identical with the visual motion-perception domain. One might think of an intelligent camera controller for a museum-guide robot, that is able to track persons even if they are occluded completely by obstacles or other persons for several time frames.
By using an egocentric space representation, it is possible to classify and to predict robot motion-paths. This work is subject in an ongoing project at our institute. In this model, the first information processing layer represents the direction and the speed of the robot, quite similar to the first layer of the visual motion-processing model. The second layer represents action combinations on an abstraction layer like qualitative-motion vectors [7] <left>, <right>, <forward>.... With the prediction ability we can react to occlusions and approaching humans in a reactive manner.

4.3 Text Processing

One important feature of our model is that context related input vectors are represented in a manner that ensures context preserving in the RSOM. If you look on the activation of the neurons of one RSOM layer you can see a "trajectory" formed by the past activated neurons. In the first layer this trajectory would represent the single word. The semantic structure of a natural language sentence is represented by a higher processing layer RSOM. We suggested the use of our model for an automatic text analysis for neuro-anatomy related paper. As the information in these papers is encoded in complex semantic structures spreaded over several sentences and sections, we suggested the use of a multi-layer spatio-temporal RSOM to preserve these complex semantic structures in the networks. The system is providing a context aware index for the retrieval of complex information out of the preprocessed data. The system is currently under development.

5 Future Work

The introduced network model is applied to robot navigation model in a student project. We are also working on a parallel implementation of the model. Our main focus still is the simulation of the motion processing area MT of the human brain. For this purpose we have to simulate at least ten million neurons for an adequate representation. Computer clusters now have the dimension to realize this task in near future. Alternate memory structures to realize the prediction net are also work in progress. Using new insights of neuro-anatomy and cognitive psychology we will refine our model to get similar performance of the biological model and the artificial model. A performance comparison between the prediction net attached

to different layers will be done to balance computational complexity with good prediction abilities.

References

[1] V. Baier, K. Schill, F. Röhrbein, W. Brauer: *Processing of spatio-temporal structures: A hierarchical neural network model,* Proc. Artificial Intell., C. Freksa (ed) Dynamische Perzeption, G. Baratoff, H. Neumann (Hrsg.). Workshop der GI-Fachgruppe Bildverstehen, pp. 223-226, AKA, Akad. Verl.-Ges., Berlin 2000.

[2] G. Briscoe: *Adaptive Behavioural Cognition,* PhD Thesis, Curtin University, 1997

[3] S. Haykin: *Neural Networks: A Comprehensive Foundation,* Prentice Hall Int., 1994

[4] T. Kohonen: *Self-Organizing Maps,* Springer, 1997

[5] T. Koskela, M. Varsta, J. Heikkonen and K. Kaski: *Time Series Prediction with Recurrent Self-Organizing Maps*

[6] T. Koskela, M. Varsta, J. Heikkonen and K. Kaski: *Temporal Sequence Processing using Recurrent SOM*

[7] A. Musto, K. Stein, A. Eisenkolb, Th. Röfer, W. Brauer and K. Schill: *From motion observation to qualitative motion representation.* In C. Habel, C. Freksa and K. Wender, editors, Spatial Cognition II, pages 115-126. Lecture Notes in Artificial Intelligence, 2000.

[8] J.G. Proakis and D. G. Manolakis: *Digital Signal Processing: Principles, Algorithms, and Applications.*, MacMillan Publishing Company, 1992

[9] K. Schill, V. Baier, F. Röhrbein and W. Brauer: *A hierarchical network model for the analysis of human spatio-temporal information processing.* In Rogowitz, B. E. and Pappas, T. N., editors, Proceedings of SPIE The International Society for Optical Engineering, volume 4299, pages 615-621. Inst. für Medizinische Psychologie, Ludwig-Maximilians-Universität. 2001

[10] K. Schill and C. Zetzsche: *A model of visual spatio-temporal memory: The icon revisited,* Psychological Research, 57:88-102, 1995

[11] J.C. Scholtes: *Recurrent Kohonen Self-Organization in Natural Language Processing,* Artificial Neural Networks, T. Kohonen, K. Mäkisara, O. Simula and J. Kangas (Editors), Elsevier Science Publishers B. V. (North-Holland), 1991

[12] C. M. Privitera and L. Shastri: *A DSOM hierarchical model for reflexive processing: an application to visual trajectory classification,* ICANN, 1996

Anwendung eines Neuronalen Netzes zur Fließkurven-bestimmung für das Setup von Walzwerken

Dr.-Ing. Andreas Hambrecht, Thomas Klawon, Nicolas Prüfer

ALSTOM Power Conversion GmbH
Culemeyerstr. 1, 12277 Berlin
Tel. (030) 7622-3751
Fax (030) 7622-3700
E-Mail: thomas.klawon@tde.alstom.com

Prof. Dr.-Ing. Michael Dlabka

FH der Deutschen Telekom AG Leipzig und FHTW Berlin
Fritschestr. 60, 10627 Berlin
Tel. (030) 3121734
E-Mail: michael.dlabka@t-online.de

1 Einleitung

Um bei kaltgewalzten Blechen eine hohe Qualität zu erreichen, ist die Erstellung von möglichst genauen Stichplänen (Beispiel Abb. 1) zur Voreinstellung der Walzstraße erforderlich. Ein Problem ist die Verteilung der Dickenabnahme auf die Walzgerüste. Dabei ist die Kenntnis der auftretenden Walzkräfte von großer Bedeutung.

Der Zusammenhang von Dickenabnahme und auftretender Walzkraft wird durch die Fließkurve beschrieben. Dieser Zusammenhang ist materialabhängig und sehr komplex. Momentan existieren keine brauchbaren physikalischen Modelle, die den Einfluß der Legierungsbestandteile einer Stahlsorte auf die Fließkurve in allen Abhängigkeiten quantitativ ausreichend genau beschreiben können. Zur Lösung dieses Problems für das Kaltwalzen wurde die Verwendung eines Neuronalen Netzes untersucht, das auch für Materialien bisher unbekannter Zusammensetzung verlässliche Walzkraftvorhersagen erlauben würde. Für das Warmwalzen wird die Grundumformfestigkeit aus relevanten Legierungsbestandteilen ermittelt sowie eine Klassierung von Stahlgruppen mit ähnlichen Eigenschaften getroffen und mit Hilfe eines Neuronalen Netzes die Fließkurvenparameter bestimmt.

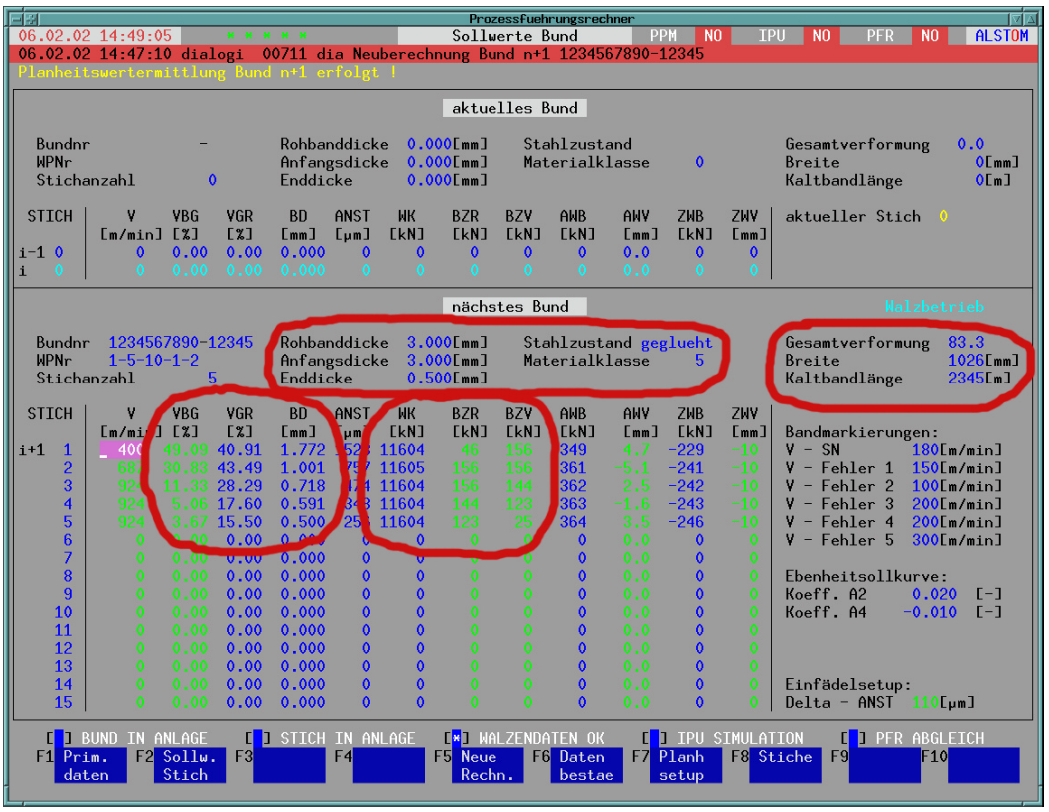


Abb. 1: Beispiel für eine Stichplanrechnung

2 Problemstellung

2.1 Formänderungsfestigkeit und Fließkurve

Die Formänderungsfestigkeit k_f ist eine Werkstoffkenngröße. Sie ist die Spannung des Werkstoffes, die aufgebracht werden muss, um den Werkstoff zum Fließen zu bringen. Unter Fließen versteht man eine plastische Formänderung. Ein Körper beginnt genau dann zu fließen, wenn eine Kombination der auf ihn einwirkenden Spannungen den kritischen Wert der Formänderungsfestigkeit k_f erreicht. Die Werkstoffkenngröße k_f kann z. B. im Versuch ermittelt werden. Sie ist beim Umformen von Metallen von dem Werkstoff, der Formänderung φ, der Formänderungsgeschwindigkeit und der Temperatur abhängig. Beim Kaltumformen überwiegt der Einfluss der Formänderung. Die Formänderung φ kann durch

$$j_{ges} = \ln\left(\frac{h_0}{h_i}\right)$$

ausgedrückt werden, wobei h_0 die Einlaufdicke (vor dem Walzgerüst) und h_i die Auslaufdicke nach dem i-ten Walzgerüst ist.

Die Fließkurve k_{fm} beschreibt die Formänderungsfestigkeit k_f in Abhängigkeit von der Formänderung φ. Nach dem Fließkurvenansatz von Ludwik kann die Fließkurve durch

$$k_{fm}(j) = k \cdot j^{n}$$

genähert werden [1].

2.2 Bestimmung der Fließkurvenparameter aus dem Walzprozess

Da die Formänderungsfestigkeit k_f und damit die Fließkurve k_{fm} materialabhängig ist, besteht die Möglichkeit, die Parameter k und n der Fließkurve k_{fm} in Beziehung zu setzen. Jedoch können die Parameter k und n nicht direkt gemessen werden. Um sie dennoch möglichst prozessnah zu ermitteln, wurden sie über eine Rückrechnung und Regression aus Messdaten ermittelt (Abb. 2).

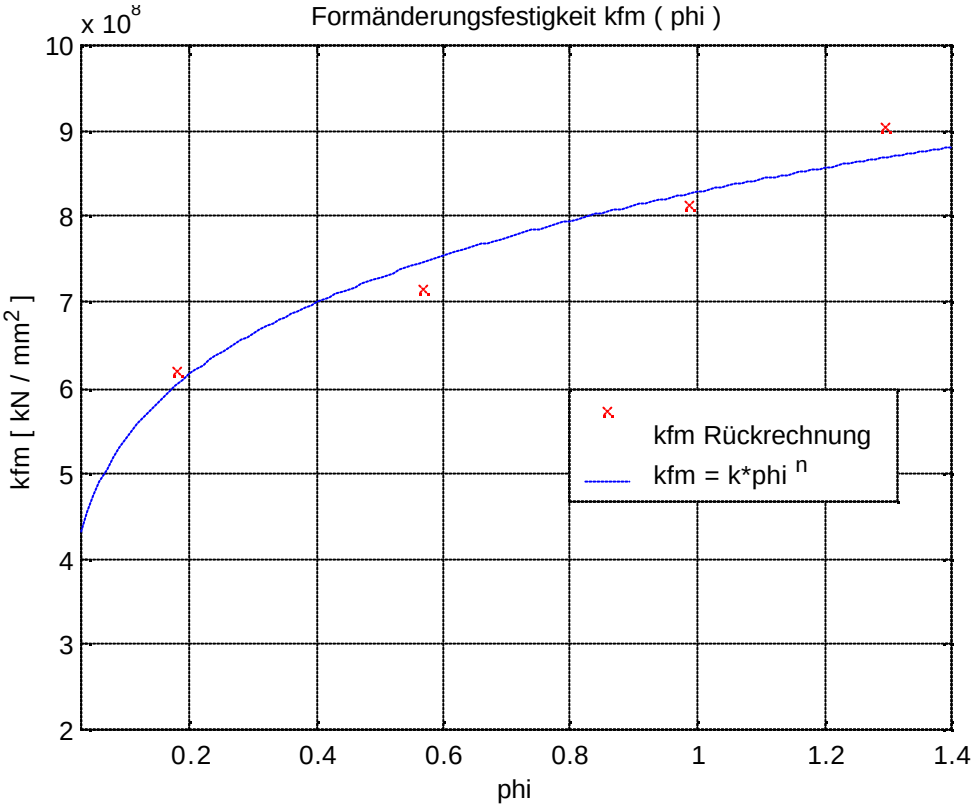


Abb. 2: Regression der Fließkurve für ein Material

3 Einsatz Neuronaler Netze

3.1 Grundkonzept und Auswahl der Eingangsparameter

Für den Umformprozess in einem Kaltwalzwerk spielen neben der Analyse der Legierungsbestandteile noch weitere Kenngrößen aus dem vorangegangenen Warmwalzprozess, wie Endwalztemperatur, Endwalzgeschwindigkeit, Haspeltemperatur und Kühlstrategie, eine Rolle. Weitere thermische Prozesse, wie das Glühen, haben ebenfalls Einfluss auf das Gefüge und damit auf die Fließkurve (Abb. 3).

Für eine erste Untersuchung wurden nur die Legierungsbestandteile des zu walzenden Materials betrachtet. Eine Erweiterung auf weitere Eingangsgrößen zu einem späteren Zeitpunkt wird angestrebt.

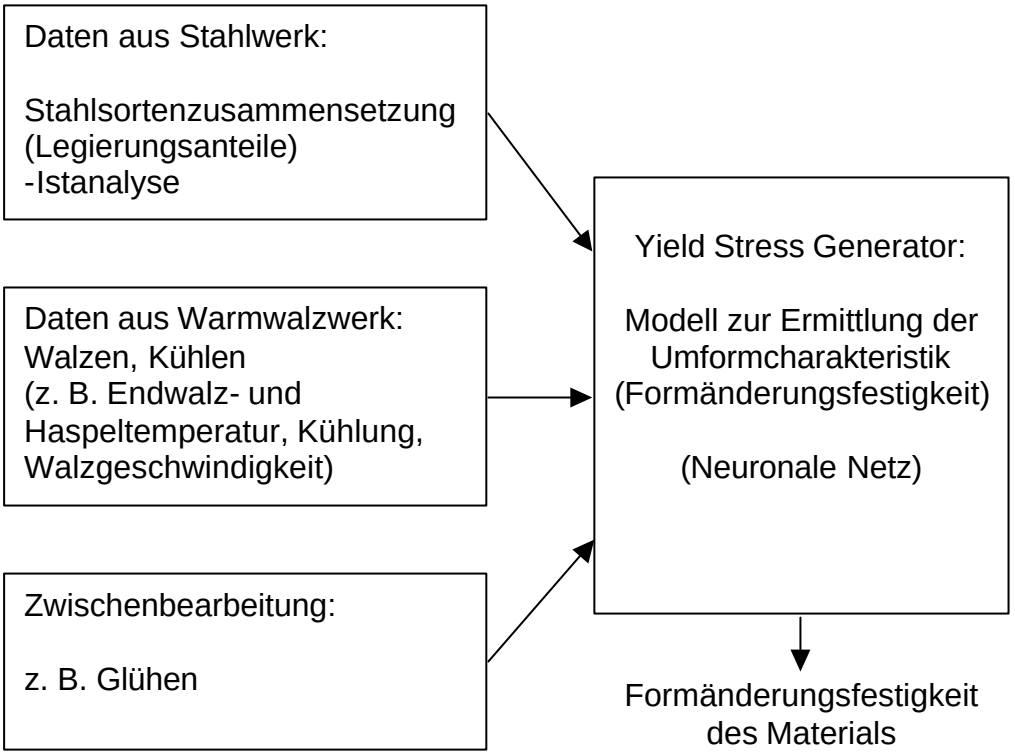


Abb. 3: Konzept zur Fließkurvenbestimmung mit Neuronalen Netzen

3.2 Gewinnung der Lerndatensätze

Die Stahlanalyse liefert die prozentualen Anteile der chemischen Elemente Aluminium, Arsen, Bor, Chlor, Chrom, Kupfer, Kohlenstoff, Mangan, Molybdän, Niob, Nickel, Stickstoff, Sauerstoff, Phosphor, Silizium, Zinn, Schwefel, Titan, Vanadium, Zink und Zirkonium. Die Parameter k und n wurden für jedes Material aus dem Walzprozess errechnet. Damit lagen die Daten vor, um ein Neuronales Netz zu trainieren, insgesamt ca. 7400 Datensätze.

Um die verschiedenen Stähle im täglichen Umgang besser handhaben zu können, werden sie in Härteklassen gruppiert. Die Verteilung der Datensätze auf die Härteklassen sind in Abb. 4 gezeigt. Je größer die Härteklasse ist, desto fester ist der jeweilige Stahl. Das Bild zeigt, dass für höherfeste Stähle eine geringere Anzahl an Lerndatensätzen zur Verfügung steht.

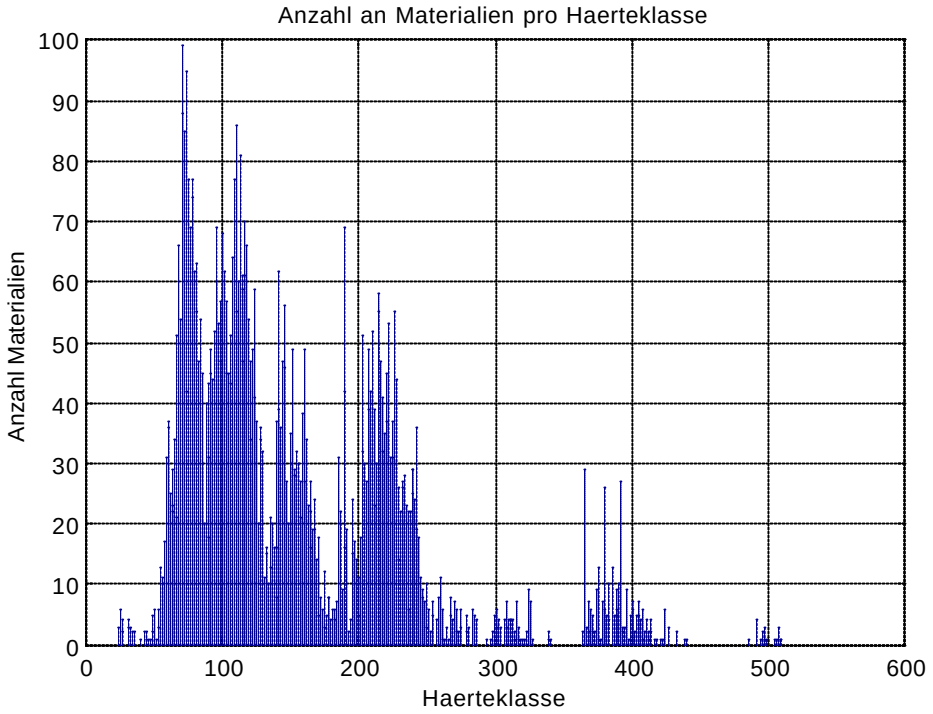


Abb. 4: Anzahl der Lerndatensätze je Härteklasse

3.3 Wahl der Netztopologie und des Lernverfahrens

Da der Zusammenhang sehr komplex ist und quantitativ verwertbares Vorwissen nicht existierte, kam ein Perzeptron-Netz mit einer verdeckten Schicht und 30 Neuronen zum Einsatz. Das Perzeptron-Netz wurde mit dem Backpropagation-Lernverfahren trainiert. Das Prinzip ist in Abb. 5 dargestellt.

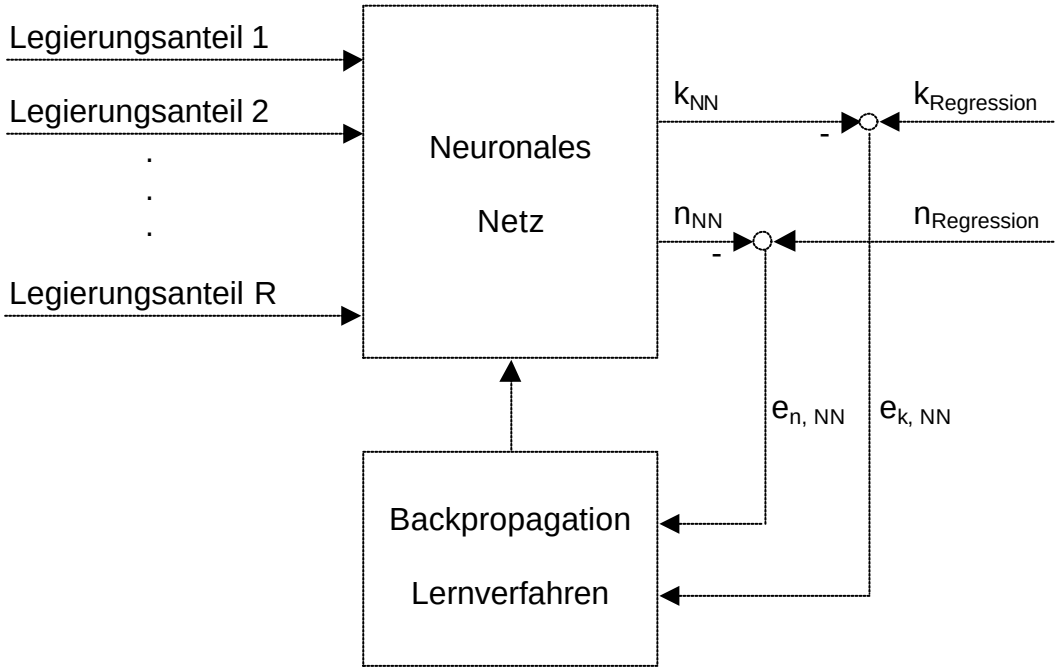


Abb. 5: Eingangs- und Ausgangsgrößen, Lernverfahren für das Perzeptron

3.4 Vergleichsmaßstab

Die Ergebnisse des Neuronalen Netzes wurden mit einem über viele Jahre entwickelten statistischen Modell verglichen.

4 Ergebnisse

Das Generalisierungsverhalten wird wie üblich bestimmt, indem die vorhandenen Daten in die Lernmenge und in die Generalisierungsmenge unterteilt werden. Um die Leistungsfähigkeit des Neuronalen Netzes besser beurteilen zu können, werden drei Situationen betrachtet.

Zunächst werden aus der Datenmenge gleichmäßig über alle Härteklassen Lern- und Generalisierungsdaten gebildet. Mit dem so parametrierten Neuronalen Netz werden die Fließkurvenparameter gebildet und mit den Werten Walzkräfte berechnet. Anschließend wird der mittlere prozentuale Walzkraftfehler ermittelt, wobei als Bezugswert die gemessene Walzkraft benutzt wird. Weiterhin werden der Walzkraftfehler der Fließkurvenparameter aus der Regression und des statistischen Modells berechnet und graphisch dargestellt. Abbildung 6 zeigt diesen unteren Grenzfehler bei idealen Bedingungen.

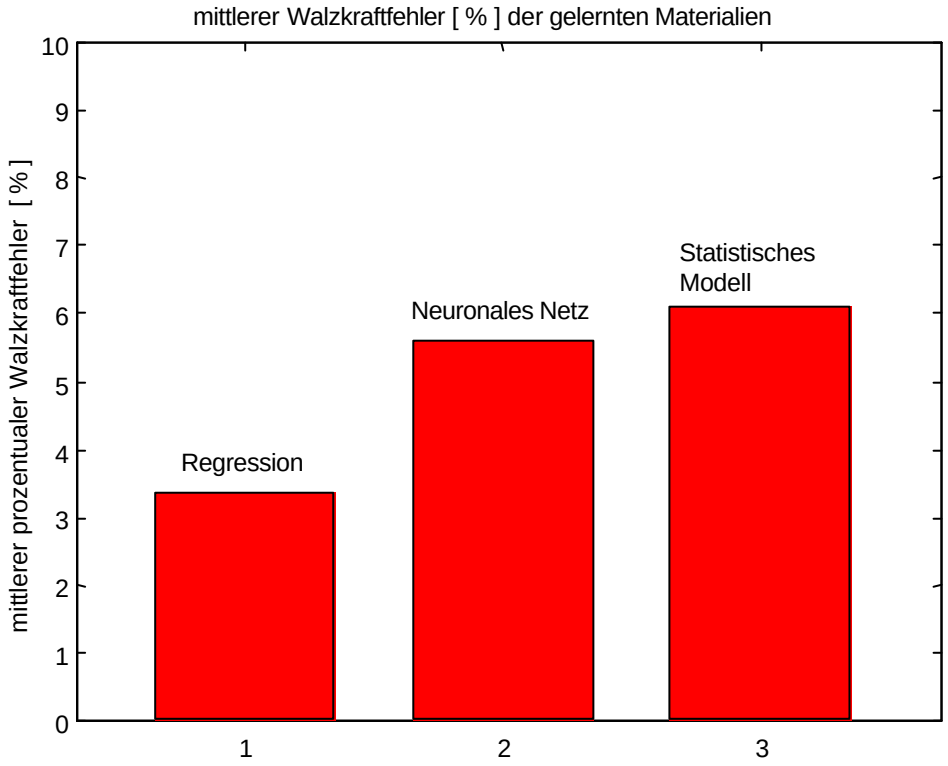


Abb. 6: Mittlerer prozentualer Walzkraftfehler aller Materialien

Der Walzkraftfehler aus der Regression entsteht wegen der Fließkurvenapproximation nach Ludwik. Dieser Fehler könnte vom Neuronalen Netz nur bei perfektem Lernen erreicht werden. Beim Lernen aller Materialien erreicht das Neuronale Netz einen Lernfehler von unter 6% und liefert damit sehr gute Fließkurvenparameter für die Walzkraftberechnung.

Um eine reale Situation im Walzwerk zu simulieren, werden nun Daten der Materialien einer gewünschten Anzahl an Härteklassen (z. B. Klassennummer 110...130) aus der Datenmenge entnommen. Dies entspricht der Situation in einem Walzwerk, dass Materialien gewalzt werden sollen, die das Neuronale Netz zuvor nicht gelernt hat. Mit den verbleibenden Materialien wird das Netz trainiert.

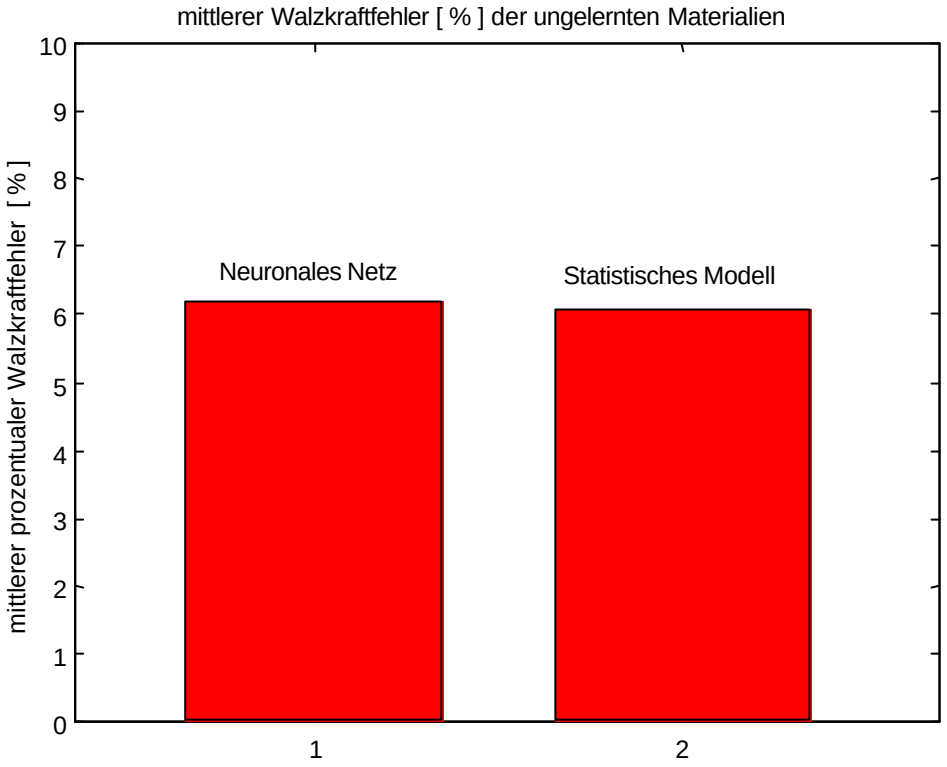


Abb. 7: Mittlerer Walzkraftfehler [%] für die dem Netz vorenthaltenen Materialien

In Abb. 7 ist zu erkennen, dass der Walzkraftfehler für die vorenthaltenen Materialien leicht über 6% gestiegen ist. Dieser Fehler ist als gut anzusehen, da er sich in der gleichen Größenordnung wie der Fehler der Vergleichsdaten des statistischen Modells bewegt.

Zur Parametrierung des Neuronalen Netzes ist es notwendig, dass Materialien des gesamten Walzspektrums als Lerndatenmenge dem Neuronalen Netz zur Verfügung stehen. Es ist nicht ausreichend, das Neuronale Netz mit Materialien einer bestimmten Gruppe an Härteklassen (z. B. Härteklassen: 0-300) zu parametrieren, da in diesem Fall das Neuronale Netz für Materialien anderer Härteklassen (z. B. >300) schlechte Generalisierungsergebnisse liefert. Zur Simulation der gerade beschriebenen extremen Verhältnisse, werden die Materialien der Härteklassen (>300) der Lerndatenmenge vorenthalten und nach dem Lernen der Walzkraftfehler für diese Materialien berechnet. In Abb. 8 ist der mittlere prozentuale Walzkraftfehler des Neuronalen Netzes für diese Materialien dargestellt, wenn sie der Lerndatenmenge vorenthalten wurden.

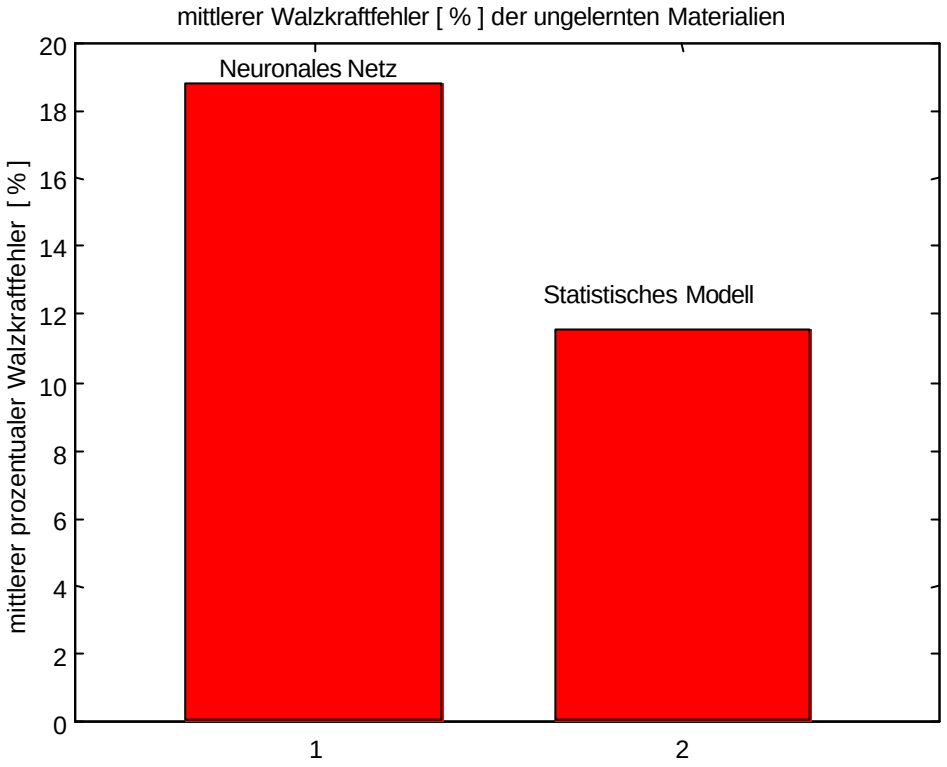


Abb. 8: Walzkraftfehler bei Entnahme von Härteklassen (> 300)

Dieses Ergebnis zeigt, dass es nicht möglich ist, das Neuronale Netz mit einem Ausschnitt an Materialien aus dem Walzspektrum zu parametrieren. Um gute Fließkurvenparameter für Materialien aller Härteklassen zu erhalten, müssen demzufolge Materialien über das gesamte Spektrum an Härteklassen als Lern-datenmenge zur Verfügung stehen. Für die Praxis der Voreinstellung der Walzstraße sei aber angemerkt, dass ein Fehler von bis zu 20% in der Walzkraftberechnung noch einen walzbaren Stichplan liefern kann, auch wenn mit diesem nicht mehr eine optimale Qualität erreicht wird. Damit kann in einem solchen Fall noch immer von der Generalisierungsfähigkeit des Neuronalen Netzes profitiert werden. Allerdings zeigt dieses Beispiel auch die Grenzen der Einsatzmöglichkeiten eines Neuronalen Netzes für diese Anwendung.

Zum Vergleich des Walzkraftfehlers aus Abb. 8, wird in Abb. 9 der Walzkraftfehler der Materialien der Härteklassen (>300) dargestellt, wenn die Daten dieser Materialien, in der Lerndatenmenge enthalten sind. Dieses Ergebnis zeigt, dass das Neuronale Netz beim Lernen des gesamten Walzspektrums gute Ergebnisse für diese Materialien liefert, trotzdem die Anzahl der zur Verfügung stehenden Lerndatensätze deutlich geringer ist. Die Ergebnisse des statistischen Modells, dem ebenfalls die Daten aus dem gesamten Walzspektrum zur Verfügung stehen, sind für die Härteklassen (>300) deutlich schlechter.

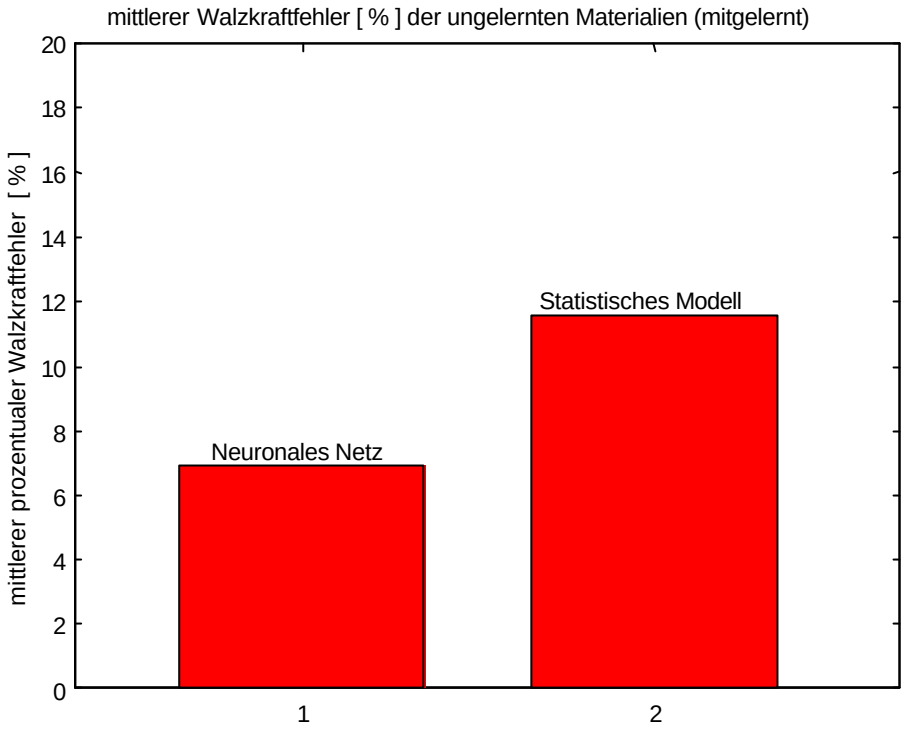


Abb. 9: Walzkraftfehler der Härteklassen (>300)

5 Praktischer Einsatz in Projekten von ALSTOM

Das hier näher beschriebene Neuronale Netz wurde bisher in einer Studie mit Daten aus von ALSTOM ausgeführten Walzwerksautomatisierungen entwickelt und getestet. In einem aktuellen Projekt wird das Neuronale Netz um die Parameter aus dem Warmwalzprozess erweitert (siehe 3.1).

Eine weitere Anwendung eines Neuronalen Netzes zur Fließkurvenbestimmung wird von ALSTOM in einem Warmwalzwerk genutzt. Diese Applikation unterscheidet sich jedoch von der hier beschriebenen dadurch, dass die Fließkurven für durch ihre standardisierte Materialzusammensetzung klassierten Stahlgruppen bekannt sind. Das Neuronale Netz hat hier die Aufgabe, die Zuordnung der Fließkurve zu dem zu walzenden Material aus der aktuellen Analyse vorzunehmen. Zusätzlich wird die Grundumformfestigkeit aus der Istanalyse bestimmt. Damit sind vom Anwender keine speziellen Kenntnisse über das zu walzende Material gefordert. Zum Einsatz kommt ein ebenfalls ein Perzeptron-Netz. Die Ergebnisse zeigen, dass Neuronale Netze, für diese mit physikalischen Modellen nur mit sehr hohem Aufwand zu lösende Aufgabe, einen interessanten und vielversprechenden Lösungsweg darstellen, der sich auch auf andere Legierungen ausdehnen lässt.

Zusammenfassend läßt sich zeigen, dass Neuronale Netze eine sinnvolle Ergänzung in Bereichen sind, für die physikalische Modelle nicht verfügbar sind. Jedoch erreichen Neuronale Netze nicht die Genauigkeit von physikalischen Modellen in Bereichen, für die keine Trainingsdaten zur Verfügung stehen. Allerdings zeigen die bisherigen Erfahrungen beim Einsatz Neuronaler Netze auch, dass mit deutlich weniger Aufwand gegenüber der Entwicklung statistischer Modelle vergleichbare Ergebnisse erreicht werden können.

6 Literatur

[1] Weber, K.: *Grundlagen des Bandwalzens.* VEB Deutscher Verlag für Grundstoffindustrie, Leipzig; 1973

[2] Prüfer, N.: *Anwendung von Methoden der künstlichen Intelligenz für das Setup von Walzwerken*, ALSTOM Power Conversion GmbH, Berlin und Fachhochschule für Technik und Wirtschaft, Berlin; 2002

[3] Klawon, T.: *Technischer Bericht: Yield Stress Generator*, ALSTOM Power Conversion GmbH, Berlin; 2001

[4] Hambrecht, A., Beckmann, T., Ullmann, T. Dlabka, M.: *Mehrgrößenregelung der Bandplanheit durch nichtlineare Vorsteuerung und Stellgliedkompensation realisiert mit Neuronalen Netzen*, VDI-Berichte 1526, S. 137-142; 2000

[5] Schulze-Ksinzyk, J.: *Fließkurvenfinder Engineering Manual,* ALSTOM Power Conversion GmbH, Berlin

Neuronale Adaptive Regelung nichtlinearer Systeme durch Feedback Linearization

D. Karimanzira, P Otto
Technische Universität Ilmenau
Fakultät für Informatik und Automatisierung
Postfach 10 0565, 98684 Ilmenau
E-mail: divas.karimanzira@systemtechnik.tu-ilmenau.de

Kurzfassung

In diesem Beitrag wird einen Feedback-Linearization-Regler (NFL) auf der Basis von Neuronalen Netzwerken zur Regelung von SISO kontinuierlichen nichtlinearen Systemen präsentiert. Die Regelungsstruktur besteht jeweils aus einer Komponente zur Feedback-Linearization und einer zur Begrenzung des Stellsignals. Bei dieser Methode wird keine off-line Trainingsphase der Neuronalen Netzwerke benötigt und die Initialisierung der Netzwerkgewichte ist einfach zu vollziehen. Die Leistungsfähigkeit des Reglers wird anhand zweier in der Regelungstechnik anerkannte Benchmarksysteme, ein Schwebekörper und ein Propellerarm erprobt.

Schlüsselwörter: Neuronale Netzwerke, Feedback linearisation

1 Einführung

Eine der gebräuchlichen Methoden zur Regelung einiger nichtlinearer Systeme besteht darin, die nichtlinearen Systeme in lineare umzuwandeln. Eine dieser Methoden ist die Linearisierung durch Rückkopplung und ordnet sich unter den geometrischen Techniken ein. Leider ist die Anwendbarkeit dieser Methode sehr begrenzt, weil sie die genaue Kenntnis der Nichtlinearitäten der Strecke voraussetzt. Um die Einschränkungen durch die Notwendigkeit eines exakten „model-matching“ zu vermindern, sind mehrere adaptive Strategien, die einige lineare [Campion und Bastin, 1990] und nichtlineare parametrische Unsicherheiten tolerieren [Marino und Tomei, 1993] vorgestellt worden.

Eine allgemeine Reglerstruktur für die Linearisierung durch Rückkopplung (Feedback Linearisation (NFL)) kann durch $u = N^{-1}(x)Z(x)$ beschrieben werden. Falls ein adaptives Schema mit einem Regler zur Berechnung von $\hat{N}(x,\theta)$ angewandt wird, muss $\hat{N}$ stets ungleich Null sein. Wie nicht anders zu erwarten, findet man das gleiche Problem auch in neuronalen Regelungssystemen, da diese als nichtlineare adaptive Systeme kategorisiert werden können. Hier wird eine Reglerstruktur, die dieses „Division durch Null“ - Problem unabhängig von den neuronalen Netzwerkgewichten in der Berechnung der Steuergröße vermeidet, vorgestellt. Es wird ein stabiles neuronales Reglerentwurfsschema vorgeschlagen, dass die in der Literatur existierenden Annahmen etwas reduziert. Für die Anwendung eines neuronalen netzwerkbasierten Reglers wird gezeigt, dass alle Signale für einige zustandsrückgekoppelte linearisierbare Systeme begrenzt sind. Außerdem kann nachgewiesen werden, dass durch die Anwendung mehrschichtiger Perzeptrons mit nichtlinearen Aktivierungsfunktionen, keine offline Trainingsphase notwendig ist und dass sich die Initialisierung der Netzwerkgewichte vereinfacht.

2 Konzept der Regelung durch Feedback-Linearization

Ursprünglich wird dieses Konzept in [Isidori, 1995] für einen unbekannten, durch Rückkopplung linearisierbaren Prozess, beschrieben. Die Grundidee besteht darin, ein Zustandsraummodell eines Prozesses in neuen Koordinaten so zu bilden, dass die Nichtlinearitäten durch Rückkopplung völlig oder teilweise ausgeschaltet werden können. Die Hauptanforderung dabei ist, dass die Modelle der Nichtlinearitäten genau bekannt sein müssen. Im vorliegenden Fall, werden die Nichtlinearitäten durch neuronale Netzwerke modelliert. Zur Darstellung der Methode wird ein adaptives Regelungsproblem für einen SISO-Prozess [Chen und Khalil, 1992] verwendet:

$$\begin{aligned} y_p(k+1) = {} & f\left[y_p(k),\ldots,y_p(k-n_y+1),u(k-1),\ldots,u(k-n_u)\right] \\ & + g\left[y_p(k),\ldots,y_p(k-n_y+1),u(k-1),\ldots,u(k-n_u)\right]u(k) \end{aligned} \qquad (1)$$

Die Zustandsvariablen werden wie folgt gewählt:

$$\begin{aligned} z_{11}(k) &= y_p(k-n_y+1) \\ &\vdots \\ z_{1n_y}(k) &= y_p(k) \\ z_{21}(k) &= u(k-n_u) \\ &\vdots \\ z_{2n_u}(k) &= u(k-1) \end{aligned} \qquad (2)$$

Damit nimmt das Zustandsraummodell folgende Form an:

$$\begin{aligned} z_{11}(k+1) &= z_{12}(k) \\ &\vdots \\ z_{1n_y}(k+1) &= f[z(k)] + g[z(k)]u(k) \\ z_{21}(k+1) &= z_{22} \\ &\vdots \\ z_{2n_u}(k+1) &= u(k) \\ y_p(k) &= z_{1n_y}(k) \end{aligned} \qquad (3)$$

Mit der Rückkopplungsregelung (Feedback control)

$$u(k) = \frac{1}{g[z(k)]}\left[w(k) - f(z(k))\right] \qquad (4)$$

ergibt sich für den Prozess die Darstellung:

$$\begin{aligned} z_1(k+1) &= Az_1(k) + Bw(k) \\ z_1(k+1) &= F[z_1(k), z_2(k), w(k)] \\ y_p(k) &= Cz_1(k) \end{aligned} \qquad (5)$$

$$\text{wo, } A=\begin{bmatrix} 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \ddots & 0 \\ 0 & 0 & 0 & \cdots & 1 \\ 0 & 0 & 0 & \cdots & 0 \end{bmatrix},\ B=\begin{bmatrix} 0 \\ 0 \\ \vdots \\ 1 \end{bmatrix}, C=\begin{bmatrix} 0 & 0 & \cdots & 0 & 1 \end{bmatrix} \qquad (6)$$

steuerbare und beobachtbare Matrizen sind. Dabei müssen die folgenden Annahmen gelten:

* $f(z)$ ist stetig und endet im Nullpunkt,
* $g(z)$ ist stetig über eine kompakte Teilmenge S von $R^{n_y+n_u}$ und ungleich Null
* der Prozess ist ein Phasenminimumsystem.

3 Feedback-Linearization mit neuronalen Netzwerken

Die Konfiguration des Konzeptes ist in Abbildung 1 dargestellt. Die Funktionen $f(z)$ und $g(z)$ werden durch zwei neuronale Netzwerke $\mathrm{NN_f}$ und $\mathrm{NN_g}$ approximiert.

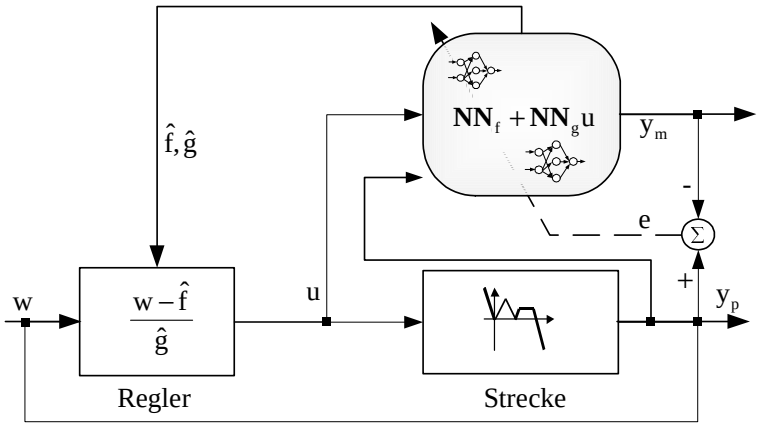


Abbildung 1: Feedback-Linearisierung mit neuronalen Netzwerken

Die Ableitung der Trainingsmethode zur simultanen Anpassung der Gewichte der zwei neuronalen Netzwerken (f- bzw. g-Netz) ist leicht nachvollziehbar. Der Unterschied zum Training eines einzelnen Netzwerkes ist, dass zwei partielle Ableitungen $\psi_f(k,\theta_f)=\frac{\partial \hat{f}}{\partial \theta_f}$ und $\psi_g(k,\theta_g)=\frac{\partial \hat{g}}{\partial \theta_g}$ der Netzwerkausgänge bezüglich der Netzwerkgewichte zuerst berechnet werden müssen, bevor sie zu einer gesamten Ableitung $\psi(k,\theta)=\frac{\partial \hat{y}(k|\theta)}{\partial \theta}=\begin{bmatrix} \psi_f(k,\theta_f) \\ \psi_g(k,\theta_g)u(k-1) \end{bmatrix}$ zusammengefasst werden. Mit dieser Ableitung kann jede der Trainingsmethoden (Backpropagation- oder Gauss-Newton Algorithmus) ohne weitere Modifikation angewendet werden.

Dieses Konzept wird weiter in [Yesildirek und Lewis, 1994], [Gawthrop,1995)], [Peel et al., 1994], [Braake et al., 1994] und [Karimanzira, 2000] behandelt.

4 Demonstration der Leistungsfähigkeit des Reglers

4.1 Beispiel 1: Schwebekörper

Als erstes Beispiel zur Demonstration der Leistungsfähigkeit des hier vorgestellten NFL-Reglers wird ein nichtlineares System (Abbildung 2), beschrieben durch die folgende Gleichung, verwendet:

$$\frac{\partial^2 y(k)}{\partial k^2} = -g + \frac{\alpha}{M}\frac{i^2(k)}{y(k)} - \frac{\beta}{M}\frac{\partial y(k)}{\partial k} \qquad (1)$$

wo $y(k)$ der Abstand des Schwebkörpers vom Elektromagnet, $i(k)$ der durch den Elektromagneten fließende Strom, M die Masse des Magneten, α die Gravitationskonstante, β der Reibungskoeffizient und $g(x)$ eine Magnetfeldkonstante sind.

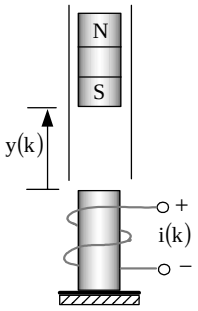


Abbildung 2: Skizze eines Schwebekörpers

Der Regler hat die Aufgabe, den Schwebekörper schnell und schwingungsarm in eine beliebige Position zu bringen. Die Neuronalen Netzwerke, die für die Approximation von $\hat{f}$ und $\hat{g}$ benutzt werden, bestehen aus 8 bzw. 3 hyperbolischen Tangens-Neuronen in der Hiddenschicht. Nach 100 Lernschritt mit dem Levenberg-Marquardt-Trainingsalgorithmus erzeugt der Regler zufriedenstellende Ergebnisse in allen untersuchten Arbeitsbereichen des Systems (siehe Abbildung 3)

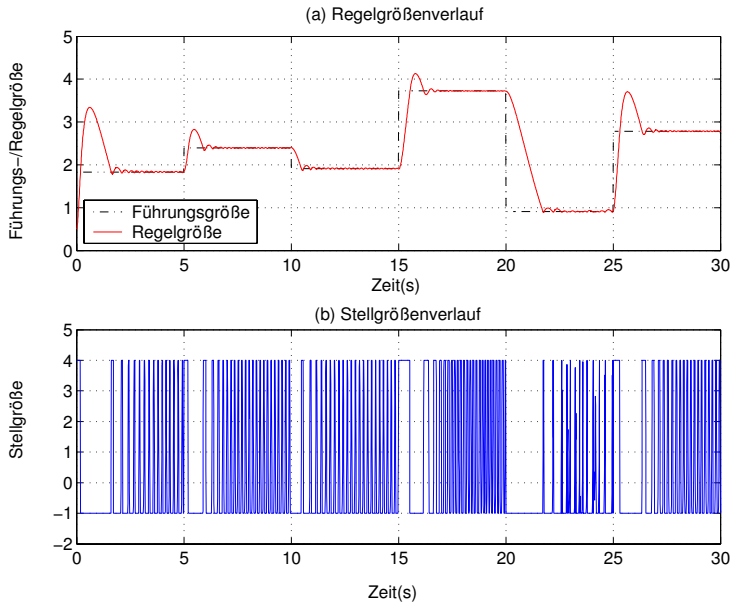


Abbildung 3: NFL-Reglerverhalten anhand eines Beispiels (Gl.(1))

4.2 Beispiel 2: Propellerarm

Das vorliegende Abschnitt behandelt den Einsatz der entwickelten Regelungskonzepte für die Echtzeitregelung eines realen Systems (Propellerarmes). Zur Demonstration der Lösung nichttrivialer regelungstechnischer Aufgabestellungen wurde das (in)stabile mechatronische System „Propellerarm“ eingesetzt [Wernstedt, 1996]. Das Propellerarm-Modell wurde im Institut Systemtechnik der Technischen Universität Ilmenau entwickelt und dient als Identifikations- und Regelungsmodell. Eine Prinzipskizze ist in Abbildung 1 gezeigt. Dieses Beispiel dient zur Untersuchung der Echtzeitfähigkeit der Regelungskonzepte mit neuronalen Netzen und dem Vergleich dieser Regelungskonzepte mit anderen herkömmlichen Strategien wie Fuzzy- und PID-Regelungen.

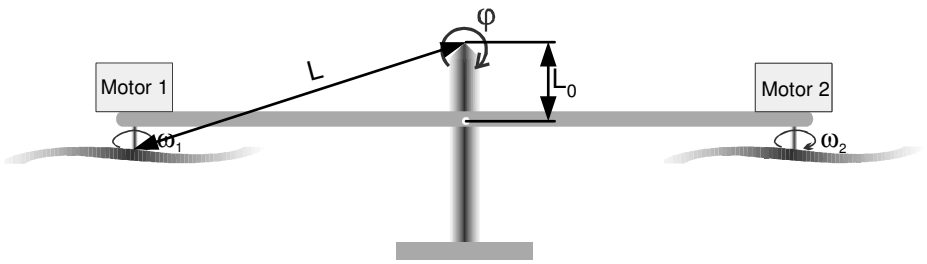


Abbildung 1: Skizze des Propellerarmes

Der Versuchsaufbau besteht aus einem im Schwerpunkt drehbar gelagerten Balken, an dessen Enden Gleichstrommotoren montiert sind. Über diese werden die an den Motorachsen befestigten Propeller angetrieben. Durch die variable Anordnung des Schwerpunktes des drehenden Teils (oberhalb ($L_o>0$) oder unterhalb ($L_o<0$) des Drehpunktes) kann ein instabiles oder stabiles Systemverhalten vorgegeben werden. Die beiden Motoren können entweder jeder separat mit einer eigenen Ansteuerspannung (bipolarer Betrieb) oder mit einer gemeinsamen Ansteuerspannung (unipolarer Betrieb) betrieben werden. Die Spannung liegt im Bereich von 0-10 V und wird zum Ansteuern der pulsweitenmodulierten Endstufe der Motoren verwendet. Zur Ermittlung der Winkellage des Propellerarms wird ein Drehpotentiometer eingesetzt. Das ausgegebene Winkelsignal liegt ebenfalls im Bereich 0-10 V. Ein Wert von 0 V entspricht dabei einen Winkel von –45° (linker Anschlag), ein Wert von 10 V entspricht einen Winkel von 45° (rechter Anschlag). Das physikalische Prinzip dieses mechatronischen Systems basiert auf dem Vorhandensein unterschiedlicher Drehmomente. Durch die Ansteuerung eines Motors dreht sich die Luftschraube und erzeugt eine senkrecht zum Propellerarm gerichtete Schubkraft. Diese Schubkraft bewirkt über den Hebel zwischen der Luftschraube und dem Drehpunkt ein resultierendes Moment. Ein weiteres Moment wird durch die Schwerkraft F_S des drehenden Systems (bestehend aus Motoren, Luftschrauben, Metallarm) über den Hebel zwischen dessen resultierenden Masseschwerpunkt zum Drehpunkt erzeugt. Da die Schwerkraft immer senkrecht nach unten gerichtet ist, besteht eine Abhängigkeit zwischen der Winkellage des drehenden Systems und dessen erzeugtem Drehmoment. Es berechnet sich unter Verwendung der allgemeinen Momentengleichung zu

$$M_S = F_S \cdot \sin(\varphi) \cdot L_0 \,. \qquad (1)$$

Abbildung 2 zeigt die Blockstruktur des mechatronischen Systems. Hieraus sind alle physikalischen Einflüsse und Zusammenhänge ersichtlich.

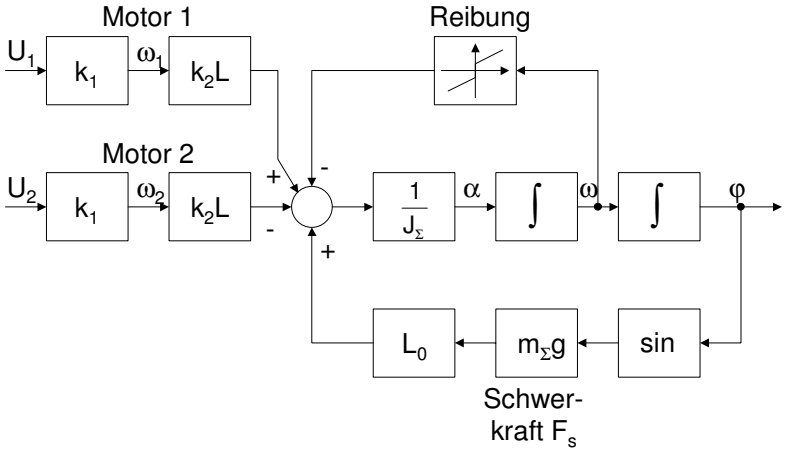


Abbildung 2: Blockstruktur des Propellerarmmodells

Durch die Sinusfunktion $\sin(\varphi)$ im Rückführzweig entsteht hier ein winkelabhängiger Verstärkungsfaktor. Betrachtet man das System um einen Arbeitspunkt, so kann dieser Verstärkungsfaktor durch die 1. Ableitung der Sinusfunktion zu $\cos(\varphi)$ gebildet werden. Fasst man die einzelnen Blöcke des Systems zu einer Übertragungsfunktion zusammen, sieht man, dass dieser Verstärkungsfaktor sowohl auf das statische als auch das dynamische Verhalten des Systems Einfluss hat.

Die Aufgabe der zu entwerfenden Regelung soll darin bestehen, den Propellerarm in jedem beliebigen Anstellwinkel zwischen –45° - +45° zu balancieren bzw. schnell und schwingungsarm in den nächsten Sollwinkel umzusteuern. Die praktische Umsetzung erfolgt hier ebenfalls, wie im vorhergehenden Abschnitt beschrieben, mit Simulink® .

4.2.1 Auswahl des Gütekriteriums

Die Güte der Regelung wurde mit einer vollständigen Sequenz von Sollwertübergängen auf verschiedenen Niveaus im Arbeitsbereich ermittelt. Die Sollwerte für die einzustellenden Werte der Regelgröße wurden für u = 5, 2, 8, 6, 3, und 7 festgelegt. Alle Sollwerte werden gleich gewichtet. Für den Fuzzy-Regler soll die gewünschte Reglereinstellung die Regelfläche und den Stellaufwand mit dem gleichen Gewicht minimieren.

Zur Regelung des Propellerarmes kommt der NFL-Regler zur Anwendung. Ihr Regelverhalten wird dann mit dem eines optimal eingestellten Fuzzy-Reglers und eines PID-Reglers verglichen, um ihre Leistungsfähigkeit einschätzen zu können.

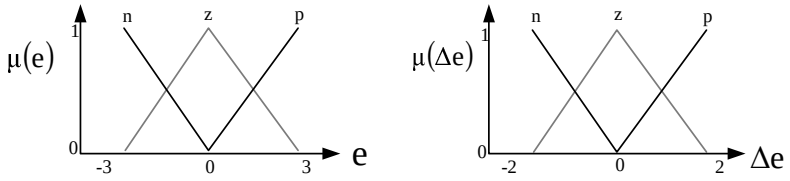
4.2.2 PID- und Fuzzy-Regelung

Die Reglerbemessung für einen PID-Regler erfolgt stets für einen im Arbeitspunkt linearisierten Prozess. Der Prozess wurde mit der Matlab®-Funktion „Linmod“ für den Arbeitspunkt $\varphi = 45°$ linearisiert und ein digitaler PID-Regler (Gl.(2)) mit dem FCD-Toolbox für Matlab® optimal eingestellt. Die ermittelten Parametern waren ($K_P = 8$, $K_I = 0.08$, $T_D = 0.8$ und $\alpha = 0.1$).

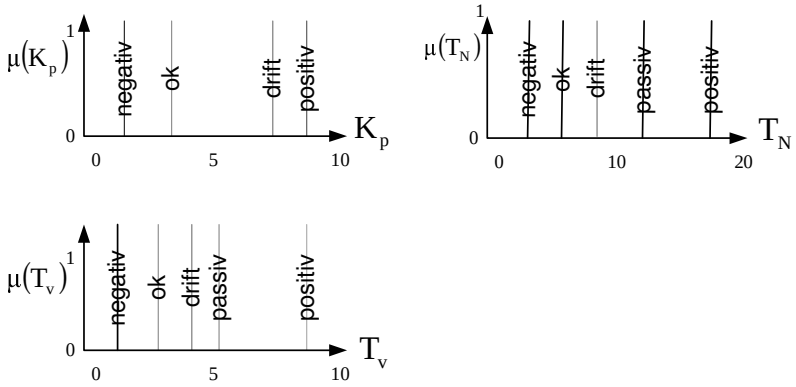
$$G(p) = K\frac{1+T_i p}{T_i p}\cdot\frac{1+T_D p}{1+\alpha T_D p} \qquad (2)$$

Vielfach reicht diese Vorgehensweise aus, um geeignete Reglerparameter zu finden, insbesondere wenn die Nichtlinearität des zu regelnden Prozesses nur schwach ausgeprägt und somit eine Linearisierung zulässig ist. Wenn jedoch, wie im vorliegenden Fall, eine Linearisierung nicht zulässig ist, wird kein zufriedenstellendes Regelverhalten im gesamten Arbeitsbereich erzielt. Die Ergebnisse der Simulation sind in Abbildung 5(a) dargestellt. Der lineare Regler kann nicht alle Arbeitspunkte dieses nichtlinearen Systems befriedigend einregeln.

Zur Verbesserung des PID-Reglers wird ein Fuzzyadapter angewandt. Zu Beginn der Optimierung wurden sämtliche Singletons für die Reglerparameter vorab auf geeignete PID-Parameter gelegt. Aufgabe des Bemessungsverfahrens war es, neben den optimalen Zugehörigkeitsfunktionen auch die optimalen Adaptionsregeln zu bestimmen. Die Optimierung mit der Evolutionsstrategie erzeugte die optimalen (in diesem Sinne) Zugehörigkeitsfunktionen in Abbildung 3 und das Regelwerk in Tabelle 1(detaillierte Beschreibung der Bezeichnungen, siehe [Wernstedt, 1996]). Die optimalen Kennfelder für die Parameteradaption K_P, T_V und T_N sind in Abbildung 4(a-c) dargestellt.



(a) Zugehörigkeitsfunktion der Eingänge des Fuzzy-Adaptors



(b) Zugehörigkeitsfunktion der Ausgänge des Fuzzy-Adaptors

Abbildung 3: Optimale Zugehörigkeitsfunktionen des Fuzzy-Adaptors

e	**N**	**P**	**Z**	**Z**	**Z**	**N**	**P**	**N**	**P**
Δe	**N**	**P**	**Z**	**P**	**N**	**P**	**N**	**Z**	**Z**
K_p	Negativ	Negativ	ok	drift	drift	passiv	passiv	positiv	positiv
T_n	Negativ	Negativ	ok	drift	drift	passiv	passiv	positiv	positiv
T_v	Negativ	Negativ	ok	drift	drift	passiv	passiv	positiv	positiv

Tabelle 1: Regelbasis des Fuzzy-Adaptors

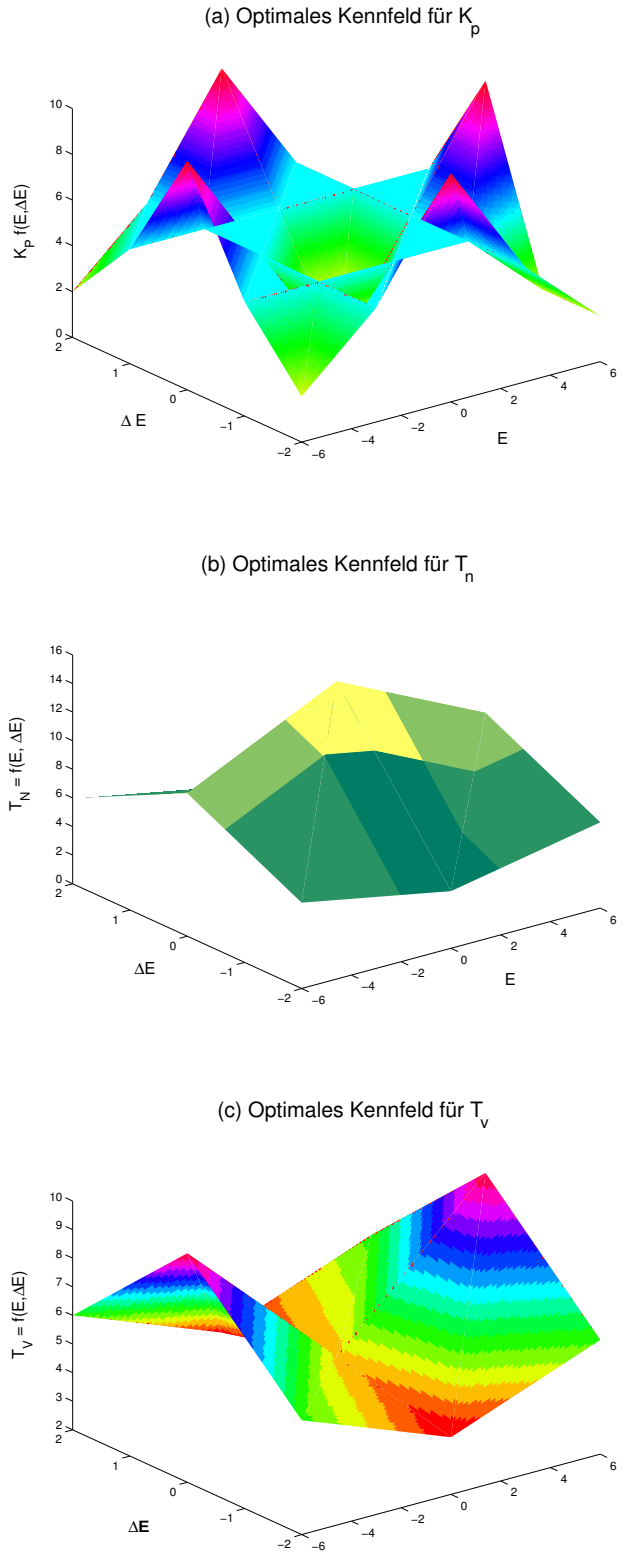
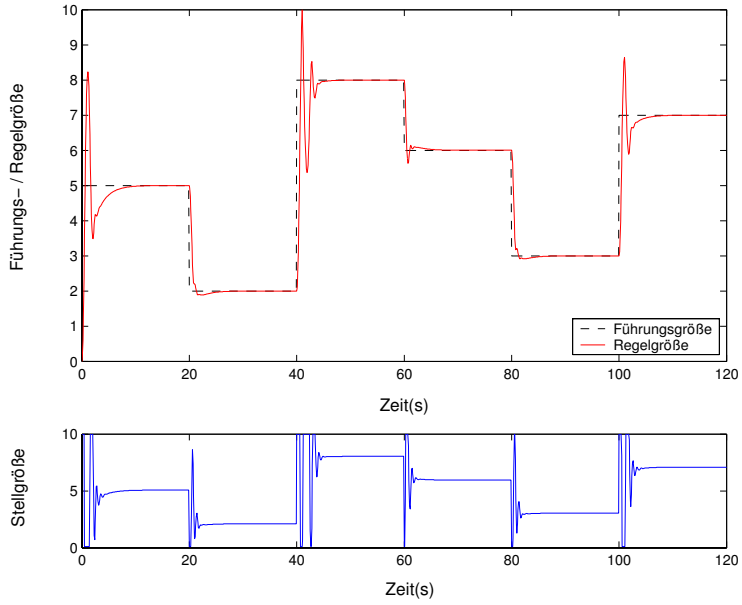
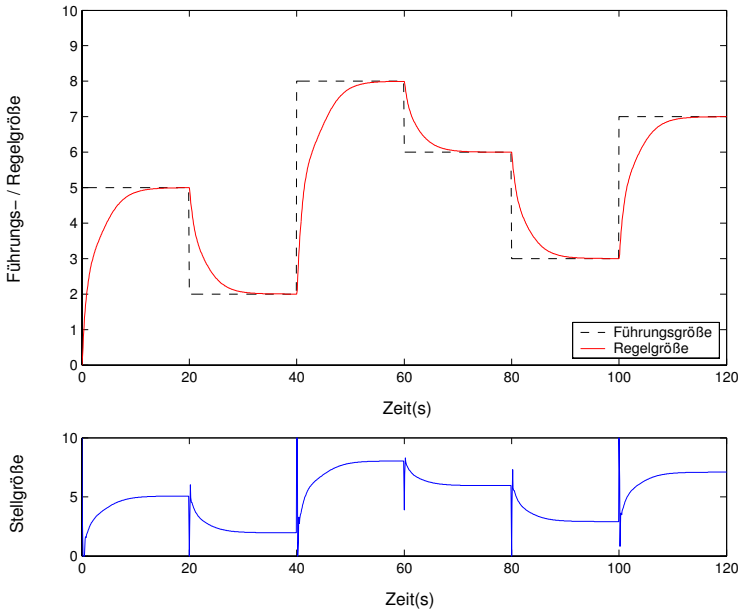


Abbildung 4: Optimales Kennfeld für die Adaption von den PID-Parametern

Abbildung 5b zeigt die Ergebnisse der Regelung mit dem optimal bemessenen fuzzyadaptierten PID-Regler. Er erreichte wesentlich bessere Ergebnisse bei der Regelung des Propellerarmes als der lineare PID-Regler (siehe Abbildung 5a). Die Ergebnisse der Regelung am realen Prozess entsprechen nicht genau denen aus der Simulation, da das System verschiedenartigen Störungen ausgesetzt ist. Allerdings werden auch am Prozess deutlich bessere Ergebnisse erzielt als mit einem PID-Regler.



(a) PID-Regler



(b) Fuzzyadaptierter PID-Regler

Abbildung 5: Regelverhalten des PID- und fuzzyadaptierten PID-Reglers

4.2.3 Regelung des Propellerarms mit dem NFL-Regler

Für das NFL-Konzept bestanden die neuronalen Netzwerke für die Funktionsapproximation von f und g aus 8 bzw. 5 Hiddenneuronen und das gesamte Netzwerk erzeugte einen maximalen Modellfehler von $3.40 \cdot 10^{-5}$ nach dem Training über 400 Epochen.

Das Führungsverhalten des Systems bei der Regelung durch den NFL Regler für die geregelte Lage des Propellerarmes bzw. entsprechende Stellgrößenverläufe sind in Abbildung 6 dargestellt. Bei der Bewertung der Regelgüte spielt auch der Stellaufwand eine große Rolle. Hier handelt sich um einen Prozess mit Ausgleich, weshalb das Kriterium für den Stellaufwand angewandt wird.

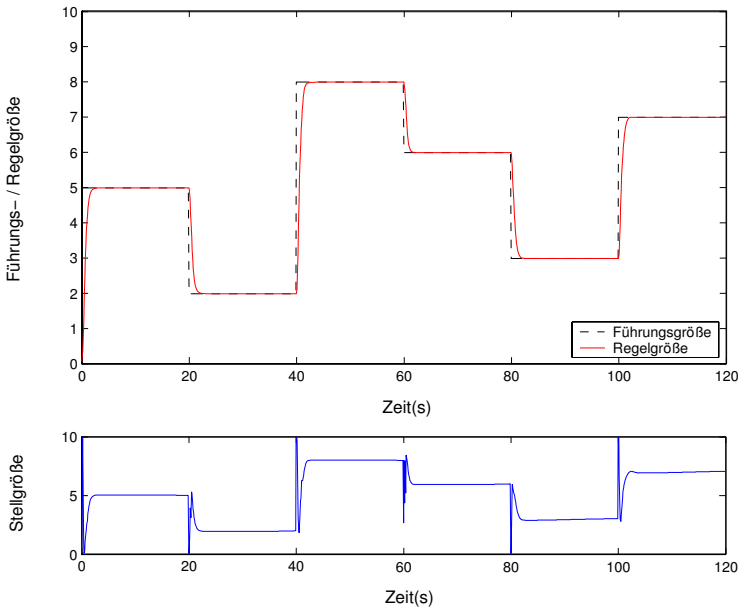


Abbildung 6:Regelverhalten des NFL-Reglers

Es ist zu sehen, dass nichtlineare neuronale Netzwerkregler ein gutes Führungsverhalten über den gesamten Arbeitsbereich der Lage des Propellerarmes aufweisen. Das ist eine deutliche Verbesserung gegenüber dem linearen PID-Regler (siehe Abbildung 5a).

5 Stabilitätseigenschaften

Im allgemeinen ist die Begrenzung von Netzwerkeingängen (φ), $\hat{f}(\theta_f, \varphi)$ und $\hat{g}(\theta_g, \varphi)$ keine Garantie für die Stabilität des geschlossenen Regelkreises, da das Regelgesetz (Gl.(4)) für $\hat{f}(\theta_f, \varphi)$ und $\hat{g}(\theta_g, \varphi) = 0$ nicht definiert ist. Es müssen Maßnahmen getroffen werden, um den Reglerausgang selbst zu begrenzen. In der Literatur [Yesidirek und Lewis, 1994] gibt es einige Techniken zur Begrenzung von Stellgrößen. Eine einfache Vorgehensweise für einige Systeme besteht darin, $\hat{g}$ als Konstante zu schätzen. Dies führt zu einer schlechten Approximation und ist nur für wenige Klassen von Systemen zulässig, da in Abhängigkeit von den Grenzen Regler entstehen können, die nicht richtig funktionieren. Falls g durch ein adaptives Schema neu berechnet wird, kann eine lokale Lösung bestimmt werden, indem angenommen wird, dass die Anfangsschätzungen sehr nahe bei den aktuellen Werten liegen und dadurch unzulässige Gebiete für $\hat{g}$ vermieden werden [Chen und Khalil, 1992]. Leider ist es auch mit guter Kenntnis des Systems nicht einfach, die Gewichte der Neuronalen Netzwerke so zu initialisieren, dass eine gute Approximation erreicht wird.

6 Vorteile und Nachteile des Konzepts

Zusammengefasst hat das NFL-Konzept die folgenden Systemeigenschaften: Es ist einfach zu implementieren, braucht nur ein Vorwärtsmodell des Systems und bietet Tuningmöglichkeiten ohne das Modell neu zu trainieren. Probleme können auftreten, falls die Inverse des Systems instabil oder in der Nähe der Stabilitätsgrenze liegt und es ist nicht einfach eine passende Modellstruktur auszuwählen, da zwei neuronale Netzwerke ausgewählt werden müssen. Das Konzept ist auf einen Typ von Systemen (Strecken der Form f()+g()u) beschränkt.

7 Diskussion

Es wurde mehrere Experimente durchgeführt. Die Ergebnisse in Abbildungen 3 und 9 zeigt das Verhalten des NFL-Reglers, wo das Regelungssystem von einer Sprungfolge mit unterschiedlichen Höhe angeregt wurde. Das System überschwingt etwas (s.h. Abbildung 3) aber kommt schnell zur Ruhelage und erzeugt keine bleibende Regelabweichung. Der direkte Vergleich mit einem PID-Regler (Abbildung 5a) zeigt die relative Leistungsfähigkeit des Reglers. Wenn man denkt, dass bei NFL kein off-line Training und keine Voreinstellungen der Entwurfsparameter (außer der Netzwerkstruktur) notwendig sind, wo im Gegensatz die PID-Parameter eingestellt werden müssen. Durch die Begrenzung von g(x) wird erreicht, dass keine „Division durch Null“ Fehler entsteht.

8 Zusammenfassung

Eine Methode zur Feedback-Linearisierung durch neuronale Netzwerk wurde vorgestellt. Die Reglerstruktur besteht aus zwei neuronale Netzwerke (NN) zur Approximation der Nichtlinearitäten und ein Teil zur Begrenzung des Steuersignals unabhängig von der NN-Gewichte. Die Ergebnisse sind zufriedenstellend, obwohl das System sehr begrenzt einsetzbar ist. Das System bedarf es eine spezielle Klasse von nichtlinearen Systemen, zumindest das System muss in die Form $f(x)+g(x)u+d$ beschreibbar sein. Das Hauptproblem ist es, dass es ist sehr schwierig zu erkennen, ob ein nichtlineares System in dieser Kategorie einfällt. Normalerweise kann nur durch Versuch der Anwendung dieses Konzept an einem System aus dem Ergebnis feststellen ob dieses Konzept einsetzbar ist oder nicht. Aber, wenn das System in der Form $f(x)+g(x)u+d$ ist dieses Konzept sehr einfach einsetzbar. Es bedarf keine offline Trainingsphase. Das untersuchte System haben wir bewusst gewählt. Es ist sehr schwierig festzustellen ob dieses System in der Kategorie $f(x)+g(x)u+d$ einfällt, aber die Ergebnisse haben gezeigt, dass dieses Konzept wohl einsetzbar ist und der Vergleich mit einer PID-Regelung bestätigt dies verstärkt.

9 Literatur

[1] Marino, R. und P. Tomei (1993). Adaptive output feedback control of nonlinear systems, part II: nonlinear parametrization. IEEE Trans. Autom. Control, AC-38, 33-48.

[2] Isidori, A. (1995). Nonlinear Control Systems. Springer-Verlag, London, UK, 3rd Edition.

[3] Taylor, D. G., P. V. Kokotovic, R. Marino und I. Kanellakopoulos (1989). Adaptive regulation of nonlinear systems with unmodeled dynamics, IEEE Trans. Autom. Control, AC-34, 405-412.

[4] Campion, G. und G. Bastin (1990), Indirekt adaptive state feedback control of linearly parametrized nonlinear systems, Int. Journal .of Control signal Process,4, 345-358.

[5] Wernstedt, J., M. Koch, T Kuhn (1996), Fuzzy Control, Optimale Nachbildung und Entwurf optimaler Entscheidungen. Oldenburg Verlag, 1996

[6] Yesildirek, A und Lewis F.L. (1994), Feedback linearisation using neural networks. In Proc. 1994. IEEE Int. Conf. On neural networks, Orlando, FL, USA, pp.2539-2544

[7] Chen, F.C. und Khalil (1992). Adaptive control of nonlinear systems using neural networks. International Journal of Control 55(6), 2861-2882.

[8] Campion, G. und G. Bastin (1990), Indirekt adaptive state feedback control of linearly parametrized nonlinear systems, Int. Journal .of Control signal Process,4, 345-358.

[9] Marino, R. und P. Tomei (1993). Adaptive output feedback control of nonlinear systems, part II: nonlinear parametrization. IEEE Trans. Autom. Control, AC-38, 33-48.

Mikrocontrollerbasiertes Softsensorsystem zur Online-Prozesszustandsdiagnose der anaeroben Biogasfermentation mit Fuzzy-Kohonen-Clustering Network

Steffen Patzwahl, Klaus-Dietrich Kramer, Thomas Nacke*

Hochschule Harz - University of Applied Sciences
Fachbereich Automatisierung und Informatik
Friedrichstraße 57-59, D-38855 Wernigerode
Tel.: +49(0)3943 / 659-317
Fax: +49(0)3943 / 659-107
E-mail: spatzwahl@hs-harz.de, kkramer@hs-harz.de
*Institut für Bioprozess- und Analysenmesstechnik e.V. (iba)
Rosenhof, D-37308 Heiligenstadt
Tel.: +49(0)3606 / 671-163
Fax: +49(0)3606 / 671-200
E-mail: thomas.nacke@iba-heiligenstadt.de

1 Einleitung

Die Softsensorik nutzt Verfahren der Computational Intelligence (CI) um bislang nur aufwändig und kostenintensiv bzw. nur offline messbare Prozessparameter auf Basis einfacher, kostengünstiger und online erfassbarer Messgrößen in Verbindung mit der „Intelligenz“ des Systems mit hinreichender Genauigkeit nachzubilden. Der vorgestellte Beitrag beschreibt ein Softsensorsystem, das in der Lage ist, den Prozesszustand der anaeroben Biogasfermentation auf Grundlage von mit Sensoren erfassten Prozesssignalen online zu klassifizieren. In Biogasanlagen wird Biogas aus organischen Reststoffen (Substrat) durch den Prozess der anaeroben Fermentation produziert. Die Substratzusammensetzung ist nicht konstant und unterliegt stofflichen und jahreszeitlichen Schwankungen. Dies führt in Verbindung mit unvollständigem Prozesswissen in der Praxis häufig zu einem suboptimalen Betrieb von Biogasanlagen und stellenweise zu einem völligen Prozessversagen. Eine Modellbildung als Grundlage für ein Prozessmonitoring bzw. eine Prozessregelung wird durch die für Bioprozesse typischen Eigenschaften von Nichtlinearitäten und kaum vorhandenen Kenntnissen über innere Prozessvorgänge erheblich erschwert. Intelligente Monitoringsysteme, die eine Online-Prozesszustandsdiagnose erlauben, können durch den Anlagenfahrer unterstützende Informationen (Entscheidungshilfesysteme) dazu betragen den Betrieb von Biogasanlagen zu optimieren.

2 Biogasanlagen

2.1 Biogaserzeugung

In Biogasanlagen lässt sich durch die anaerobe Vergärung von Substraten (beispielsweise organische Abfälle, Gülle aus landwirtschaftlichen Betrieben oder nachwachsende Rohstoffe in Form von Energiepflanzen (Mais, Raps, etc.)) Biogas gewinnen, dass

sich aufgrund seiner Eigenschaften zur Elektroenergie- und/oder Wärmeerzeugung eignet.

2.2 Probleme bei der Prozessführung / Stand der Technik

Die Erzeugung von Biogas ist besonders bei der anaeroben Co-Fermentation komplex und geschieht über eine Vielzahl von Zwischenprodukten. Bei der Co-Fermentation, wie diese bei vielen Biogasanlagen zur Anwendung kommt, wird dem Bioreaktor ein Substratgemisch aus Grundsubstrat (häufig Gülle) und verschiedenen Co-Substraten (z. B. Mais oder organische Abfälle) zugeleitet. Anaerobe Fermentationen sind immer dann problematisch, wenn die Substratzufuhr, wie im Fall der Co-Substratvergärung, hinsichtlich Raumbelastung und Zusammensetzung nicht homogen ist. In einer komplexen Mischkultur, wie sie bei der Biogaserzeugung aus organischen Reststoffen vorliegt, kann es dabei zu Schwankungen innerhalb einer adaptierten Population und damit in der Gesamtstoffwechselleistung kommen [1]. Der worst-case beim Betrieb von Biogasanlagen ist das „Umkippen“ des Fermenters. Vom „Umkippen“ wird gesprochen, wenn der Säuregehalt im Fermenter einen kritischen Wert erreicht, so dass keine Methanbildung mehr möglich ist. Der Gärprozess kommt damit zum Stillstand. Hauptgrund ist eine Überladung des Fermenters mit organischer Substanz, z. B. durch Schwankungen der Zudosiermengen und der Substratzusammensetzungen. Besonders bei Co-Fermentationen ist die Gefahr des Prozessversagens sehr hoch [6]. Eine suboptimale Substratbeschickung bzw. Prozessführung führt weiterhin zu einer sinkenden Gasproduktionsrate und einem zu niedrigen Anteil von Methan im Biogas.

In landwirtschaftlichen Betrieben lokalisierte Biogasanlagen sind i. d. R. einer von mehreren Betriebszweigen. Der für Anlagenbetreuung und Wartung mögliche Zeitaufwand ist dadurch begrenzt. Aus diesem Grund müssen landwirtschaftliche Anlagen funktionssicher konzipiert werden [10]. Der Stand der Technik von Biogasanlagen ist derzeit in Bezug auf MSR-Technik sehr unterentwickelt. Die Prozessführung findet fast ausschließlich durch Einflussnahme auf die Zudosiermenge und damit auf die hydraulische Verweilzeit sowie über die Substratzusammensetzung statt. Überwiegend erfolgt die Zudosierung nach Erfahrung des Anlagenfahrers oder durch einfache Zeitsteuerungen, welche dem Fermenter in bestimmten Intervallen eine definierte Substratmenge (auch Co-Substrate) zuführen und Reststoffe gleichzeitig abführen. Vielfach definiert sogar das Aufkommen bestimmter Substrate die Zufuhr und damit die Prozessführung. Die Folgen davon sind die angesprochenen Probleme im Hinblick auf Nichtausschöpfung der Energiebildungspotenziale und Anlagenausfälle, welche ein kostenintensives Wiederanfahren der Anlage, gekoppelt mit erneutem Animpfen des Substrates oder gar einem Austausch des Fermenterinhaltes, nach sich ziehen.

Seit mehreren Jahren wird mit verschiedenen mathematischen Modellansätzen versucht, eine geeignete Prozessstrategie zur Prozessoptimierung anaerober Fermentationsabläufe zu entwerfen. Grundlage dieser Modelle bildeten die genaue physikalische und chemische Analyse der Ausgangsmaterialien. Diese wurden in Verbindung mit der genutzten Reaktortechnik zur Bildung kinetischer Modelle mit Wachstums- und Ertragskoeffizienten genutzt. Die Überführung in die industrielle Anwendung erwies sich oft als sehr schwierig, da diese Systeme nur begrenzt auf Schwankungen in der Substratzusammensetzung reagieren konnten. Auch neuere u. a. CI-basierte Strategien zur Prozessführung von Biogasanlagen weisen Defizite im Hinblick auf die Berücksichtigung realer Anlagenverhältnisse (Substrataufkommen, Betriebsweisen, Adaptionsmöglichkeiten, preiswerte und robuste Zielhardware) auf.

Die modernen Fermentationsprozesse zur anaeroben Biogasgewinnung werden bei großtechnischen Anlagen durch eine große Anzahl von Messungen überwacht und entsprechend gesteuert. Dabei werden als wesentliche Kenngrößen für die Beurteilung des Prozesses, die Messung des zeitlichen Verlaufes der Messwerte für den pH-Wert, des Redox-Potenzials, der Konzentration wichtiger Substrate in Zuführung und Abführung aus dem Biogasreaktor und Metabolite bzw. Produkte des Stoffwechsels oder deren Verknüpfung betrachtet. Gerade die Stoffkonzentrationen sind jedoch messtechnisch nicht oder nur schwer als Online-Messgrößen zu erfassen, so dass diese Prozesskennwerte nicht in die Regelung der Anlage einfließen können. Kleine und mittlere Biogasanlagen sind i. d. R. unterinstrumentiert.

3 Konzeption und Realisierung des Softsensorsystems für Biogasanlagen

3.1 Möglichkeiten der Softsensorik bei der Prozessführung

Zentrale Aufgabe der zielgerichteten Prozessbeeinflussung in Bezug auf technische Prozesse und Systeme ist es, das dynamische (Zeit-)Verhalten in gezielter Weise zu beeinflussen. Zeitveränderliche Ausgangsgrößen des Systems sollen dabei durch Veränderung von Eingangsgrößen in einer gewünschten Weise variiert werden. Steuer- und/oder Regelaufgaben des Prozessführungssystems (PFS) können durch automatisierte Einheiten (Steuereinheit/Regler) oder auch vom Menschen (Anlagenfahrer/Bediener) ausgeführt werden. Bei komplexen Prozessen ist es aus verschiedenen Gründen zweckmäßig, die zielgerichtete Beeinflussung durch eine Symbiose zwischen Mensch und Automaten zu erreichen (operative Steuerung/Regelung bzw. Monitoring) [3]. Softsensorsysteme können neben der in der Einleitung erörterten Möglichkeit zur Nachbildung bestimmter Prozessparameter in der Realisierung als Monitor auf Basis online messbarer Größen den Prozesszustand in bestimmte vordefinierte Prozesszustandsgruppen klassifizieren. Damit ist ein Einsatz derartiger Systeme in der Prozessführung möglich.

Das im Weiteren beschriebene Softsensorsystem für Biogasanlagen ist in Form eines Prozesszustandsmonitors konzipiert. Das System klassifiziert online den Prozesszustand der Fermentation und unterstützt damit den Anlagenfahrer bei der Prozessführung, wodurch auch eine Prozessoptimierung erzielbar ist. Das PFS wurde mit Hilfe eines Designprozesses für PFS unter Einsatz der Software EmSoft (3.2) erstellt.

3.2 Material und Methoden

Labor-Biogasanlage

Im Rahmen der Arbeiten zu diesem Forschungsthema wurden zwei Laborbiogasanlagen (LBA) aufgebaut und mit umfangreichem Messequipment zur Erfassung von Prozessgrößen ausgestattet. Diese Anlagen ermöglichen es, Versuche zur anaeroben Fermentation durchzuführen, wobei insbesondere auch kritische Prozesszustände gezielt herbeigeführt und datentechnisch dokumentiert werden können. Bild 1 zeigt die Labor-Biogasanlage LBA-1 im Versuchsbetrieb zusammen mit einem mikrocontrollerbasierten Kompaktgerät. Das Fermentervolumen der LBA-1 beträgt 6 Liter.

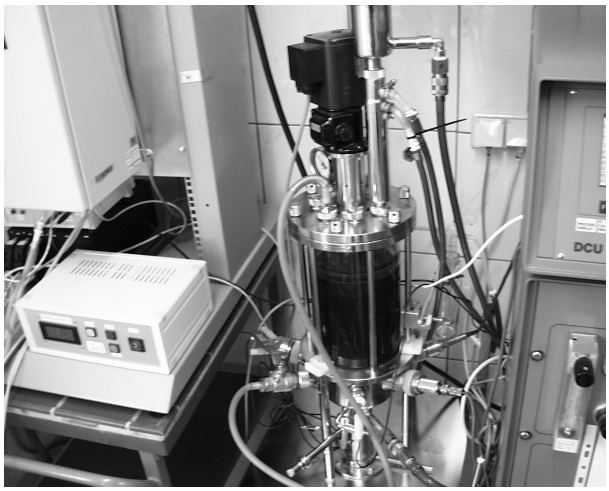


Bild 1: Laborbiogasanlage mit mikrocontrollerbasierten Kompaktgerät im Versuchsbetrieb

Softwaresytem EmSoft

Der PC-gestützte Entwurf des Softsensorsystems erfolgte unter Anwendung der Software EmSoft [4, 8] der Hochschule Harz. Den Hauptschwerpunkt des Softwaretools bildete die Anwendern gebotene Möglichkeit, direkt am Prozess einzusetzende PFS im Rahmen des Designprozesses zu realisieren. Dabei sollte auch Anwendern ohne Spezialkenntnisse eine leichte Bedienung ermöglicht werden. In Abgrenzung zu kommerziell verfügbaren Softwarewerkzeugen stellt EmSoft ein Rahmenwerkzeug dar, welches die gebräuchlichsten daten- und wissensbasierten Verfahren der CI, sowie einige klassische Verfahren in Form von Strategien zur Erstellung von PFS und/oder Modellen auf Zielhardwareeinheiten mit Mikrocontrollern (µC) und digitalen Signalprozessoren (DSP) beinhaltet und damit ein recht vollständiges Instrumentarium bildet. Der Anwender wird dadurch in die Lage versetzt, für spezielle Aufgabenstellung aus der Vielfalt der gebotenen Möglichkeiten die optimale Variante auszuwählen, zu testen und als PFS zu realisieren.

EmSoft fügt sich in den Designprozess zur Erstellung von PFS (Bild 2) [5] unter konsequenter Berücksichtigung von vier Designschritten (DS) ein. Dazu erlaubt die Software, neben dem Entwurf und der Erstellung des Reglers/Monitors, auch die Verwendung von Datenkonditionierungsstrategien mit der Möglichkeit der Merkmalsbildung sowie die Definition der Signalausgabeart um das PFS zu realisieren. Ein erfolgreich auf dem PC erstelltes PFS kann in einen Run-Time(RT)-C-Code überführt und nach der Compilierung in den Maschinencode auf das Zielhardwaresystem transferiert und auf diesem als Stand-Alone-Applikation eingesetzt werden. Unabhängig von Prozessführungsstrategien bzw. Softsensorsystemen können mit EmSoft auch Modelle zur Beschreibung von Zusammenhängen zwischen Ein- und Ausgangsgrößen (Modellbildung) erstellt und simulativ getestet werden.

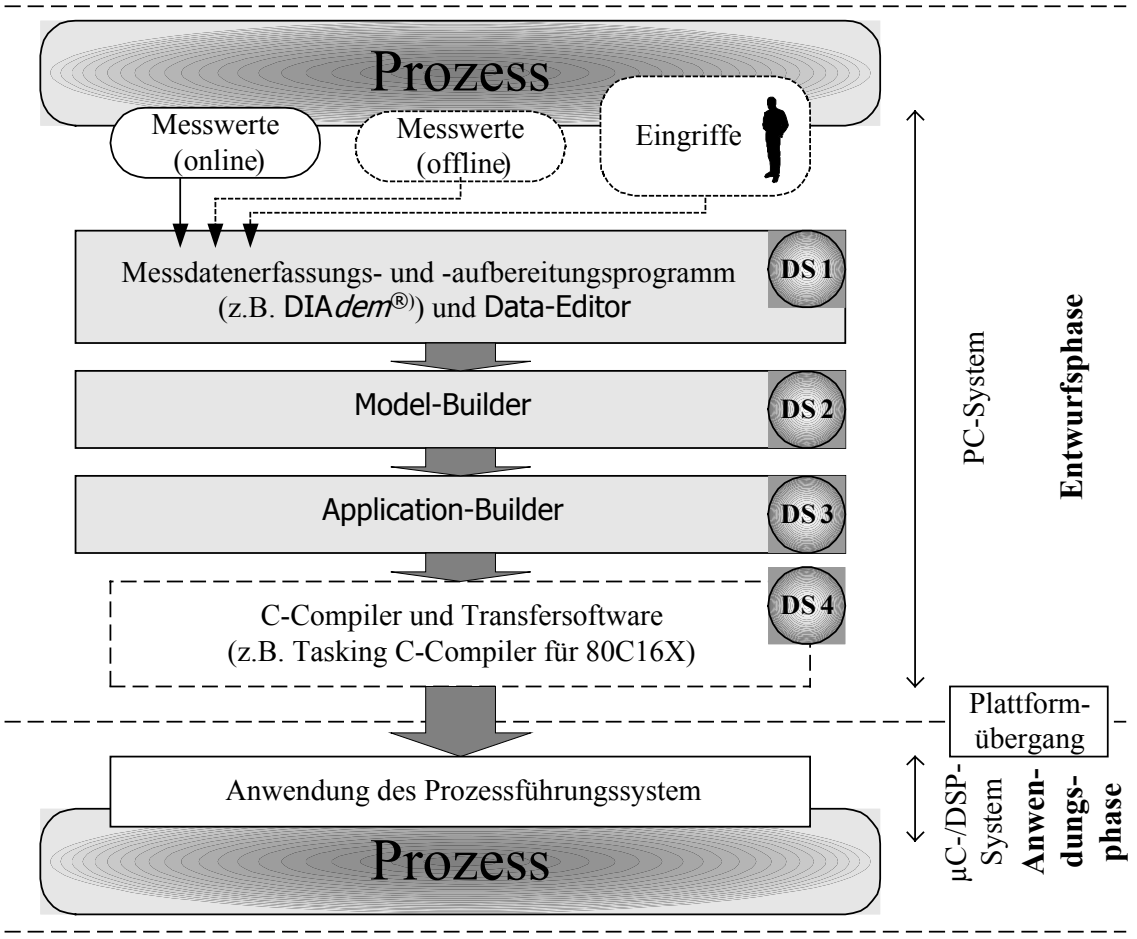


Bild 2: EmSoft im Designprozess für Prozessführungssysteme mit Darstellung der notwendigen Softwarewerkzeuge

3.3 Online-Parameter und Prozessmerkmale

Nach der Prozess- und Datenanalyse in Verbindung mit Korrelationsverfahren wurden, bezogen auf die Aufgabenstellung, die nachfolgend aufgeführten online erfassbaren Prozessparameter der anaeroben Biogasfermentation als prozess- und modellrelevant eingestuft.

- Gasproduktion (*GP*) [l/d]
- Methankonzentration im entstandenen Gas (*CH4*) [Vol.- %]
- Kohlendioxidkonzentration im entstandenen Gas (*CO2*) [Vol.-%]
- pH-Wert im Fermenter (*pH*) [-]
- Redoxpotenzial im Fermenter (*Redox*) [mV]

Diese Prozessparameter wurden in der Entwurfsphase durch einen Mess-PC mit einer Abtastzeit von t = 10 min aufgezeichnet. Der Versuchszeitraum erstreckte sich über 59 Tage. Durch die zyklische Substratzu- und -abfuhr, die für Biogasanlagen kleinerer und mittlerer Größe typisch ist [10], entstehen charakteristische, sich zyklisch wiederholende Verläufe der Online-Prozessparameter. Bild 3 zeigt dies am Beispiel der Größe *CH4*. Als Grundlage für die Merkmalsbildung wurde jeweils ein Intervall von 24 h (Zeitraum zwischen zwei Substratdosierungen im Versuch) betrachtet, das einen Zyklus repräsentiert.

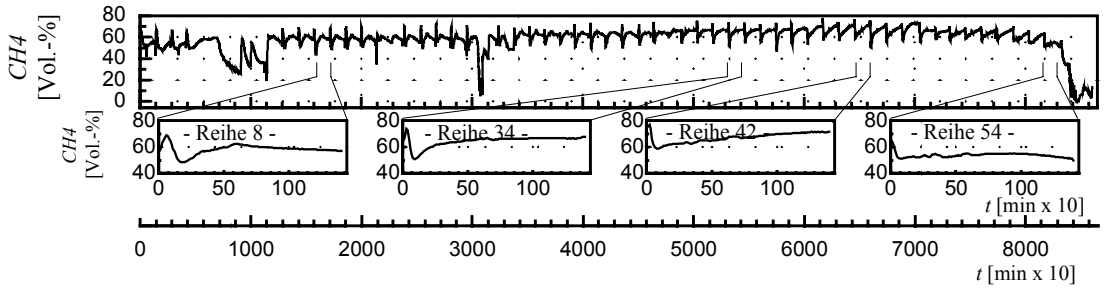


Bild 3: Verdeutlichung des Separationsvorgangs des Gesamtdatensatzes der Online-Prozessmessgrößen in Zykluskurven, die als Betrachtungszeitraum die Zeitspanne zwischen zwei Substratbeschickungen zu Grunde legen

Das Softsensor-/Monitoringsystem nutzt nachfolgend genannte Prozessmerkmale:

- Gasproduktionsrate pro Zykluszeit bzw. produziertes Volumen an Biogas bezogen auf das Fermentervolumen (*GPR*) [Nl/(l*d)]
- Konzentration an Methan im Gesamtvolumen des entstandenen Biogases pro Zykluszeit (*CH4_d*) [%]
- Arithmetisches Mittel der Zykluskurve des pH-Wertes (*pH_m*) [-]
- Arithmetisches Mittel der Zykluskurve des Redox-Wertes (*Redox*) [mV]
- Quotient der Gasvolumina von Methan und Kohlendioxid (*CH4_CO2_d*) [-]
- Änderung des Merkmals *CH4_d*, aktueller Zyklus in Bezug zum vorhergehenden (dCH4) [%]
- Quotient aus Maximalwert und 0.5-Quantil der Zykluskurve des pH-Wertes (*pH50_max*) [-]

3.4 Systemstruktur

Den Kernbestandteil des Prozessmonitors für Biogasanlagen bildet der Fuzzy-Kohonen-Clustering-Network(FKCN)-Algorithmus in dessen Anwendung als Klassifikator. Das von BEZDEK ET AL. [2] 1992 vorgestellte FKCN verbindet selbstorganisierende neuronale Netze (SOM) und den Fuzzy-C-Mean-Algorithmus. Im Vergleich zu anderen Klassifikatorstrukturen (statistische Klassifikatoren, SOM) konnten mit dem FKCN die besten Ergebnisse bezogen auf die eingesetzten Datensätze erzielt werden.

Die Struktur des FKCN-Systems zur Klassifikation des Prozesszustands (PZ) zeigt mit Benennung der benutzen Ein- und Ausgangsgrößen schematisiert Bild 4. Das FKCN klassifiziert unter Verarbeitung der Prozessmesssignale die online in jedem Zyklus berechneten Merkmale in eine vordefinierte Klasse des Prozesszustandes (PZ). Dabei wird innerhalb des Prozess- bzw. Beschickungszyklus (T_{PFS} = 24 h, bei täglicher Substratzugabe) ein Klassifikationsergebnis auf einem digitalen Ausgangsport (Byte-Format) der Implementierungshardware mit µC ausgegeben und auf einem Display angezeigt (Monitoringsystem).

Das Softsensorsystem ist für Biogasanlagen kleinerer Leistungsklasse mit unterschiedlicher Co-Substratzusammensetzung, aber relativ gleichbleibendem Substratinput, konzipiert.

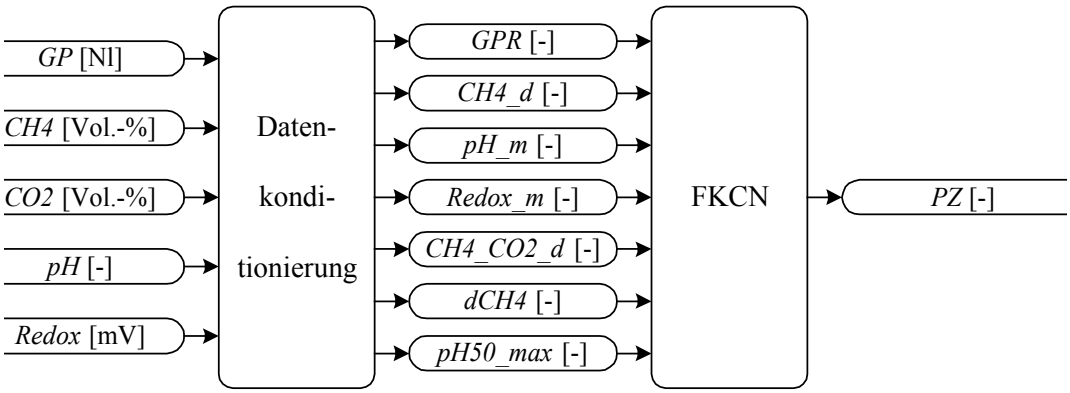


Bild 4: Struktur des Softsensors/Prozessmonitors mit FKCN, Merkmale normiert

3.5 Versuchsdurchführung / Systemerstellung

In der Phase der Erstellung des Softsensors spielt die Generierung einer Prozessdatenbasis eine signifikante Rolle. Im Rahmen der vorgestellten Arbeiten wurde an der Laborbiogasanlage (LBA-1) ein Komplexversuch durchgeführt, bei dem zur gezielten Einleitung von Prozesszuständen ein 2^n-Faktorversuchplan [9] mit sechs Teilversuchen zur Anwendung kam. Entsprechend des Zielanlagentyps und dessen Betriebsweise wurde mit einer konstanten Substratzu- und -abfuhr (QS = 275 ml/d (Dosierrate), HRT = 20 d (hydraulische Verweilzeit)) gearbeitet. Durch die Variation der Menge an organischer Trockensubstanz (oTS) im zugegebenen Co-Substrat konnte die Prozessbeeinflussung durch Einflussnahme auf die Raumbelastung (BR) erzielt werden.

Als Co-Substrate mit variabler Zugabemenge (veränderlicher Parameter des Versuchsplans) wurden Hühnerei (Protein) und Sonnenblumenöl (Fettquelle) aus industrieller Produktion verwendet. Zusätzlich kam das Basis-Co-Substrat Katzenfutter als Modellsubstanz für Schlachtabfälle zum Einsatz. Das Grundsubstrat, mit dem die Mischung aus Co-Substraten und Basis-Co-Substrat aufgefüllt wurde, bildete Rindergülle. Die Substratzu- und -abfuhr erfolgte einmal täglich.

Zur Beurteilung des Prozesszustandes in der Phase der Systemerstellung wurde ein Entscheidungsbaum [12] entworfen. Mit Hilfe dieses Entscheidungsbaumes kann der Prozesszustand durch eine Klassifizierung von Prozessdaten (Online-Merkmale nach 4.2.3 und Offline-Analyseergebnisse) in die vier Klassen beschrieben werden (Tafel 1).

Tafel 1: Mögliche Prozesszustände für die anaerobe Biogasfermentation in Bezug auf das Softsensorsystem mit Angabe zugehöriger Prozessführungsempfehlungen

Prozesszustandsklasse		Prozessführungsempfehlung
Nr.	Name	
1	Unterlast	*QS* erhöhen
2	Stabiler Betrieb	*QS* beibehalten oder ggf. leicht erhöhen
3	Leichte Versäuerung	*QS* beibehalten oder ggf. leicht absenken
4	Säureakkumulation	*QS* zwingend senken und Prozess beobachten

Der Entscheidungsbaum nutzt die Prozessparameter:

- *GPR* (online)
- *CH4_CO2_d* (online)
- $c_{GS,Dest.}$ (Konzentration an Gesamtsäure, offline ermittelt mit Hilfe des Destillationsverfahrens [12])
- $c_{ES,HPLC}$ (Konzentration an Essigsäure, offline ermittelt durch Einsatz der Hochleistungsflüssigkeitschromatographie (HPLC))
- $c_{PS,HPLC}$ (Konzentration an Propionsäure, offline ermittelt durch Einsatz der HPLC))

Die Offline-Analysen von Fermenterproben sind nur im Rahmen der Generierung des Trainingsdatensatzes notwendig (für den Recalldatensatz als Überprüfungsgrundlage). Aufgabe des FKCN als Kern des Softsensors ist eine Klassifizierung der Merkmalswerte auf Basis der online erfassten Prozessparameter in eine Prozesszustandsklasse (unüberwachtes Lernen). Die Labelingphase ordnet den Klassen deren Bedeutung zu. In der Anwendung (Recall) klassifiziert das FKCN die Online-Merkmalswerte eigenständig ohne die Verwendung des Entscheidungsbaumes und ohne Offline-Prozessparameter in eine der vier Prozesszustandsklassen. Die EmSoft-Realisierung des FKCN arbeitet mit einer zweidimensionalen Neuronenkarte mit 12 Neuronen je Dimension.

3.6 Zielhardware

Als Zielhardware kam für das Softsensorsystem der Mikrocontroller 80C167 der Infineon Technologies AG [11] zum Einsatz. Dieser Controller weist eine Verarbeitungsbreite von 16 Bit und eine Taktfrequenz von 20 MHz auf.

Der RT-C-Code des Application-Builder, als Ergebnis des Designprozesses, wurde unter Verwendung eines C-Compilers (Tasking Embedded Development Environment® (EDE), Toolchain: C/C++ for C166/ST10®) in den entsprechenden Maschinencode übersetzt und auf den Flash-Speicher der Zielhardware übertragen.

Ein Kompaktgerät, das mit dem µC 80C167 arbeitet und um einige Features erweitert wurde, zeigt Bild 1 an der Laborbiogasanlage.

3.7 Testergebnisse

Die Systemtests des Softsensors wurden durch die Beaufschlagung der Eingänge des Systems mit den Online-Prozessmerkmalen (PC-Simulation (EmSoft)) bzw. Online-Prozessdaten (µC-Simulation mit Hilfe eines Simulationssystems mit hardwaremäßiger Signalausgabe) des Trainings- und des Recalldatensatzes sowie an der Laborbiogasanlage durchgeführt. Bild 5 zeigt relevante Merkmalsverläufe, die Vorgaben für die zu detektierende Prozesszustandsklasse und die Klassifizierungsergebnisse des FKCN als Monitor zusammen mit einer Darstellung von Fehlklassifikationen. Das FKCN eignet sich sehr gut zur gewünschten Prozesszustandsklassifikation auf Grundlage der erläuterten Online-Merkmale. Sowohl im Test mit Trainingsdaten als auch bei der Anwendung des Systems auf Datensätze, die nicht für den Lernvorgang verwendet wurden (Recall), konnte der Prozesszustand fehlerfrei detektiert werden und entsprach damit der Einschätzung, die mit Hilfe des Entscheidungsbaumes getroffen wurde.

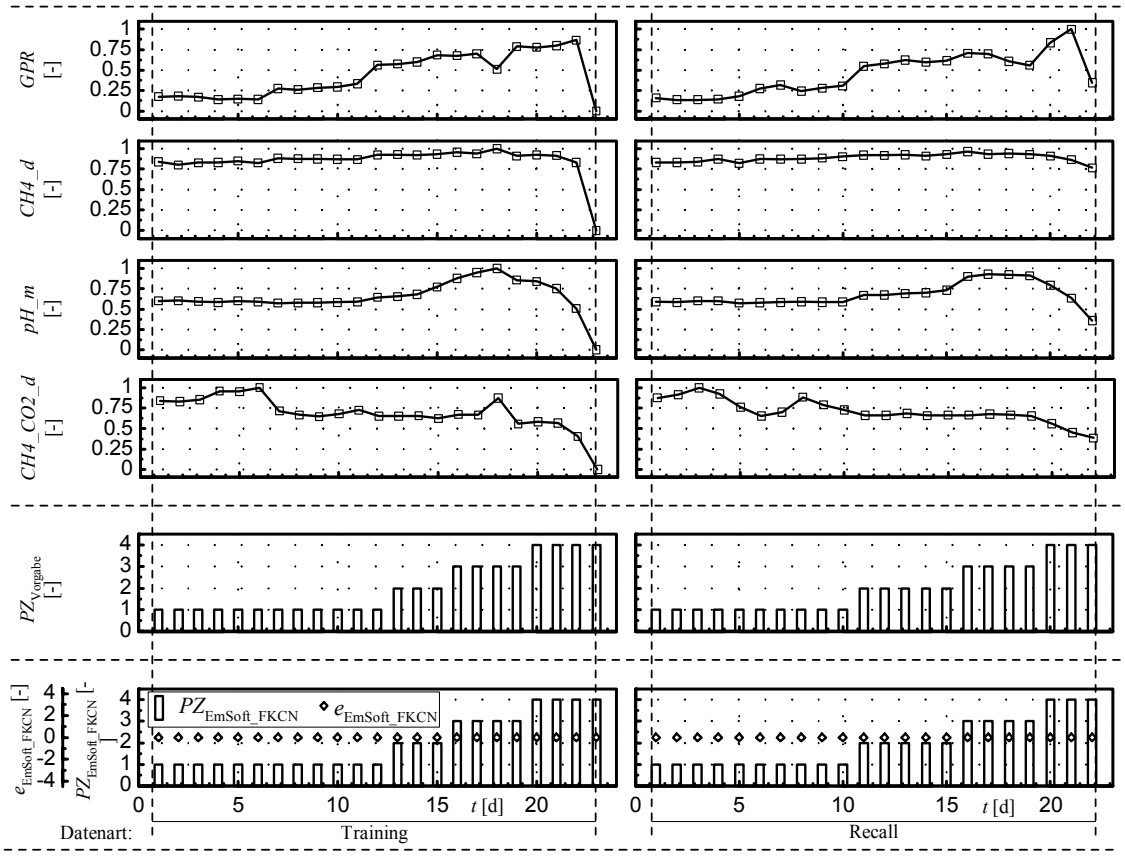


Bild 5: Testergebnisse des Softsensors in Form ausgewählter Merkmalsverläufe (Eingänge des Klassifikators), der gewünschten Prozesszustandsklasse (PZ), der durch das FKCN errechneten PZ sowie einer Fehlerdarstellung (Unterteilung in Trainings- und nicht gelernte Recalldaten)

4 Zusammenfassung und Ausblick

Die Testläufe (Simulation und Test an der Laborbiogasanlage) des mikrocontrollerbasierten Softsensors in dessen Konzeption als Prozessmonitor verliefen erfolgreich und stellten die Eignung zur Prozessführung von Biogasanlagen unter Beweis. Eine Überführbarkeit des Monitors an reale Biogasanlagen mit vergleichbarer Betriebsart und ähnlichen Substraten ist unter Anwendung entsprechender Scale-up Vorschriften gegeben.

EmSoft verwendet ein Konzept, welches im generierten RT-C-Code die gleichen Anwendungsalgorithmen nutzt, die auch in den Programmbausteinen für die Simulationsphase des Programms auf dem PC implementiert wurden. Dadurch ergaben sich nur sehr geringe und zu vernachlässigende Abweichungen der Ergebnisse der PC-Softwaresimulation mit den Merkmalswerten (Model-Builder) im Vergleich zu den Ergebnissen die nach Überführung auf den µC bzw. DSP (Application-Builder, C-Compiler der Zielhardware, Transfersoftware) unter Einsatz eines Simulationssystems mit hardwaremäßiger Signalausgabe der online erfassten Prozessgrößen erzielt wurden.

Neben der vorgestellten Systemvariante für Prozessführungssysteme an Biogasanlagen sind auch Realisierungen anderer Strategien zur Prozessführung denkbar und möglich. Die Flexibilität, die Bereitstellung einer Vielzahl von Algorithmen für Merkmalsextraktionen, Regler und Monitore, sowie die Einordnung in den transparenten und

universellen Designprozess für PFS des Softwaresystems EmSoft erleichtert dabei die Entwicklung und Umsetzung weiterer Systemrealisierungen in erheblichem Umfang.

In weiteren Arbeiten ist geplant, das vorgestellte Softsensor-/Monitoringsystem an realen Biogasanlagen zu installieren um die Praxistauglichkeit unter Beweis stellen zu können. Neben dem Monitoringsystem werden zusätzliche PFS für Biogasanlagen unter Anwendung des Designprozesses unter Anwendung von EmSoft erstellt, von denen ein Fuzzy-Control-System zur automatischen Regelung der Substratzufuhr bereits realisiert wurde [7, 8]. Ein Vorhersagesystem für den Gasertrag der Anaerobfermentation ist derzeit in der Entwicklungsphase.

Die Arbeiten des vorgestellten Themas wurden im Rahmen eines Forschungsprojektes von der Arbeitsgemeinschaft industrieller Forschungsvereinigungen „Otto von Guericke“ e.V. (aif) – Projektträger des BMBF im Rahmen des Programms aFuE (Förderkennzeichen: 17.113.01) sowie durch das Kultusministerium des Landes Sachsen-Anhalt (Förderkennzeichen: 3498A/0203L) gefördert.

5 Literatur

[1] BEHMEL, U., MEYER-PITTROFF, R.: Risiken bei der Cofermentation organischer Reststoffe in Biogasanlagen. *Korrespondenz Abwasser* 43(12):2172-2179, 1996.

[2] BEZDEK, J. C., TSAO, E.C.-K., PAL, N.R.: Fuzzy Kohonen Clustering Network, in *Proceedings of 1992 IEEE International Conference on Fuzzy Systems*, S. 1035-1043, New York, IEEE Press, 1992.

[3] KOCH, M., KUHN, T., WERNSTEDT, J.: *Fuzzy Control - Optimale Nachbildung und Entwurf optimaler Entscheidungen*. München, Oldenbourg Verlag, 1996.

[4] KRAMER, K.-D., PATZWAHL, S., NACKE, T., FRENSE, D.: Computational Intelligence - Design Tool for Use in Biotechnological Process Monitoring, in *EUNITE 2002, European Symposium on Intelligent Technologies, Hybrid Systems and their Implementation on Smart Adaptive Systems - Programme, Abstracts of the Papers and Proceedings on CD-Rom*, S. 47, Aachen, Verlag Mainz - Wissenschaftsverlag, ISBN: 3-89653-919-1, 2002.

[5] KRAMER, K.-D., PATZWAHL, S., NACKE, T., FRENSE, D.: Complete Algorithm to Realise CI Modelbased Control and Monitoring Strategies on Microcontroller Systems, in *Soft Computing Systems: Design, Management and Applications*, S. 785-795, Abraham A., Ruiz-del-Solar, J., Köppen, M. (Hrsg.), Amsterdam, IOS Press, ISBN: 1 58603 297 6, 2002.

[6] NACKE, O.: *Biogas von A bis Z*, Firmendruck. Bielefeld, Borsig Energy GmbH, 2001.

[7] PATZWAHL, S., NACKE, T., FRENSE, D., BECKMANN, D., VOLLMER, G.-R., TAUTZ, T., KRAMER, K.-D.: Microcontroller-based Fuzzy System to Optimize the Anaerobic Digestion in Biogas Reactors, in *Computational Intelligence. Theory and Applications. International Conference, 7th Fuzzy Days Dortmund, Proceedings (Lecture Notes in Computer Science Vol. 2206)*, S. 2-10, Reusch B. (Hrsg.), Berlin, Heidelberg, u.a., Springer Verlag, ISBN: 3-540-42732-5, 2001.

[8] PATZWAHL, S., KRAMER, K.-D., NACKE, T.: Computational Intelligence auf Mikrocontrollersystemen am Beispiel eines Fuzzy-Systems zur Optimierung von Biogasanlagen, in *Proceedings 12. Workshop Fuzzy Systeme, Wissenschaftliche*

Berichte FZKA 6767, Karlsruhe: Institut für Angewandte Informatik, S. 201-210, Mikut R., Reischl, M. (Hrsg.), Karlsruhe, Eigenverlag des Forschungszentrum Karlsruhe GmbH, ISSN: 0947-8620, 2002.

[9] RETZLAFF, G., RUST, G., WAIBEL, J.: *Statistische Versuchsplanung - Planung naturwissenschaftlicher Experimente und ihre Auswertung mit statistischen Methoden*. Weinheim, New York, Wiley/VCH, 1987.

[10] SCHATTNER, S.: Methanbildung verschiedener Substrate - Kenntnisstand und offene Fragen, in *Energetische Nutzung von Biogas: Stand der Technik und Optimierungspotenzial*, Gülzower Fachgespräche, Band 15, S. 28-39, Gülzow, Fachagentur Nachwachsende Rohstoffe e.V. (FNR), 2001.

[11] SCHULTES, R., POHLE, I.: *80C166 Mikrocontroller - Einführung, Applikation, Programmierung der C167/C165-Bausteine*. Poing, Franzis Verlag, 1994.

[12] SEIFERT, T.: *Klassifikation von anaeroben Biogasprozessen mit Modellsubstraten durch die Erfassung von Online/Offline- Daten und Verwendung neuronaler Netze*. Diplomarbeit, Jena, Fachhochschule Jena, FB Medizintechnik, 2003.

Computational Intelligence-Regler (CI-Controller)

Ulrich Lehmann, Jörg Krone, Johannes Brenig, Udo Reitz,
Fachhochschule Südwestfalen Campus Iserlohn, CIC.Lab
lehmann@fh-swf.de

1. Ausgangslage bei der Planung des Projektes

Von der Industrie Systeme für Advanced Control (Fortschrittliche Regelungstechnik) basierend auf linearen Prozessmodellen und konventioneller Prozessidentifikation angeboten (L1).
Die Methoden der Computational Intelligence haben in der Industrie, dies war gut auf den einschlägigen Messen (L4, L5) zu beobachten, noch keine große Verbreitung gefunden.
Im Bereich Fuzzy-Control (L2) gibt es eine Durchdringung der industriellen Anwendungspraxis im Bereich Verbrennungsregelung für Müllheizkraftwerke. Neuronale Netze werden häufig für die Mustererkennung (Bildverarbeitung, Sensortechnik allgemein) eingesetzt (L3).
Die erfolgversprechende Kombination aus Fuzzy-Logik, Neuronale Netzen und evolutionären Algorithmen für die Automatisierung, insbesondere Regelung, von Prozessen findet man bisher nur vereinzelt in Forschungslabors, aber eher selten in praktischen Anwendungen.

2. Darstellung der Ziele;

Im gleichnamigen Projekt wird ein Computational Intelligence-Regler (CI-Regler) entwickelt. Der CI-Regler soll sich ohne Expertenwissen an die Regelstrecke adaptieren und auch bei nichtlinearen und zeitvarianten Regelstrecken ständig auf optimale Regelgüte einstellen.
Kommerzielle Tools zur Reglerentwicklung und Simulation (Matlab/Simulink oder Winfact) werden eingesetzt, um den CI-Regler sowohl in der Simulationsumgebung als auch in der Zielumgebung betreiben zu können. Darüber hinaus ist die Entwicklungsumgebung im Arbeitskreis COIN abgestimmt und eignet sich als Plattform für den Austausch und gegenseitigen Test von Zwischenergebnissen.
Zur praktischen Erprobung der selbsttätigen regelungstechnischen Optimierung und Adaption wird der CI-Regler an einer realen nichtlinearen, zeitvarianten Regelstrecke (Belichtungsregelung) in Betrieb genommen und getestet.

3. Darstellung der technisch/wissenschaftlichen Ergebnisse;

Das Ergebnis (siehe Abb. 1) des Projektes ist ein funktionsfähiger Prototyp eines CI-Reglers (L4) mit Komponenten zur Identifikation von Regelstrecken, Modellbildung des Prozesses, Evolutionärer Optimierung des CI-Reglers im Offline-Betrieb, Sammlung von Wissen über die Regelstrecke (Trainingsdaten) und zur Optimierung der Regelgüte mit einem Künstlich Neuronalen Netz (KNN) im Online-Betrieb in Echtzeit. Alle Module des CI-Reglers arbeiten im Zeitbereich und sind daher für den Anwender und den Systemtechniker maximal transparent. Der Kern des CI-Reglers besteht aus einem industriellen PID-Reglers. Dadurch erhält er hervorragende Notlaufeigenschaften. Dies ist wichtig für den Fall, wenn die Adaptionsmechanismen bedingt durch unzulässige Störsignale zu einer ungünstigen Prozessgüte führen. Ein wichtiges Argument für die Akzeptanz eines CI-Reglers in industriellen sicherkritischen Anwendungen.

Es wurde darüber hinaus ein Konzept entwickelt, um die CI-Regler-Architektur auf einen Kompaktregler mit Netzwerkanbindung zu übertragen. In diesem Fall wäre ein Server/Client-

Verbund zur Regelung mit CI-Unterstützung erforderlich. Die Echtzeitfunktionen liegen im Kompaktregler, so dass durchaus Abtastzeiten im Millisekundenbereich erreichbar sind.

4. Verwertung der Ergebnisse;

Durch die Einbeziehung der mittelständischen Industriepartner in das Projekt ergeben sich vielfältige Möglichkeiten zur Umsetzung der Forschungsergebnisse in die industrielle Anwendung.

Die Vollmer Messtechnik GmbH in Hagen ist ein mittelständisches Unternehmen. Das Unternehmen, das im Bereich der Mess- und Automatisierungstechnik weltweit tätig ist, kann durch die Verwendung der Ergebnisse aus diesem Projekt wartungsärmere Produkte für den Weltmarkt entwickeln und dadurch die Inbetriebnahme verkürzen sowie Maintenance-Aufwände und Service-Kosten für komplexe Regelsysteme einsparen.

Die SIHK Hagen unterstützt nachdrücklich die Aktivitäten der FH Südwestfalen im Bereich der CI-Technologien. Es ist geplant, an der FH Südwestfalen (Iserlohn, Hagen, Meschede, Soest) standortübergreifend eine Kompetenzplattform Computer Vision and Computational Intelligence einzurichten.
Durch die Präsentation der COIN-Forschungsergebnisse auf der Hannover Messe 2003 sind neben mittelständischen auch große Industrieunternehmen auf das TRAFO-Verbundprojekt aufmerksam geworden und suchen den Kontakt zu den einzelnen Projektgruppen an den beteiligten Hochschulen.

5. Darstellung der Ergebnisse des Projektes;

Die FuE-Strukturen wurden an der Fachhochschule Südwestfalen (FH SW) nachhaltig gestärkt, es wurde ein Konzept für eine Computer Vision and Computational Intelligence (CV&CI)-Kompetenzplattform beim Ministerium für Wissenschaft und Forschung NRW vorgestellt.

Ein neues Wahlfach interdisziplinäres Studienfach Neuro-Fuzzy-Systeme an der FH SW stellt die Aktualität und Praxisorientierung der Lehre in diesem Bereich sicher. Das TRAFO-Verbundvorhaben COIN hat die interdisziplinäre wissenschaftliche Zusammenarbeit der Fachhochschulen Köln mit Gummersbach, Bielefeld, Bochum sowie Südwestfalen mit Iserlohn und Hagen gestärkt.

Durch die Messeteilnahme Hannover Messe 2003 und zahlreichen Veröffentlichungen (u.a. L1, L3, L4), sowie die Eröffnung eines neuen Labors CIC.Lab (Computational Intelligence and Control Laboratory) an der FH Südwestfalen in Iserlohn Ende Juni 2003 wurde die Attraktivität und Konkurrenzfähigkeit der Fachhochschule in der anwendungsorientierten Forschung und Lehre gesteigert

Zahlreiche Mitarbeiter, Diplomanden und studentische Hilfkräfte waren direkt in dem Forschungsvorhaben eingebunden. Die Studierenden der höheren Semester indirekt über Vortäge und Ausarbeitungen zu Detailaufgaben des Forschungsvorhabens CI-Regler.
Ein Projektmitarbeiter ist direkt zum kooperierenden Unternehmen gewechselt, dies ist der Idealfall für den Technologietransfer.

6. Am Projekt beteiligte industriellen und wissenschaftliche Partner;

- Volmer Messtechnik GmbH - Hagen
- Prof. Dr. Stefan Dormeier, AK COIN – FH Bielefeld
- Prof. Dr. Michael Bongards, AK COIN (Sprecher) - FH Köln Campus Gummersbach
- Prof. Dr. Hermann Johannes, AK COIN - FH Südwestfalen Campus Hagen
- Prof. Dr. Jörg Krone und Burkhard Neumann, FB Informatik und Naturwissenschaften, FH Südwestfalen Campus Iserlohn

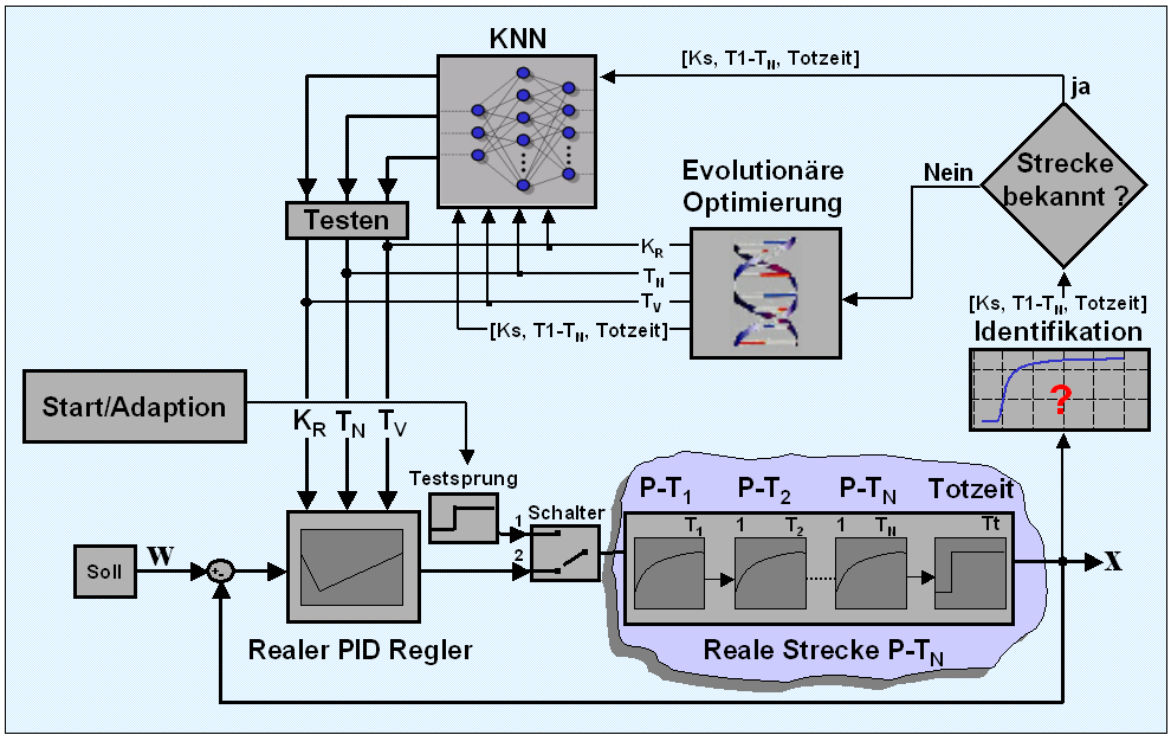


Abbildung 1: Architektur des CI-Controllers (CI-Regler) mit realem PID-Regler im Kern, Identifikation von unbekannten Regelstrecken, evolutionärer Optimierung der Reglerparameter und KNN (Künstlich Neuronales Netz) für die Echtzeit-Adaption

7. *Literaturverzeichnis*

L1 Michael Bongards, Hartmut Westenberger: Regelungstechnische Optimierung von verfahrenstechnischen Anlagen. TRAFO-Verbundprojekt COIN (Computational Intelligence for Industry), Zwischenbericht Juni 2003, Gummersbach

L2 Pfeiffer, B.-M.; Jäkel, J.; Kroll, A.; Kuhn, C.; Kuntze, H.-B.; Lehmann, U.; Slawinski, T.; Tews, T.: Erfolgreiche Anwendungen von Fuzzy Logik und Fuzzy Control (Teil 1+2). Automatisierungstechnik at 10 und 11 (2002), Oldenburg Verlag, S. 461 –471.

L3 Lehmann, U.; Krone, J.; Neumann, B. et. al.: Computer Vision and Computational Intelligence, Präsentation eines Antrages für eine Kompetenzplattform vor der Jury des Ministeriums für Forschung und Wissenschaft NRW in Köln im Juli 2002

L4 Lehmann, U.; Krone, J.; Brenig, J.; Reitz, U.: CI-Controller für nichtlineare Regelstrecken. Gemeinschaftsstand des Arbeitskreises COIN (Computational Intelligence for Industrie) auf der Hannover Messe Industrie 2003 Bongards, M.

(Sprecher); Johannes, H.; Lehmann, U. et. al.: CI-Systeme, Katalog Forschungsland Nordrhein-Westfalen zur Hannover Messe 2003

L5 Lehmann, U. (Hsg.); Bongards, M.; Johannes, H.: Neuronale Fuzzy-Logik, Neuro-Fuzzy-Regler, Mustererkennung, Identifikation, Adaption, Data Mining; Reihe Informatik/Kommunikation. VDI-Fortschrittsbericht, Düsseldorf, VDI-Verlag (2000); ISBN 3-18-3648 10-5.

L6 Lehmann, U.; Dormeier, S. u.a.: Trainierbarer Neuro-PID-Regler für hohe Regelgüte. GMA Workshop Fuzzy-Control, Dortmund/Witten, November 2000.

DynStar – Ein Simulationssystem für Verfahrens- und Automatisierungstechniker

D. Fiß, N. Chaker, R. Hampel

„Institut für Prozeßtechnik, Prozeßautomatisierung und Meßtechnik“ (IPM)
„Hochschule Zittau / Görlitz (FH)“
02763 Zittau, Theodor-Körner-Allee 16
Tel. (03583) 611287
Fax (03583) 611288
E-Mail: d.fiss@hs-zigr.de

1 Einführung

DynStar ist ein Programm zur Simulation des dynamischen Verhaltens technologischer Prozesse, für grafikorientierte Betriebssysteme. Es wird speziell für die Anforderungen der Automatisierungstechnik entwickelt. Das Simulationssystem ermöglicht die Steuerung und Regelung von Prozessen in Echtzeit mit Reaktionszeiten bis zu einer Millisekunde. DynStar unterstützt verteiltes Rechnen, d.h. es können Simulationen auf mehrere miteinander vernetzte Rechner aufgeteilt werden. Neben den Grundübertragungsgliedern sind arithmetische Funktionsblöcke, Signalgeneratoren, Logik- und Schaltfunktionen, Funktionsblöcke zur Visualisierung und Archivierung von Daten, Kraftwerkskomponenten, eine Fuzzy-Shell implementiert. Durch die Auslagerung der Funktionsblöcke in Software-Bibliotheken (FBLs) ist das Simulationssystem DynStar leicht erweiterbar. Die Einsatzmöglichkeiten reichen von der Verwendung für Lehrzwecke bis hin zur Lösung von Industrieaufgaben.

DynStar stellt Entwicklern, Herstellern und Planern von Kraftwerken und Kraftwerkskomponenten ein Werkzeug zur Simulation von Kraftwerksprozessen für die Projektvorbereitung, die Projektierung und der Inbetriebsetzung von Kraftwerken zur Verfügung. Kraftwerksbetreiber können das Simulationssystem u.a. für die Betriebsführung und Ausbildung einsetzen. Das Simulationssystem DynStar ist in der Lage, stationäre Zustände einer Anlage oder einzelner Komponenten sowie die Dynamik bei Übergangsprozessen zu berechnen.

Durch Testrechnungen und automatisierte Variationsrechnungen können sowohl optimale Strukturen entwickelt als auch Parameteroptimierungen durchgeführt werden. Durch Einstellen beliebiger (auch kritischer) Betriebszustände ist die Analyse von Störfällen möglich.

2 Allgemeine Beschreibung

2.1 Übersicht

DynStar ist modular aufgebaut und setzt sich aus dem Hauptprogramm und Funktionsblockbibliotheken(FBL) zusammen. Die Funktionsblockbibliotheken enthalten ein großes Spektrum an Funktionselementen, z.B.:

- Proportionale Elemente, Differentierer, Integrierer, Totzeitglied, etc.
- Arithmetik-Blöcke wie z.B.: Addition, Subtraktion, Multiplikation, etc.
- PT1-, PT2-Elemente
- Nichtlineare Funktionsblöcke wir z.B.: Hysterese, etc.
- Fuzzy-Systeme, Linguistische Variablen, Fuzzy-Sets, Regeln, etc.
- Kraftwerkskomponenten (Turbinen, Pumpen, Ventile, etc.)

Simulationsmodelle werden schnell mit Maus-Operationen (Drag&Drop) erstellt. Oft verwendete Funktionsblockstrukturen können zu Makros zusammengefaßt werden.

Die Vorgehensweise der Modellbildung in DynStar wird am Beispiel der Van der Pol Gleichung, ein einfaches nichtlineares Modell zweiter Ordnung, gezeigt. Die Grundlage der Modellbildung in DynStar sind die Differentialgleichungen. Die Van der Pol Gleichung lautet:

$$\ddot{y} + (y^2 - 1) \cdot \dot{y} + y = 0$$

Sie kann auch wie folgt beschrieben werden:

$$\ddot{y} = (1 - y^2) \cdot \dot{y} - y$$

Das entsprechende Funktionsblockdiagramm wird in Abbildung 1 gezeigt.

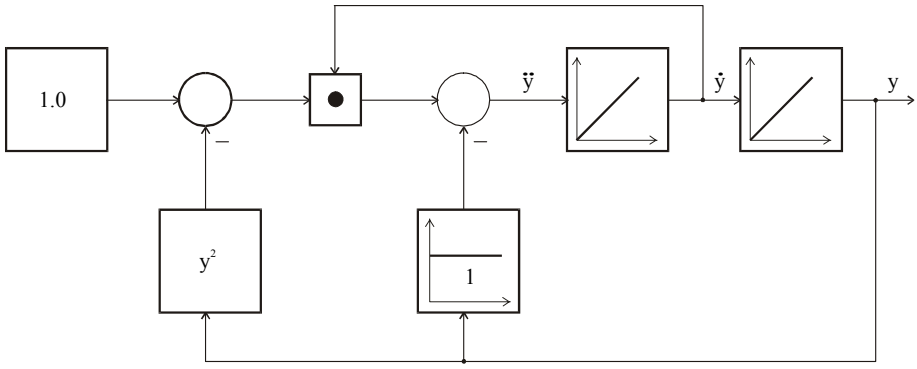


Abbildung 1: Funktionsblockdiagramm der Differentialgleichung

Auf der Grundlage des Funktionsblockdiagrammes wählt man die entsprechenden Funktionsblöcke aus den FBLs aus und verbindet sie mit Hilfe von Mausklicks. Abbildung 2 zeigt die Umsetzung in Dynstar und das Simulationsergebnis.

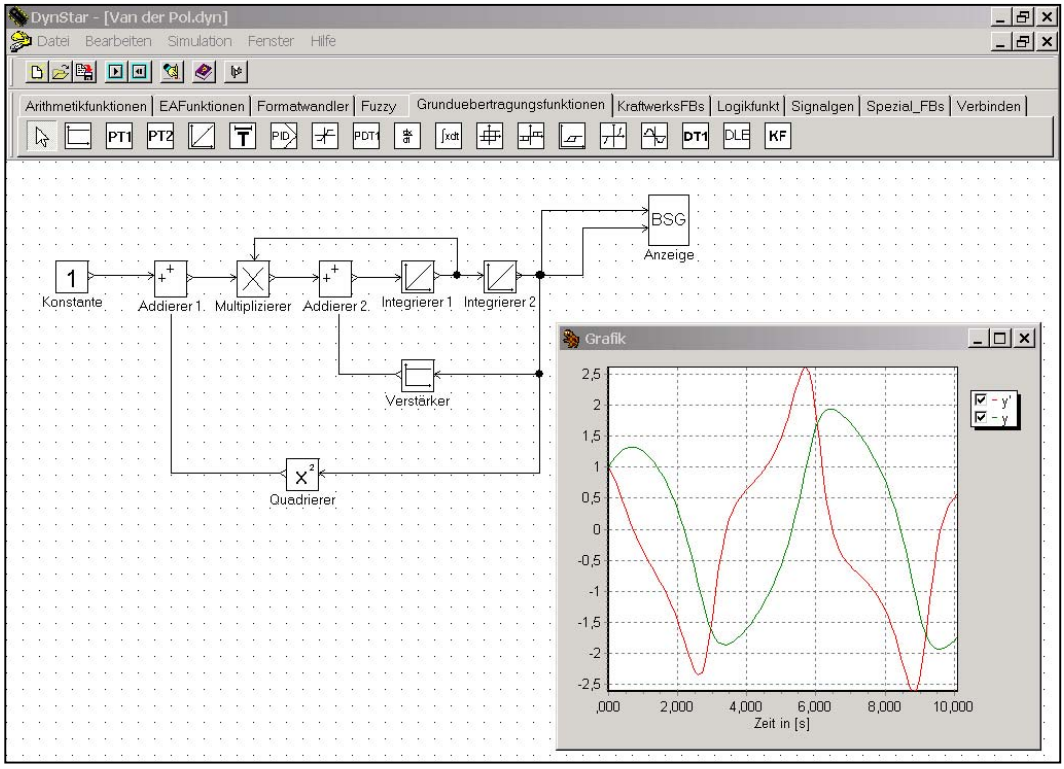


Abbildung 2: Differntialgleichung in DynStar

2.2 Oberfläche

Die graphische Oberfläche setzt sich aus mehreren Komponenten mit bestimmten Aufgaben zusammen. Die in Abbildung 3 dargestellt und in der Tabelle 1 näher beschrieben sind.

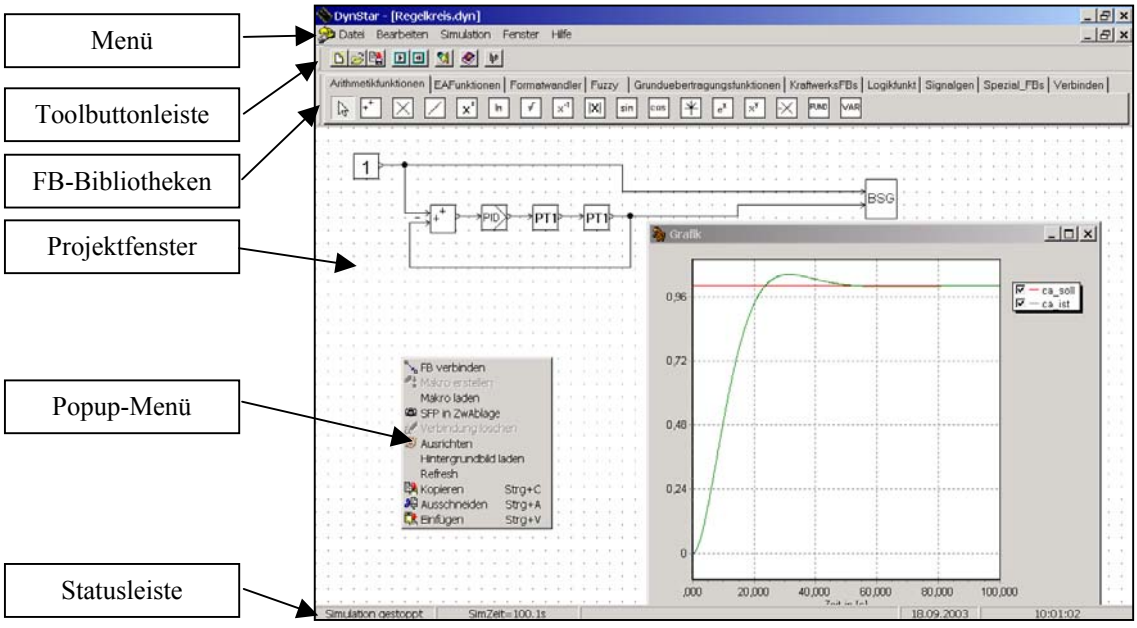


Abbildung 3: Oberfläche

Die benötigten Objekte und deren Aufgabe sind in Tabelle1 zusammengestellt:

Tabelle 1: Aufgaben der einzelnen Komponenten

Komponente:	Aufgabe:
Hauptfenster	Enthält das Menü, Toolbuttonleiste, Statusleiste, Funktionsblockbibliotheken (FBL) und alle Projektfenster
Menü	Enthält windowtypische Befehle
Toolbuttonleiste	Enthält leicht erreichbar sehr häufig verwendete Befehle, z.B. Simulation Start/Stop
FBL-Leiste	Enthält alle verfügbaren FBLs und deren Funktionsblöcke
Statusleiste	Informiert über den Simulationszustand und zeigt Zeit und Datum an
Projektfenster	Enthält alle Verbindungen und Funktionsblöcke eines Modells
Popup-Menü	Enthält schnellverfügbare und kontextbezogene Funktionen

Menüstruktur

Die Funktionen des Menüs sind in Tabelle 2 aufgelistet.

Tabelle 2: Menü

Menüpunkt	Kurze Beschreibung
Datei	Befehle zur Projektdateiverwaltung (Neu, Öffnen, Speichern, Speichern unter und Beenden)
Bearbeiten	Befehle zur Projektbearbeitung (z.B. Kopieren, Ausschneiden, Einfügen,...)
Simulation	Befehle zur Simulationssteuerung
Fenster	Befehle zur Fensterverwaltung
Hilfe	Online-Hilfe und Programminformationen

Popup-Menü

Das Popup-Menü wird windowstypisch mit der rechten Maustaste aktiviert. Die verfügbaren Funktionen richten sich danach über welcher graphischen Komponente sich die Maus befindet.

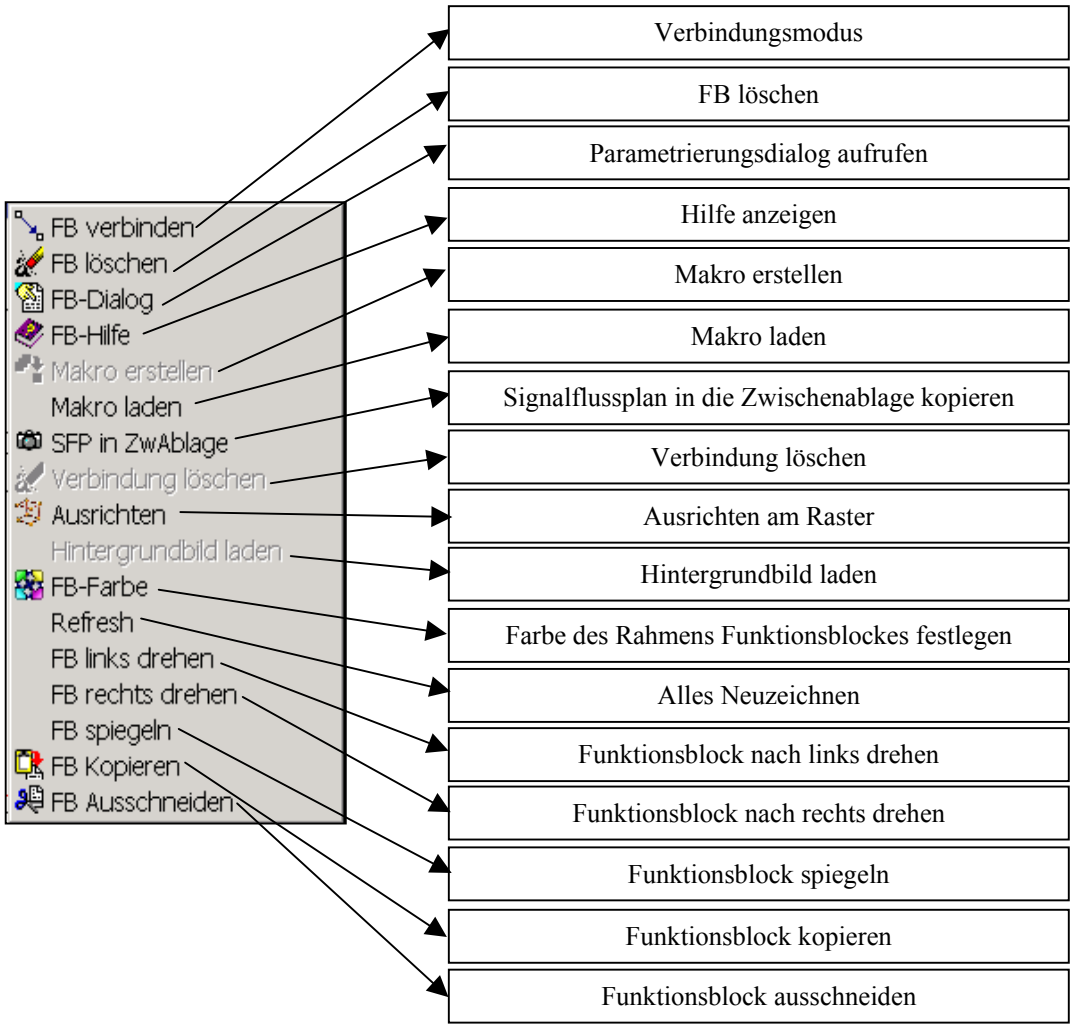


Abbildung 4: Popup-Menü

2.3 Fuzzy-Shell

Übersicht

Das Konzept von Dynstar ist es einen Fuzzy-Controller genau so wie einen Funktionsblock zu benutzen. Der erste Schritt ist es einen Controller anzulegen und dann die benötigten linguistischen Variablen für die Eingänge bzw. Ausgänge des Controllers festzulegen. Dazu benutzt man die Fuzzy-Shell, die man durch doppelklicken auf den Fuzzy-Block öffnet. Die Fuzzy-Shell wird benutzt für die:

- Definition der Anzahl der Eingänge und Ausgänge der Fuzzy-Controller (Zweidimensionaller Controller, Mehrdimensionaler Controller)
- Definition der linguistischen Variablen
- Definition der Anzahl und des Typs der Fuzzy-Sets jeder linguistischer Variable
- Definition der Bezeichnung des Fuzzy-Controllers
- Definition der Regelbasis für den Controller
- Definition der Fuzzy-Operatoren und Defuzzifizierungsmethoden

Die Shell ist in 4 Bereiche (Controller, linguistische Variablen, Regeln und Kennfeld) unterteilt. Jedem Teilbereich wird ein Arbeitsblatt zugeordnet.

Im Arbeitsblatt „Controller“ erstellt oder löscht man Fuzzy-Controller. Aus der Liste der existierenden Controller, wählt man einen Controller aus und weist ihn auch dem Fuzzy-Block zu.

Bearbeitung der Linguistischen Variablen

Das Arbeitsblatt „Linguistische Variable“ verwaltet alle erzeugten linguistischen Variablen. Sie sind in drei Gruppen eingeteilt. Die erste Gruppe enthält die Eingangsvariablen des ausgewählten Fuzzy-Controllers. Die zweite Gruppe entsprechen den Ausgängen und die dritte Gruppe linguistischer Variablen stehen zur freien Verfügung und sind dem ausgewählten Controller nicht zugeordnet (Abbildung 5).

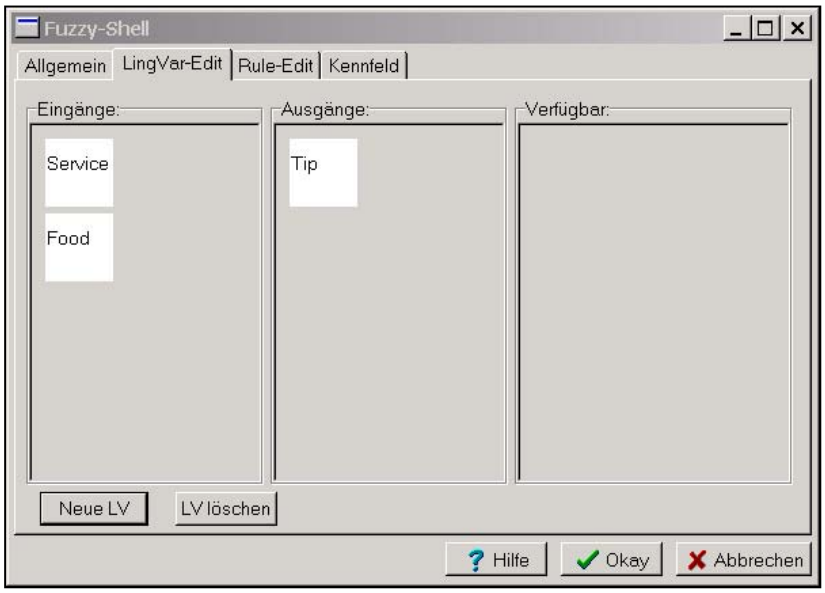


Abbildung 5: Arbeitsblatt für linguistische Variable

Fuzzy-Set-Editor

Der Fuzzy-Set-Editor wird gestartet indem man mit der Maus auf die entsprechende linguistische Variable doppelklickt. Folgende Möglichkeiten stehen zur Verfügung:

- hinzufügen von Fuzzy-Sets
- modifizieren von Fuzzy-Sets

- umbenennen von Fuzzy-Sets
- löschen von Fuzzy-Sets
- ändern des Typs der Fuzzy-Sets

Die Abbildung 6 zeigt beispielhaft die Fuzzy-Set-Definition von der linguistischen Variable „Service“. Die Spreizung, die Verteilung, die Form der Fuzzy-Sets, etc. können mit wenigen Mausklicken schnell geändert werden. Jeder Fuzzy-Set kann über die Maustaste einzeln ausgewählt werden und auf der rechten Fensterseite separat bearbeitet werden.

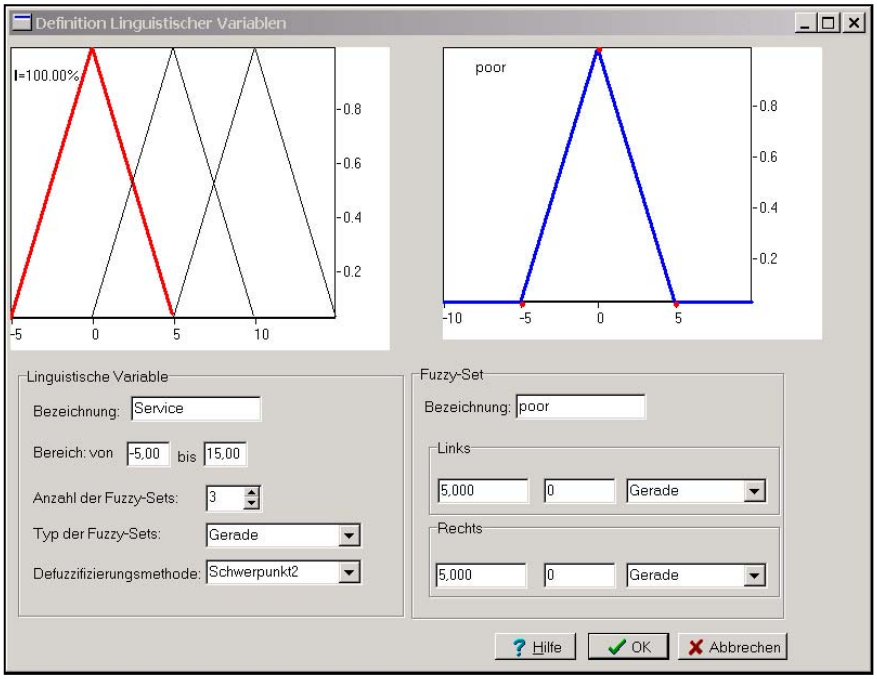


Abbildung 6: Fuzzy-Set-Editor

Eine Zusammenstellung der möglichen Formen der Fuzzy-Sets werden in Abbildung 7 dargestellt. Andere Fuzzy-Set-Formen können durch Kombination der unterschiedlichen Typen erzeugt werden. Folglich ist es möglich, verschiedene Formen der Fuzzy-Sets zu generieren wie z.B.: trapezförmige, dreieckige, rechteckige, normalverteilte Fuzzy-Sets, usw..

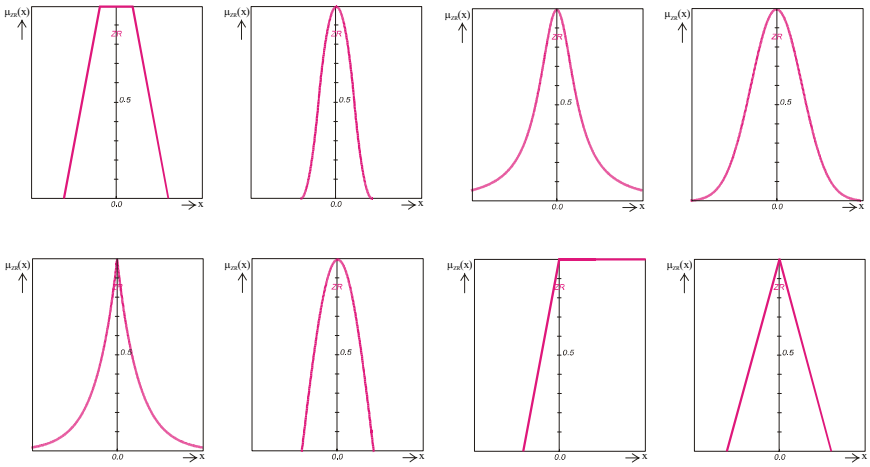


Abbildung 7: Mögliche Varianten von Fuzzy-Sets

Das Arbeitsblatt „Regeln“ teilt sich in zwei Bereiche. Im oberen Teil sind alle gültigen Regeln aufgelistet, die bereits erstellt wurden und im unteren Bereich werden sie bearbeitet. Man stellt für jede verwendete linguistische Variable die gültige Bedingung ein und wählt einen Fuzzy-Operator aus, um die Variablen zu verknüpfen. Jeder Regel wird ein Wichtungsfaktor zwischen 0 und 1 zugewiesen. Die editierte Regel wird mit einem Klick auf den Button „Hinzufügen“ dem Fuzzy-Controller zugeordnet.

Bestehende Regeln können gegebenenfalls auch geändert bzw. gelöscht werden. Zum Löschen wählt man die gewünschte Regel aus und klickt auf den Button „Löschen“. Zum Ändern klickt man ebenfalls auf die gewünschte Regel und ändert im Editorbereich die entsprechenden Parameter und betätigt den Button „Ändern“, um die Änderungen zu übernehmen (Abbildung 8).

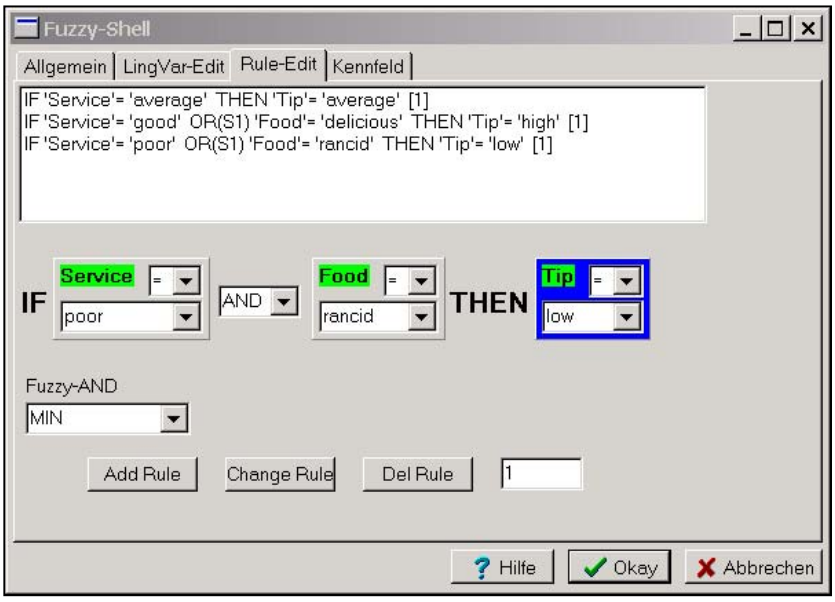


Abbildung 8: Arbeitsblatt für die Regeln

Das Arbeitsblatt „Kennfeld“ stellt das resultierende Kennfeld dar. Werden dem Fuzzy-Controller mehr als zwei Eingänge bzw. mehr als ein Ausgang zugeordnet, müssen die darzustellenden Eingänge bzw. Ausgänge ausgewählt werden.

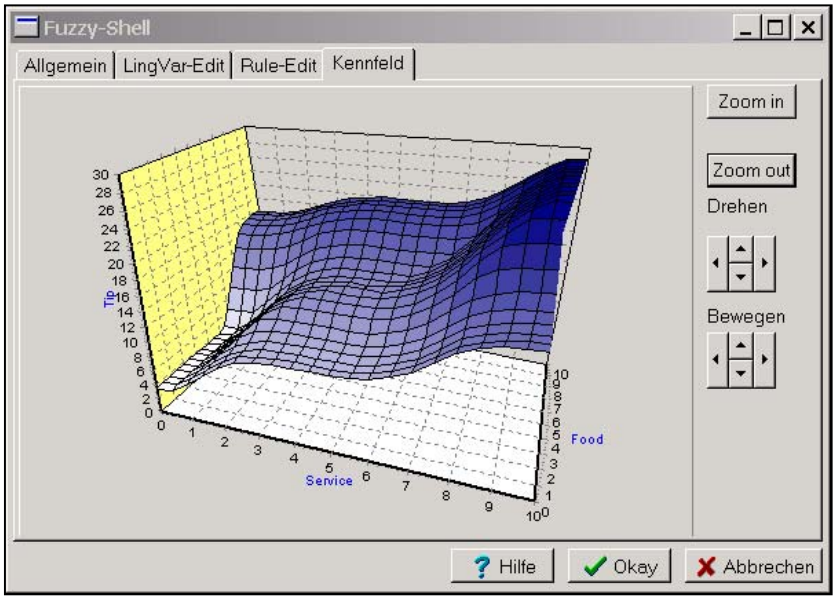


Abbildung 9: Arbeitsblatt Kennfeld

2.4 Kraftwerkskomponentenbibliothek

Systemkonzept

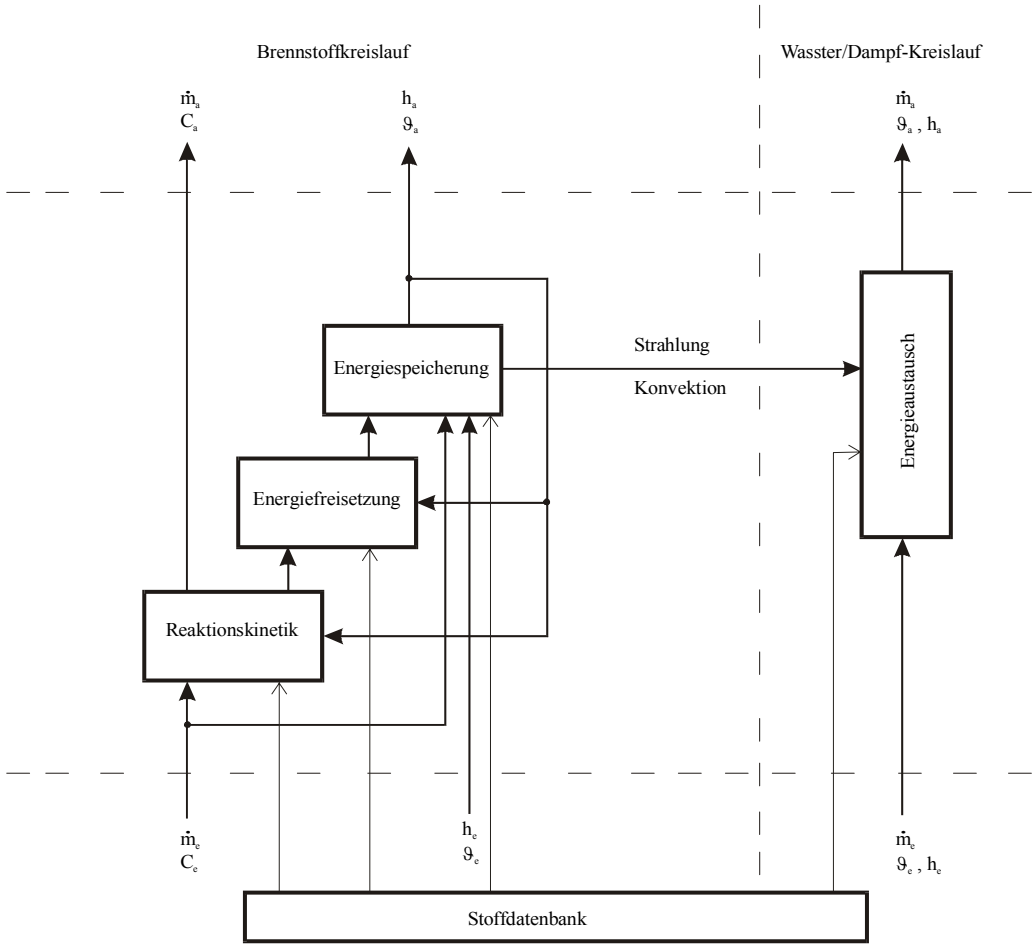
Die grundlegende Struktur eines Kraftwerkmoduls (Verbrennung und Energieerzeugung) wird in Abbildung 10 gezeigt. Der Verbrennungprozeß tritt in drei Hauptstadien auf, die abhängig von den resultierenden Temperatur- und Brennstoffeigenschaften sind. Die erzeugte Hitze wird durch Strahlung, Konvektion und Übertragung dem Arbeitsmittel übermittelt.

Die Kraftwerksfunktionsblockbibliothek enthält spezifische Funktionen und Module, um Kraftswerksanlagen oder Teilanlagen zu simulieren (Tabelle 12). Sie enthält Module wie:

- Dampferzeuger
- Zwischenüberhitzer
- Ventile
- Turbine, Turbinenstufen
- Rohrleitungen
- Kondensator
- Pumpen
- etc.

Die Lösungen der Gleichungen der Masse und Energiebilanzen erfordert die thermophysikalischen Eigenschaften des Mediums (Wasser/Dampf, Luft, Rauchgas). Dafür wurde eine leistungsfähige Funktionsbibliothek entwickelt und in DynStar integriert. Sie ermöglicht die Berechnung der thermodynamischen Zustandsvariablen, wie Enthalpie, Entropie, die Wärmekapazität, etc., sowie die Transportvariablen (Reynoldszahl, Wärmeleitfähigkeit, etc.) eines Arbeitsmittels.

Die Kraftwerkskomponenten enthalten stark nichtlineare dynamische Modelle. Dabei handelt es sich um Gleichungen für Massen- und Energiebilanzen. Die wichtigen Variablen sind der Druck, die Enthalpie und die Massenstrom. Die Entwicklung des Drucks, Massenstrom und Enthalpie (Temperatur) werden zu jedem Zeitpunkt berechnet. Sind die Werte der wichtigen Zustandsgrößen X(t) zum Zeitpunkt t bekannt, kann man ebenfalls $X(t+\Delta t)$ berechnen. Δt ist der Berechnungszeitschritt, der leicht geändert werden kann.



ṁ	Massenstrom	a	Ausgang
C	Konzentration	e	Eingang
h	Enthalpie		
ϑ	Temperatur		

Abbildung 10: Modulstruktur für Verbrennung und Energieerzeugung

Beispielanwendung

In Abbildung 11 ist das Modell einer Turbine mit Dampferzeuger und Generator dargestellt. Dieses Turbinenmodell läuft auf einem Rechner und der zugehöriger Regler nach der DVG-Struktur (Abbildung 12) wird auf einem anderen Rechner ausgeführt.

In Abbildung 13 ist der Leistungsverlauf der 500MW Turbine für den Schnellschlussfall dargestellt. Es werden die zwei Turbinenventile geschlossen und der Dampf strömt nicht mehr durch die einzelnen Turbinenstufen.

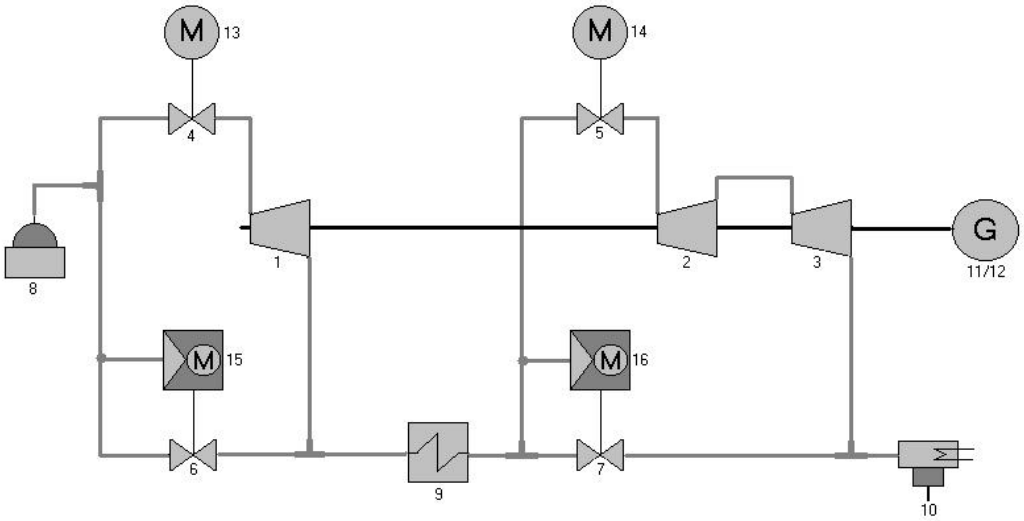


Abbildung 11: Turbinenmodell

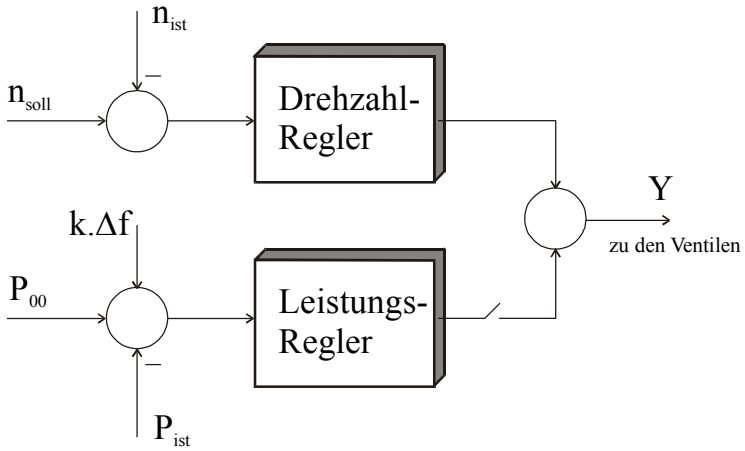


Abbildung 12: Regelstruktur gemäß DVG-Anforderung

(1) Hochdruckturbine

(2) Mitteldruckturbine

(3) Niederdruckturbine

(4) Hochdruckregelventil (HDRV)

(5) Mitteldruckregelventil (MDRV)

(6) Hochdruckumleitregelventil (HDURV)

(7) Mitteldruckumleitregelventil (MDURV)

(8) Dampferzeuger

(9) Zwischenüberhitzer

(10) Kondensator

(11) Generator

(12) Netz

(13) Antrieb Hochdruckregelventil

(14) Antrieb Mitteldruckregelventil

(15) Regler und Antrieb Hochdruckumleitregelventil

(16) Regler und Antrieb Mitteldruckumleitregelventil

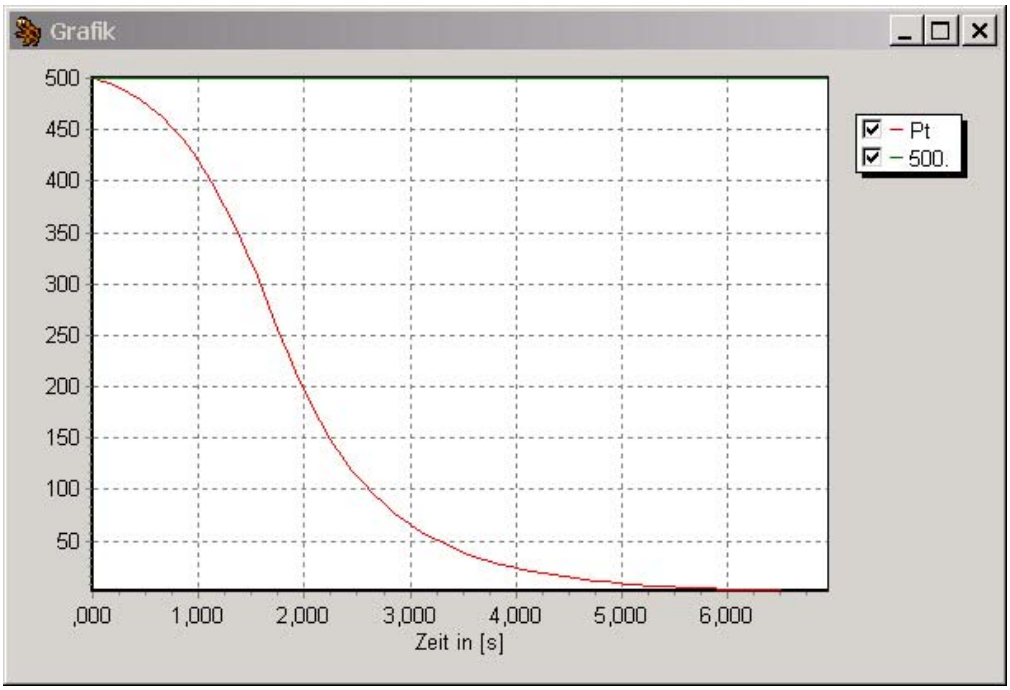


Abbildung 13: Zeitlicher Verlauf der Turbinenleistung für den Schnellschlussfall

3 Zusammenfassung

Das Programm DynStar wird am IPM entwickelt. Die Einsatzgebiete reichen von der Lehre und Forschung bis zu Optimierungsaufgaben in Kraftwerken. Aufgrund der Anwendungsgebiete werden folgende Anforderungen gestellt:

- modulare Struktur, um einen hohen Grad an Flexibilität zu erreichen
- einfache Handhabung,

Das Softwaresystem befindet sich in einem ständigen Entwicklungsprozeß und der Funktionsumfang nimmt immer mehr zu.

Konzept zur Regelung und Stabilitätsüberwachung beim Greifen mit flexiblen Robotergreifern

Arne Lehmann, Ralf Mikut, Jan Martin, Georg Bretthauer

Forschungszentrum Karlsruhe GmbH, Institut für Angewandte Informatik,
D-76021 Karlsruhe, Postfach 3640,
Telefon: (07247) 82-5749, Fax: (07247) 82-5786, E-Mail: arne.lehmann@iai.fzk.de

Abstract

Neben dem Einsatz von Robotergreifern mit steifen Achsen und Gelenken nimmt derzeit das Interesse an flexiblen Greifern zu. Ein solcher Greifer ist der am Forschungszentrum Karlsruhe entwickelte Robotergreifer mit flexiblen Fluidaktoren. Anforderungen an die Regelung dieses Systems sind, neben der Positionier- und Kraftgenauigkeit, die Gewährleistung von Sicherheit und Stabilität bei Greifvorgängen. Da aufgrund der Komplexität solcher Greifvorgänge ein formaler Stabilitätsnachweis nahezu unmöglich ist, wird als Lösungsvorschlag eine Fuzzy-Überwachungsebene vorgestellt, die auftretende Gefahrensituationen erkennt und somit das Einleiten von Gegenmaßnahmen ermöglicht. Die Analyse komplexer Greifszenarien erfordert eine leistungsfähige Simulationsumgebung, die sowohl das Antriebssystem des Robotergreifers (Hydraulik), die Handmechanik (Fingerglieder, Gelenke) als auch das zu greifende Objekt beinhaltet. Im Weiteren werden durch Simulationsstudien die entwickelten Algorithmen zur Regelung sowie zur Sicherheits- und Stabilitätsüberwachung von Greifvorgängen getestet und die erzielten Ergebnisse vorgestellt.

1 Einleitung

Ein wichtiges Entwicklungsziel heutiger Robotergreifer stellt die Gewichtsreduktion dar. Erst extremer Leichtbau ermöglicht den Einsatz z. B. in der Raumfahrt, der Servicerobotik oder der Medizintechnik (Handprothese). Im Gegensatz zu herkömmlichen Robotergreifern mit steifen Achsen und Gelenken [1, 2, 3] führt der angestrebte Leichtbau bei flexiblen Robotergreifern zum Auftreten von Elastizitäten in den Achsen und/oder Gelenken. Diese Elastizitäten können zu unerwünschtem Schwingen führen. Dadurch stehen den Vorteilen des geringen Gewichts und der bereits durch die Konstruktion bedingten Nachgiebigkeit des Greifers (erwünscht in bestimmten Greifsituationen) die Nachteile der komplexeren Modellbildung und Regelung entgegen. So hat die Regelung die Aufgabe, ggf. den Einfluss der Elastizitäten zu kompensieren und gleichzeitig das geforderte Greiferverhalten zu garantieren. Dazu gehört neben der Positionsregelung auch die Regelung der Nachgiebigkeit bzw. der Steifigkeit des Robotergreifers.

Ein solcher elastischer Greifer ist der am Forschungszentrum Karlsruhe entwickelte Robotergreifer, bei dem sich die Elastizität durch die Verwendung hydraulisch versorgter flexibler Fluidaktoren ergibt [4]. Um mit einem solchen flexiblen Robotergreifer Gegenstände zu greifen, werden von einer übergeordneten Bahnplanung die

erforderlichen Solltrajektorien (Winkelverläufe, Minimal- und Maximalmomente pro Gelenk über der Zeit) generiert. Regler auf Gelenkebene haben dann die Aufgabe, die geforderten Positionen bzw. Momente einzuregeln. Zu beachten ist hierbei, dass sich bei hydraulischer Fluidversorgung u. U. zusätzliche Einschränkungen ergeben, wenn für die Regelung *mehrerer* Gelenke nur *eine* unabhängige Druckversorgung zur Verfügung steht.

Bei Objektkontakt wirken auf die Gelenkregelungen *Störungen* in Form von externen Kräften und Momenten. Diese *Störungen* hängen sowohl von den meist unbekannten Objekteigenschaften (Lage, Geometrie, Verformbarkeit) als auch von den Zuständen des Greifers selbst ab. Folglich korrelieren sie mit den Stell- und Regelgrößen. Ein formaler Stabilitätsnachweis solcher Regelkreise ist wegen der schwer modellierbaren Störeinflüsse und der kontinuierlich-diskreten Systemeigenschaften für praktisch relevante Aufgaben nahezu unmöglich. Aufgrund der Komplexität sind Instabilitäten und daraus resultierende Sicherheitsprobleme allerdings nicht auszuschließen. Ein Lösungsansatz ist der Einsatz einer Fuzzy-Überwachungsebene [5], die auftretende Abweichungen vom Sollverhalten und potenzielle Gefahrensituationen auf der Ebene der Basisregler erkennt und sowohl durch eigene Modifikation der Reglereinstellungen, als auch durch Weiterleiten der detektierten Gefahrensituation an die übergeordnete Bahnplanung, geeignete Gegenmaßnahmen veranlasst.

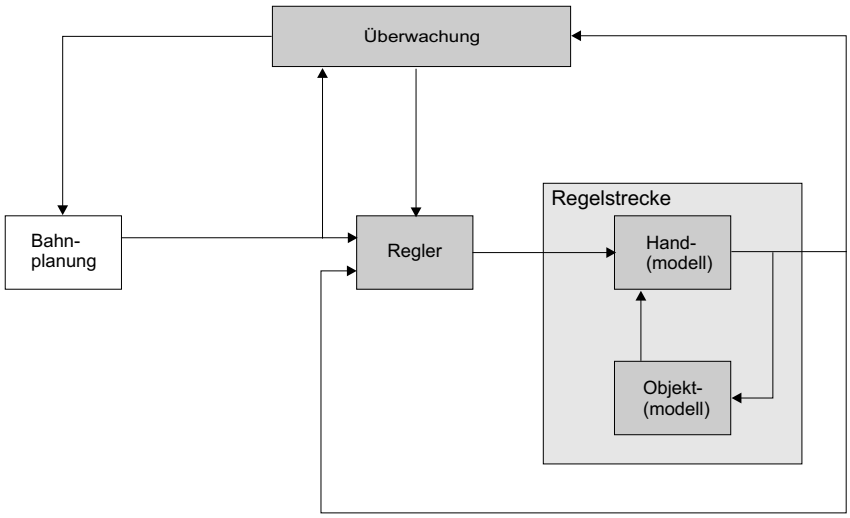


Bild 1: Modellstruktur der Simulationsumgebung (Bahnplanung, Regler, Regelstrecke und Überwachung)

Um die bei einem Greifvorgang auftretende Effekte zunächst in einer vereinfachten Umgebung zu analysieren, werden Simulationsstudien durchgeführt. Das Simulationsmodell beinhaltet das hydraulische System des Robotergreifers, die Handmechanik und das zu greifende Objekt. Die beim Greifen des Objekts auftretenden Kräfte werden durch ein zweidimensionales Kontaktmodell (Flächen- und Punktkontaktmodell) ermittelt. Mit dieser Simulationsumgebung werden in einem ersten Schritt die entwickelten Algorithmen zur Stabilitätsüberwachung sowohl bei freier Bewegung des Systems als auch bei Kontakt mit einem ortsfesten, verformbaren Objekt (Petri-Netz mit zustands-individuellen Reglern) getestet. Benötigte Teilmodelle für die Simulation und Analyse von Greifvorgängen sind somit, neben der Bahnplanung,

der *Regler*, die Regelstrecke bestehend aus *Handmodell* und *Objektmodell* sowie die *Überwachung*. Bild 1 zeigt die sich ergebende Gesamtstruktur des vorgeschlagenen Konzepts.

Auf diesem Konzept aufbauend werden in Kapitel 2 das Modell des Robotergreifers sowie das Modell des zu greifenden Objekts (incl. Kontaktmodell) diskutiert. In Kapitel 3 und 4 werden das verwendete Regelungskonzept und die theoretischen Grundlagen der Fuzzy-Überwachung vorgestellt. Kapitel 5 zeigt Ergebnisse von simulierten Greifszenarien mit einem ortsfesten, verformbaren Objekt sowie ihre Stabilitätseinschätzung durch die Fuzzy-Überwachungsebene. Die von der Bahnplanung generierten Solltrajektorien sind dabei für das erforderliche Griffmuster vorgegeben.

2 Modellbildung

Handmodell

Um die Dynamik einer Roboterhand zunächst ohne Interaktion mit der Umwelt zu modellieren, wird der Greifer als Verbund von n unabhängigen Fingern mit jeweils m Gelenken aufgefasst (Gelenkwinkel $\varphi_{Finger,Gelenk}$), die sich als offene Ketten durch den Arbeitsraum bewegen. Wenn dann Fingerglieder, Gelenke und Achsen als starr angenommen werden, ergeben sich folgende aus der Literatur [6, 7, 8] bekannte dynamische Bewegungsgleichungen für einen einzelnen Finger

$$\underline{\tau} = M(\underline{q})\ddot{\underline{q}} + C(\underline{q}, \dot{\underline{q}})\dot{\underline{q}} + F(\dot{\underline{q}}) + G(\underline{q}) \tag{1}$$

mit:

$$\begin{array}{lcll} \underline{q} & = & (\varphi_{1,1} \ldots \varphi_{n,m})^T & = \text{ Vektor der verallg. Gelenkwinkel} \\ \dot{\underline{q}} & = & (\dot{\varphi}_{1,1} \ldots \dot{\varphi}_{n,m})^T & = \text{ Vektor der Gelenkgeschwindigkeiten} \\ \ddot{\underline{q}} & = & (\ddot{\varphi}_{1,1} \ldots \ddot{\varphi}_{n,m})^T & = \text{ Vektor der Gelenkbeschleunigungen} \\ \underline{\tau} & = & (M_{Aktor(1,1)} \ldots M_{Aktor(n,m)})^T & = \text{ Vektor der Aktormomente.} \end{array}$$

Weiterhin bezeichnen $M(\underline{q})$ die $(n \cdot m \times n \cdot m)$ Trägheitsmatrix, $F(\dot{\underline{q}})$ den $(n \cdot m \times 1)$ Vektor der Reibungskräfte, $G(\underline{q})$ den $(n \cdot m \times 1)$ Vektor der Gravitationskräfte und $C(\underline{q}, \dot{\underline{q}})$ den $(n \cdot m \times 1)$ Vektor der Coriolis- und Zentripetalkräfte.

Für Robotergreifer mit elastischen Gelenken und elektrischen Antrieben muss das Dynamikmodell des starren Robotergreifers (1) erweitert werden. Die Bewegungsgleichungen solcher Roboter werden z. B. in [9, 10, 11] aufgestellt. Im Gegensatz zum Starrkörpermodell wird hierbei jedes Gelenk mit dem darauffolgenden Fingerglied als Zweimassensystem modelliert [12]. Zur Beschreibung der Bewegung werden dann zwei verallgemeinerte Koordinaten $\underline{q} = (\underline{q}_1, \underline{q}_2)^T$ benötigt, wobei $\underline{q}_1$ der Vektor der Antriebspositionen und $\underline{q}_2$ der Vektor der Fingergliedpositionen ist. $\underline{\tau}$ bezeichnet das Drehmoment der Gelenkfeder und $\underline{\tau}_{Antrieb}$ das vom Antrieb erzeugte Moment

$$\begin{array}{lcl} \underline{\tau}_{Antrieb} & = & J_{Antrieb}\,\ddot{\underline{q}}_1 + \underline{\tau} \\ \underline{\tau} & = & M(\underline{q}_2)\ddot{\underline{q}}_2 + C(\underline{q}_2, \dot{\underline{q}}_2)\dot{\underline{q}}_2 + F(\dot{\underline{q}}_2) + G(\underline{q}_2) \\ \underline{\tau} & = & K(\underline{q}_1 - \underline{q}_2). \end{array} \tag{2}$$

Das Gleichungssystem (2) entspricht prinzipiell Gleichung (1), jedoch wird das Eingangsdrehmoment über die Gelenkfeder mit der Federkonstanten K übertragen. Der

Motor ist über seine eigene Dynamik an das Gelenkmoment gekoppelt. Bei dem Robotergreifer des Forschungszentrum Karlsruhe wird das Eingangsdrehmoment von einem fluidischen System, bestehend aus Pumpe, Ventilbank, 2/2-Wege-Ventilen und flexiblen Fluidaktoren erzeugt. Dieses Eingangsdrehmoment $\underline{\tau}_{Antrieb}$ hängt zwar vom aktuellen Gelenkwinkel ab, wird aber im Unterschied zu Gleichung (2) direkt übertragen [13, 14]. Somit lässt sich die Dynamik des Robotergreifers, trotz der durch die flexiblen Fluidaktoren erzeugten Elastizität, nach Gleichung (1) mit $\underline{\tau}(\underline{q})$ modellieren.

Nach der Modellierung des mechanischen Systems erfolgt nun die Modellbildung des hydraulischen Systems. Die mathematischen Modelle des hydraulischen Systems (Pumpe, Ventile, Aktoren) wurden für ein *einzelnes* Fingergelenk bereits in [14] vorgestellt. Für den Mehraktorfall werden diese Modelle zu einem Gesamtmodell integriert (vgl. Bild 2).

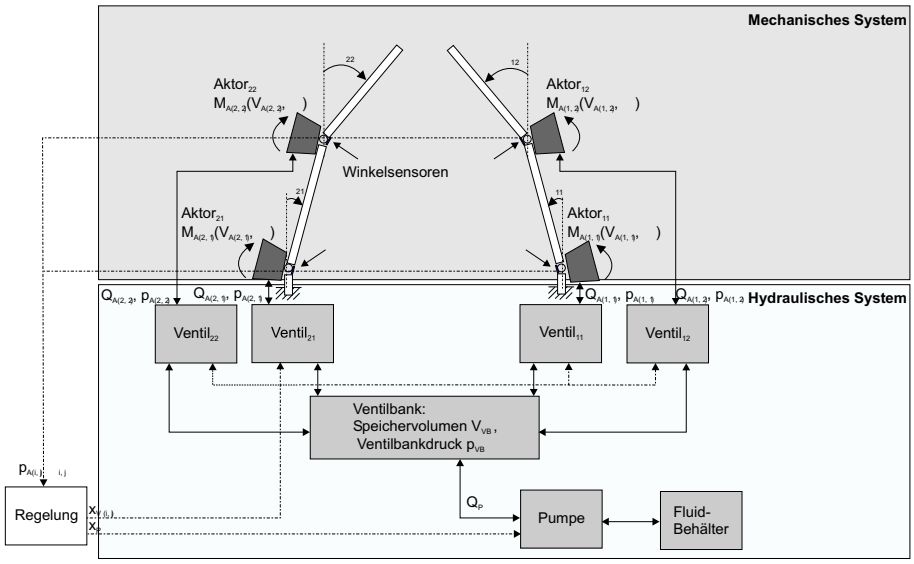


Bild 2: Schematische Darstellung des Handmodells für vier Aktoren

Bild 2 zeigt in schematischer Darstellung den Aufbau eines mit fluidischen Aktoren betriebenen Robotergreifers für $n = 2$ Finger und $m = 2$ Gelenken pro Finger. Für den Einsatz in der Robotik ist ein Greifer mit $n = 5$ Fingern und $m = 3$ (Daumen) bzw. $m = 2$ (andere Finger) Gelenken pro Finger vorgesehen [15]. **Eine** Pumpe fördert das Arbeitsfluid aus einem Behälter zunächst in das Speichervolumen V_{VB} der Ventilbank. Von dort gelangt es zu den Ventilen. Für jeden der Aktoren existiert ein separates 2/2-Wege-Ventil. Für das Gleichgewicht der Volumenströme gilt bei inkompressiblen Fluiden:

$$Q_{VB} = \underbrace{Q_P(x_P, p_{VB}(V_{VB}))}_{Pumpenkennfeld} - \sum_{i=1}^{n}\sum_{j=1}^{m} \underbrace{Q_{A(i,j)}(x_{V(i,j)}, p_{VB}(V_{VB}) - \overbrace{p_{A(i,j)}(V_{A(i,j)}, \varphi_{i,j})}^{Aktor-Druckkennfeld})}_{Ventilkennfeld} \tag{3}$$

Darin bezeichnet Q_{VB} den Volumenstrom in die Ventilbank, Q_p den Förder-Volumenstrom der Pumpe, x_P die Steuerspannung der Pumpe in Form der Pulsweite, $p_{VB}(V_{VB})$ den Druck in der Ventilbank in Abhängigkeit vom Fluidvolumen in der

Ventilbank, $Q_{A(i,j)}$ die Volumenströme in die Aktoren, $x_{V(i,j)}$ die Steuerspannungen der Ventile in Form der Pulsweite sowie $p_{A(i,j)}(V_{A(i,j)}(\varphi_{i,j}), \varphi_{i,j})$ die Drücke in den Aktoren in Abhängigkeit von den in den Aktoren befindlichen Fluidvolumina $V_{A(i,j)}(\varphi_{i,j})$. Aus diesen Fluidvolumina $V_{A(i,j)}(\varphi_{i,j})$ berechnen sich dann die in den Gelenken erzeugten Aktormomente $M_{A(i,j)}$ in Abhängigkeit vom derzeitigen Gelenkwinkel $\varphi_{i,j}$ $(M_{A(i,j)}(p_{A(i,j)}(V_{A(i,j)}(\varphi_{i,j})), \varphi_{i,j})$ [14].

Die Stellgrößen des Reglers sind die pulsweitenmodulierten Steuerspannungen der Pumpe x_P und der Ventile $x_{V(i,j)}$. Zu beachten ist dabei, dass bei einer negativen Steuerspannung $x_P < 0$ und etwa ausgeglichenen Druckverhältnissen die Pumpe das Arbeitsfluid in Richtung des externen Fluidbehälters und bei einer positiven Steuerspannung $x_P > 0$ in Richtung der Ventilbank fördert (vgl. Bild 2).

Kontaktmodell

Um Greifvorgänge mit dem in Bild 1 gezeigten Konzept zu simulieren und zu analysieren, sind neben dem Robotergreifer (Kinematik, Dynamik) auch das zu greifende Objekt (Lage im Raum, Geometrie, mechanische Eigenschaften) sowie die Kontaktsituation zu modellieren.

In einem ersten Schritt sind die modellierten Objekte ausschließlich zylindrisch. Die Projektion in die x-y-Ebene ergibt eine kreisförmige Geometrie der Objekte. Die Objektposition und -ausrichtung in Bezug zum WKS (Welt-Koordinaten-System, Ursprung O_0) werden durch das OKS (Objektkoordinatensystem, Ursprung O_B) bestimmt. Der Ursprung des OKS ist der Mittelpunkt M_{Objekt} des Objekts im WKS. Die Objektgröße wird durch den Radius r festgelegt. Für die Kontaktmodellierung werden im Weiteren geometrische Betrachtungen erforderlich. Aus diesem Grund werden die Gelenkwinkel $\varphi_{i,j}$ unter Verwendung der direkten Kinematik der Finger bezüglich des Ursprungs der Fingerkoordinatensysteme $O_{i,j}$ und der homogenen Koordinatentransformation ${}^0T_{i,j}$ in kartesische Koordinaten der Fingerspitze ${}^0x_i^{tip}$ umgerechnet (vgl. Bild 3).

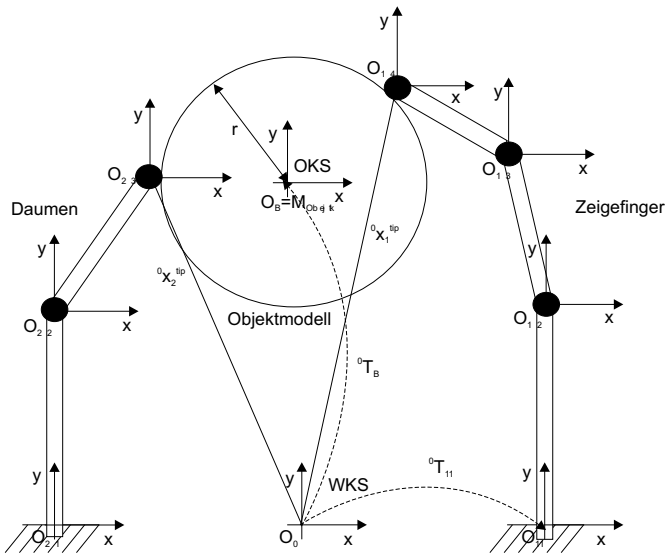


Bild 3: Koordinatensysteme im Kontaktfall

Die Objekteigenschaften werden durch die Masse m und die Materialkonstante k definiert (Feder-Masse-System). Daraus berechnet sich die resultierende Kontaktkraft F radial zum Objektmittelpunkt M_{Objekt}:

$$F = \begin{cases} m\ d\ddot{x} + k\ dx & \text{für} \quad dx = r - d_s > 0 \\ 0 & \text{sonst} \end{cases} \tag{4}$$

Darin stellt dx die Objektverformung infolge der Kontaktsituation dar.

In der Literatur existieren zahlreiche Ansätze [6, 16, 17] zur Kontaktmodellierung. In dieser Arbeit wird ein reibungsfreies Kontaktmodell verwendet, das sowohl Flächen- als auch Punktkontakt beinhaltet. Bild 4 zeigt beispielhaft zwei verschiedene Kontaktsituationen (Flächenkontakt (l), Punktkontakt (r)) des Robotergreifers mit einem ortsfesten, verformbaren Objekt. Dargestellt ist die geometrische Berechnung der Objektverformung dx.

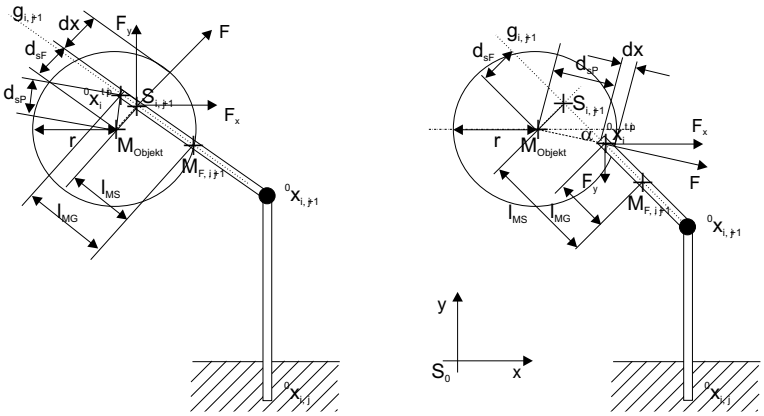


Bild 4: Flächenkontakt (l), Punktkontakt (r) des Robotergreifers mit einem ortsfestem, verformbaren Objekt

Als Schnittpunkt $S_{i,j}$ wird die Position mit dem minimalen Abstand des Objektmittelpunkts M_{Objekt} zu der Geraden $g_{i,j}$ definiert. Die Gerade wird dabei mit Hilfe der kartesischen Koordinaten zweier aufeinander folgender Gelenkpunkte ${}^0x_{i,j}$ und ${}^0x_{i,j+1}$ bestimmt. Mit d_{sF} wird dann das lokale Minimum des Abstandes (M_{Objekt}, S) und mit d_{sP} das lokale Minimum des Abstandes $(M_{Objekt}, {}^0\,x_i^{tip})$ bezeichnet:

$$\begin{aligned} d_{sF} &= \|M_{Objekt}^{WKS} - S_{i,j}^{WKS}\|_2 \\ d_{sP} &= \|M_{Objekt}^{WKS} -{}^0\,x_i^{tip}\|_2 \end{aligned} \tag{5}$$

Mit l_{MS} wird weiterhin der Abstand des Gelenkmittelpunkts $M_{F,(i,j)}$ vom Schnittpunkt $S_{i,j}$ und mit l_{MG} der Abstand des Gelenkmittelpunkts $M_{F,(i,j)}$ von den Koordinaten der Fingerspitze ${}^0x_i^{tip}$ bezeichnet:

$$\begin{aligned} l_{MS} &= \|M_{F,(i,j)}^{WKS} - S_{i,j}^{WKS}\|_2 \\ l_{MG} &= \|M_{F,(i,j)}^{WKS} -{}^0\,x_i^{tip}\|_2 \end{aligned} \tag{6}$$

Die Gleichungen (5) und (6) führen auf das zur Berechnung der Objektverformung dx benötigte Abstandsmaß d_s:

$$d_s = \begin{cases} d_{sF} & \text{für} \quad l_{MS} \leq l_{MG} \\ d_{sP} & \text{für} \quad l_{MS} > l_{MG} \end{cases} \tag{7}$$

Mit Hilfe des Abstandsmaßes d_s aus Gleichung (7) lässt sich dann die Kontaktart bestimmen:

$$\text{Kontaktart} = \begin{cases} \text{Flächenkontakt} & \text{für} \quad d_s \leq r \wedge l_{MS} \leq l_{MG} \\ \text{Punktkontakt} & \text{für} \quad d_s \leq r \wedge l_{MS} > l_{MG} \\ \text{kein Kontakt} & \text{sonst} \end{cases} \tag{8}$$

Bei beiden Modellen erfolgt die Berechnung der Kontaktkraft F nach Gleichung (4). Der Unterschied zwischen beiden Modellen ergibt sich aus der geometrischen Berechnung der Objektverformung dx und der Wirkungsrichtung der Kontaktkraft F (vgl. Bild 4).

Weiterhin muss bei Objektkontakt die Kopplung der Gelenkmomente $M_{i,j}$ und $M_{i,j+1}$ berücksichtigt werden, da die an einem distalen Fingerglied angreifenden externen Kräfte auch die proximalen Gelenke beeinflussen (Bild 4). Durch Freischneiden des Robotergreifers lassen sich die in den Gelenken auftretenden Kräfte und Momente berechnen [18]. Bild 5 zeigt das Gesamtmodell eines Fingers mit zwei Gelenken unter Einwirkung einer externen Kraft F am distalen Fingerglied (Teilkörper 2).

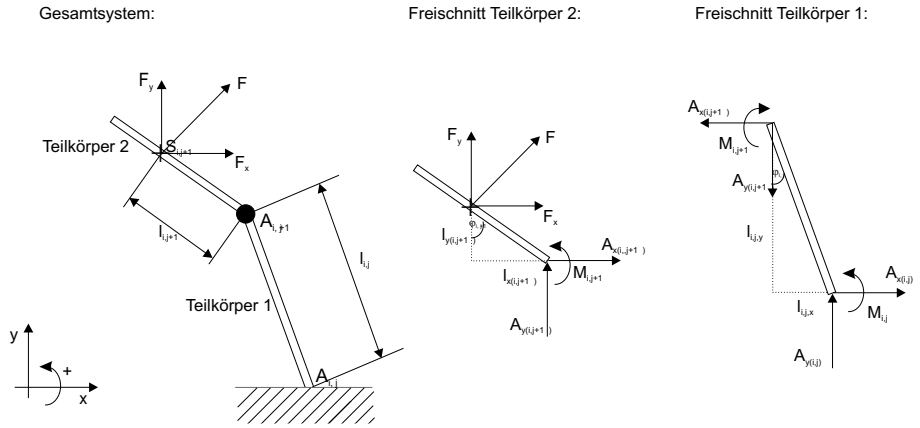


Bild 5: Gesamtmodell (l) und Freischnitte (m), (r) eines Fingers mit zwei Gelenken unter Einwirkung einer externen Kraft F

Allgemein gelten folgende Gleichgewichtsbedingungen für die freigeschnittenen Teilkörper:

$$\sum F_{x(i,j)} = 0 \quad \sum F_{y(i,j)} = 0 \quad \sum M_{A(i,j)} = 0 \tag{9}$$

Mit den Strecken $l_{y(i,j+1)}$ und $l_{x(i,j+1)}$ aus Bild 5 (m) berechnet sich das resultierende Moment $M_{i,j+1}$ am distalen Gelenk zu:

$$M_{i,j+1} = F_x \underbrace{l_{i,j+1} \cos(\varphi_{i,j+1})}_{l_{y(i,j+1)}} + F_y \underbrace{l_{i,j+1} \sin(\varphi_{i,j+1})}_{l_{x(i,j+1)}} \tag{10}$$

Wird die Gleichung (9) auch auf den zweiten Teilkörper angewendet, so ergibt sich unter Verwendung der Strecken $l_{y(i,j)}$ und $l_{x(i,j)}$ aus Bild 5 (r) das gesuchte Moment $M_{i,j}$ am proximalen Fingergelenk in Abhängigkeit von der Fingerkinematik, der aktuellen Gelenkstellung und der einwirkenden Kraft.

$$M_{i,j} = M_{i,j+1} + F_x \underbrace{l_{i,j}\cos(\varphi_{i,j})}_{l_{y(i,j)}} + F_y \underbrace{l_{i,j}\sin(\varphi_{i,j})}_{l_{x(i,j)}} \tag{11}$$

F, F_x und F_y bezeichnen dabei die Kontaktkraft und ihre x und y-Komponente in der Ebene. $A_{i,j}, A_{x(i,j)}$ und $A_{y(i,j)}$ bezeichnen die Lagerkräfte in den Gelenken sowie ihre x- und y-Komponente in der Ebene. $M_{i,j}$ ist das resultierende Moment am Gelenk $A_{i,j}$. Die Strecken $l_{i,j}$ bezeichnen die Abstände von $\|^0 x_{i,j} -^0 x_{i,j+1}\|_2$ (Punktkontakt) bzw. $\|^0 x_{i,j+1} - S_{i,j+1}\|_2$ (Flächenkontakt).

3 Reglerstruktur

Um sicheres Greifen zu ermöglichen, ist die Interaktion zwischen Robotergreifer und Umgebung (Objekt) auf gewünschte Weise zu regeln. Anforderungen an den Regler sind dabei, Führungsgrößenvorgaben $\varphi_{soll(i,j)}, M_{soll(i,j)}$ in Gelenkwinkel $\varphi_{ist(i,j)}$ und Gelenkmomente $M_{ist(i,j)}$ umzusetzen. Es existieren zahlreiche Ansätze [6, 17], u. a. hybride Kraft-Positions-, Nachgiebigkeits- und Impedanzregelungen, um dieses Ziel zu erreichen. Als Stand der Technik sind hierbei vor allem Nachgiebigkeits- und Impedanzregelungskonzepte hervorzuheben. Bild 6 zeigt das in dieser Arbeit entwickelte hybride Kraft-Positions-Regelkonzept, welches auf der Vorgehensweise in [15] aufbaut.

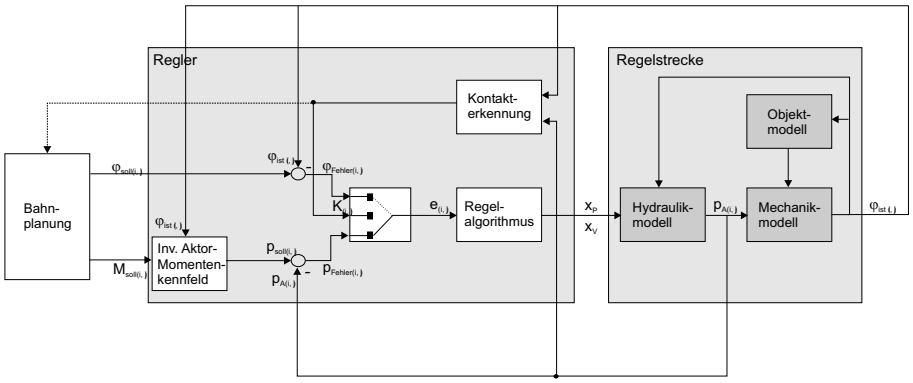


Bild 6: Modellstruktur des hybriden Kraft-Positions-Regelkonzepts (ohne Überwachungsebene)

Zu beachten ist dabei, dass nur eine unabhängige Druckversorgung zur Verfügung steht. Dadurch ergeben sich bei der Realisierung einer Kraft- und Positionsregelung Restriktionen, welche bei der Auswahl des Regelungskonzepts zu berücksichtigen sind. So kann sich der Regelalgorithmus pro Abtastintervall nur für eine Pumpenrichtung und damit für ein entsprechendes pulsweitenmoduliertes Steuersignal x_P entscheiden. Damit ist es nur in Sonderfällen möglich, in einem Intervall gegenläufige Bewegungen verschiedener Fingergelenke zu realisieren.

Um nun eine definierte Kraft auf ein zugreifendes Objekt auszuüben, muss im Kontaktfall ein vorgegebenes Gelenkmoment $M_{soll(i,j)}$ vom Regelalgorithmus eingeregelt werden. Um dies zu realisieren, wird ausgehend von einem bekannten $\varphi_{ist(i,j)}$ und dem vorgegebenen Gelenkmoment $M_{soll(i,j)}$ über das inverse Aktor-Momentenkennfeld $p_{soll(i,j)}(M_{soll(i,j)}, \varphi_{ist(i,j)})$ der benötigte Aktordruck $p_{soll(i,j)}$ berechnet. Die Differenz aus benötigtem Aktordruck $p_{soll(i,j)}$ und aktuellem Aktordruck $p_{A(i,j)}$ ist dann die vom Regelalgorithmus zu minimierende Regelabweichung $p_{Fehler(i,j)}$:

$$p_{Fehler(i,j)}[k] = p_{soll(i,j)}[k] - p_{A(i,j)}[k] \tag{12}$$

Bei freier Bewegung ist die zu minimierende Regelabweichung die Differenz der Führungsgrößenvorgabe $\varphi_{soll(i,j)}$ und der momentanen Gelenkwinkel $\varphi_{ist(i,j)}$:

$$\varphi_{Fehler(i,j)}[k] = \varphi_{soll(i,j)}[k] - \varphi_{ist(i,j)}[k] \tag{13}$$

Die Regeldifferenz (Eingangsgröße des Regelalgorithmus) berechnet sich dann mit den Wichtungsparametern α und β nach folgender Gleichung:

$$e_{i,j}[k] = \alpha[k]\ \varphi_{Fehler(i,j)}[k] + \beta[k]\ p_{Fehler(i,j)}[k] \tag{14}$$

Das Erreichen der Kontaktsituation wird der Bahnplanung mitgeteilt, die daraufhin die entsprechenden Führungsgrößen festlegt, um die Finger zu koordinieren. Der Finger mit dem *ersten* Kontakt wartet mit dem Zugreifen, bis *alle* am Griff beteiligten Finger Kontakt haben, damit später ein koordiniertes Zugreifen erfolgen kann. In Abhängigkeit von der aktuellen Kontaktsituation $K_{i,j}$ erfolgt dabei, im einfachsten Fall, eine scharfe Umschaltung vom Positions- in den Kraftregelmodus:

$$K_{i,j}[k] = \begin{cases} 1 = \text{Kontakt} & \Rightarrow \quad \alpha[k] = 0 \wedge \beta[k] = 1 \\ 0 = \text{kein Kontakt} & \Rightarrow \quad \alpha[k] = 1 \wedge \beta[k] = 0 \end{cases} \tag{15}$$

Optional können auch Druckbegrenzungen bei der freien Bewegung berücksichtigt werden, um die Aufprallgeschwindigkeit eines Fingers auf ein empfindliches Objekt und damit die Auswirkungen einer fehlerhaften Kontakterkennung zu reduzieren [14].

Die Idee des verwendeten Regelalgorithmus beruht darauf, zu jedem Abtastzeitpunkt eine Gruppe von Aktoren mit ähnlichen Anforderungen an die Pumpe zusammenzufassen und die zugehörigen Ventile zu öffnen. Die Pumpe wird dann entsprechend eines Kompromisses zwischen diesen Aktoren angesteuert. Wenn die Aktoren bzw. die Gelenke sich innerhalb von vorgegebenen Totzonenbereichen $e_{tot_{aus}}$ und $e_{tot_{ein}}$ mit $e_{tot_{aus}} > e_{tot_{ein}}$ (Hysterese) befinden, scheiden sie aus der Gruppe aus. Für die Berechnung der Totzonenbereiche ergibt sich wiederum in Abhängigkeit von der Kontaktsituation $K_{i,j}$:

$$e_{tot_{aus}}[k] = \alpha[k]\ \varphi_{tot} + \beta[k]\ p_{tot} \qquad e_{tot_{ein}}[k] = \frac{e_{tot_{aus}}[k]}{2} \tag{16}$$

Die vom Regelalgorithmus zu tolerierenden Regelgrößenabweichungen werden mit p_{tot} und φ_{tot} bezeichnet. Mit den Ergebnissen aus Gleichung (16) berechnet sich dann die Regeldifferenz $e^*_{i,j}$:

$$e^*_{i,j}[k] = \begin{cases} 0 & \text{für} \quad (|e_{i,j}[k]| < e_{tot_{ein}}[k]) \\ & \vee \quad (|e_{i,j}[k]| < e_{tot_{aus}}[k] \wedge |e^*_{i,j}[k-1]| = 0) \\ e_{i,j}[k] & \text{sonst} \end{cases} \tag{17}$$

Die Gruppenbildung erfolgt über alle Aktoren bzw. Gelenke mit positiver bzw. negativer Regeldifferenz:

$$E_{pos}[k] = \sum_{i=1}^{n} \sum_{j=1}^{m} \begin{cases} e^*_{i,j}[k] & \text{für} \quad e^*_{i,j}[k] > 0 \\ 0 & \text{sonst} \end{cases} \tag{18}$$

$$E_{neg}[k] = \sum_{i=1}^{n} \sum_{j=1}^{m} \begin{cases} e^*_{i,j}[k] & \text{für} \quad e^*_{i,j}[k] < 0 \\ 0 & \text{sonst} \end{cases} \tag{19}$$

Der Regelalgorithmus schaltet dann ereignisdiskret zu jedem Abtastzeitpunkt T_A für jedes Verhältnis von E_{pos} zu E_{neg} in Abhängigkeit von dem Schaltfaktor k_{switch} ($k_{switch} > 1$) und seinem derzeitigen Zustand. Bild 7 zeigt die Transitionen des implementierten Zustandsautomaten. Im Pumpenzustand 1 werden alle Aktoren der E_{pos}-Gruppe, im Pumpenzustand -1 alle Aktoren der E_{neg}-Gruppe geregelt. Im Pumpenzustand 0 wird die Pumpe ausgeschaltet.

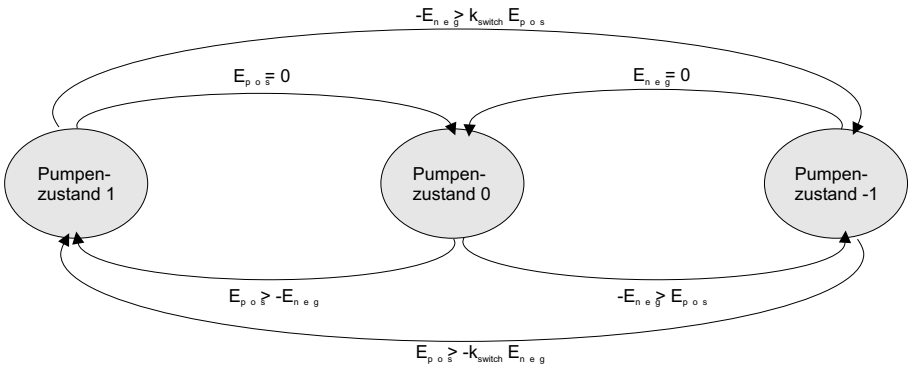


Bild 7: Zustandsautomat des Regelalgorithmus mit k_{switch} als Konstante zur Prioritätenumschaltung

In Abhängigkeit vom geschalteten Pumpenzustand berechnet sich die Steuerspannung der Pumpe mit Hilfe der Regeldifferenzen E_{pos} bzw. E_{neg} nach folgendem zeitdiskreten PI-Algorithmus:

$$x_P[k+1] = K_R[k+1] E_{(pos,neg)}[k+1] + u_I[k+1] \tag{20}$$

$$u_I[k+1] = u_I[k] + \frac{K_R[k]}{T_N[k]} T_A \; E_{(pos,neg)}[k] \tag{21}$$

Die Adaption der Reglerparameter K_R und T_N aus den Gleichungen (20) und (21) erfolgt wiederum in Abhängigkeit des Kontaktzustands $K_{i,j}$ und kann später um weitere Größen erweitert werden:

$$K_R[k] = \alpha[k] \; K_{RF} + \beta[k] \; K_{RP} \qquad T_N[k] = \alpha[k] \; T_{RF} + \beta[k] \; T_{RP} \tag{22}$$

K_{RF} und K_{RP} bezeichnen die Reglerverstärkung und T_{RF} und T_{RP} die Zeitkonstante des PI-Regelalgorithmus im positions- bzw. im kraftgeregelten Modus.

Die Steuerspannung der Ventile $x_{V_{(i,j)}}$ ergibt sich in Abhängigkeit von der Regeldifferenz und dem derzeitigen Pumpenzustand:

$$x_{V_{(i,j)}}[k] = \begin{cases} 1 = \text{Ventil offen} & \text{für} \quad e^*_{i,j}[k] > 0 \wedge \text{Pumpenzustand} = 1 \\ & \vee \quad e^*_{i,j}[k] < 0 \wedge \text{Pumpenzustand} = -1 \\ 0 = \text{Ventil geschlossen} & \text{sonst} \end{cases} \tag{23}$$

Druckdifferenzen zwischen Eingang und Ausgang der Pumpe führen dazu, dass sich trotz einer positiven Steuerspannung x_P ein negativer Durchfluss ergibt, wenn die Pumpe gegen einen zu hohen Druck p_{VB} fördert. Aus diesem Grund sollen in einem weiteren Schritt in die Entscheidung des Reglers, ob die Ventile geöffnet oder geschlossen werden, nicht nur die Regelabweichungen, sondern auch die aktuell anliegenden Druckdifferenzen zwischen Pumpe, Ventilen und Aktoren einbezogen werden.

Bei Handvarianten ohne integrierte Druck- und Kraftsensorik ist zwar eine Positionsregelung möglich, die Leistungsfähigkeit bei der Kraft-Momentenregelung und Kontakterkennung aber sehr eingeschränkt. Hier besteht lediglich die Möglichkeit, entsprechende Grobinformationen aus den zeitlichen Winkelverläufen bei bekannten Pumpen und Ventilzuständen zu extrahieren.

4 Überwachungsebene

Die Idee des Überwachungskonzepts besteht darin, die für das sichere Durchführen eines Greifvorgangs benötigten Regelgrößentrajektorien $(\varphi_{ist(i,j)}, M_{ist(i,j)})$ in einem vorgegebenen Führungsgrößenintervall $(\varphi_{max(i,j)},\ \varphi_{min(i,j)},\ M_{max(i,j)},\ M_{min(i,j)})$ zu überwachen. Deshalb wird eine Fuzzy-Überwachungsebene eingesetzt, die auf einer Erweiterung und Verallgemeinerung des Konzepts aus [5] beruht. Befinden sich die Regelgrößentrajektorien $(\varphi_{ist(i,j)}, M_{ist(i,j)})$ zu allen Zeitpunkten innerhalb ihrer vorgegebenen Intervalle, so wird die Situation von der Überwachungsebene als 'STABIL' eingestuft. Bei Verlassen dieser Intervalle werden die Ursachen mit in die Beurteilung einbezogen. Vom Regelalgorithmus wird gefordert, dass er das Gesamtsystem nach äußeren Störungen oder 'schnellen' Veränderungen der Intervallgrenzen wieder in Richtung des Sollverhaltens ausregelt. Gelingt dies nicht, wird die Gesamtsituation als 'INSTABIL' eingeschätzt. Weiterhin sind Zustände mit Stabilitätsgraden *zwischen* 'STABIL' und 'INSTABIL' möglich.

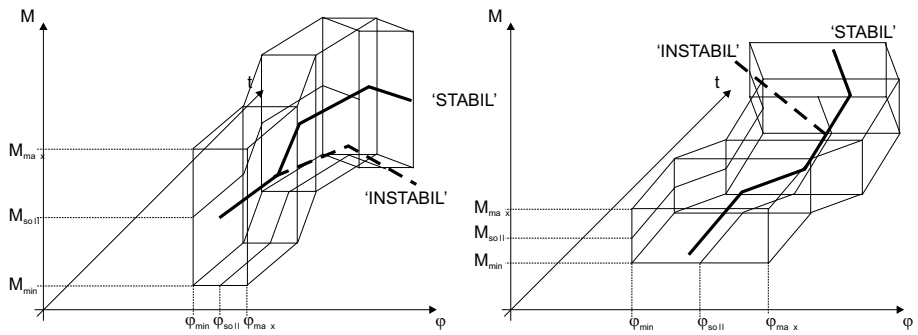


Bild 8: Grafische Darstellung der Führungsgrößenintervalle für die freie Bewegung (l) und bei Objektkontakt (r)

Durch Aufteilen der Greifbewegung in verschiedene Bewegungsphasen lässt sich dabei der Umschaltvorgang von der Positions- zu der Kraftregelung während eines Greifvorgangs erfassen. Im Zustand der *freien Bewegung* wird durch ein sehr schmales Führungsgrößenintervall für die Gelenkwinkel ($\varphi_{max(i,j)}$, $\varphi_{min(i,j)}$) und ein breites Führungsgrößenintervall für die Gelenkmomente ($M_{max(i,j)}$, $M_{min(i,j)}$) ein geringes Abweichen von den Sollpositionen stark gewichtet (Bild 8(l)). Im *Kontaktfall* dagegen werden durch ein schmales Führungsgrößenintervall für die Gelenkmomente ($M_{max(i,j)}$, $M_{min(i,j}$) und ein breites Führungsgrößenintervall für die Gelenkwinkel ($\varphi_{max(i,j)}$, $\varphi_{min(i,j)}$) bereits geringe Abweichungen der Gelenkmomente von den Sollmomenten als 'INSTABIL' eingestuft (Bild 8(r)).

Bild 1 zeigt die Eingliederung der Überwachung in das in dieser Arbeit vorgestellte Gesamtkonzept. Bild 9 zeigt die detaillierte Struktur der Überwachungsebene. Eingangsgrößen der Überwachungsebene sind die Führungsgrößentrajektorien ($\varphi_{soll(i,j)}, M_{soll(i,j)}$), die Regelgrößentrajektorien ($\varphi_{ist(i,j)}, M_{ist(i,j)}$) sowie die Führungsgrößenintervalle ($\varphi_{max(i,j)}$, $\varphi_{min(i,j)}$) und ($M_{max(i,j)}$, $M_{min(i,j)}$). Ausgangsgröße ist die von der Überwachungsebene berechnete Stabilitätsbeurteilung in Form des Stabilitätsgrades S. Teilmodelle, auf die im Weiteren näher eingegangen wird, sind die *Bewertungsfunktion* L, die *Korrektur der Bewertungsfunktion* L_{Extern}, die *korrigierten und gefilterten Bewertungsfunktionen* L_M, L_N, die *mittlere Bewertungsfunktion* L_{Mean}, der *Trend* D, der *gewichtete Trend* D_{gew} sowie der *Stabilitätsgrad*.

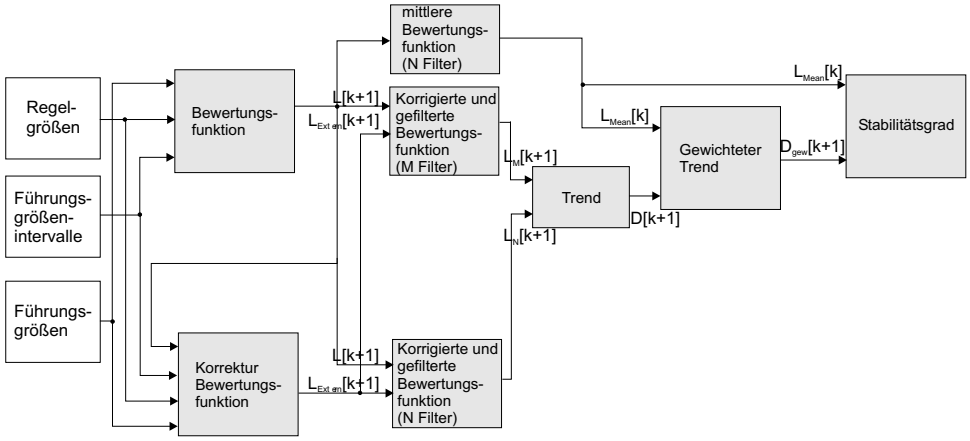


Bild 9: Struktur der Überwachungsebene

Ausgehend von einem unscharfen Stabilitätsbegriff nach [5, 19] werden bei diesem Ansatz Ljapunov-ähnliche Funktionen [5] als Bewertungsmaß eingesetzt, die die gewünschte Beurteilung tolerierbarer Ausgangsgrößen erlauben. Dieses Bewertungsmaß soll einen umso kleineren Wert liefern, je näher der aktuelle Zustand des Systems dem gewünschten Zustand ist. Allerdings ergeben sich bei der Anwendung von Ljapunov-ähnlichen Funktionen zur Überwachung von Regelkreisen Probleme, von denen die wichtigsten, sowie die dazu gehörenden Lösungsansätze, kurz dargestellt werden.

Nicht alle Prozesszustände sind immer mess- oder beobachtbar. Deshalb wird anstelle einer Ljapunov-Funktion eine *Bewertungsfunktion* L verwendet, die nicht unbedingt alle Prozesszustände, sondern nur die verfügbaren Größen enthält. Ein Beispiel

für eine solche Bewertungsfunktion ist:

$$L(e[k]) = L(\underline{w}[k], \underline{y}[k], \underline{w}_{max}[k], \underline{w}_{min}[k]) = \underline{e}[k]^T \; P \; \underline{e}[k] \tag{24}$$

mit der positiv definiten Matrix P:

$$W = \begin{pmatrix} W_{1,1} & 0 & \cdots & 0 \\ 0 & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & W_{n,m} \end{pmatrix} \quad G = \begin{pmatrix} G_{1,1} & 0 & \cdots & 0 \\ 0 & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & G_{n,m} \end{pmatrix}$$

$$P = \begin{pmatrix} W & 0_{n,m} \\ 0_{n,m} & G \end{pmatrix}$$

$(W_{1,1} \ldots W_{n,m})^T$ = Vektor der Wichtungselemente für die Gelenkwinkel
$(G_{1,1} \ldots G_{n,m})^T$ = Vektor der Wichtungselemente für die Momente

Der Vektor $\underline{e}[k]$ berechnet sich dann in Abhängigkeit von $\underline{y}$ und $\underline{w}_{max}$ bzw. $\underline{w}_{min}$:

$$\underline{e}[k] = \begin{cases} \underline{y}[k] - \underline{w}_{max}[k] & \text{für} \quad \underline{y} > \underline{w}_{max} \\ \underline{y}[k] - \underline{w}_{min}[k] & \text{für} \quad \underline{y} < \underline{w}_{min} \\ 0 & \text{sonst} \end{cases} \tag{25}$$

mit:

$\underline{w}$ $=(\varphi_{soll_{1,1}} \ldots \varphi_{soll_{n,m}}, M_{soll_{1,1}} \ldots M_{soll_{n,m}})^T$ =Vektor der Führungsgrößen
$\underline{y}$ $=(\varphi_{ist_{1,1}} \ldots \varphi_{ist_{n,m}}, M_{ist_{1,1}} \ldots M_{ist_{n,m}})^T$ =Vektor der Regelgrößen
$\underline{w}_{max}$ $=(\varphi_{max_{1,1}} \ldots \varphi_{max_{n,m}}, M_{max_{1,1}} \ldots M_{max_{n,m}})^T$ =Vektor der max. Führungsgrößen
$\underline{w}_{min}$ $=(\varphi_{min_{1,1}} \ldots \varphi_{min_{n,m}}, M_{min_{1,1}} \ldots M_{min_{n,m}})^T$ =Vektor der min. Führungsgrößen

Auf Stabilität eines autonomen Systems kann geschlossen werden, wenn L alle Zustände erfasst und

$$L[k+1] = \begin{cases} < L[k] & \text{für} \quad L[k] > 0 \\ 0 & \text{für} \quad L[k] = 0 \end{cases} \tag{26}$$

gilt.

Freie und erzwungene Signale sind zu separieren, da sonst Änderungen der Führungsgrößen $\Delta\underline{w}[k]$ zu einer schlechteren Stabilitätsbeurteilung durch die Bewertungsfunktion $L[k]$ führen. Deshalb werden die Führungsgrößenänderungen in der Bewertungsfunktion kompensiert, was im Weiteren zu einer *Korrektur der Bewertungsfunktion* $L_{Extern}[k]$ führt:

$$L_{Extern}[k+1] = L[k+1] - L(\underline{w}[k], \underline{y}[k+1], \underline{w}_{max}[k], \underline{w}_{min}[k]) \tag{27}$$

Der zweite Term in $L_{Extern}[k]$ stellt die Bewertung durch die Überwachungsebene zum Zeitpunkt [k+1] dar, falls die Führungsgrößen gleich geblieben wären ($\underline{w}[k+1] = \underline{w}[k]$, $\underline{w}_{max}[k+1] = \underline{w}_{max}[k]$, $\underline{w}_{min}[k+1] = \underline{w}_{min}[k]$). Im Gegensatz zu [5] werden in dieser Arbeit die Störgrößenänderungen nicht kompensiert, da sie von den Regelgrößen selbst abhängen, und nicht unabhängig am Ausgang der Strecke auftreten.

Da Gleichung (27) nur eine Beurteilung der Zeitpunkte k und $k+1$ ermöglicht, ist die Erweiterung um eine Rückwärtskorrektur erforderlich. Die sich dadurch ergebende Funktion wird als *korrigierte Bewertungsfunktion* bezeichnet. Bei der Bewertung ist weiterhin nicht eine kurzzeitige Verschlechterung des Bewertungsmaßes, sondern die längerfristige Entwicklung entscheidend. Deshalb erfolgt die Berechnung eines *Trends* durch *Filterung* der *korrigierten Bewertungsfunktion* anstelle der zeitlichen Ableitung

$$L_F[k+1] = a_F(L_F[k] + L_{Extern}[k+1]) + (1-a_F)L[k+1], \quad 0 < a_F < 1 \tag{28}$$

anhand von zwei Zeitfenstern M ($a_F = a_M$) und N ($a_F = a_N < a_M$). Ist der Mittelwert des Zeitfensters N kleiner als der Mittelwert des Zeitfensters M, so ist der Trend der Funktion negativ. Die Funktion D wird als *Trend* der *korrigierten und gefilterten Bewertungsfunktion* L_F bezeichnet:

$$D[k] = \frac{L_N[k] - L_M[k]}{T_N - T_M} \quad \text{mit} \quad T_N - T_M = T_A \frac{(a_M - a_N)}{(1-a_N)(1-a_M)} \tag{29}$$

Zusätzlich ergibt die Filterung der unkorrigierten *Bewertungsfunktion* L an, wie weit das System durchschnittlich von seinem Sollzustand entfernt war. Diese Größe wird als *mittlere Bewertungsfunktion* L_{Mean} bezeichnet:

$$L_{Mean}[k+1] = a_F L_{Mean}[k] + (1-a_F)L[k+1], \quad 0 < a_F < 1 \tag{30}$$

Weiterhin wird ein *gewichteter Trend* D_{gew} eingeführt, der den *Trend* D in Bezug zur *mittleren Bewertungsfunktion* L_{Mean} setzt. Dies ermöglicht eine Einschätzung des Systemverhaltens unabhängig von der Größenordnung der Bewertungsfunktion:

$$D_{gew}[k+1] = \frac{D[k+1]}{L_{Mean}[k]} \tag{31}$$

Die *mittlere Bewertungsfunktion* L_{Mean} und der *gewichtete Trend* D_{gew} werden von einem Fuzzy-System linguistisch interpretiert. Stützstellen der Zugehörigkeitsfunktionen sind NEG (NEGATIV), N (NULL), PK (POSITIV KLEIN), PG (POSITIV GROSS) (Bild 10).

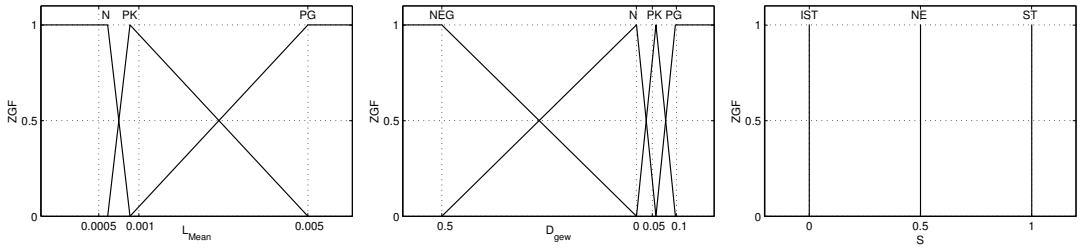


Bild 10: Zugehörigkeitsfunktionen zur Beurteilung des Stabilitätsgrades

Tabelle 1 zeigt die Fuzzy-Regelbasis zur Beurteilung des Stabilitätsgrades.

Als Ergebnis ergibt sich eine qualitative Stabilitätsbeurteilung mit den linguistischen Werten STABIL (ST), INSTABIL (IST) und NICHT ENTSCHEIDBAR (NE).

L_{Mean} / D_{gew}	N	PK	PG
NEG	STABIL	STABIL	STABIL
N	STABIL	NICHT ENTSCHEIDBAR	NICHT ENTSCHEIDBAR
PK	STABIL	NICHT ENTSCHEIDBAR	INSTABIL
PG	STABIL	INSTABIL	INSTABIL

Tabelle 1: Fuzzy-Regelbasis zur Beurteilung des Stabilitätsgrades

5 Ergebnisse

Um das vorgestellte Konzept zu testen, wird zunächst der Annäherungsvorgang eines einzelnen Fingerglieds an ein ortsfestes, verformbares Objekt mit einer anschließenden definierten Kontaktphase simuliert. Anhand von drei Beispielen (A, B, C) mit unterschiedlichen Materialkonstanten k (Gleichung (4)) und Reglerparametern K_{RP} und T_{NP}, aber sonst identischen Simulationsparametern, soll die Funktion des entwickelten Regelungs- und Überwachungskonzepts verdeutlicht werden. Um die Funktionsfähigkeit des Konzepts auch im Mehraktorfall nachzuweisen, wird als viertes Beispiel (D) der Annäherungsvorgang eines Fingers mit drei Gelenken an ein Objekt mit einer anschließenden Kontaktphase vorgestellt. Anhand dieses Beispielszenarios soll die Reaktionen der Überwachungsebene auf nicht zu realisierende Vorgaben der Bahnplanung dargestellt werden. Tabelle 2 zeigt die Parameter der untersuchten Szenarien.

Parameter / *Beispiel*	*Anzahl Gelenke*	k	K_{RP}	T_{NP}
A	1	0.5	0.2	0.2
B	1	0.5	**20**	**0.08**
C	1	**5**	**20**	**0.08**
D	**3**	0.5	0.2	0.2

Tabelle 2: Parameter der Beispielszenarien A, B , C und D

Die Bilder 11, 12 und 13 zeigen jeweils den Führungs- und Regelgrößenverlauf des Gelenkwinkels $\varphi_{soll(i,j)}, \varphi_{ist(i,j)}$ sowie die dazugehörenden Führungsgrößenintervalle $\varphi_{max(i,j)}$, $\varphi_{min(i,j)}$ in Abhängigkeit von der Kontaktsituation $K_{i,j}$ (oben links), den Aktordruck $p_{A(i,j)}$ und den Ventilbankdruck p_{VB} (oben rechts), den Führungs- und Regelgrößenverlauf des Gelenkmoments $M_{soll(i,j)}, M_{ist(i,j)}$ sowie die dazugehörenden Führungsgrößenintervalle $M_{max(i,j)}$, $M_{min(i,j)}$ in Abhängigkeit von der Kontaktsituation $K_{i,j}$ (unten links) und dem berechneten Stabilitätsgrad S (unten rechts) über der Zeit t. Die Stabilitätseinschätzung bezieht sich dabei stets auf die Mehrgrößenregelung von Position und Moment.

Deutlich zu erkennen ist der Umschaltvorgang der Führungsgrößenintervalle zum Zeitpunkt $t = 1.44$ s, hier findet der Objektkontakt statt. Der Regler schaltet vom positions- in den kraftgeregelten Modus. Während der *freien Bewegung* beträgt das Gelenkwinkel-Führungsgrößenintervall $\pm 3°$ Grad und das Momenten-Führungsgrößenintervall ± 0.25 Nm. Im *Kontaktfall* ist das Gelenkwinkel-Führungsgrößenintervall auf $\pm 17.5°$ Grad und das Momenten-Führungsgrößenintervall auf ± 0.025

Nm eingestellt. Ab dem Zeitpunkt 1.44 s arbeitet der Regler als reiner Momentenregler und regelt das von der Führungsgröße vorgegebene Moment $M_{soll(i,j)}$ ein.

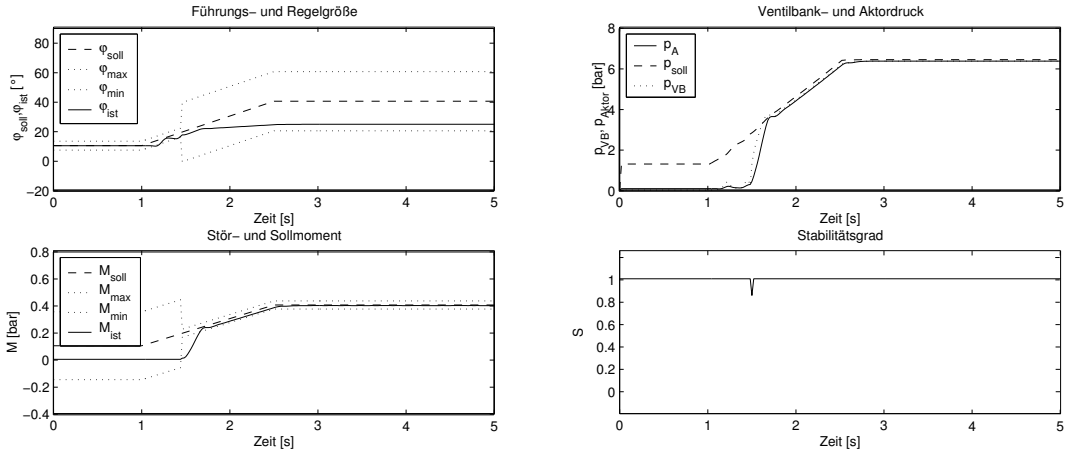


Bild 11: Führungs- und Regelgrößenverlauf, Druck im Aktor und der Ventilbank und Stabilitätsgrad über der Zeit (Beispiel A)

In Beispiel A beträgt die Materialkonstante $k = 0.5$, d. h. das Objekt ist leicht verformbar. Die Regelgröße $M_{ist(i,j)}$ befindet sich bereits kurz nach Objektkontakt, bedingt durch die in diesem Beispiel eingestellte geringe Objektsteifigkeit und 'gut eingestellten' Reglerparametern (Tabelle 2), innerhalb ihres vorgegebenen Intervalls $M_{max(i,j)}$, $M_{min(i,j)}$. Weiterhin bleibt auch die Regelgröße $\varphi_{ist(i,j)}$ innerhalb ihres vorgegebenen Intervalls $\varphi_{max(i,j)}$, $\varphi_{min(i,j)}$ (Bild 11).

Bild 11 (unten rechts) zeigt den Verlauf des Stabilitätsgrades S. Die Überwachungsebene detektiert die leicht verzögerte Reaktion des Reglers auf das Verlassen der Führungsgrößenintervalle ab $t = 1.44$ s mit einem Stabilitätsgrad von $S = 0.85$, d. h mit einer Stabilitätseinschätzung zwischen 'STABIL' und 'NICHT ENTSCHEIDBAR'. Ab dem Simulationszeitpunkt $t = 1.5$ s baut der Regler die Regeldifferenz ab, und es ergibt sich ein Stabilitätsgrad von $S = 1$, d. h. die Situation wird wieder als vollständig 'STABIL' eingestuft.

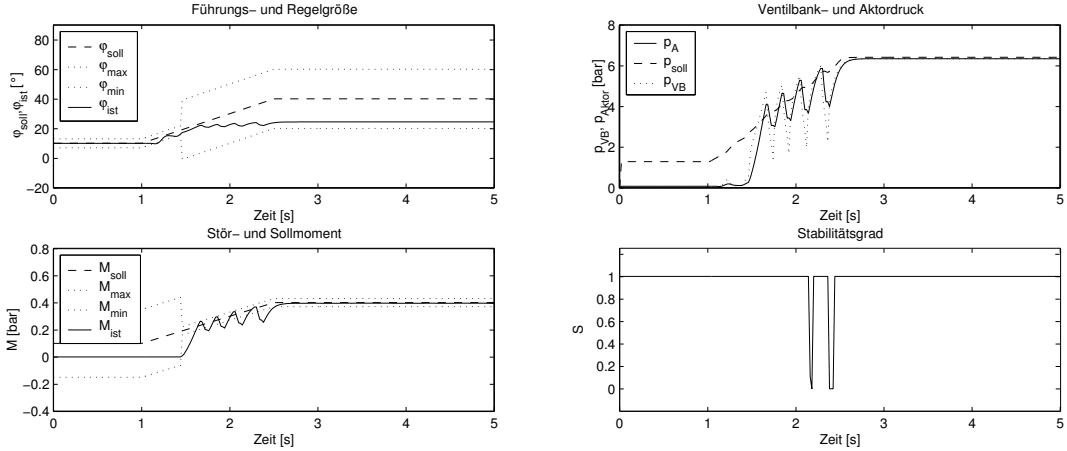


Bild 12: Führungs- und Regelgrößenverlauf, Druck im Aktor und der Ventilbank und Stabilitätsgrad über der Zeit (Beispiel B)

In Beispiel B werden bei unveränderter Materialkonstante die Reglerparameter K_{RP} und T_{NP} verstellt (Tabelle 2). Als Folge der zu hohen Reglerverstärkung von $K_{RP} = 20$ und der zu kleinen Zeitkonstante $T_{NP} = 0.08$ beginnt das System zum Zeitpunkt $t = 1.68$ s zu schwingen.

Die sich durch das Reglerfehlverhalten ergebenden Abweichungen des Systems vom Sollverhalten zu den beiden Zeitpunkten $t = 2.14$ s und $t = 2.36$ s werden von der Überwachungsebene erkannt und mit 'INSTABIL' beurteilt ($S = 0$). Ab dem Zeitpunkt $t = 2.68$ s ist der Regler in der Lage das System zu stabilisieren, der Stabilitätsgrad beträgt wieder $S = 1$ (Bild 12 unten rechts).

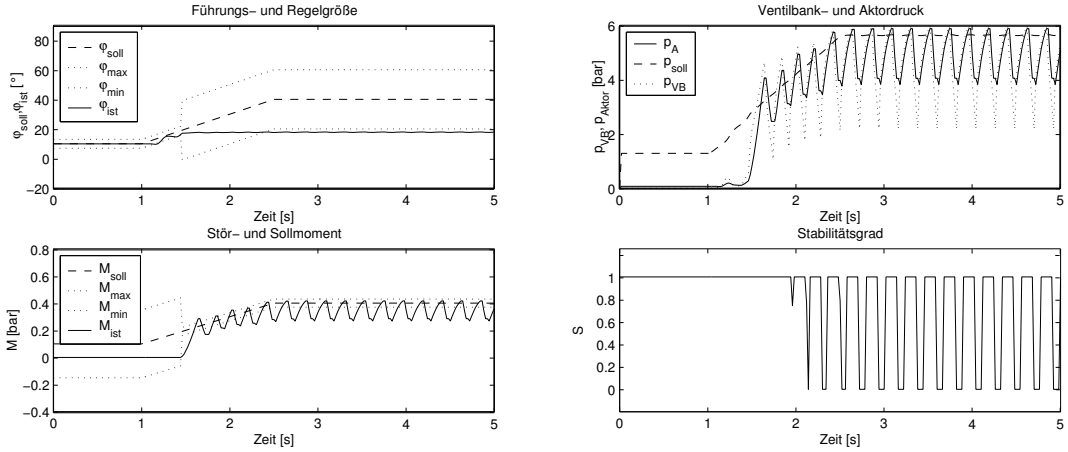


Bild 13: Führungs- und Regelgrößenverlauf, Druck im Aktor und der Ventilbank und Stabilitätsgrad über der Zeit (Beispiel C)

In Beispiel C wird bei unverändert 'falschen' Reglerparametern K_{RP} und T_{NP} zusätzlich die Materialkonstante auf $k = 5$ erhöht (Tabelle 2). Es ergibt sich bedingt durch das steifere Objekt ein größerer permanenter Positionsfehler $\varphi_{Fehler(i,j)}$ als bei den Beispielen A und B. Da der Regelalgorithmus im Kontaktfall nur das vorgegebene Moment $M_{soll(i,j)}$ einregelt, bleibt die Regelgröße $\varphi_{ist(i,j)}$ jetzt außerhalb des vorgegeben Intervalls $\varphi_{max(i,j)}$, $\varphi_{min(i,j)}$ (Bild 13 (links oben)).

Als Folge der großen Objektsteifigkeit und der (im Vergleich zu Beispiel B) noch schlechter an die Regelstrecke angepassten Reglerparameter, beginnt das System wiederum zum Zeitpunkt $t = 1.68$ s zu schwingen. Im Gegensatz zu Beispiel B ist der Regler diesmal allerdings nicht in der Lage, das System zu stabilisieren. Die Überwachungsebene detektiert das nicht zu tolerierende Systemverhalten mit periodisch wechselnden Stabilitätseinschätzungen zwischen 'STABIL' und 'INSTABIL' (Bild 13 rechts unten).

In Beispiel D wird die Anzahl der Gelenke, bei sonst gleichen Simulationsparametern wie in Beispiel A, auf 3 erhöht (Tabelle 2). Bild 14 zeigt die Führungs- und Regelgrößenverläufe sowie die dazugehörenden Führungsgrößenintervalle in Abhängigkeit von der Kontaktsituation, den Pumpenzustand, das Steuersignal der Pumpe und die Ventilzustände sowie den berechneten Stabilitätsgrad über der Zeit.

Der Regler arbeitet bis zum Zeitpunkt $t = 1.80$ s als Positionsregler. An den Regelgrößenverläufen für die Gelenkwinkel $\varphi_{ist,(i,j)}$ zwischen $t = 1.04$ s und $t = 1.80$ s ist deutlich die bereits diskutierte Problematik bei der Realisierung von gegenläufigen

Fingergelenkbewegungen zu erkennen, da der Regler sich nur für eine Pumpenrichtung und damit für ein entsprechendes pulsweitenmoduliertes Steuersignal pro Abtastintervall entscheiden kann. Aus diesem Grund ist der Regler nur wechselseitig in der Lage, Streck- und Beugebewegungen zu realisieren. Ab dem Zeitpunkt $t = 1.80$ s ergibt sich Objektkontakt. Der Regler schaltet vom positions- in den kraftgeregelten Modus und regelt die von den Führungsgrößen vorgegebenen Momente $M_{soll(i,j)}$ ein (Bild 14 (l)).

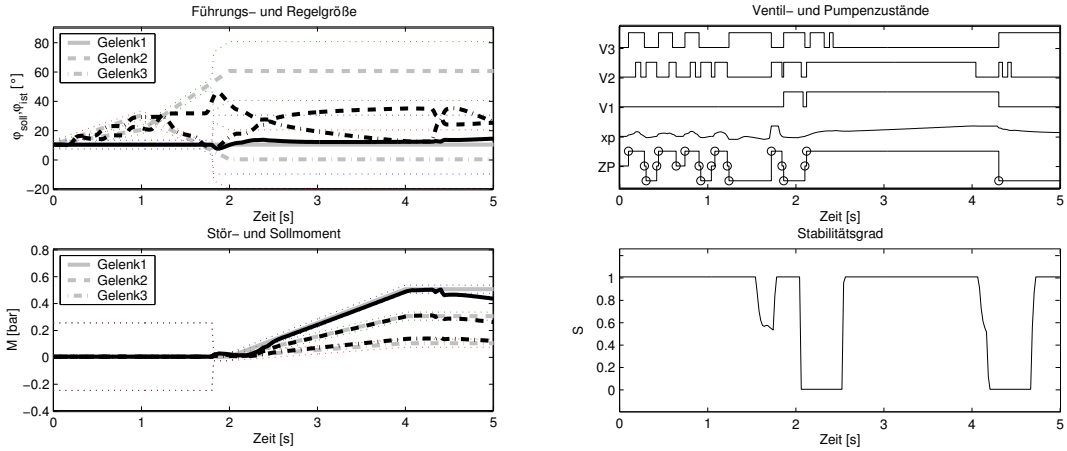


Bild 14: Führungs (grau)- und Regelgrößenverläufe (schwarz), Pumpenzustand (ZP), Steuersignal der Pumpe (x_P), Ventilzustände (V1-V3) und Stabilitätsgrad über der Zeit (Beispiel D)

Die Überwachungsebene detektiert das Verlassen des Führungsgrößenintervalls, bedingt durch die gegenläufige Führungsgrößenvorgabe ab $t = 1.36$ s, mit Stabilitätsgraden zwischen $S = 0.6$ und $S = 0.5$, d. h mit einer Stabilitätseinschätzung zwischen 'STABIL' und 'NICHT ENTSCHEIDBAR'. Resultierend aus dem Objektkontakt, ergeben sich zwischen den Zeitpunkten $t = 2.02$ s und $t = 2.54$ s sowie $t = 4.24$ s und $t = 4.66$ s vom Regler nicht auszuregelnde Abweichungen vom Gelenkwinkel-Sollverhalten $\varphi_{soll(i,j)}$, da der Regelalgorithmus im Kontaktfall nur das vorgegebene Moment $M_{soll(i,j)}$ einregelt. Die Überwachungsebene detektiert das nicht zu tolerierende Systemverhalten zu diesen Zeiten mit Stabilitätsgraden von $S = 0$, d. h. mit der Stabilitätseinschätzung 'INSTABIL' (Bild 13 rechts unten) und erkennt somit die nicht zu realisierenden Vorgaben der Bahnplanung.

Fazit: Die Überwachungsebene ist in der Lage, eine zusammengefasste Bewertung der Situation der Basisregler vorzunehmen. Solange sich die Basisregler im Sollbereich befinden oder für eine Annäherung an den Sollbereich sorgen, erfolgt eine positive Beurteilung der Gesamtsituation.

6 Zusammenfassung und Ausblick

In dieser Arbeit wurde ein Algorithmus zur Regelung und Überwachung von Greifvorgängen mit einem elastischen Robotergreifer vorgestellt und simulativ erprobt. Das Simulationsmodell beinhaltet sowohl das Antriebsystem des Robotergreifers (Hydraulik), die Handmechanik (Fingerglieder, Gelenke) als auch das zu greifende

Objekt. Zur Realisierung von Greifbewegungen wurde ein hybrides Kraft-Positionsregelkonzept entwickelt. Um die Stabilität und Sicherheit eines Greifvorgangs zu überwachen, wurde das Konzept der Fuzzy-Überwachungsebene aus [5] erweitert. Anhand von vier Beispielen wurde die Funktion des hybriden Regelungskonzepts und der modifizierten Fuzzy-Überwachungsebene diskutiert und bewertet.

In einem nächsten Schritt soll das vorgestellte hybride Regelungskonzept durch Parameteradaption, Modifikation von Totzonenbereichen sowie *unscharfe* Steuerung der Umschaltbedingungen des Reglers (positionsgeregelt - kraftgeregelt) verbessert werden. Weiterhin sollen andere aus der Literatur bekannte Regelungskonzepte wie z. B. Nachgiebigkeits-, Steifigkeits-, Impedanz- und Admittanzregelungen untersucht, verglichen und bewertet werden.

Das vorgestellte Überwachungskonzept ist dahingehend zu erweitern, dass neben dem Erkennen von Gefahrensituationen auch das Einleiten von Gegenmaßnahmen durch gezielte Beeinflussung des Reglers (Parameteradaption, Schaltbedingung, Reglerkonzept, 'NotAus') und der Bahnplanung möglich wird. Zudem erlaubt das Konzept die Einbeziehung weiterer Planungsinformationen in die Gesamteinschätzung wie z.B. erwartete Übergangszeiten zwischen Zuständen und Erreichen von Kontaktzuständen.

Die Autoren bedanken sich bei der DFG für die finanzielle Unterstützung der Forschungsarbeiten im Rahmen des SFB 588 "Lernende und kooperierende multimodale Roboter" [20] und bei unseren Kollegen Sebastian Beck und Thomas Lotz (Forschungszentrum Karlsruhe) sowie Dirk Osswald und Tamim Asfour (Universität Karlsruhe) für viele anregende Diskussionen.

Literatur

[1] Butterfass, J.; Grebenstein, M.; Liu, H.; Hirzinger, G.: DLR Hand II: Next generation of a dextrous robot hand. In: *International Conference on Robotics and Automation, Seoul, Korea*. 2001.

[2] Lovchik, C. S.; Diftler, M. A.: The Robonaut Hand: A Dexterous Robot Hand for Space. In: *Proceedings of the IEEE ICRA*, S. 907–913. Detroit. 1999.

[3] Fukaya, N.; Toyama, S.; Asfour, T.; Dillmann, R.: Design of the TUAT/Karlsruhe Humanoid Hand. *IEEE/RSJ ICIRS* (2000).

[4] Schulz, S.; Pylatiuk, C.; Bretthauer, G.: Entwicklung einer künstlichen Hand mittels flexibler Fluidaktoren. *VDI Bericht Nr. 1680, GMA Kongress* (2001), S. 143ff.

[5] Mikut, R.: *Modellgestützte on-line Stabilitätsüberwachung komplexer Systeme auf der Basis unscharfer Ljapunov-Funktionen*. Dissertation, TU Karlsruhe. Fortschr.-Ber. VDI Reihe 8 Nr. 757. Düsseldorf: VDI Verlag 1999; ISBN 3-18-375708-7, ISSN 0178-9546. 1999.

[6] Siavicco, L.; Siciliano, B.: *Modelling and Control of Robot Manipulators*, Bd. 2. 2001.

[7] Paul, R.: Robot Manipulators: Mathematics, Programming, and Control. *MIT Press* (1981).

[8] Craig, J. J.: *Introduction to Robotics*. Addison-Wesley. 1986.

[9] Readmann, M. C.: Flexible Joint Robots. *CRC Press* (1994).

[10] Spong, M.: Modelling and Control of Elastic Joint Robots. *IEEE Journal of Robotics and Automation* (1987), S. 291–300.

[11] R. Kelly, V. S.: Global Regulation of Elastic Joint Robots Based on Energy Shaping. *IEEE Transactions on Automatic Control* Vol.43, No.10 (1998), S. 1451–1456.

[12] Albu-Schäffer, A.: *Regelung von Robotern mit elastischen Gelenken am Beispiel der DLR-Leichtbauarme*. Dissertation, TU München. 2002.

[13] Beck, S.; Lehmann, A.; Martin, J.; Mikut, R.: Modellbildung und Fuzzy-Gelenkpositionsregelung für eine 5-Finger-Roboterhand mit flexiblen Fluidaktoren. In: *Proceedings 12. Workshop Fuzzy Systeme* (Mikut, R.; Reischl, M., Hg.), S. 177–191. Forschungszentrum Karlsruhe, FZKA 6767. 2002.

[14] Beck, S.; Lehmann, A.; Lotz, T.; Martin, J.; Mikut, R.: Modellgestützte adaptive Regelungskonzepte für eine fluidisch betriebene Roboterhand. In: *Proceedings GMA-Kongress*, S. 65–72. VDI Verlag. 2003.

[15] Osswald, D.; Martin, J.; Burghart, C.; Mikut, R.; Wörn, H.; Bretthauer, G.: Integrating a Robot Hand into the Control System of a Humanoid Robot. In: Humanoids [21]. 2003.

[16] Mirza, K.; Hanes, M.; Orin, D.: Dynamic Simulation of Enveloping Power Grasps. *IEEE Journal of Robotics and Automation* (1993), S. 430–435.

[17] Schlegl, T.: *Diskret-kontinuierliche Regelung mehrfingriger Roboterhände zur robusten Manipulation von Objekten*. Dissertation, TU München. Fortschr.-Ber. VDI Reihe 8 Nr. 928. Düsseldorf: VDI Verlag 2002; ISBN 3-18-392808-2, ISSN 0178-9546. 2001.

[18] Gross, D.; Hauger, W.; Schnell, W.: *Technische Mechanik*. Springer Lehrbuch. 1992.

[19] Bandemer, H.; Hartmann, S.: A Fuzzy Approach to Stability of Fuzzy Controllers. *Fuzzy Sets and Systems* 96 (1998), S. 161–172.

[20] Dillmann, R.: Towards Humanoid Robots in Human-Centered Environments. In: Humanoids [21]. 2003.

[21] *Humanoids 2003 International Conference on Humanoid Robots, Karlsruhe und München (Conference Documentation und CD-ROM)*. 2003.

Mobile Roboter im Wettkampf

Ivan Masár und Michael Gerke

Prozesssteuerung und Regelungstechnik
FernUniversität in Hagen
Universitätsstr. 27, 580 97 Hagen
Tel. (02331) 987 1117
Tel. (02331) 987 354
Ivan.Masar@FernUni-Hagen.de, Michael.Gerke@FernUni-Hagen.de

1 Kurzfassung

Mobile Roboter stellen sich dem Wettkampf. Mit der zunehmenden Miniaturisierung von Systemkomponenten und der gleichzeitigen Steigerung von verfügbaren System-Ressourcen (z.B. CPU-Leistung, Speicherplatz) wachsen die Einsatzmöglichkeiten mobiler Kleinroboter beständig an. Die Wahrnehmung der Umwelt über Sensorik, die Auswertung der Eingangsdaten auf leistungsfähiger Hardware und die autonome on-board Aktionsplanung durch intelligente Strategien (z.B. „computational intelligence“) ermöglicht inzwischen selbst komplexe Anwendungsszenarien. Gleichzeitig ergeben sich im Forschungsbereich durch zahlreiche Roboterwettkämpfe sowohl Synergien als auch Innovationsschübe. Ein spektakuläres Beispiel dazu sind die Wettbewerbe im Roboter-Fußball, durch welche die Arbeiten auf dem Gebiet der „verteilten künstlichen Intelligenz“ erhebliche Impulse erhalten haben.

In unserem Beitrag stellen wir den mobilen Kleinroboter F.A.A.K. vor, der entsprechend der Ausschreibung zum Roboterwettbewerb ISTROBOT entwickelt wurde. Dieser Wettbewerb findet jährlich in der Hauptstadt der Slowakei Bratislava statt, und er ist heutzutage eine der größten Veranstaltungen dieser Art in den osteuropäischen Ländern. Der ISTROBOT-Wettbewerb ist hauptsächlich für mobile Roboter bestimmt, die in zwei Hauptkategorien (Linienverfolgung – ‚Pathfinder‘ einerseits, andererseits Labyrinth-Durchquerung – ‚Maus im Labyrinth‘) teilnehmen können. Eine dritte Kategorie ist der sogenannten ‚Free Run‘; hierbei werden jährlich neben neuartigen mobilen Robotern auch verschiedene stationäre Systeme und Laufroboter präsentiert.

Obwohl die Aufgabenstellungen der einzelnen Kategorien relativ einfach sind und bis jetzt dazu schon zahlreiche Lösungen publiziert wurden, ist es immer sehr interessant, die kleineren oder größeren Modifikationen mit dem Ziel mindestens einen kleineren Vorsprung vor der Konkurrenz bei der Endabrechnung zu gewinnen, zu beobachten. Dass häufig nur minimale konstruktive Vorteile über das Ergebnis des Wettbewerbes entscheiden, bezeugen die manchmal nur hauchdünnen Zeitunterschiede zwischen den siegreichen Robotern. Weiterhin muss man sich vergegenwärtigen, dass - obwohl die Roboter kleine Abmessungen haben - es sich bei der Steuerung dieser Systeme keineswegs um eine einfache Aufgabe handelt, da die Roboter sich mit einer relativ hohen Geschwindigkeit im Vergleich zu ihrer Umgebung bewegen. Die kleinen Abmessungen komplizieren zudem die Ausrüstung des Roboters mit Steuerungs- und Sensorenelektronik und beschränken das Gewicht der mitgeführten Energiequellen.

Aus diesen Gründen spielen die Steuerungsalgorithmen des Roboters die größte Rolle beim Wettbewerb. Obwohl man die akzeptablen Ergebnisse bei der Aufgabendurchführung auch

mit ziemlich einfachen Steuerbefehlen erreichen kann, muss man für eine wirklich dynamisch leistungsfähige und präzise Steuerung sehr fortgeschrittene Strategien verfolgen. Als ein Beispiel der Anwendung der intelligenten Steuerungsstrategien präsentieren wir in unserem Beitrag einen selbstlernenden Neuro-Fuzzy-Regler für die Linienverfolgung.

2 Beschreibung des Roboters

2.1 Kinematische Struktur des Roboters

Der mobile Roboter F.A.A.K. wurde komplett am Lehrgebiet Prozesssteuerung und Regelungstechnik der FernUniversität in Hagen gebaut. Die beim Entwurf und der Konstruktion der kinematischen Struktur an den Robotern gestellten Anforderungen waren:

- kleine Abmessungen,
- ein niedriger Preis,
- eine hohe Manövrierfähigkeit
- und ausreichender Raum für die Platzierung der Sensoren, Akkus und der Steuerungselektronik.

Obwohl man die obigen Anforderungen auch mit einem klassischen kinematischen Aufbau mit zwei angetriebenen Rädern und mit einem/mehreren Stützrädern erfüllen könnte, wurde letztendlich eine Konstruktion mit vier gelenkten Rädern gewählt, wobei zwei davon angetrieben sind. Diese Lösung wurde vor allem wegen der besseren Stabilität bei schnelleren Fahrten und der Möglichkeit zum Testen von nicht-konventionellen Bewegungsarten gewählt. Andererseits benötigt eine solche Antriebsanordnung durchdachte Regelungsalgorithmen für die Synchronisation der Radbewegungen (der Drehwinkel der gelenkten und der Geschwindigkeiten der angetriebenen Räder), denn die spezielle Kinematik kompliziert das mathematische Modell und erhöht die Anforderungen an die CPU-Leistung und die sensorische Ausrüstung (Positions- und Geschwindigkeitsgeber) des mobilen Roboters. Der entworfene F.A.A.K.-Roboter ist in Bild 1 dargestellt.

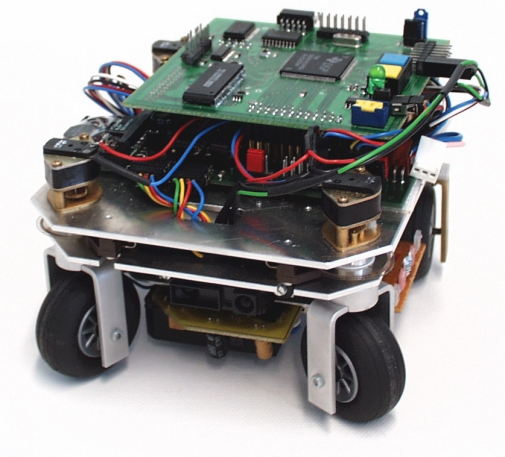


Bild 1 Der F.A.A.K. Roboter

Verschiedene Bewegungsarten des Roboters in Abhängigkeit vom Lenkungsverfahren des Roboters sind im Bild 2 dargestellt. Gezeigt sind zwei verschiedene Wendefahrten; bei der synchronen Lenkung (a) ändert sich nur der Radius der Kurve, während sich bei der asynchronen Lenkung (b) neben dem Radius auch der Drehpunkt – ICR (Instantaneous Center of Rotation) ändert. Eine weitere wichtige Bewegungsart bei der Fahrt in einem Raum mit Hindernissen ist die Drehung auf der Stelle (c), eine eher untypische Bewegung ist die Querfahrt (d).

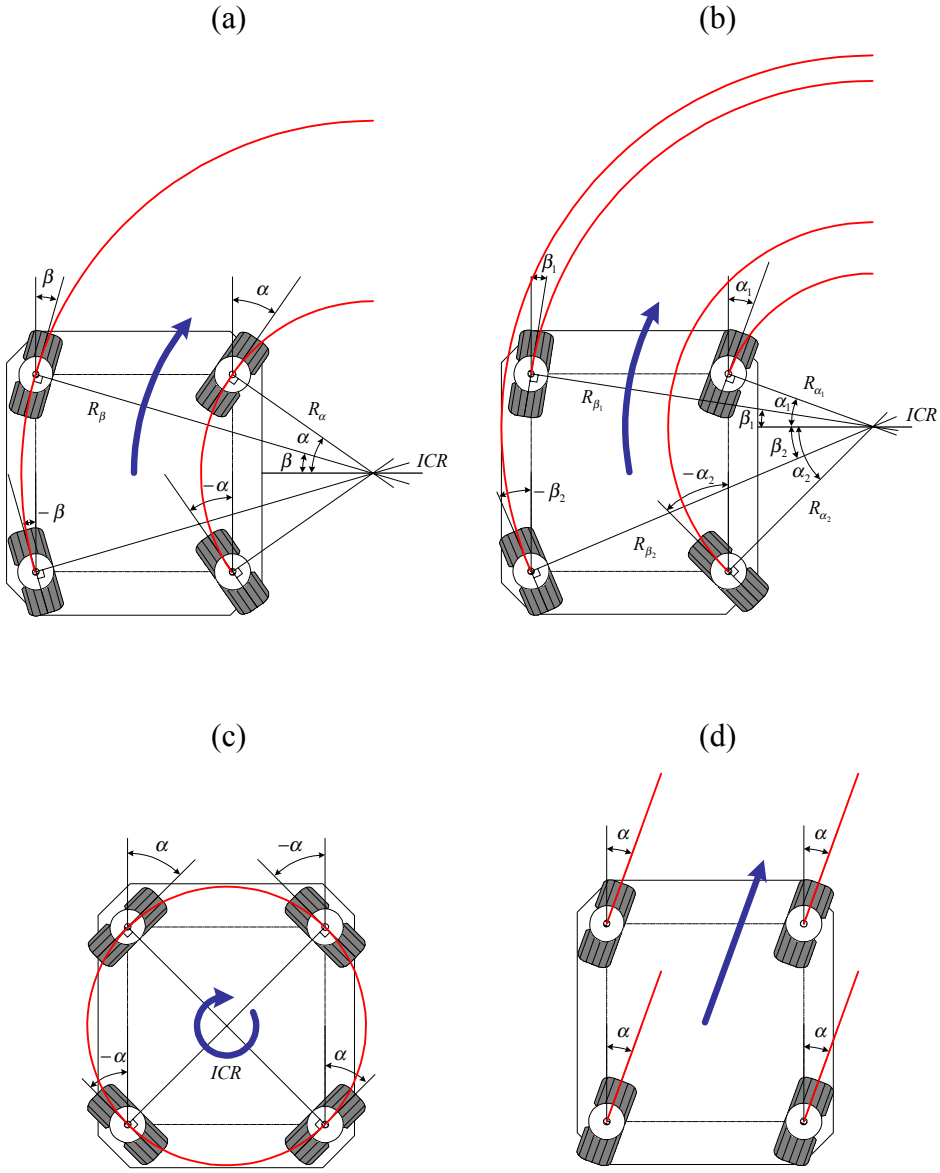


Bild 2 Die Bewegungsarten des Roboters F.A.A.K.: (a) und (b) – Wenden, (c) – Drehung, (d) – Querfahrt

In Abhängigkeit vom jeweiligen Lenkungsverfahren ändert sich auch die Zahl der Freiheitsgrade und der Steuerbarkeitsgrad des Roboterfahrzeugs. Nach [1] kann man diese Konfigurationen wie folgt aufteilen:

- Roboterart (1,2) – der Roboter hat einen Freiheitsgrad (die Geschwindigkeit) und zwei Steuerbarkeitsgrade (die Lenkungswinkel der vorderen/hinteren Rädern). Sofern nur ein Räderpaar gelenkt wird, reduziert sich die Roboterart auf (1,1).

- Roboterart (1,0) – in diesem Fall hat der Roboter einen Freiheitsgrad (die Rotation um seinen Mittelpunkt), aber keinen Steuerbarkeitsgrad, deshalb ist seine kinematische Struktur *degeneriert*.
- Roboterart (2,1) – der Roboter hat zwei Freiheitsgrade und nur einen Steuerbarkeitsgrad, da alle Räder um den gleichen Winkel gedreht werden.

Die Wahl der momentanen Konfiguration der Räder hängt von der Art der durchzuführenden Aufgabe und vom Ziel der Steuerung ab. Er kann auch eventuell gemäß einem Optimierungskriterium (z.B. Geschwindigkeitsoptimierung der Verfolgung einer definierten Trajektorie, Minimierung des Energieaufwandes bei den Bewegungen der hinteren Räder mit den Antriebsmotoren bei Einhaltung der Genauigkeit der Trajektorienverfolgung usw.). Der Entwurf eines Algorithmus für die optimierte Umschaltung zwischen den Räderkonfigurationen gehört zu den nächsten Zielen des Projektes.

2.2 Die Antriebe

Die Lenkung des Roboters sowie sein Antrieb wird mittels Getriebe-Gleichstrommotoren (FTB-Bertsch) realisiert. Es handelt sich um die 6V Motoren FTB 00-30 mit dem Getriebe M 11.00 mit der Übersetzung 176:1 für die Lenkungen und FTB 20-08 mit dem Getriebe M 12.00 mit der Übersetzung 23:1 für die Antriebe. Diese Motoren ermöglichen es dem Roboter, die Vorwärtsgeschwindigkeiten von maximal 0,6 m/s zu erreichen.

Für die Drehwinkelerkennung der Räder werden inkrementalen Geber (Agilent) vom Typ HEDS-9140 für die vorderen Räder bzw. HEDS-9730 für die angetriebenen Räder, beide mit einer Auflösung von 500 Impulsen pro Radumdrehung, benutzt. Für die Auswertung ihrer Ausgangssignale werden 24-Bit Zähler vom Typ LS7166 (Quadratur-Verfahren) angewendet, die bereits direkt mit dem Steuerungsprozessor mittels eines Datenbusses kommunizieren.

Die Motoren werden durch PWM-Signale gesteuert, die im Steuerungsprozessor generiert werden. Als Leistungsstufen werden L293D mit H-Brücken benutzt.

2.3 Die Sensoren

Die Anzahl, der Typ und die Fertigungsqualität der angewendeten Sensoren beeinflussen sehr stark die gesamten Fähigkeiten des Roboters und seine Verwendbarkeit in verschiedenen Umgebungen sowie beim Erfüllen von mannigfaltigen Aufgaben. Die jetzige sensorische Ausrüstung wurde so gewählt, dass der Roboter zwei elementare Fahrmissionen erfüllen kann – die Linienverfolgung und die Labyrinth-Durchquerung. Beide Aufgaben stellen sich in diversen Roboter-Wettbewerben, so z.B. bei der ISTROBOT in Bratislava, Slowakei.

Damit die Möglichkeit besteht, eine schwarze Linie auf dem weißen Boden zu verfolgen, ist der Roboter mit einem CCD-Zeilensensor Sony ILX551 mit insgesamt 2048 aktiven Pixeln ausgerüstet. Die Ansteuerimpulse für den Sensor werden im Steuerprozessor generiert, und das analoge Ausgangssignal wird von einem 8-Bit CCD-Digitizer MAX 1101 umgewandelt. Die Kommunikation zwischen diesem Chip und der CPU erfolgt mittels eines synchronen seriellen Links und dient zur Einstellung der Parameter des AD-Umwandlers sowie zur Übertragung des digitalisierten Bildes in den Steuerungsprozessor.

Für die Hindernisdetektion werden IR-Sensoren Sharp GP2D120 eingesetzt: Die Auswertung beruht auf dem Triangulationsprinzip, und es können damit Hindernisse im Abstand von 4 bis 30 cm erkannt werden. Ihr Ausgang ist ein Analogsignal, welches aber zum Abstand nicht proportional ist.

Damit auch solche Hindernisse erkannt werden, die näher als 4 cm entfernt sind (was eine notwendige Bedingung bei einer Labyrinth-Durchquerung ist), wurden zusätzliche IR-Sensoren auf den Roboterseiten angebracht. Diese Sensoren bestehen aus einer Photodiode und einem Phototransistor. Mit ihnen kann man den Arbeitsbereich der gesamten Sensorik von 0 bis 30 cm erweitern und auswerten.

2.4 Der Steuerungsprozessor

Ein wichtiger Teil des Entwurfes war die Auswahl eines geeigneten Steuerungsprozessors. Da der Roboter autonom arbeiten muss, sollte seine CPU in Echtzeit folgende Aufgaben durchführen:

- Abtastung und Auswertung der Sensorensignale,
- Berechnung der Steuersignale,
- Navigation in der Umgebung mittels der on-board Odometrie,
- Kommunikation mit einem Überwachungssystem einschließlich der Visualisierung seiner Bewegung auf diesem externen Rechner (ein PC mit den Entwicklungstools Matlab/Simulink und der Virtual Reality Toolbox),
- Durchführung von anspruchsvollen „intelligenten“ Handlungsstrategien wie z.B. Kollisionsvermeidung, Selbstlernen, Kartenbildung, Bahnplanung usw.

Neben den obigen Anforderungen sind bei einem mobilen System auch die Größe und der Stromverbrauch des Prozessors, die Kapazität und der Typ der Speicherbausteine, die Anzahl und der Typ der on-chip verwendbaren Peripherien (AD- und DA-Umsetzer, Zähler, Kommunikationsports usw.) sowie die Einsatzmöglichkeiten von hochwertigen Entwicklungswerkzeugen wichtig.

Unsere Anforderungen werden weitgehend durch den DSP-Prozessor TMS32LF2407A (Texas Instruments) erfüllt, der zu der sogenannten C2000-Entwicklungsreihe gehört. Diese Baureihe ist fokussiert auf leistungsfähige fixed-point Prozessoren für die Anwendungen in Steuerungsapplikationen. Diese Prozessoren besitzen einen DSP-Berechnungskern und dazu eine vielfältige Peripherie direkt auf dem Chip.

Der DSP-Prozessor TMS32LF2407A ist ein 40MHz 16-Bit Prozessor; er ist ausgerüstet mit insgesamt 2,5K RAM-Datenspeicher und einem 32K Flash-Programmspeicher, mit einer Schnittstelle zum externen Speicher, mit vier Timer/Counter-Modulen, einem Watchdog-Modul, einem 16-Kanal AD-Umwandler mit 10 Bit Auflösung, einem CAN-Modul, mit Modulen für die asynchrone (SCI) sowie auch für die synchrone (SPI) serielle Kommunikation und mit digitalen I/O-Pins und JTAG-Port für die Emulation des Prozessors.

Das Schema der Anbindung des Prozessors an die weiteren Subsysteme des mobilen Roboters ist im Bild 3 skizziert. Da in naher Zukunft eine Erweiterung des Roboters um eine visuelle Erfassungssensorik mit anschließender Bildbearbeitung mittels eines übergeordneten DSP-Prozessors vorgesehen ist, besitzt die Steuerungskarte des Roboters eine Schnittstelle über sogenannten Dual-Port RAM-Speicher für die Kommunikation zwischen beiden Prozessoren.

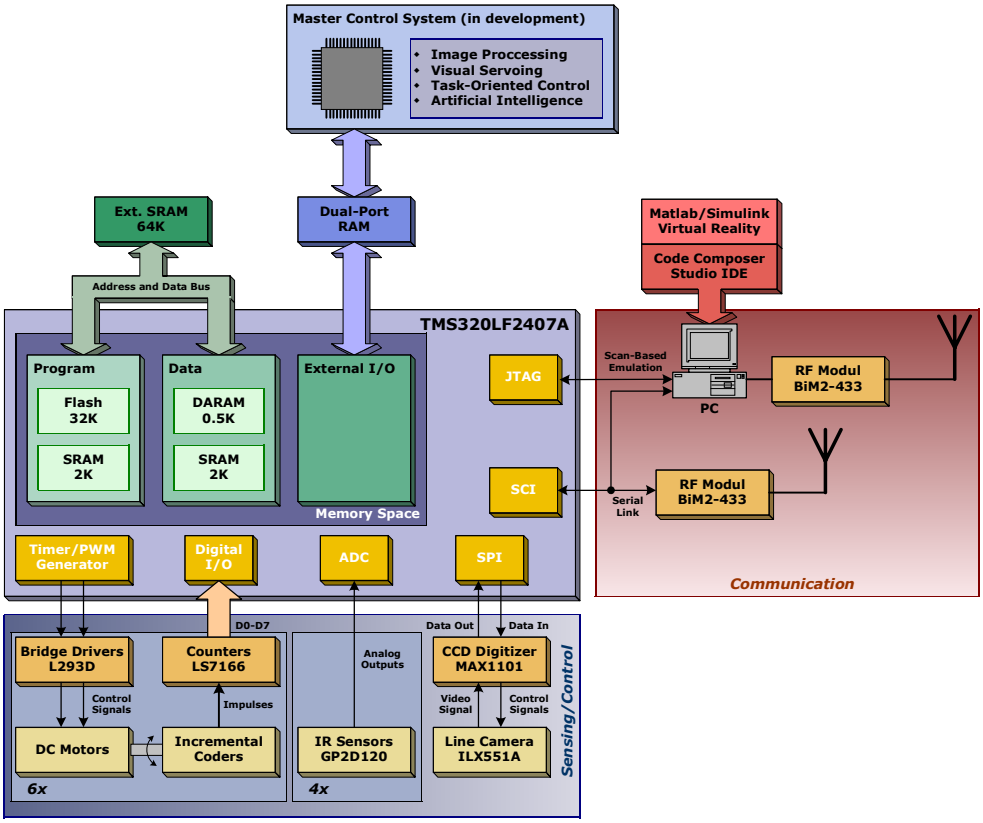


Bild 3 DSP TMS320LF2407A und seine Anbindung an die Subsysteme des Roboters

2.5 Die Kommunikation und die Überwachung des Roboters

Obwohl der Roboter die Aufgaben selbstständig durchführt, verfügt er über ein Funkmodul vom Typ Radiometrix BiM2-433 für die drahtlose serielle Kommunikation mit einem externen Überwachungssystem. Damit kann man im Halbduplexbetrieb die Daten mit einer Übertragungsgeschwindigkeit von bis zu 64 kbps übertragen. Die Daten werden auf dem externen PC dann in Matlab/Simulink bearbeitet und entweder in einem eigens entworfenen Auswertungsprogramm oder mittels der Virtual Reality Toolbox und des 3D-Robotermodells visualisiert.

Verschiedene Möglichkeiten der Visualisierung der Roboterdaten sind in Bild 4 dargestellt. Je nach Bedarf kann man dem Roboter vom externen Überwachungsrechner auch Steuerbefehle oder zusätzliche Informationen über seine lokale Umgebung senden.

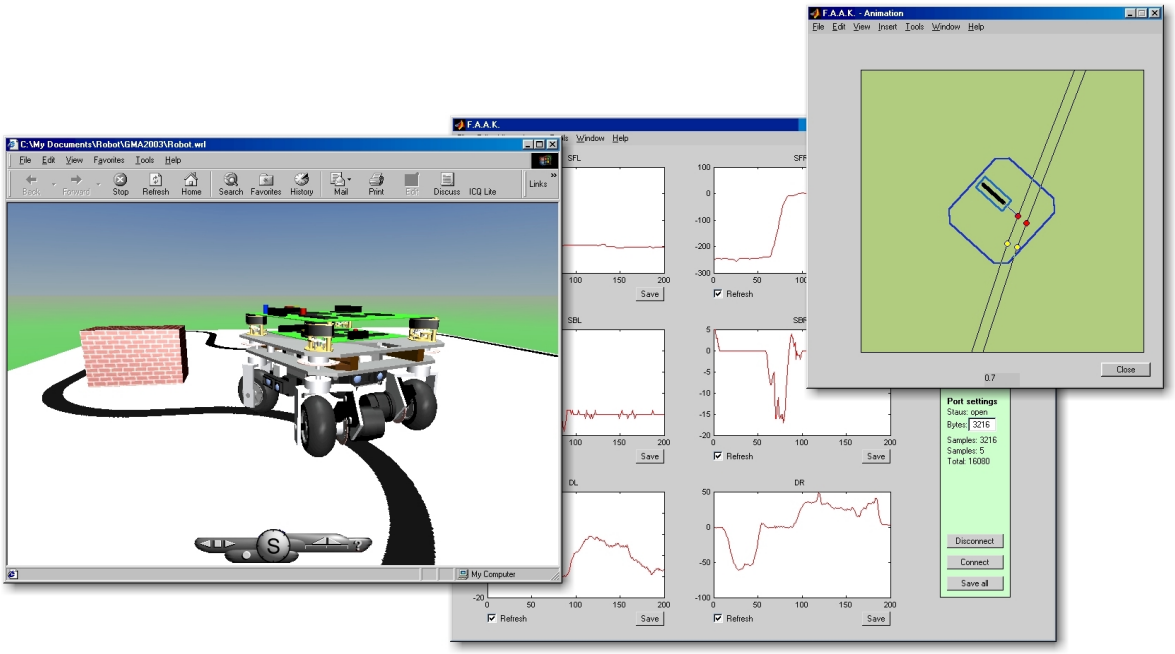


Bild 4 Visualisierung der vom Roboter gesendeten Daten

2.6 Das mathematische Modell des Roboters

Damit bei der Regelung des Roboters fortgeschrittene Steuerungsstrategien (Selbstlernen aus den verschiedenen Fahrsituationen, Navigation mit der Hilfe einer lokalen Karte usw.) angewandt und die Algorithmen bereits in Simulationen verifizieren werden können, wird ein möglichst vollständiges mathematisches Modell des Roboters benötigt. Derzeit benutzen wir für die Regelung, die Beobachtung der Roboterposition und die Simulationen ein kinematisches Modell des Roboters, aber für zukünftige Anwendungen soll auch ein dynamisches Modell entwickelt und durch dessen Verwendung die Regelungsqualität noch weiter erhöht werden.

Für die Ableitung eines kinematischen Modells ist es sinnvoll, die Struktur des Fahrgestelles des realen Roboters zunächst auf einen Fahrzeug gemäß Bild 5 mit nur zwei Rädern zu reduzieren, d.h. für jede Lenkachse wird nur ein Ersatzrad in der Mitte der Achse betrachtet. Die Rechtfertigung dafür ist die, dass die Lenkwinkel der beiden Räder auf einer Achse sowieso abhängig gesteuert werden müssen, damit kein Schlupf entsteht. Durch diese berechtigte Reduktion wird auch der Berechnungsaufwand für die CPU bei Verwendung des dynamischen Modells in der Echtzeitregelung des realen Fahrzeuges vermindert.

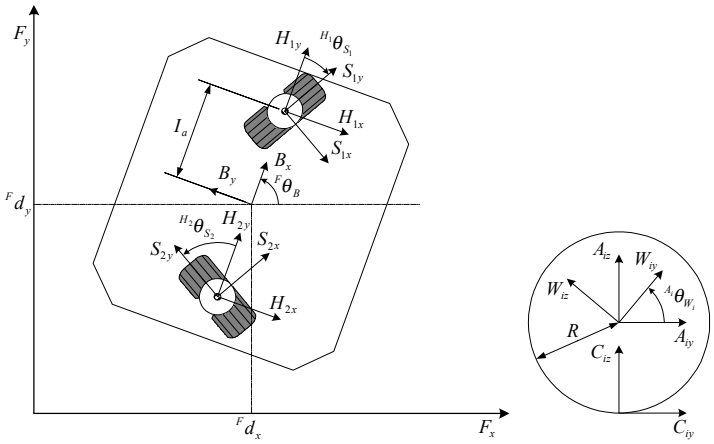


Bild 5 Die Koordinaten des Robotermodells

Bei der Ableitung des kinematischen Modells gemäß [2] werden dem mobilen Roboter folgende Koordinatensysteme zugeordnet:

- F – Floor: ein stationäres Koordinatensystem mit der Z-Achse senkrecht zum Boden,
- B – Body: ein Koordinatensystem, welches sich mit dem Roboter bewegt,
- H_i – Hip: dieses Koordinatensystem bewegt sich auch mit dem Roboter, sein Ursprung liegt in der Mitte der Lenkachse,
- S_i – Steering: dieses Koordinatensystem bewegt sich mit dem gelenkten Rad, der Winkel ${}^{H_i}\theta_{S_i}$ ist der Lenkungswinkel des *i*-tes Rades,
- C_i - Contact Point: ein Koordinatensystem, welches sich mit dem *i*-ten Rad bewegt, mit dem Ursprung im Kontaktpunkt zwischen dem Boden und dem Rad,
- A_i - Axle: dieses Koordinatensystem fällt mit der Rotationsachse des *i*-tes Rades zusammen,
- W_i - Wheel: dieses Koordinatensystem dreht sich mit dem *i*-ten Rad um die Achse A_i.

Das „gemessene“ kinematische Modell des Roboters ist dann

$$
{}^F\dot{\mathbf{p}}_B = \begin{pmatrix} {}^F v_{Bx} \\ {}^F v_{By} \\ {}^F \omega_B \end{pmatrix} = \mathbf{M} \begin{pmatrix} -R\dfrac{\sin({}^{H_3}\theta_{S_3})\cos({}^{H_1}\theta_{S_1}) + \sin({}^{H_1}\theta_{S_1})\cos({}^{H_1\,H_3}\theta_{S_3})}{2\cos({}^{H_1}\theta_{S_1})} \\ R\cos({}^{H_3}\theta_{S_3}) \\ R\dfrac{\sin({}^{H_1}\theta_{S_1})\cos({}^{H_1\,H_3}\theta_{S_3}) - \sin({}^{H_3}\theta_{S_3})\cos({}^{H_1}\theta_{S_1})}{2I_a\cos({}^{H_1}\theta_{S_1})} \end{pmatrix} {}^{A_3}\omega_{W_3},
$$

wobei der Vektor ${}^F\dot{\mathbf{p}}_B$ der relativen Bewegung des Roboters zum stationären Koordinatensystem entspricht. M ist die Rotationsmatrix zwischen dem Boden- und Roboterkoordinatensystem und somit definiert als

$$\mathbf{M} = \begin{pmatrix} \cos({}^{F}\theta_{B}) & -\sin({}^{F}\theta_{B}) & 0 \\ \sin({}^{F}\theta_{B}) & \cos({}^{F}\theta_{B}) & 0 \\ 0 & 0 & 1 \end{pmatrix}.$$

Die Geschwindigkeit des angetriebenen (hinteren) Rades des Roboters wird mit ${}^{A_3}\omega_{W_3}$ bezeichnet.

Mit Hilfe dieses Modells kann man einen Positionsbeobachter (Dead-Reckoning-Verfahren) für den Roboter ableiten:

$${}^{F}\tilde{\mathbf{p}}_{B}[kT] = \begin{pmatrix} {}^{F}\tilde{d}_{Bx} \\ {}^{F}\tilde{d}_{By} \\ {}^{F}\tilde{\theta}_{B} \end{pmatrix} = {}^{F}\tilde{\mathbf{p}}_{B}[(k-1)T] + \frac{T}{2}\mathbf{M}[(k-1)T]\left({}^{A_3}\omega_{W_3}[(k-1)T] + {}^{A_3}\omega_{W_3}[kT]\right),$$

wobei T die Abtastzeit bezeichnet.

3 Neuro-Fuzzy-Regler für eine Linienverfolgung

Als erste Anwendung der Werkzeuge der künstlichen Intelligenz für unseren Rennroboter entwarfen wir einen Neuro-Fuzzy-Regler für die Linienverfolgung.

Bei dieser Disziplin muss der Roboter eine 15 mm breite schwarze Linie auf dem weißen Boden verfolgen. Die Linie kann an beliebiger Stelle unterbrochen sein, und verschiedene Hindernisse können auf ihr aufgestellt sein. Außerdem führt die Linie auch durch einen Tunnel, in welchem beschränkten Lichtbedingungen herrschen. Bei dieser Disziplin hat jeder Roboter drei Fahrversuche, und am Ende wird die schnellste Zeit berücksichtigt, die der Roboter für die Verfolgung des kompletten Linienzuges braucht.

Aus der Aufgabenstellung resultieren folgende strategische Überlegungen:

- Eine große Rolle spielt die Qualität der Verfolgung selbst, d.h. mit welchem Geschwindigkeitsprofil kann der Roboter die Linie abfahren, wie reagiert er in den Kurven, was zieht er vor – ein Geschwindigkeitsprofil ohne überlagerte Oszillation (Zittern) oder eine präzise Positionierung bei der Linienverfolgung.
- Da der Roboter drei Versuche hat, kann er am Anfang die Linie explorativ (und damit langsamer) abfahren, die Strecke und die Hindernisse abspeichern, geeignete Geschwindigkeitsprofile berechnen und danach die Linie schneller abfahren.

Die Anwendung eines Neuro-Fuzzy-Regelungsverfahrens war in diesem Fall sehr günstig aus diesen Gründen:

- Obwohl der Roboter bei der Linienverfolgung mit einem einfachen PD-Regler steuerbar ist, ist die resultierende Qualität der Regelung dann nicht sehr hoch. Das Problem dabei ist, dass ein solcher Regler nur in bestimmten Arbeitsbereichen der Roboterbewegung gut arbeitet, aber bei anderen Geschwindigkeiten und/oder Kurvenradien kann der Roboter die Linie verlieren oder nur mit großen Überschwingungen (Pendeln) verfolgen. Eine Lösung ist, die maximale Geschwindigkeit des Roboters für den schlimmsten Fall (die schärfste Kurve) zu beschränken, was aber nicht wirklich wünschenswert ist.

- Man kann eventuell mehrere PD-Regler entwerfen (leider oft nur mit einer trial-and-error-Methode) und dann zwischen diesen Reglern umschalten; aber auch dies führt sehr oft zu Oszillationen und zu Sprüngen der Stellgrößen. Deshalb ist hier eine Fuzzy-Umschaltung geeigneter.
- Einige Trainingsversuche bieten bereits ideale Möglichkeiten, die Parameter des Reglers mit selbstlernenden Algorithmen einzustellen.

Der Entwurf des Neuro-Fuzzy-Reglers für die Linienverfolgung besteht aus zwei Schritten:

- der Implementierung eines Takagi-Sugeno Fuzzy-Reglers, der zwischen verschiedenen PD-Reglern umschaltet gemäß dem auftretenden Verfolgungsfehler und den jeweiligen Kurvenparametern und dessen Ausgänge die Lenkwinkel sowie die Robotergeschwindigkeit sind, und
- einem Adaptieren der Parameter der Fuzzy-Regler während der ersten Testfahrten des mobilen Roboters.

Die Struktur des Regelkreises mit dem Neuro-Fuzzy-Regler ist in Bild 6 dargestellt. Weitere Informationen über die Regler-Struktur, den benutzten Lern-Algorithmus sowie die Roboter-Demonstration werden beim Workshop präsentiert.

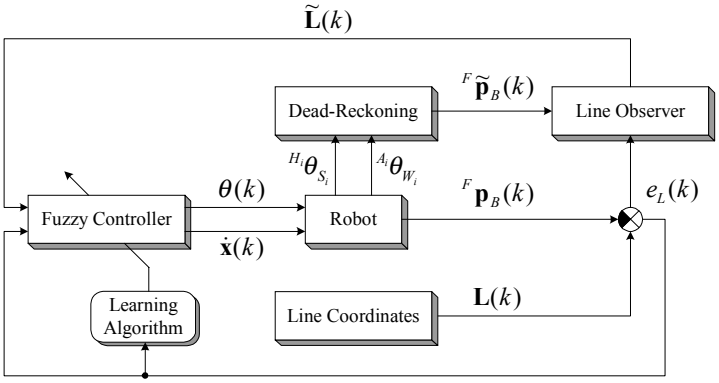


Bild 6 Neuro-Fuzzy-Regler für die Linienverfolgung

4 Literatur

[1] Canudas de Wit, C., Siciliano, B., Bastin, G.: *Theory of Robot Control.* Springer-Verlag, London; 1996

[2] Muir, P. F.: *Modeling and Control of Wheeled Mobile Robots. Ph.D. Thesis.* UMI, Michigan; 1990

[3] Jang, J.-S. R., Sun, C.-T., Mizutani, E.: *Neuro-Fuzzy and Soft-Computing.* Prentice-Hall, Upper Saddle River; 1997

[4] Internet: http://www.robotika.sk/mains.htm

Adaptive modellbasierte Weißeregelung für die Altpapieraufbereitung

Thomas Runkler[1], Rainer Palm[1], Klaus Villforth[2],
Markus Dinkel[3], Thomas Schmidt[3], Reinhold Götz[4]

[1]) Siemens AG, Corporate Technology, Information and Communications,
Neural Computation, CT IC 4, 81730 München,
Tel.: (089) 636–45372, Fax: (089) 636–49767, E-Mail: Thomas.Runkler@siemens.com

[2]) Technische Universität Darmstadt, FG Papierfabrikation und Mechanische Verfahrenstechnik, Alexanderstraße 8, 64283 Darmstadt

[3]) Gebr. Lang GmbH Papierfabrik, Fabrikstraße 4, 86833 Ettringen

[4]) Siemens AG, Industrial Solutions and Services,
I&S IP PEP PM, Schuhstraße 60, 91050 Erlangen

Kurzfassung

Ein wichtiger Teilprozess der Altpapieraufbereitung ist die Flotation, in der Druckfarben aus der Faserstoffsuspension entfernt werden. Dieser Prozess ist stark totzeitbehaftet, stark gestört und engen technologischen Begrenzungen unterworfen. Darüber hinaus ist die Effektivität der Druckfarbenentfernung nicht direkt messbar. Zur Regelung der Flotation wurden daher Beobachter (Soft–Sensoren) für die Schätzung der Qualität sowie Prozessmodelle für die Steuerung der Flotationsprozesse entwickelt. Diese Modelle basieren auf Mess– und Labordaten, aus denen mit Fuzzy–Clustermethoden adaptive Takagi–Sugeno–Modelle gewonnen wurden. Die Flotationsregelung SiFlot ist in WinCC integriert und seit Mai 2003 bei der Firma Lang Papier in Ettringen im Einsatz. Sie ermöglicht eine deutliche Reduktion der Qualitätsschwankungen.

Stichworte: Flotationsregelung, Soft–Sensor, Fuzzy–Clustering, Takagi–Sugeno–Modell

1 Einführung

Ein wichtiger Prozess-Schritt beim Recycling von Altpapier für graphische Papiere ist die Druckfarbenentfernung. Beim sogenannten Flotationsdeinking strömen Luftblasen durch die Faserstoffsuspension, an die sich durch Adhäsionseffekte Druckfarbenpartikel anlagern und mit dem Schaum ausgetragen werden. Ziel ist es, diesen Prozess so zu regeln, dass der Faserstoff eine vorgegebene Zielweiße oder Helligkeit erreicht und einhält. Diese Regelung wird im wesentlichen durch vier Aspekte erschwert:

a) Der Sollwert der Regelung bezieht sich auf die Weiße des fertigen Papiers, die vorab im Labor an aufwändig hergestellten Probeblättern gemessen wird. Die Druckfarbenentfernung ist nur ein Schritt in der Aufbereitungskette. Tatsächlich vergehen viele Stunden zwischen dem Eintrag des Altpapiers und der Papierherstellung. Dies erschwert die gezielte Beeinflussung der Papierqualität durch die Anlagenfahrer erheblich.

b) Die Durchlaufzeit der Fasern durch die Druckfarbenentfernung beträgt etwa 10-30 Minuten. Eine reine Regelung auf das Resultat der Druckfarbenentfernung (Akzept) kann daher nur langsam reagieren.

c) Der Prozess der Druckfarbenentfernung unterliegt sehr starken Störungen, die im wesentlichen durch die schwankende Zusammensetzung des Rohstoffs Altpapier verursacht werden. Diese Zusammensetzung ist zeitnah praktisch nicht erfassbar oder gar steuerbar.

d) Der Wertebereich der Stellgrößen ist aus technologischen Gründen stark eingeschränkt. Die erreichbare Regelgüte ist daher begrenzt.

Es wurde ein adaptiver modellbasierter Regler entworfen, der diese Schwierigkeiten folgendermaßen umgeht:

a) Um eine quasi totzeitfreie Regelung der Druckfarbenentfernung zu ermöglichen, wurden Beobachter (Soft-Sensoren) entwickelt, die auf Messungen optischer Spektren, Stoffdichten, sowie der Fein- und Füllstoffgehalte von Faserstoffsuspensionen basieren. Aufgrund der Nichtlinearität zwischen den optischen Eigenschaften in der Suspension und auf dem Blatt werden hierzu Takagi-Sugeno-Modelle [23] verwendet, die mit Gustafson-Kessel-Clustering [6] trainiert werden.

b) Für den Prozess der Druckfarbenflotation selbst wurde ein datenbasiertes Modell entwickelt, das eine Vorwärtssteuerung und damit ein schnelles Reagieren auf Eintragsschwankungen ermöglicht. Durch Einbeziehung der Akzeptqualität sowie durch integratorische Anteile im Regler werden bleibende Regelabweichungen vermieden.

c) Zum Ausgleich der Störungen erfolgt eine Adaption des Prozessmodells bzw. der Vorwärtssteuerung.

d) Die Stellgrößen werden mit Sigmoid-Funktionen beschränkt. Dadurch erfolgt auch in der Nähe der Begrenzungen ein gewisses Eingreifen des Reglers.

Die Ergebnisse von ersten Ansätzen zu diesen Arbeiten wurden bereits in [17, 18, 19, 20] vorgestellt. In diesem Artikel wird das auf der Anlage bei Lang Papier in Ettringen dauerhaft in Betrieb befindliche Optimierungssystem detailliert präsentiert. Der Artikel ist wie folgt gegliedert: In Abschnitt 2 wird ein Überblick über die Altpapieraufbereitung und speziell die Druckfarbenentfernung in der Flotation gegeben. In Abschnitt 3 wird das Konzept der Soft–Sensoren und die modellbasierte Regelung der Flotation beschrieben. In Abschnitt 4 werden einige Grundlagen der Clusteranalyse vorgestellt, auf denen die hier eingesetzte Fuzzy–Modellierung basiert. In Abschnitt 5 wird gezeigt, wie sich die

Ergebnisse der Clusteranalyse verwenden lassen, um Fuzzy–Modelle (Takagi–Sugeno–Modelle) aus Daten zu erstellen. In den Abschnitten 6 und 7 werden die Methoden zur Modelladaption, –inversion und zur Stellgrößenbegrenzung vorgestellt, die in dieser Anwendung genutzt werden. Die Ergebnisse und Erfahrungen im laufenden Betrieb sind schließlich in Abschnitt 8 zusammengefasst.

2 Altpapieraufbereitung

Abbildung 1 zeigt den vereinfachten Ablauf der Aufbereitung von Altpapier [4].

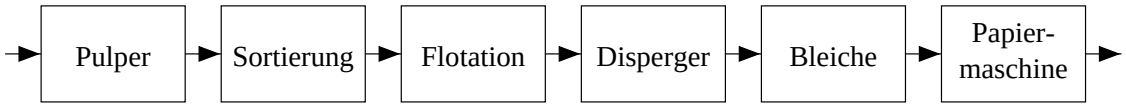


Abbildung 1: Ablauf der Altpapier–Aufbereitung.

Die erste Stufe ist der *Pulper*, in dem durch das Auflösen des Altpapiers in Wasser eine Suspension erzeugt wird. Dieses Auflösen erfolgt zumeist chargenweise, während die folgenden Prozess–Schritte kontinuierlich arbeiten. Die Durchmischung der Pulper–Füllungen ist jedoch gering, so dass in den folgenden Prozess–Schritten die Chargen–Grenzen noch deutlich erkennbar sind. Nach der Auflösung werden zunächst grobe Verunreinigungen in der Fasersuspension im Zentrifugalfeld von ihr getrennt *(Sortierung)*. Die folgenden Prozess–Schritte beziehen sich hauptsächlich auf die Herstellung graphischer Papiere. Nach dieser Grobreinigung des Faserstoffs werden in der *Flotation* [2, 5] abgelöste Druckfarbenpartikel ausgetragen. Danach werden die in der Suspension verbliebenen Druckfarbenpartikel in einem *Disperger* zerkleinert, so dass sie für das Auge nicht mehr sichtbar sind. Eine Erhöhung der Weiße des Faserstoffs wird anschließend durch eine *Bleiche* erzielt. Schließlich wird in der *Papiermaschine* aus der Faserstoffsuspension Papier hergestellt. In vielen Anlagen werden die Prozess–Schritte Flotation, Dispergierung und Bleiche mehrfach hintereinander ausgeführt. In der hier betrachteten Anlage bei der Firma Lang Papier in Ettringen wird eine zweistufige Druckfarbenflotation eingesetzt. Die einzelnen Stufen werden als Vorflotation und Nachflotation bezeichnet, die in sich in Primär- und Sekundärstufe kaskadiert sind. Die in diesem Artikel beschriebene Weißeregelung [9] bezieht sich auf die Regelung von Vor– und Nachflotation.

Abbildung 2 zeigt einen schematische Längsschnitt durch eine Flotationsstufe [5]. Von links durchströmt die Stoffsuspension eine Reihe von Flotationszellen. In den Flotationszellen wird die Faserstoffsuspension mit Luftblasen angereichert. Durch Adhäsionseffekte lagern sich Druckfarbenpartikel an die Luftblasen an und werden mit ihnen an die Oberfläche getragen. An der Oberfläche bildet sich ein schwarzer Schaum (Rejekt), der entfernt und stofflich oder thermisch verwertet wird. Die verbleibende Suspension (Akzept) wird zu den folgenden Prozess–Stufen weiter geleitet.

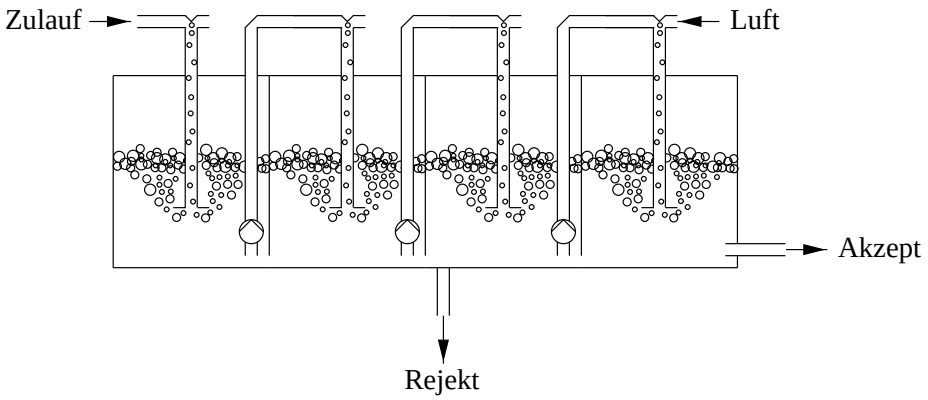


Abbildung 2: Längsschnitt einer Flotation (Schema).

Der Wirkungsgrad der Flotationszelle hängt nichtlinear von zahlreichen verschiedenen Prozessparametern ab, z.B. von den Volumenströmen im Zulauf, Akzept und Rejekt, von der (praktisch nicht messbaren) Faserstoffzusammensetzung, von der Deinking–Chemie, von den optischen Eigenschaften der Fasersuspension, von der Stoffdichte im Zulauf und vom Fein– und Füllstoffgehalt der Suspension. Wegen dieser vielfältigen Abhängigkeiten ist es Stand der Technik, Flotationszellen mit konstanter Einstellung zu betreiben.

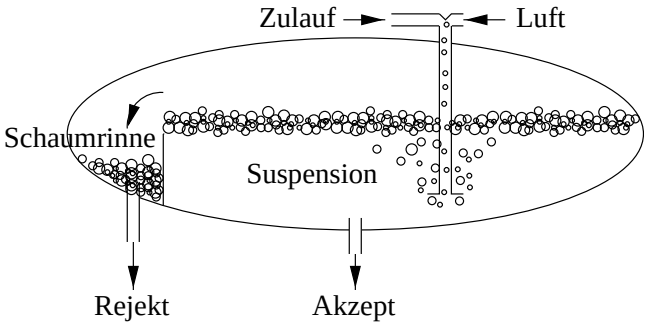


Abbildung 3: Querschnitt einer Flotation (Schema).

Abbildung 3 zeigt den Querschnitt einer Flotationszelle. Der an der Oberfläche angesammelte Schaum fällt links in die sogenannte Schaumrinne und wird dort als Rejekt ausgetragen. Das übliche Regelungskonzept misst die Höhe des in der Schaumrinne befindlichen Schaums und regelt den Akzeptfluss über ein Stellventil so, dass das Schaumrinnen–Niveau konstant bleibt. Der Nachteil dieses Regelungskonzepts ist, dass bei guter Eintragsqualität zu viel und bei schlechter Eintragsqualität zu wenig Rejekt produziert wird. Dies führt zu unnötig hohen Produktionskosten und starken Schwankungen in der Akzeptqualität.

3 Soft–Sensorik und Weißeregelung

Ein wesentlicher Qualitätsparameter von graphischen Papieren ist die Weiße, definiert als spektraler Reflexionsfaktor bei einer Schwerpunktswellenlänge von 457 nm. In der Fasersuspension lassen sich neben der Stoffdichte und dem Fein–/Füllstoffgehalt die optischen Reflektionen bei verschiedenen Wellenlängen (rot, grün, blau, Infrarot) messen. Die optischen Eigenschaften des Papierblatts unterscheiden sich jedoch wesentlich von den optischen Eigenschaften der Fasersuspension. Zur Bestimmung der Akzeptqualität werden

daher Stoffproben aus dem Akzeptstrom entnommen, daraus Probeblätter hergestellt und diese dann optisch vermessen. Dies ist ein sehr zeit– und kostenaufweniger Vorgang. Um diese Vorgehensweise zu vereinfachen, wurden auf der Basis von vermessenen Probeblättern und den zugehörigen Stell– und Messwerten des Prozesses Beobachtermodelle (Soft–Sensoren) entwickelt, die eine On–Line–Schätzung der Weiße ohne Labormessungen durchführen und somit die Basis für eine On–Line–Weißeregelung bilden. Die Daten für diese Soft–Sensoren sind stark geclustert, weil zur Herstellung verschiedener Papiersorten Altpapier unterschiedlicher Stoffzusammensetzung verwendet wird. Aus diesem Grund wurde hier ein clusterbasierter Modellierungsansatz gewählt, der in den folgenden beiden Abschnitten beschrieben ist.

Abbildung 4 zeigt das Regelungskonzept für die Flotation. Wie in der Einleitung bereits erwähnt, besitzt die Flotation eine schwankende Totzeit von etwa 10 bis 30 Minuten. Der Takt der Pulper–Füllungen liegt ebenfalls in der Größenordnung von 30 Minuten. Eine reine Regelung auf die Akzept–Weiße würde daher nur einen geringen Anteil des Stoffes mit dem richtigen Schaumrinnen–Niveau flotieren. Die hier verfolgte Alternative ist eine modellbasierte Vorwärtssteuerung, die die Schaumrinne auf der Basis der Zulaufweiße und der internen Volumenströme steuert. Das Modell für diese Vorwärtssteuerung wurde auf der Basis von Daten erstellt, die während gezielter Versuchsläufe mit Variationen der Schaumrinne gewonnen wurden. Ebenso wie die Daten für die Soft–Sensoren sind auch die Daten für das Flotationsmodell stark geclustert, was sich auch hier auf die unterschiedlichen Papiersorten und Eintragsqualitäten zurückführen lässt. Daher wurde auch zur Erstellung des Flotationsmodells eine clusterbasierte Modellierung gewählt (siehe Abschnitt 4 und 5) [11]. Die in Abbildung 4 eingezeichnete Adaption des Modells ist in Abschnitt 6 beschrieben.

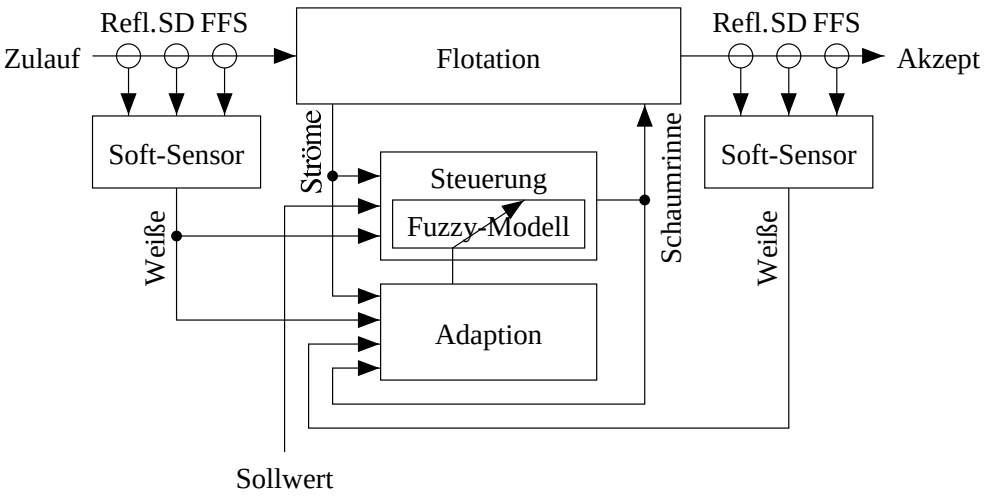


Abbildung 4: Weißeregelung mit Soft–Sensoren.

4 Clusteranalyse

Die zur Modellierung verwendeten Daten lassen sich allgemein schreiben als

$$X = \{x_1, \ldots, x_n\} \subset \mathbb{R}^p, \tag{1}$$

wobei einige der p Komponenten Eingangsgrößen und die übrigen Komponenten Ausgangsgrößen sind. Wir unterscheiden hier nicht nach Eingangs– oder Ausgangsgrößen, d.h. die Clusteranalyse erfolgt im Produktraum der Eingangs– und Ausgangsgrößen [10, 12]. Aus den für die Modellierung genutzten Daten werden zunächst die fehlerhaften Daten entfernt, also Ausreißer von Messungen oder Daten, die während Produktionsstörungen der Anlage aufgenommen wurden. Die verbleibenden Daten werden anschließend auf den Mittelwert Null und die Standardabweichung Eins normalisiert. Zur Bestimmung der Cluster mininieren wir die Kostenfunktion des Fuzzy c–Means Modells (FCM) [1]

$$J_{FCM}(U, V; X) = \sum_{i=1}^{c} \sum_{k=1}^{n} u_{ik}^m \, d_{ik}^2 \tag{2}$$

mit den Abständen d_{ik} zwischen den Punkten x_k und den Clusterzentren v_i, $i = 1, \ldots, c$, $k = 1, \ldots, n$, dem Unschärfeparameter $m > 1$ und den Nebenbedingungen

$$\sum_{k=1}^{n} u_{ik} > 0, \qquad i = 1, \ldots, c, \tag{3}$$

$$\sum_{i=1}^{c} u_{ik} = 1, \qquad k = 1, \ldots, n. \tag{4}$$

Bedingung (3) verhindert leere Cluster, und Bedingung (4) verlangt, dass die Summe der Zugehörigkeiten jedes Elements zu allen Clustern gleich eins ist. Sind beide Bedingungen (3) und (4) erfüllt, dann heißt U eine *Fuzzy Partitionsmatrix*. Zur Optimierung von J_{FCM} (2) mit der Nebenbedingung (4) definieren wir die Lagrange–Funktion

$$F_{FCM}(U, V, \lambda; X) = \sum_{k=1}^{n} \sum_{i=1}^{c} u_{ik}^m \, d_{ik}^2 - \sum_{k=1}^{n} \lambda_k \cdot \left(\sum_{i=1}^{c} u_{ik} - 1 \right). \tag{5}$$

Die notwendigen Bedingungen für lokale Extrema von F_{FCM} liefern (nach einigen Umformungen) die notwendigen Gleichungen zur Berechnung der Clusterzentren V und der Partitionsmatrix U

$$\left.\begin{array}{l} \dfrac{\partial F_{FCM}}{\partial \lambda_k} = 0 \\[2ex] \dfrac{\partial F_{FCM}}{\partial u_{ik}} = 0 \end{array}\right\} \Rightarrow \quad u_{ik} = 1 \Big/ \sum_{j=1}^{c} \left(\frac{d_{ik}}{d_{jk}} \right)^{\frac{2}{m-1}}, \tag{6}$$

$$\frac{\partial J_{FCM}}{\partial v_i} = 0 \quad \Rightarrow \quad v_i = \frac{\sum\limits_{k=1}^{n} u_{ik}^m \, x_k}{\sum\limits_{k=1}^{n} u_{ik}^m}. \tag{7}$$

Offensichtlich werden zur Berechnung der Zugehörigkeiten u_{ik} mit (6) die Clusterzentren v_i benötigt — und zur Berechnung der Clusterzentren v_i mit (7) umgekehrt die Zugehörigkeiten u_{ik}. Es werden daher zunächst die Clusterzentren v_i zufällig initialisiert. Dann

werden abwechselnd mit (6) die Zugehörigkeiten u_{ik} und mit (7) die Clusterzentren v_i berechnet, bis entweder eine maximale Anzahl von Schritten erreicht ist, oder die Änderung der Clusterzentren zwischen zwei Schritten eine gegebene Grenze unterschreitet. Dieses effiziente Verfahren heißt *alternierende Optimierung* [15].

Wir wollen hier mit Hilfe der Clusteranalyse Modelle mit lokal linearem Verhalten erstellen. Fuzzy c–Means liefert bei Verwendung des Euklid'schen Abstands jedoch nur punktförmige Cluster ohne lineare Vorzugsrichtung. Um lineare Modelle zu erzeugen, verwenden wir als Abstandsmaß den (lokalen) Mahalanobis–Abstand

$$d_{ik} = \sqrt{(x_k - v_i) A_i (x_k - v_i)}. \tag{8}$$

Die Abstandsmatrizen A_i können aus den Kovarianzmatrizen S_i der Cluster $i = 1, \ldots, c$ berechnet werden.

$$S_i = \sum_{k=1}^{n} u_{ik}^m \, (x_k - v_i)(x_k - v_i)^T. \tag{9}$$

$$A_i = \sqrt[p]{\rho_i \det(S_i)} \, S_i^{-1} \tag{10}$$

Der Vorfaktor ρ_i in (10) dient der Normalisierung, so dass jedes Cluster das Hypervolumen $\det(A_i) = \rho_i$ besitzt. Wir wählen hier die Hypervolumina $\rho_1 = \ldots = \rho_c = 1$. Fuzzy c–Means Clustermodelle mit solchen lokalen Mahalobis–Abständen heißen auch Gustafson–Kessel Modelle [6].

5 Takagi–Sugeno–Modellierung

Die mit Hilfe der im vorigen Abschnitt beschriebenen Clusteranalyse gewonnenen Clusterinformationen können zum Erstellen von regelbasierten Modellen [3] verwendet werden. Die mit Gustafson–Kessel Clustering gewonnenen Modelle sind lokal linear und unscharf, d.h. sie gehen kontinuierlich ineinander über. Diese Art von regelbasierten Modellen lässt sich als *Takagi–Sugeno* (TS) System schreiben [23]

$$\begin{array}{lllll} R_1: & \text{If} & \mu_1(x) & \text{then} & y = f_1(x). \\ R_2: & \text{If} & \mu_2(x) & \text{then} & y = f_2(x). \\ \vdots & & & & \\ R_c: & \text{If} & \mu_c(x) & \text{then} & y = f_c(x). \end{array} \tag{11}$$

Dabei spezifizieren die Funktionen $f_i : \mathbb{R}^q \to \mathbb{R}^{p-q}$ die lokalen Modelle für q–dimensionale Eingänge und $(p-q)$–dimensionale Ausgänge, und die Zugehörigkeitsfunktionen $\mu_i : \mathbb{R}^q \to [0,1]$ legen die Gültigkeitsbereiche der lokalen Modelle fest. Für einen gegebenen Eingangsvektor $x \in \mathbb{R}^q$ erfolgt die Auswertung eines Takagi–Sugeno–Systems durch konvexe Kombination der Einzelmodelle.

$$y(x) = \frac{\sum_{i=1}^{c} \mu_i(x) \cdot f_i(x)}{\sum_{i=1}^{c} \mu_i(x)}. \tag{12}$$

Im Trivialfall, dass nur eine einzige Regel R_j aktiv ist, gilt $y(x) = f_j(x)$. Ansonsten ist $y(x)$ eine Kombination aus mehreren oder allen einzelnen lokalen Modellen.

Im vorigen Abschnitt wurde beschrieben, wie sich mit Gustafson–Kessel–Clustering aus einem (Eingangs–Ausgangs–)Datensatz eine Partitionsmatrix U, eine Menge von Clusterzentren V sowie eine Menge von Abstandsmatrizen $\{A_1, \ldots, A_c\}$ berechnen lässt. Die für ein TS System benötigten Zugehörigkeitsfunktionen $\mu_i(x)$, $i = 1, \ldots, c$, lassen sich unter Verwendung der Clusterzentren V und Abstandsmatrizen $A_1, \ldots, A_c$ in Analogie zu (6) definieren.

$$\mu_i(x) = 1 \Bigg/ \sum_{j=1}^{c} \left(\frac{(x - v_i^{(1,\ldots,p)})^T \; A_i^{(1,\ldots,p)\times(1,\ldots,p)} \; (x - v_i^{(1,\ldots,p)})}{(x - v_j^{(1,\ldots,p)})^T \; A_i^{(1,\ldots,p)\times(1,\ldots,p)} \; (x - v_j^{(1,\ldots,p)})} \right)^{\frac{1}{m-1}} . \qquad (13)$$

Hierbei werden die Clusterzentren auf den Eingangsraum projiziert, als nur die ersten p Komponenten $v_i^{(1,\ldots,p)}$ verwendet. Ebenso werden nur die ersten p Zeilen und p Spalten der Abstandsmatrizen A_i verwendet, also $A_i^{(1,\ldots,p)\times(1,\ldots,p)}$. Gleichung (13) ist eine Erweiterung von (6) mit (8) auf kontinuierliche Werte $x \in \mathbb{R}$. Insbesondere gilt für $q = 0$: $\mu_i(x_k) = u_{ik}$ für alle $i = 1, \ldots, c$ und $k = 1, \ldots, n$.

Wir betrachten hier TS Systeme mit lokal linearen Modellen, d.h. jede lokale Funktion lässt sich schreiben als $f_i(x) = B_i \cdot x + c_i$, $B_i \in \mathbb{R}^{p\times p}$, $c_i \in \mathbb{R}^p$. Zur Bestimmung der Modellmatrizen B_i und der Offsets c_i lassen sich die Clusterzentren V sowie die $p - 1$ größten Eigenvektoren der lokalen Kovarianzmatrizen S_i verwenden. Im einfachsten Fall sind Ein– und Ausgänge eindimensional, also $p = 2$ und $q = 1$ und wir erhalten

$$f_i(x) = v_i^{(2)} + \frac{e_i^{(2)}}{e_i^{(1)}}(x - v_i^{(1)}), \quad i = 1, \ldots, c, \qquad (14)$$

wobei e_i der größte Eigenvektor von S_i ist. Im allgemeinen Fall $1 \leq q < p$ gilt

$$f_i(x) = v_i^{(q+1,\ldots,p)} + (E_i^T E_i)^{-1} F_i \, (x - v_i^{(1,\ldots,q)}), \quad i = 1, \ldots, c, \qquad (15)$$

wobei E_i die Matrix der ersten q Komponenten der $p - 1$ größten Eigenvektoren von S_i ist und F_i die Matrix der restlichen $p - q$ Komponenten. Detailliertere Beschreibungen der Modellierung mit Clustermethoden sind in [7, 13, 14] zu finden.

Wie in Abschnitt 3 angedeutet wurden die oben beschriebenen Methoden der Modellierung mit Gustafson–Kessel–Clustering und Takagi–Sugeno–Systemen im Rahmen der Weißereglung benutzt, um aus Messdaten Modelle für die Soft–Sensoren und Flotationsmodelle zu generieren. Da die hier betrachtete Anlage bei der Firma Lang Papier in Ettringen eine zweistufige Flotation besitzt, wurden insgesamt sechs verschiedene Modelle erstellt: vier Soft–Sensoren am Zulauf Vorflotation, Akzept Vorflotation, Zulauf Nachflotation und Akzept Nachflotation, sowie zwei Flotationsmodelle für die Vorflotation und die Nachflotation. Die Erstellung und Auswertung der Modelle für die Soft–Sensoren und für Vor–/Nachflotation sind in den folgenden beiden Teilabschnitten beschrieben.

5.1 Soft–Sensor–Modelle

Für die Modelle der Soft–Sensoren wurden in einem Zeitraum von etwa einer Woche aus dem laufenden Betrieb Stoffproben entnommen und vermessen. Den Messergebnissen wurden anschließend die entsprechenden On–Line–Werte der Messungen in der Stoffsuspension zugeordnet und daraus ein Datensatz X für die oben beschriebene Modellierung

erstellt. Nach der Erstellung der Modelle wurden die Soft–Sensoren validiert. Hierzu wurden im laufenden Betrieb in einem Zeitraum von etwa zwei Wochen erneut Stoffproben entnommen und deren Weiße mit den berechneten On–Line–Werten der Soft–Sensoren verglichen. Abbildung 5 zeigt die am Zulauf von Vorflotation (links) und Nachflotation (rechts) gemessenen Werte (schwarz) und die entsprechenden von den Soft–Sensoren berechneten Werte (grau). Die Modellwerte am Zulauf der Nachflotation (rechts) stimmen sehr gut mit den Laborwerten überein. Am Zulauf der Vorflotation ist die Modellgüte etwas schlechter, was auf die stärker schwankenden Eigenschaften der Fasersuspension und einer inhomogenen Verteilung von Druckfarbenpartikeln im Probeblatt zurückzuführen ist.

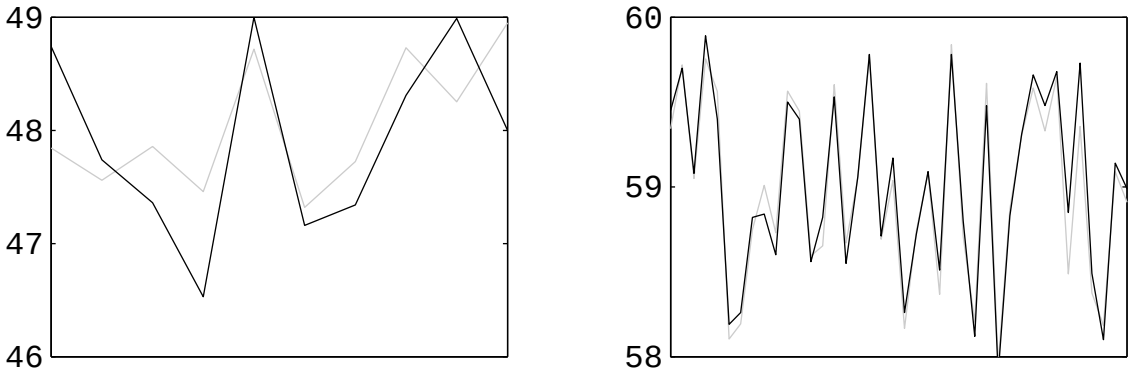


Abbildung 5: Gemessene (schwarz) und Soft–Sensor–Werte (grau) der Weiße am Zulauf von Vorflotation (links) und Nachflotation (rechts). Die Einzelmessungen wurden in einem Zeitraum von etwa 2 Wochen aufgenommen.

5.2 Flotationsmodelle

Für die Modellierung der Vorflotation und der Nachflotation müssen die durch die Fließverzögerungen der Fasern bedingten Totzeiten berücksichtigt werden. Die Flotation lässt sich als statischer totzeitbehafteter Prozess modellieren.

$$y_k = f(w_{k-T_w}, u_{k-T_u}) \tag{16}$$

Die Regelgröße y ist die Weiße im Akzept und die Stellgröße u ist das Schaumrinnen–Niveau. Zu den externen Einflussgrößen w zählen beispielsweise die Volumenströme und die Zulauf–Weiße. Die Totzeiten T_u und T_w wurden experimentell ermittelt. Dabei zeigte sich, dass die Totzeiten T_w je nach externer Einflussgröße unterschiedlich sind, d.h. T_w ist tatsächlich ein Vektor von Totzeiten für die einzelnen Komponenten von w. Wir benutzen hier jedoch die vereinfachte Schreibweise wie in (16). Für das Training der Prozessmodelle werden die totzeitbereinigten Eingangswerte w und u mit den Ausgangswerten y zu einem Datensatz X zusammengefasst und für die oben beschriebene Modellierung benutzt. Zur Validierung der Modelle werden anschließend die Ausgangswerte der Modelle mit den tatsächlichen Prozessmesswerten verglichen. Abbildung 6 zeigt die nach dem Training im laufenden Betrieb aufgenommen Soft–Sensor–Messungen (schwarz) und die entsprechenden Werte der Flotationsmodelle (grau) für Vorflotation (links) und Nachflotation (rechts). Beide Kurven zeigen Werte im Minutentakt und überstreichen einen Zeitraum von einer Woche. Die relativen Bewegungen der Weiße werden von beiden Modellen gut erfasst, allerdings bildet sich ein bleibender Offset zwischen Soft–Sensor– und

Modellweißen aus. Dieser Offset ist in diesem Beispiel besonders stark in der Vorflotation ausgeprägt (Abbildung 6 links), er tritt jedoch auch in der Nachflotation auf. Verursacht wird dieser Offset offenbar durch Änderungen von nicht erfassten Prozessparametern und anderen Eingriffen in den Prozess, wie z.B. Reinigungen von Rohren oder Sensoren. Zur Vermeidung dieses Offsets wurde das Regelungskonzept wie in Abbildung 4 gezeigt um eine Adaptionskomponente erweitert. Die Funktionsweise dieser Adaptionskomponente ist im folgenden Abschnitt beschrieben.

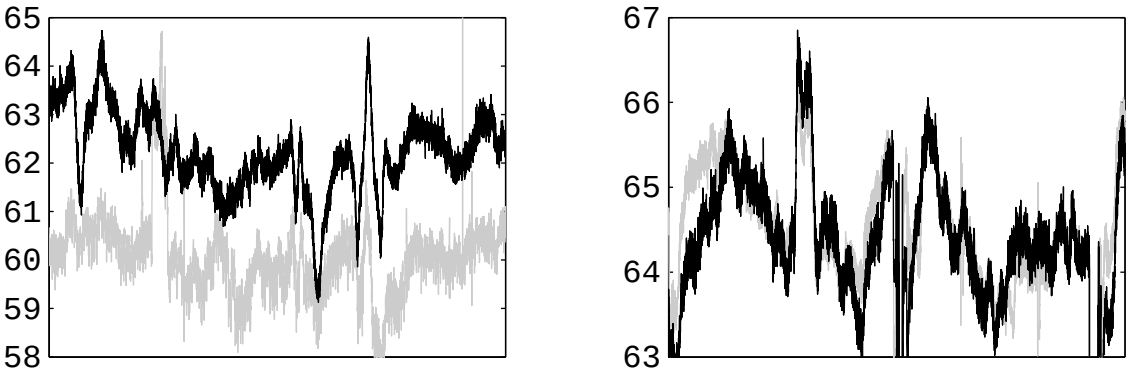


Abbildung 6: Soft–Sensor– (schwarz) und Modell–Werte (grau) der Weiße im Akzept von Vorflotation (links) und Nachflotation (rechts) über einen Zeitraum von einer Woche.

6 Modelladaption und –inversion

Zur Kompensation der in Abbildung 6 erkennbaren Offsets werden Offset–Terme δ zu den Ausgabewerten der Takagi–Sugeno–Modelle addiert.

$$\hat{y}_k = \hat{f}(w_{k-T_w}, u_{k-T_u}) = TS(w_{k-T_w}, u_{k-T_u}) + \delta_k \tag{17}$$

Die beiden Modelle TS bzw. $\hat{f}$ für Vorflotation und Nachflotation sind unterschiedlich und verwenden unterschiedliche Parameter und Variablen y, u, w und δ. Zur besseren Lesbarkeit werden hier jedoch nicht nach Vor– und Nachflotation unterschieden und nur einer der beiden strukturell identischen Fälle beschrieben. Eine Adaption des Offsets erfolgt durch Vergleich der Modellweiße $\hat{y}$ mit der gemessenen Weiße y. Dieser Modellfehler wird in jedem Schritt mit c_I multipliziert und zum aktuellen Offset addiert (zeitdiskrete Integration).

$$\delta_{k+1} = \delta_k + c_I \cdot (y_k - \hat{y}_k) \tag{18}$$

Für die Integrationskonstante wurde experimentell ein für diese Anwendung geeigneter Wert von $c_I = 0.5$ ermittelt. Abbildung 7 zeigt die Verläufe von Modell– und Messweiße wie in Abbildung 6, jedoch hier mit Modelladaption. Die Modellfehler sind offensichtlich sehr gering.

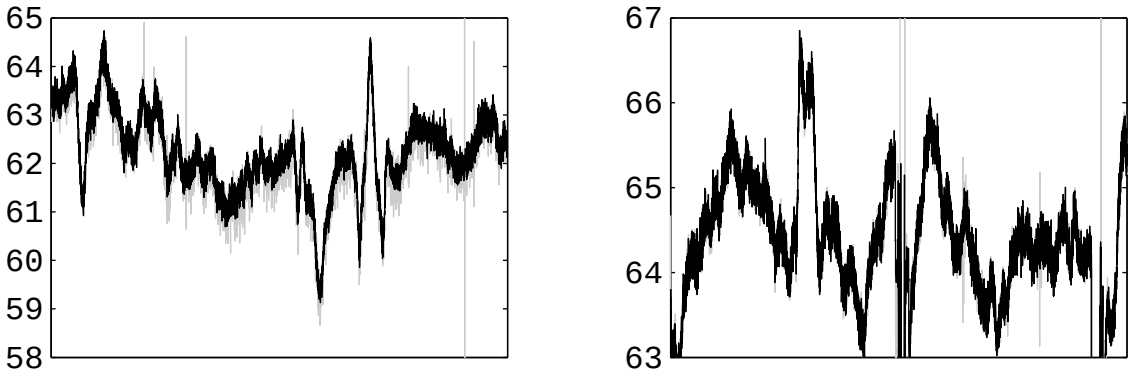


Abbildung 7: Kurven wie in Abbildung 6, jedoch mit Modelladaption.

Die Berechnung der Stellgröße u erfolgt formell durch Inversion des Modells $\hat{f}$. Hierbei werden die Totzeiten zwischen w und u berücksichtigt.

$$u_k = \hat{f}^{-1}(y_k, w_{k-T_w+T_u}) \tag{19}$$

Takagi–Sugeno–Modelle lassen sich nicht explizit invertieren. Die Modellinversion erfolgt daher in jedem Schritt numerisch durch gradientenbasierte Minimierung des Fehlerfunktionals

$$J(u_k; y_k, y_{\text{soll}}, w_{k-T_w+T_u}) = \left(\hat{f}(u_k, w_{k-T_w+T_u}) - y_{\text{soll}}\right)^2. \tag{20}$$

Das hier beschriebene Regelungskonzept lässt sich charakterisieren als eine modellbasierte Vorwärtssteuerung mit einer additiven Modellkompensation. Aufgrund des integrativen Charakters der Modellkompensation lässt sich dieses Konzept auch als modellbasierte PI–Regelung interpretieren.

7 Nichtlineare Stellgrößenbegrenzung

Die Stellbereiche der Schaumrinnenregler sind aus technologischen Gründen begrenzt. Sowohl ein zu geringes als auch ein zu hohes Schaumrinnen–Niveau verursacht unerwünschte Störungen in den folgenden Prozess–Stufen. Für das mit der Weißeregelung vorgegebene Schaumrinnen–Niveau u wurden daher in Absprache mit der Produktionsleitung die Grenzwerte $u_{\min}$ und $u_{\max}$ festgelegt. In einem ersten Versuch wurde eine harte Begrenzung gemäß

$$u^* = \begin{cases} u_{\min} & \text{falls } u < u_{\min} \\ u_{\max} & \text{falls } u > u_{\max} \\ u & \text{sonst} \end{cases} \tag{21}$$

verwendet, die jedoch in Perioden mit besonders guter oder besonders schlechter Eintragsqualität zu einem Verharren am oberen und unteren Grenzwert führte. In solchen Fällen blieb das Schaumrinnen–Niveau über längere Zeiträume konstant, und dadurch fand keinerlei Ausgleich von Weißeschwankungen mehr statt. Darüber hinaus interpretierten die Anlagenfahrer dieses Verharren in einem Fall als ein Fehlverhalten und schalteten die

automatische Regelung wieder aus. Aus diesen Gründen wurde die harte Stellgrößenbegrenzung (21) durch folgende Sigmoid–Funktion ersetzt:

$$u^* = \frac{1}{2} \cdot \left(u_{\max} + u_{\min} + (u_{\max} - u_{\min}) \cdot \tanh \left(\frac{2u - u_{\max} - u_{\min}}{u_{\max} - u_{\min}} \right) \right) \qquad (22)$$

Der Verlauf dieser weichen Dämpfungsfunktion ist in Abbildung 8 dargestellt. Die Begrenzungen $u_{\min}$ und $u_{\max}$ des Schaumrinnen–Niveaus sind gestrichelt eingezeichnet. Offensichtlich werden die Grenzen in allen Fällen eingehalten, jedoch wird die Reglerverstärkung hier im Gegensatz zur harten Begrenzung niemals zu null. Auch in extremen Eintragssituationen erfolgt also stets eine (wenn auch begrenzte) Reaktion des Reglers auf Qualitätsschwankungen.

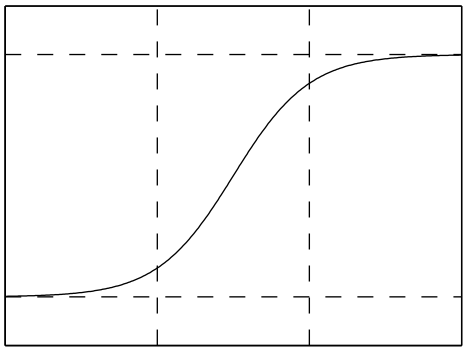


Abbildung 8: Sigmoid–Funktion zur Stellgrößenbegrenzung (gestrichelte Linien: $u_{\min}$, $u_{\max}$).

8 Ergebnisse

Alle beschriebenen Module zur modellbasierten Weißeregelung wurden als Produkt SiFlot in das Siemens Automatisierungs- und Visualisierungstool WinCC integriert. Ein erster Einsatz der Soft–Sensoren erfolgte bei der Optimierung der Druckfarbenentfernung bei Kübler + Niethammer Papierfabrik Kriebstein AG [17, 18, 19]. Die weiterentwickelten Soft–Sensoren sind seit Juni 2001 bei Lang Papier in Ettringen im Dauereinsatz. Die modellbasierte Weißeregelung wurde im Mai 2003 bei Lang Papier in Betrieb genommen.

Zur Validierung der Regelungsgüte wurden zwei Ansätze vefolgt: Im ersten Validierungs–Ansatz wurden die Werte der Standardabweichung der Akzeptqualität mit und ohne Weißeregelung über längere Zeiträume verglichen. Je nach betrachtetem Zeitraum wurde die Standardabweichung mit Hilfe der Weißeregelung um 30 bis 70% reduziert. Der Nachteil dieser Analyse ist, dass den verglichenen Standardabweichungen unterschiedliche Betriebssituationen zu Grunde liegen. So könnten beispielsweise im ungeregelten Fall die Eintragsschwankungen größer gewesen sein als im geregelten Fall und dadurch eine größere Standardabweichung in der Akzeptqualität verursacht haben. Im zweiten Validierungs–Ansatz wurde daher versucht, den ungeregelten und den geregelten Fall unter gleichen Voraussetzungen zu vergleichen. Hierzu wurden die Verläufe der Akzeptqualität in Phasen den geregelten Betriebs betrachtet. Die entsprechenden Verläufe der Akzeptqualität im ungeregelten Betrieb wurden dann mit Hilfe des Flotationsmodells berechnet, in dem das Schaumrinnen–Niveau auf einen konstanten Wert gesetzt wurde.

Abbildung 9 zeigt die Ergebnisse dieser Analyse mit Daten über einen Zeitraum von einem Tag, links für die Vorflotation und rechts für die Nachflotation. Die tatsächlichen Verläufe der Akzeptweiße im geregelten Fall sind in schwarz gezeichnet, die mit konstanter Schaumrinne simulierten Verläufe in grau. Die Sollwertvorgaben für die Akzeptweiße sind gestrichelt gezeichnet. In keinem der beiden Fälle wird dieser Sollwert präzise erreicht und eingehalten. Der Grund dafür ist, dass die Akzeptweiße im wesentlichen von der Eintragsweiße bestimmt wird und durch Variationen der Prozessparameter eine nur vergleichsweise geringe Beeinflussung der Akzeptweiße möglich ist. Der Prozess ist also stark gestört und nur schwach steuerbar. Dennoch zeigt das linke Bild eine deutliche Reduktion der Weißeschwankungen in der Vorflotation. Die Standardabweichung der Weiße reduziert sich hier von 1.12 auf 0.34, also um etwa 70%. In der Nachflotation (rechtes Bild) reduziert sich die Standardabweichung der Weiße von 9.73 auf 8.87, also nur um etwa 8.8%. Das Regelungspotential der Vorflotation ist also höher als das der Nachflotation.

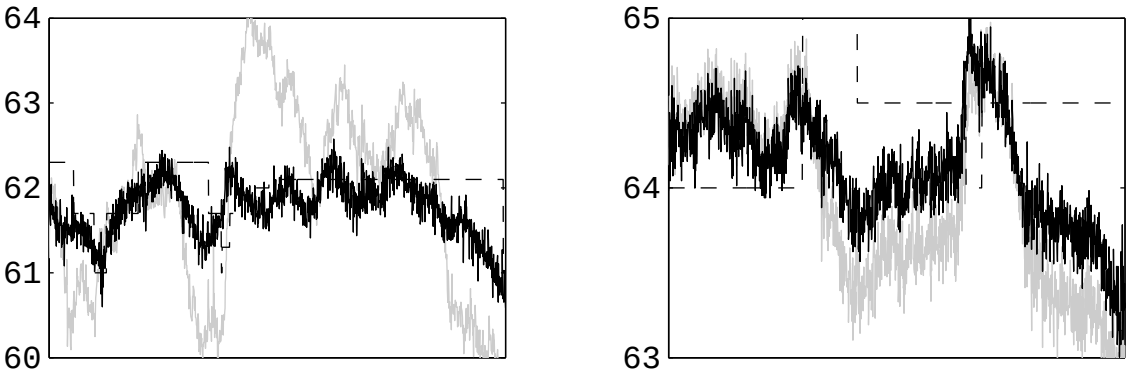


Abbildung 9: Verläufe der Akzeptweiße von Vorflotation (links) und Nachflotation (rechts) mit Weißeregelung (schwarz) und ohne Weißeregelung (grau, simuliert) über einen Zeitraum von einem Tag. Die Sollwertvorgaben sind gestrichelt einzeichnet.

Die Weißeregelung führt zu einer gleichmäßigeren Produktqualität. Dies ist aus produktionstechnologischer Sicht ein Vorteil. Darüber hinaus ermöglicht die Weißeregelung jedoch auch eine Kosteneinsparung, in dem bei guter Eintragsqualität das Schaumrinnen–Niveau abgesenkt und dadurch die Rejektmenge reduziert wird. Die untere Grenze der produzierten Qualität, also die Garantie–Qualität, kann dabei unverändert bleiben. Durch die Vermeidung unnötiger Rejektmengen verringern sich die Faserverluste und die Abfallverwertungskosten. Zur Zeit wird versucht, die Höhe der durch die Weißeregelung erzielten Einsparungen abzuschätzen.

Die beschriebenen Methoden zur Qualitätsoptimierung lassen sich auch auf weitere Stufen der Papierherstellung übertragen. Diese Arbeiten sind Gegenstand laufender Projekte. Mit einem ähnlichen Ansatz wurde bereits die Fasermahlung (Refining) in der Spanplattenproduktion [16] sowie zahlreiche Prozesse in der Stahlproduktion [8, 21, 22] optimiert.

Literatur

[1] BEZDEK, J. C.: *Pattern Recognition with Fuzzy Objective Function Algorithms*. Plenum Press, New York, 1981.

[2] CATSBURG, R. W.: *Flotation de–inking research of the future: Parallel process simulation.* Paper Technology, 40(9):35–39, 1999.

[3] DRIANKOV, D., H. HELLENDOORN und M. REINFRANK: *An Introduction to Fuzzy Control*. Springer, Berlin, 1995.

[4] FELSCH, W., W. KILPPER und W. WERNISCH: *Papiermacher–Taschenbuch*. Curt Haefner Verlag, Heidelberg, 5. Auflage, 1989.

[5] GROSSMANN, H., E. HANECKER und G. SCHULZE: *Flotation von Altpapierstoff im Zentrifugalfeld*. Wochenblatt für Papierfabrikation, 3:94–100, 1997.

[6] GUSTAFSON, E. E. und W. C. KESSEL: *Fuzzy Clustering with a Covariance Matrix*. In: *IEEE International Conference on Decision and Control, San Diego*, Seiten 761–766, 1979.

[7] HÖPPNER, F., F. KLAWONN, R. KRUSE und T. A. RUNKLER: *Fuzzy Cluster Analysis — Methods for Image Recognition, Classification, and Data Analysis*. Wiley, 1999.

[8] LANG, B., R. DÖLL, M. JANSEN, T. POPPE, T. A. RUNKLER und K. WEINZIERL: *Application of Artificial Intelligence in Steel Processing.* In: *Fachtagung Automatisierung in der Metallurgie (Metallurgisches Seminar des GDMB–Fachausschusses für Metallurgische Aus– und Weiterbildung)*, Band 89, Grevenbroich, März 2001.

[9] LEIVISKÄ, K.: *Process Control*, Band 14 der Reihe *Papermaking Science and Technology*. Fapet Oy, Helsinki, 1998.

[10] PAL, K., J. C. BEZDEK, N. R. PAL und T. A. RUNKLER: *Some Notes on Fuzzy Rule Extraction by Clustering*. In: *SIARP Ibero–American Symposium on Pattern Recognition*, Seiten 19–27, Lisbon, September 2000.

[11] PAL, K., N. R. PAL, T. A. RUNKLER und J. C. BEZDEK: *Fuzzy Rule Extraction by Clustering: The Role of Tendency Assessment.* In: *CoIl Symposium on Computational Intelligence and Learning*, Seiten 3–16, Chios, Greece, Juni 2000.

[12] PAL, N. R., K. PAL, J. C. BEZDEK und T. A. RUNKLER: *Some Issues in System Identification using Clustering*. In: *IEEE International Conference on Neural Networks*, Seiten 2524–2529, Houston, Juni 1997.

[13] RUNKLER, T. A.: *Automatic Generation of First Order Takagi–Sugeno Systems using Fuzzy c–Elliptotypes Clustering*. Journal of Intelligent and Fuzzy Systems, 6(4):435–445, 1998.

[14] RUNKLER, T. A.: *Nonlinear System Identification with Global and Local Soft Computing Methods*. In: *GMA-Workshop Fuzzy Control*, Seiten 163–176, Dortmund, Oktober 2000.

[15] RUNKLER, T. A., J. C. BEZDEK und L. O. HALL: *Clustering Very Large Data Sets: The Complexity of the Fuzzy c–Means Algorithm*. In: *European Symposium on Intelligent Technologies, Hybrid Systems and their implementation on Smart Adaptive Systems (eunite), Albufeira, Portugal*, Seiten 420–425, September 2002.

[16] RUNKLER, T. A., E. GERSTORFER, M. SCHLANG, E. JÜNNEMANN und J. HOLLATZ: *Modeling and Optimization of a Refining Process for Fiber Board Production*. IFAC Control Engineering Practice, (in press), 2003.

[17] RUNKLER, T. A., E. GERSTORFER, M. SCHLANG, E. JÜNNEMANN und K. VILLFORTH: *Data Compression and Soft Sensors in the Pulp and Paper Industry*. In: *European Control Conference '99, Karlsruhe*, Band CM–2, Seiten 1–5, August 1999.

[18] RUNKLER, T. A., E. GERSTORFER, M. SCHLANG, E. JÜNNEMANN und K. VILLFORTH: *Fuzzy Clustering for Data Compression, Modeling, and Optimization in Recovered Paper Industry*. European Journal of Control, 7(1):67–74, 2001.

[19] RUNKLER, T. A., J. HOLLATZ, H. FURUMOTO, E. JÜNNEMANN und K. VILLFORTH: *Compression of Industrial Process Data Using Fast Alternating Cluster Estimation (FACE)*. In: *GI Jahrestagung Informatik (Workshop Data Mining)*, Seiten 71–80, Magdeburg, September 1998.

[20] RUNKLER, T. A., K. VILLFORTH und E. JÜNNEMANN: *Prozessmodellierung und –optimierung mit Fuzzy–Clustering am Beispiel der Altpapieraufbereitung*. Künstliche Intelligenz, 3:34–37, 2001.

[21] SCHLANG, M., B. FELDKELLER, B. LANG, T. POPPE und T. RUNKLER: *Neural Computation in Steel Industry*. In: *European Control Conference '99, Karlsruhe*, Band BP–1, Seiten 1–6, August 1999.

[22] SCHLANG, M., M. JANSEN, B. LANG, T. POPPE, T. A. RUNKLER und K. WEINZIERL: *Current and Future Development in Neural Computation in Steel Processing*. IFAC Control Engineering Practice, 9(9):975–986, 2001.

[23] TAKAGI, T. und M. SUGENO: *Fuzzy Identification of Systems and Its Application to Modeling and Control*. IEEE Transactions on Systems, Man, and Cybernetics, 15(1):116–132, 1985.

Optimierte Klassifikation für Mehrklassenprobleme am Beispiel der Bewegungssteuerung von Handprothesen

Markus Reischl, Lutz Gröll, Ralf Mikut

Forschungszentrum Karlsruhe GmbH, Institut für Angewandte Informatik,
D-76021 Karlsruhe, Postfach 3640,
Telefon: (07247) 82-5749, Fax: (07247) 82-5786, E-Mail: Markus.Reischl@iai.fzk.de

1 Motivation

Als Schnittstelle zwischen Mensch und Prothese haben sich myoelektrische (EMG-, Muskel-) Signale etabliert [1]. Entsprechende Sensoren sind einfach handhabbar und unbedenklich in der Anwendung. Der Prothesenträger ist in der Lage, durch reproduzierbare Muskelkontraktionen seine Bewegungsabsicht (Griffart und -stärke) mitzuteilen und die Prothese zu steuern [2].

Das Erkennen und Auswerten einer Bewegungsabsicht findet durch eine Signalvorverarbeitung der EMG-Signale, eine anschließende Merkmalsextraktion und eine Klassifikation statt. Obwohl insbesondere das Auswerten der Bewegungsabsicht aufgrund verrauschter Signale, Messwertdrift und patientenindividueller Unterschiede kompliziert ist, liefern bisher angewandte statistische Verfahren wie Diskriminanzanalyse und Maximum-Likelihood-Klassifikatoren gute Ergebnisse [3, 4, 5]. Mitunter können jedoch nur einige Griffarten sicher erkannt werden, während bei Griffarten, bei denen sich die EMG-Signale ähneln, häufiger Fehlauswertungen auftreten. Eine Ursache dafür liegt darin, dass herkömmliche Entwurfsverfahren für Mehrklassenprobleme tendenziell zu Merkmalskombinationen neigen, mit denen sich eine Klasse von den anderen besonders gut trennen lässt, die aber für das Trennen der anderen Klassen gemeinhin weniger geeignet sind.

Um dieses Manko zu beseitigen, wurden zwei neue Ansätze erarbeitet und untersucht. Der eine Ansatz modifiziert den Entwurf von Transformationsmatrizen für die Diskriminanzanalyse, indem er sich auf eine Minimierung eines Kriteriums für die Klassifikationsgüte stützt. Der andere Ansatz ist ein hierarchischer Klassifikatorentwurf, der ein sukzessives Abtrennen einzelner Klassen verfolgt. Beide Ansätze eignen sich nicht nur zum Auswerten der Bewegungsabsicht, sondern sind ganz allgemein für Mehrklassenprobleme einsetzbar. In dieser Arbeit werden sie jedoch überwiegend an Patientendaten validiert.

Zur Gliederung: Abschnitt 2 beschreibt eine Möglichkeit, wie der Patient durch Muskelkontraktionen seiner Handprothese eine Bewegungsabsicht mitteilen kann. Abschnitt 3 zeigt, wie die aus der Kontraktion resultierenden Signale verarbeitet werden, während sich Abschnitt 4 damit befasst, wie sich die aus der Signalverarbeitung in Form von Merkmalen gewonnene Information durch eine Klassifikation einer Bewegungsabsicht zuordnen lässt. Die Verfahren werden in Abschnitt 5 modifiziert, um deren Güte zu steigern. Abschließend findet in Abschnitt 6 ein Vergleich mit Fuzzy-Verfahren [6] statt.

2 Mensch-Prothesen-Schnittstelle

2.1 Steuerungsschema für myoelektrische Handprothesen

Im Falle steuerbarer Handprothesen werden mittels EMG-Sensoren Muskelkontraktionen im Armstumpf des Patienten detektiert. Das entstehende Signal wird einer Auswerteeinheit zugeführt, die eine erkannte und bewertete Bewegungsabsicht einer nachgeschalteten Ansteuerelektronik mitteilt.

Arbeiten auf diesem Gebiet befassen sich hauptsächlich mit dem Erkennen von Bewegungen in Laborumgebungen. Dabei werden gesunde Probanden untersucht und Leitmuskeln für entsprechende Handbewegungen abgetastet [4, 7]. Bei Amputierten fehlen in der Regel einige Leitmuskeln, andere sind nur noch teilweise vorhanden. Aufgrund der damit verbundenen eingeschränkten Kontraktionsmöglichkeiten wird in dieser Arbeit abweichend vom menschlichen Griffverhalten eine Unterteilung in Vorbewegungs- (Schaltsignale) und Bewegungszustand (Bewegungssignale) unternommen.

Nach Bild 1 lernt der Anwender k; (hier:$k \leq 8$) beliebige vorgegebene oder gewählte Signalverläufe (EMG-Spannung über Zeit) zu reproduzieren, die dann Griffarten zugewiesen werden können. Diese Signalverläufe dienen als *Schaltsignale*. Nach abgeschlossenem Schaltsignal nimmt die Handprothese einen für die folgende Bewegung optimalen Zustand (sog. Preshape) ein. An der Stellung der Finger in diesem Preshape-Zustand erkennt der Anwender, dass ein Schaltsignal erkannt und richtig interpretiert wurde.

Anschließend generiert der Anwender ein *Bewegungssignal*, das die Finger für die gewählte Griffart schließt bzw. öffnet. Die Bewegungsrichtung und -geschwindigkeit hängt dabei von der Art des kontrahierten Muskels sowie der Stärke der Kontraktion ab. Eine gleichzeitige Kontraktion beider Muskeln, d. h. eine sog. Kokontraktion, führt in den neutralen Zustand [9] zurück. Die Prothese öffnet sich wieder und wartet auf ein weiteres Schaltsignal.

Durch diesen Ansatz kann jegliches Schaltsignal implementiert werden, sofern

- Schaltsignale vom Patienten gut reproduziert werden können und
- die betrachteten Eigenheiten des Signals zu seiner Beschreibung ausreichen.

Zwar wirkt sich die Verzögerung zwischen Aktivierung der Muskeln und Bewegung der Prothese nachteilig aus, doch kann der Patient gerade für die häufig auftretenden Bewegungen die ihm angenehmsten Bewegungssignale (kurz, einfach, mit wenig Anstrengung) zum Anlernen und damit für den späteren Gebrauch der Prothese wählen.

Neben der Möglichkeit, Schalt- und Bewegungssignal zeitversetzt und separat vorzugeben (Schalten mit Zeitversatz, vgl. Abschnitt 3.1), kann auch ein einzelner, zusammenhängender Signalverlauf die vollständige Schalt- und Bewegungsinformation enthalten (Schalten ohne Zeitversatz, vgl. Abschnitt 3.2). In diesem Fall wird der Beginn der Muskelkontraktion als Schaltsignal gewertet und die nachfolgende myoelektrische Amplitude als Bewegungssignal interpretiert. Vorteil des Schaltens ohne Zeitversatz ist die vom Anwender als angenehmer empfundene Ansteuerung, nachteilig ist der höhere Klassifikationsfehler.

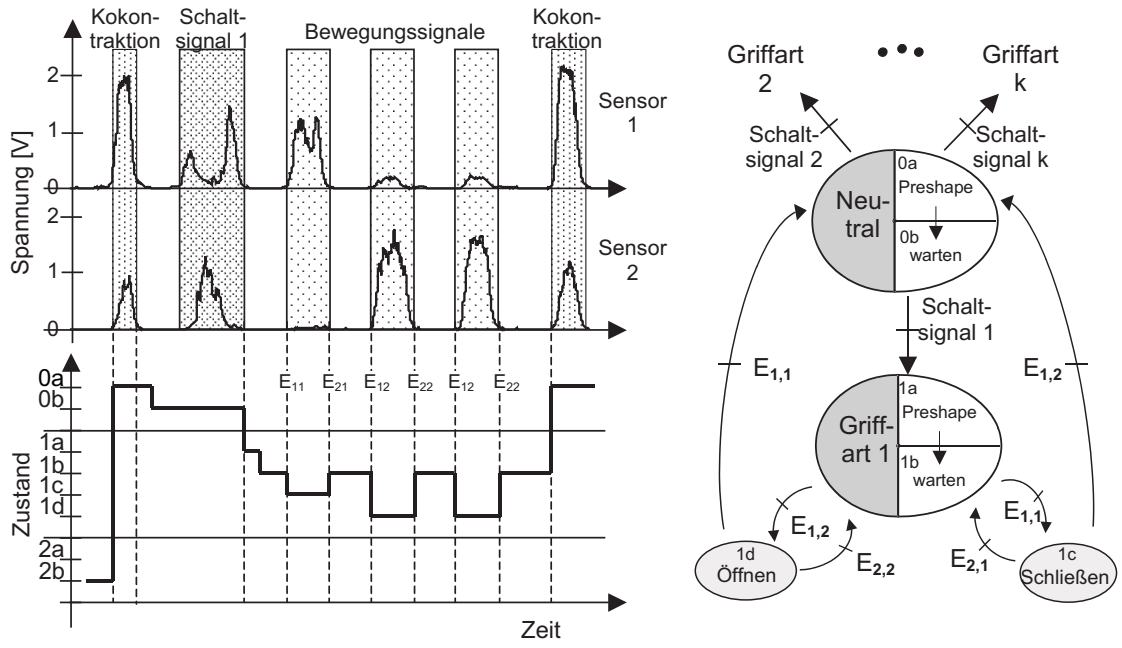


Bild 1: links: Myoelektrisches Signal zur Umsetzung von Bewegungsmustern unter Verwendung von zwei Sensoren (Schalten mit Zeitversatz), rechts: Automatengraph einer künstlichen Ansteuerung zur Erkennung von k Griffarten unter Verwendung von zwei EMG-Sensoren. E_{11}: Kontraktion von Muskel 1, E_{12}: Kontraktion von Muskel 2, E_{21}: Relaxation von Muskel 1, E_{22}: Relaxation von Muskel 2, $E_{11} \wedge E_{12}$: Kokontraktion, siehe [10].

2.2 Vorgehen zur Klassifikation von Schaltsignalen

Während Bewegungssignale durch einfache Schwellwerte zu detektieren sind, differieren Schaltsignale in Abhängigkeit von Anatomie und Bewegungsgewohnheiten sowohl zwischen Patienten als auch in den Realisierungen selbst (Bild 2). Entsprechend sind Auswertestrategien in Abhängigkeit der Signal-/Systemeigenschaften (Rauschen, Hintergrundpotenziale) zu entwerfen. Notwendige Parameter müssen patientenindividuell und automatisch angepasst werden.

Die Charakterisierung eines Schaltsignals geschieht durch die Einführung von Ersatzgrößen, eine so genannte Merkmalsextraktion (Abschnitt 3), die die zeitdiskreten Sensordaten (Spannung $U(t)$ in Volt) in einen Merkmalsraum transformieren

$$S_1 : \; U(t) \mapsto (x_1, ..., x_s)^T, \quad \mathbf{x} \in \mathbb{R}^s. \tag{1}$$

Ziel der Klassifikation ist es, einem Merkmalssatz eine der k Griffarten zuzuordnen

$$S_2 : (x_1, ..., x_s)^T \mapsto \{1, ..., k\}, \quad k \in \mathbb{N}^+, \quad \mathbf{x} \in \mathbb{R}^s. \tag{2}$$

Gesucht sind also Abbildungen S_1 und S_2, wobei S_1 auch als *Merkmalsextraktor* und S_2 als *Klassifikator* bezeichnet wird. Es sei jedoch bemerkt, dass sich die Bezeichnung Klassifikator hier auf Realisierungen und nicht wie in der Statistik üblich auf Zufallsvariablen bezieht.

Im folgenden Abschnitt wird der Entwurf des Merkmalsextraktors erklärt. Das ist möglich, da die Abbildung S_1 in eine Vielzahl von elementaren Teilabbildungen zerfällt. Im Abschnitt 4 wird detailliert auf den Entwurf der Abbildung S_2 eingegangen.

3 Merkmalsextraktion

3.1 Schalten mit Zeitversatz

Das Schaltsignal besteht aus einer abgeschlossenen Kontraktionssequenz. Zur Klassifikation wird ein repräsentativer Merkmalssatz generiert, indem das Signal in charakteristische Bereiche[1] unterteilt wird. Für jeden Bereich und jeden Sensor wird ein Merkmalssatz generiert[2], bestehend aus:

- Amplitude des lokalen Extremums eines tiefpassgefilterten Signals,
- zeitlicher Ausdehnung des gewählten Bereichs,
- Anzahl von Schnittpunkten zwischen Signal und einer Filterung desselben,
- Mittelwert der Signale im jeweiligen Zeitbereich sowie
- gefilterter Standardabweichung zum Zeitpunkt lokaler Extrema [10].

Einzelne Griffarten unterscheiden sich somit in der Anzahl an Merkmalen sowie in den Merkmalen selbst[3]. Ein Beispiel für ein Schaltsignal mit Zeitversatz kann eine einzelne Kontraktion für Griffart 1, zwei Kontraktionen für Griffart 2, etc. sein. Die Anzahl der verwendeten Sensoren ist dabei beliebig. Bild 2 stellt zwei Beispiele für Schaltsignale dar (links), die mehrfache Realisierung eines Schaltsignaltyps (Mitte) und eine beispielhafte Auswertung eines Schaltsignals (rechts).

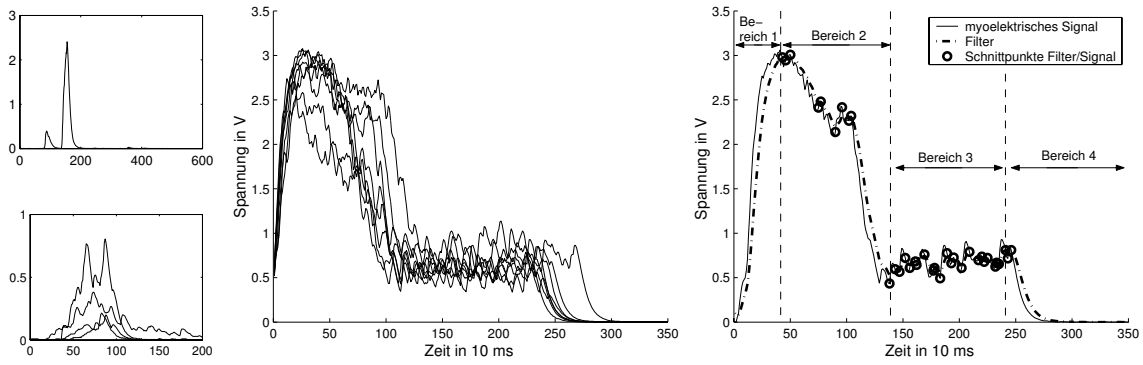


Bild 2: Typische Schaltsignale zum Schalten mit Zeitversatz: links: ein Sensor bzw. mehrere Sensoren, Mitte: Mehrfache Realisierung eines Schaltsignals (ein Sensor), rechts: Auswertung des Schaltsignals

3.2 Schalten ohne Zeitversatz

Ziel des Schaltens ohne Zeitversatz ist es, einer Muskelkontraktion bereits während ihres Auftretens eine Griffart zuzuordnen (Bild 3). Es wird je ein Merkmalssatz zu den Zeiten t_{Anfang} und t_{Ende} bestehend aus

[1] Die Bereiche sind durch lokale, ausgeprägte Extrema gegeben. Der erste Bereich erstreckt sich bis zum ersten Maximum, der zweite bis zum darauffolgenden Minimum usw. Die Anzahl der Bereiche wird auf vier begrenzt.

[2] Ähnliche Ansätze finden sich in [11].

[3] Sollte ein Signal nur aus einem Peak bestehen, so wird für die Bereiche drei und vier $U(t) = \text{konst.} = 0\text{V}$ gesetzt.

- Momentanwerten der über Tief-, Band- und Hochpässe gefilterten Signale und ihrer Ableitungen,
- Produkten der letzten drei Momentanwerte der ungefilterten Signale sowie
- gefilterten Standardabweichungen

gebildet. Bei kleinem $t_{\text{Auswahl}} = t_{\text{Ende}} - t_{\text{Anfang}}$ wird keine Zeitverzögerung zwischen Schaltsignal und Prothesenbewegung empfunden.

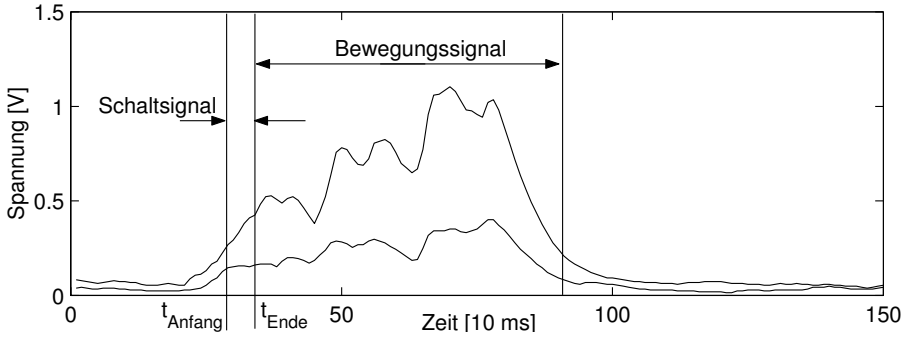


Bild 3: Typisches Schalt- und Bewegungssignal zum Schalten ohne Zeitversatz auf der Basis von zwei Sensorsignalen

3.3 Auswahl repräsentativer Daten

In der Anlernphase besteht die Möglichkeit, dass Datensätze unbrauchbar sind, z. B. bei Sensorausfällen oder wenn die Kontraktion nicht mit der notwendigen Sorgfalt durchgeführt wurde. Um den Klassifikator nicht mit falschen Beispielen anzulernen, sind diese auszusortieren. Unter der Annahme normalverteilter Daten können Ausreißer durch Konfidenzbereiche erkannt und eliminiert werden. In [12] wurde exemplarisch belegt, dass hierdurch die Klassifikationsgüte steigt.

4 Klassifikatorentwurf

4.1 Einteilung von Klassifikatorentwurfsverfahren

Ein Klassifikatorentwurf besteht aus den Schritten

- Merkmalsauswahl,
- Merkmalsaggregation und
- Konstruktion der Entscheidungsregel.

Diese schrittweise Strategie hat sich bewährt, da es überaus schwer ist, direkt einen Klassifikator S_2 zu ermitteln. Suboptimalität wird dabei in Kauf genommen.

Oft ist der Merkmalsraum redundant und enthält bei unglücklicher Wahl Merkmale, die de facto keine für die Klassifikation taugliche Information besitzen. Dieser Problematik nimmt sich die Merkmalsauswahl an. Selbst in dem Fall, dass die ausgewählten Merkmale für eine Klassifikation geeignet sind, muss mit diesen der Entwurf einer Entscheidungsregel nicht zu einem guten Klassifikator führen. Ursache

ist die hohe Dimension des Merkmalsraums, der eine geringe Anzahl von Lerndaten gegenübersteht. Eine zuverlässige Parametrisierung gelingt dann nicht und macht eine Merkmalsaggregation erforderlich.

Die Konstruktion der Entscheidungsregel kann nach unterschiedlichen Prinzipien erfolgen. Diesen Prinzipien ist es mehr oder weniger gleichgültig, ob zuvor eine Merkmalsauswahl und/oder -aggregation vorgenommen wurde. Die Entscheidungsregel bildet gewissermaßen die finale Abbildung, die letztlich die Klassenzuordnung bewerkstelligt. Sie ist somit im Sinne von (2) selbst ein Klassifikator, aber gleichzeitig auch Teil eines Klassifikators.

Neben der hier vorgestellten dreistufigen Vorgehensweise, fassen viele Klassifikatorentwurfsverfahren die Merkmalsaggregation und die Konstruktion der Entscheidungsregel in einem Schritt zusammen. Andere verzichten auf die Merkmalsauswahl.

Ein Unterscheidungsmerkmal für die Entwurfsverfahren stellen die dem Entwurf zu Grunde liegenden mathematischen Modellen dar, vgl. Bild 4.

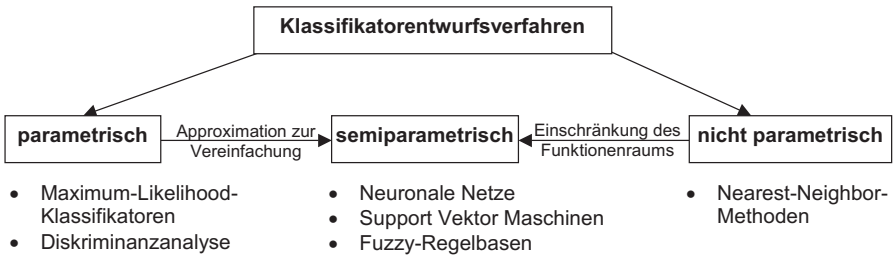


Bild 4: Klassifikatorentwurfsverfahren

Parametrisch werden Modelle genannt, wenn sie einen funktionalen Zusammenhang durch einen Formelausdruck beschreiben, in dem eine endliche Anzahl an Parametern vorkommt. Üblicherweise wird dabei die Struktur des Zusammenhangs als bekannt angenommen (als wahr postuliert), weshalb die Parameter gemeinhin interpretierbar sind. Falls aus Gründen einer einfacheren mathematischen Behandlung statt des parametrischen Modells eine Approximation desselben benutzt werden muss, sind die Parameter des approximierenden Systems nicht mehr interpretierbar, weshalb sie als Semiparameter bezeichnet werden.

Nichtparametrisch werden Modelle genannt, wenn sie einen funktionalen Zusammenhang durch eine unendliche Menge (z. B. Funktionsgraphen), eine endliche Menge (Menge, die durch Aufzählungen charakterisiert wird) oder eine Menge von Kennwerten (z. B. Werte von Funktionalen) beschreiben. Da das Arbeiten mit unendlichen und großen endlichen Mengen schwer fällt, wird durch den Übergang auf eine endliche parametrische Approximation ausgewichen. Das nichtparametrische Modell wird dadurch zu einem semiparametrischen Modell.

4.2 Merkmalsauswahl durch MANOVA

In einem ersten Schritt sind aus den extrahierten s Merkmalen die s_m für eine Klassifikation wichtigsten herauszufinden. Hierzu kommt eine multivariate Analyse der Varianzen (MANOVA, [6, 13]) zum Einsatz

$$S_{21} : \mathbf{x} = (x_1, ..., x_s)^T \mapsto \mathbf{x}|_{\mathcal{I}} = (x_{i_1}, ..., x_i \quad)^T, \quad s > s_m. \tag{3}$$

$\mathcal{I}$ steht hierbei für die Indexmenge der ausgewählten Merkmale.

4.3 Merkmalsaggregation durch Diskriminanzanalyse

Das Ziel der Diskriminanzanalyse besteht ganz allgemein darin, eine Transformation vom s_m-dimensionalen Merkmalsraum in einen s_d-dimensionalen Raum, $s_m > s_d$, mit einem geringen Verlust an Diskriminanzinformation zu finden

$$S_{22} : \mathbf{x}|_{\mathcal{I}} = (x_{i_1}, ..., x_i \quad)^T \mapsto (z_1, ..., z_{s_d})^T, \quad s_m > s_d. \tag{4}$$

Im Rahmen der Griffartenerkennung wird eine lineare Transformation gesucht. Zur Vereinfachung der Schreibweise von $\mathbf{x}|_{\mathcal{I}}$ wird im Folgenden auf die Notation $|_{\mathcal{I}}$ verzichtet. Weiterhin wird vereinbart:

n_j	Anzahl an Datentupeln pro Klasse j
$n = \sum_{j=1}^{k} n_j$	Gesamttupelzahl
$\mathbf{X} = (\mathbf{x}_1, ..., \mathbf{x}_n)^T$	Datenmatrix
$\mathbf{y} = (y_1, ..., y_n)^T;\ y_i \in \{1, \ldots, k\}$	Vektor der Klassenzuordnungen
$\mathbf{Z} = (\mathbf{z}_1, ..., \mathbf{z}_n)^T$	aggregierte Datenmatrix
$\overline{\mathbf{x}} = \frac{1}{n} \sum_{i=1}^{n} \mathbf{x}_i$	Gesamtmittelwert
$\overline{\mathbf{x}}_j = \frac{1}{n_j} \sum_{i:y_i=j} \mathbf{x}_i$	Klassenmittelwerte
$\hat{\mathbf{\Sigma}}_j = \frac{1}{n_j} \sum_{i:y_i=j} (\mathbf{x}_i - \overline{\mathbf{x}}_j)(\mathbf{x}_i - \overline{\mathbf{x}}_j)^T$	geschätzte Klassenkovarianzmatrizen
$\mathbf{B} = \sum_{j=1}^{k} n_j (\overline{\mathbf{x}}_j - \overline{\mathbf{x}})(\overline{\mathbf{x}}_j - \overline{\mathbf{x}})^T$	Zwischenklassenvariationsmatrix
$\mathbf{W} = \sum_{j=1}^{k} n_j \cdot \hat{\mathbf{\Sigma}}_j$	Innerklassenvariationsmatrix.

Bei Verwendung einer linearen Transformation

$$\mathbf{Z} = \mathbf{XA} \tag{5}$$

mit $\mathbf{A} \in \mathbb{R}^{s \ \times s_d}$ besteht das Ziel, die Zwischenklassenvariationsmatrix der aggregierten Merkmale $\mathbf{A}^T\mathbf{BA}$ bei Fixierung der Innnerklassenvariationsmatrix $\mathbf{A}^T\mathbf{WA}$ zu maximieren. Grob gesprochen bedeutet das, dass das Verhältnis aus Klassenabständen und Klassenstreuungen im niederdimensionalen Raum maximiert wird.

Konkret ist zu lösen:

$$\mathrm{sp}(\mathbf{A}^T\mathbf{BA}) \rightarrow \max_{\mathbf{A}} \quad \text{bez.} \quad \mathbf{A}^T\mathbf{WA} = \mathbf{I}_{s_d}. \tag{6}$$

Die Lösung lautet

$$\mathbf{A}_{\text{opt}} = (\mathbf{a}_1, \ldots, \mathbf{a}_{s_d}), \tag{7}$$

wobei die $\mathbf{a}_i$ die zu den geordneten Eigenwerten $(\lambda_1 > \lambda_2 > ... > \lambda_{s_d} > ... > \lambda_s\ )$[4] korrespondierenden Eigenvektoren von $\mathbf{W}^{-1}\mathbf{B}$ bezeichnen.

4.4 Konstruktion der Entscheidungsregel

4.4.1 Begriffsbestimmungen

Eine Entscheidungsregel ist eine Abbildung aus einem Merkmalsraum $\mathcal{M}$ in eine Menge von k möglichen Klassen (z. B. {rot, gelb, grün}, {klein, mittel, groß}), die durch natürliche Zahlen $1, \ldots, k$ spezifiziert seien:

$$S_{23} : \mathbf{x} \in \mathcal{M} \mapsto y \in \{1, ..., k\}. \tag{8}$$

Bei der Klassifikation der Griffarten werden als Elemente des Merkmalsraums $\mathcal{M}$ die aggregierten Datentupel $\mathbf{z}_i \in \mathbb{R}^{s_d}$ verwendet. Dennoch sollen im Weiteren die Elemente des Merkmalsraums durch $\mathbf{x}_i$ bezeichnet bleiben, um mit den in der Literatur üblichen Darstellungen konform zu gehen.

Die Abbildung (8) basiert entweder auf

- Entscheidungsmaßen $d_j(\mathbf{x})$ (statistisch bzw. geometrisch motiviert) oder
- einer Zuordnung zu Entscheidungsbereichen $\mathcal{G}_j$ mit $\mathcal{G}_i \cap \mathcal{G}_j = \emptyset$ für $i \neq j$ und $\bigcup_{j=1}^{k} \mathcal{G}_j = \mathcal{M}$ im Merkmalsraum.

Bei der Klassifikation mittels Entscheidungsmaß ist ein Datentupel $\mathbf{x}$ derjenigen Klasse y zuzuordnen, für die das Maß minimal wird, d. h.

$$y(\mathbf{x}) = \underset{1 \leq j \leq k}{\operatorname{argmin}}\, d_j(\mathbf{x}; \boldsymbol{\theta}_j). \tag{9}$$

Die Beziehung (9) ist eine Entscheidungsregel. $\boldsymbol{\theta}_j$ fasst die Parameter des Entscheidungsmaßes in einem Vektor zusammen. Als Parameter kommen beispielsweise die Klassenmittelwerte und die geschätzten Klassenkovarianzmatrizen in Frage.

Bei der Klassifikation mittels Entscheidungsbereichen ist ein Tupel $\mathbf{x}$ derjenigen Klasse y zuzuordnen, in deren Entscheidungsbereich sich $\mathbf{x}$ befindet. Die Entscheidungsregel lautet

$$y(\mathbf{x}) = j \qquad \text{für } \mathbf{x} \in \mathcal{G}_j. \tag{10}$$

Für einfache Fälle lassen sich die Gebiete $\mathcal{G}_j$ durch klassenspezifische Ungleichungssysteme darstellen, d. h.

$$y(\mathbf{x}) = j, \text{ wenn } \mathbf{g}_j(\mathbf{x}) \geq \mathbf{0} \text{ und } \mathbf{g}_i(\mathbf{x}) \ngeq \mathbf{0} \text{ für } i = 1, \ldots, k,\ i \neq j. \tag{11}$$

[4] Die Eigenwerte genügen zwar genau genommen der Relation $\lambda_1 \geq ... \geq \lambda_{s_m}$, doch gilt die strenge Ungleichheit mit Wahrscheinlichkeit 1.

Eine einzelne Ungleichung markiert eine Bereichsgrenze zwischen zwei Klassen und wird als Diskriminanzfunktion bezeichnet.

Der Vorteil der Entscheidungsmaße ist, dass die aufwändige Auswertung der Ungleichungssysteme entfällt. Eine explizite Formulierung der Diskriminanzfunktionen und damit der Entscheidungsbereiche wird nicht benötigt. Dennoch ist es möglich, Diskriminanzfunktionen zur Trennung der Klassen i, j über $d_i(\mathbf{x}) = d_j(\mathbf{x})$ zu definieren. Allerdings lassen sich diese nicht zwingend durch geschlossene Formelausdrücke darstellen.

Als Vorteil der Entscheidungsbereiche erweist sich die Umsetzbarkeit beliebig komplexer Klassentrennungen. Eine gleichwertige Darstellung durch Entscheidungsmaße gelingt häufig nicht.

4.4.2 Maximum-Likelihood-Entscheidungsregel

Bei der Konstruktion der Entscheidungsregel nach dem Maximum-Likelihood(ML)-Prinzip wird

- die Existenz von k Klassen vorausgesetzt,
- eine Verteilungsannahme für jede Klasse getroffen,
- eine Beschreibung der j-ten Klassenverteilung durch eine über $\boldsymbol{\theta}_j$ (deterministisch, aber unbekannt) parametrisierte reguläre Dichtefunktion unterstellt,
- häufig (wie auch hier) stochastische Unabhängigkeit der Realisierungen gefordert,
- die Zuordnung y_i; $i = 1, \ldots, n$ der Lerndaten $\mathbf{x}_i$ als bekannt, fest und richtig angenommen und
- die diesen Annahmen entsprechende Likelihood-Funktion

$$L(\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_k, y; \mathbf{x}_1, \ldots, \mathbf{x}_n; \mathbf{x}) = f_y(\mathbf{x}; \boldsymbol{\theta}_y) \prod_{i=1}^{n} f_{y_i}(\mathbf{x}_i; \boldsymbol{\theta}_{y_i}). \tag{12}$$

  bezüglich der Parameter $\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_k$ und der Klassenzuordnung y des zu klassifizierenden Tupels $\mathbf{x}$ maximiert[5].

Zum Zwecke einer vereinfachten Darstellung wird im Folgenden von einer Likelihood-Funktion ausgegangen, in der die n Lerndaten den Zuordnungen $\mathbf{x}_1, \ldots, \mathbf{x}_{n_1}$ gehöre zur Klasse 1 und $\mathbf{x}_{n_1+1}, \ldots, \mathbf{x}_{n_1+n_2}$ gehöre zur Klasse 2 usw. genügen. Das Umordnen ist wegen der stochastischen Unabhängigkeit zulässig und schränkt das Problem in keiner Weise ein. Somit hat die Klassifikationslikelihood L_K die Gestalt

$$L_K(\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_k, y; \mathbf{x}_1, \ldots, \mathbf{x}_n; \mathbf{x}) = f_y(\mathbf{x}; \boldsymbol{\theta}_y) \prod_{j=1}^{k} \prod_{i=1}^{n} f_j(\mathbf{x}_\nu; \boldsymbol{\theta}_j); \quad \nu = i + \sum_{l=1}^{j-1} n_l, \tag{13}$$

[5]Im Gegensatz zur Klassifikation ist bei der Clusterung von Daten die Likelihood über den $\boldsymbol{\theta}_j; j = 1, \ldots, k$ und allen y_i zu maximieren.

aus der sich eine ML-Schätzung für die Klassenzugehörigkeit von $\mathbf{x}$ unter Lerndaten $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)$ errechnet

$$\hat{y}^{ML}(\mathbf{x}) = \underset{\substack{y\in\{1,\ldots,k\}\\ \theta_1\in\mathbb{R}^{\ 1},\ldots,\theta\ \in\mathbb{R}}}{\operatorname{argmax}} \quad f_y(\mathbf{x};\boldsymbol{\theta}_y)\prod_{j=1}^{k}\prod_{i=1}^{n} f_j(\mathbf{x}_\nu;\boldsymbol{\theta}_j) \tag{14}$$

$$= \underset{y\in\{1,\ldots,k\}}{\operatorname{argmax}} \Big(\max_{\boldsymbol{\theta}\ \in\mathbb{R}} f_y(\mathbf{x};\boldsymbol{\theta}_y)\prod_{i=1}^{n} f_y(\mathbf{x}_\nu;\boldsymbol{\theta}_y)\Big)\Big(\prod_{\substack{=1\\ \neq}}^{k}\max_{\boldsymbol{\theta}\ \in\mathbb{R}}\ \prod_{i=1}^{n} f_j(\mathbf{x}_\nu;\boldsymbol{\theta}_j)\Big) \tag{15}$$

$$\approx \underset{y\in\{1,\ldots,k\}}{\operatorname{argmax}} f_y(\mathbf{x};\hat{\boldsymbol{\theta}}_y^{ML|n}\ ) \qquad \text{für } n_j \gg \max_{1\le j\le k} p_j. \tag{16}$$

In (14) bezeichnet p_j die Dimension des Parametervektors der Verteilungsdichte der Klasse j. Wird für alle Klassen dieselbe Familie von Verteilungen unterstellt, so sind alle p_j gleich. Speziell für den überwiegend betrachteten Fall der Normalverteilung gilt $p = s + s(s+1)/2$, wobei der erste Summand die Komponenten des Erwartungswertes und der zweite die freien Parameter der Kovarianzmatrix angibt.
Der Darstellung in (15) liegt eine Aufspaltung der Optimierung über y und über die $\boldsymbol{\theta}_j$ zu Grunde. Es werden k optimale Parametersätze $\hat{\boldsymbol{\theta}}_j^{ML|(\mathbf{x},y)}$; $j = 1, \ldots, k$ geschätzt, wobei $|(\mathbf{x}, y)$ meint, dass $\mathbf{x}$ der Klasse y zugeordnet sei. Für $\hat{y}^{ML}(\mathbf{x})$ wird das $y \in \{1, \ldots, k\}$ gewählt, dessen Parametersatz mit dem größten der k Maxima korrespondiert. Ferner wird in (15) von der Regel Gebrauch gemacht, dass das Maximum eines Produktes positiver Funktionen mit disjunkten Parametern gleich dem Produkt der Maxima der Funktionen ist.
Die Approximation in (16) basiert auf der Konsistenzeigenschaft von ML-Schätzern. Für große n_j werden die Schätzungen der Klassenparameter durch das Tupel $\mathbf{x}$ nur noch unwesentlich beeinflusst. Damit können die ML-Schätzungen $\hat{\boldsymbol{\theta}}_j^{ML|(\mathbf{x},y)}$ durch die ML-Schätzungen $\hat{\boldsymbol{\theta}}_j^{ML|n}$ auf der Basis der n_j Lerndaten genähert werden. Mit dieser Näherung zerfällt das Problem in eine Auswertung bezüglich der Werte der klassenspezifischen Dichten an der Stelle $\mathbf{x}$. Im Fall der k-Klassen-Normalverteilung lautet die approximative ML-Entscheidungsregel (AML-Regel) somit

$$\hat{y}^{ML'}(\mathbf{x}) = \underset{1\le j\le k}{\operatorname{argmax}}\ f_j(\mathbf{x};\overline{\mathbf{x}}_j,\hat{\boldsymbol{\Sigma}}_j) \tag{17}$$

$$= \underset{1\le j\le k}{\operatorname{argmax}}\ \frac{1}{(2\pi)^{\overline{2}}\sqrt{\det\ \hat{\boldsymbol{\Sigma}}_j}} e^{-\frac{1}{2}(\mathbf{x}-\overline{\mathbf{x}}\ )^T\hat{\boldsymbol{\Sigma}}^{-1}(\mathbf{x}-\overline{\mathbf{x}}\ )} \tag{18}$$

$$= \underset{1\le j\le k}{\operatorname{argmin}} \Big(-\ln f_j(\mathbf{x};\overline{\mathbf{x}}_j,\hat{\boldsymbol{\Sigma}}_j)\Big) \tag{19}$$

$$= \underset{1\le j\le k}{\operatorname{argmin}}\ \ln\det\hat{\boldsymbol{\Sigma}}_j + (\mathbf{x}-\overline{\mathbf{x}}_j)^T\hat{\boldsymbol{\Sigma}}_j^{-1}(\mathbf{x}-\overline{\mathbf{x}}_j). \tag{20}$$

Für einen 3-Klassen-Normalverteilungsfall sind in Bild 5 die klassenspezifischen Dichten dargestellt. Bild 6 zeigt die Höhenlinien des Entscheidungsmaßes und die Graphen der Diskriminanzfunktionen (dicke Linien), also jene Linien, in denen das Minimum über j nicht eindeutig ist.

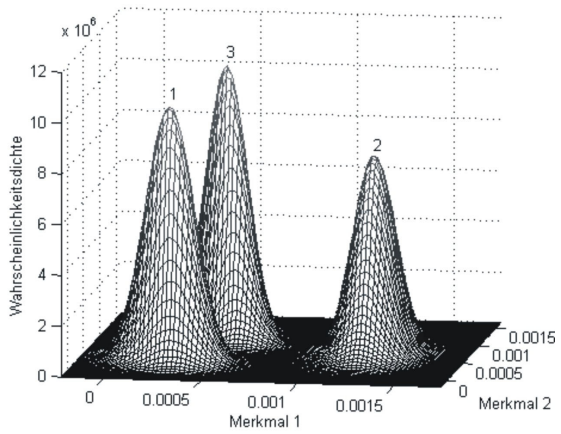


Bild 5: Wahrscheinlichkeitsdichten

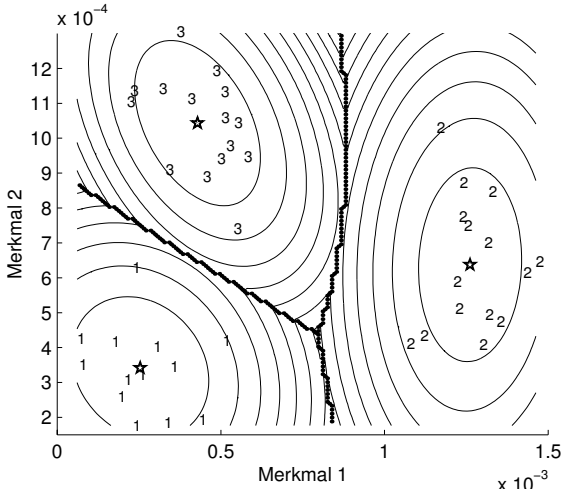


Bild 6: Höhenlinien des Entscheidungsmaßes

4.5 Fuzzy-Regelbasen

In diesem Abschnitt soll kurz auf ein in [6] beschriebenes Verfahren eingegangen werden, um die in dieser Arbeit vorgestellten Verfahren einem alternativen Entwurfskonzept vergleichend gegenüberzustellen.

Beim Entwurf mittels Fuzzy-Regelbasen werden durch baumorientiertes Vorgehen (WENN, DANN)-Regeln aus Beispieldaten abgeleitet. Im Ergebnis entstehen relevante Einzelregeln, die zu einer relativ kleinen Anzahl kooperierender Regeln in einer Regelbasis zusammengefasst werden. Im Gegensatz zu Abschnitt 4.4.2 bietet dieses Verfahren einen interpretierbaren Zugang in Form von linguistischen Ausdrücken. Diskriminanzfunktionen und Entscheidungsregeln ergeben sich aus dem Zusammenwirken aller gefundenen Regeln. Eine explizite Darstellung ist zwar möglich, jedoch nicht angestrebt, da hierdurch die Interpretierbarkeit der Regeln verloren ginge.

Anwendung findet dieses Verfahren u. a. bei medizinischen Diagnoseproblemen.

5 Modifikationen

Das Kriterium der Diskriminanzanalyse (6) bevorzugt meist Merkmale, die eine einzelne Klasse weit von den übrigen positionieren, ohne dabei auf die Trennbarkeit näher zusammenliegender Klassen Rücksicht zu nehmen. Dies erhöht i. Allg. den Klassifikationsfehler. Ebenso ergeben sich Fehlklassifikationen durch schlechte Schätzungen (z. B. von Mittelwerten und Kovarianzmatrizen) bedingt durch eine geringe Datenanzahl oder eine Verletzung der Normalverteilungsannahme.

Verbessern lässt sich das Klassifikationsschema neben den bereits angesprochenen Verfahren der Merkmalskombination (Abschnitt 3.2) und der Detektion von Ausreißern und fehlerhaften Daten (Abschnitt 3.3) durch eine modifizierte Diskriminanzanalyse oder eine hierarchische Vorgehensweise zur Klassifikation.

Alle im Folgenden vorgeschlagenen Algorithmen wurden als MATLAB-Toolbox implementiert. Die Datenaufnahme und der Export der Steuerungssoftware als Mikrocontroller-Code erfolgt durch das Programmpaket DAVE [9, 14].

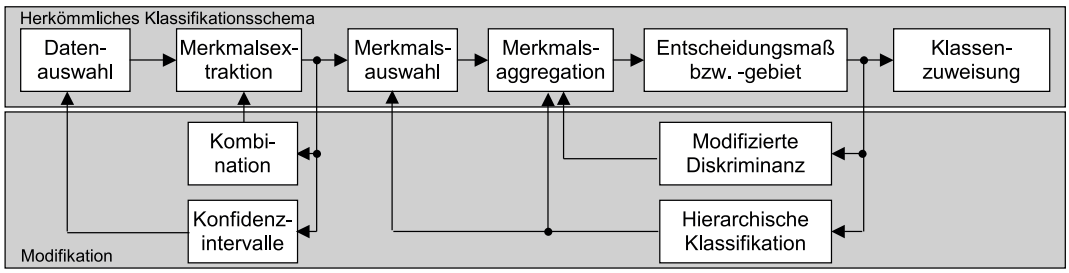


Bild 7: Möglichkeiten der Modifikation des Klassifikationsschemas

5.1 Modifizierte Diskriminanzanalyse

In die AML-Entscheidungsregel gemäß (19) gehen bei der Griffartenklassifikation die transformierten Merkmale $\mathbf{A}^T\widetilde{\mathbf{x}}_i$ ein. Die hierfür bestimmte Transformationsmatrix $\mathbf{A}$ ist bezüglich des Klassifikators nur suboptimal, da sie nach dem Kriterium (6) berechnet wurde. Ein pragmatischer Zugang besteht darin, alle Lerndaten auf (19) anzuwenden und die Fehlklassfikationsrate bezüglich $\mathbf{A}$ zu minimieren. Nachteil dieses Zugangs ist es, dass die Zielfunktion unstetig ist. Um ein stetiges Kriterium zu erhalten, wird der Wert des Entscheidungsmaßes in (19) für die wahre Klassenzuordnung zur Summe der Entscheidungsmaße aller Klassenzuordnungen ins Verhältnis gesetzt

$$p_i(\mathbf{A}) = \frac{\ln f_{y_i}(\mathbf{A}^T\mathbf{x}_i; \mathbf{A}^T\overline{\mathbf{x}}_{y_i}, \mathbf{A}^T\hat{\mathbf{\Sigma}}_{y_i}\mathbf{A})}{\sum_{j=1}^{k} \ln f_j(\mathbf{A}^T\mathbf{x}_i; \mathbf{A}^T\overline{\mathbf{x}}_j, \mathbf{A}^T\hat{\mathbf{\Sigma}}_j\mathbf{A})}; \qquad p_i(\mathbf{A}) \in [0,1]. \tag{21}$$

Die Summe aller Verhältnisse $p_i(\mathbf{A})$ über den Lerndaten liefert eine stetige Approximation für die Klassifikationsgüte, die Summe der Werte $1-p_i(\mathbf{A})$ ein Kriterium für die Fehlklassifikationsgüte. Im ersten Fall ist bezüglich $\mathbf{A}$ zu maximieren, im zweiten zu minimieren. In der praktischen Anwendung hat sich jedoch gezeigt, dass es zweckmäßig ist, von einer Gleichgewichtung der Fehlklassifikationswerte $1-p_i(\mathbf{A})$ abzuweichen. Fehlklassifikationen implizieren stets Verhältnisse $p_i(\mathbf{A}) < 0.5$; die Umkehrung gilt nicht. Werte $p_i(\mathbf{A}) < 0.5$ weisen aber auf einen nicht unbeträchtlichen Einfluss der anderen Klassen hin.

In der Simulation hat sich folgendes Kriterium bewährt

$$\mathbf{A}_{\text{opt}} = \underset{\mathbf{A}}{\operatorname{argmin}} \sum_{i=1}^{n} \psi(p_i(\mathbf{A})) \qquad \text{mit} \qquad \psi(w) = \begin{cases} 2-3w & \text{für } w < 0.5 \\ 1-w & \text{für } w \geq 0.5. \end{cases} \tag{22}$$

$\mathbf{A}_{\text{opt}}$ wird per numerischer Optimierung bestimmt. Als Startwert dient die aus der Diskriminanzanalyse stammende Matrix (7). Die Bilder 8 und 9 zeigen die Ergebnisse der modifizierten Diskriminanzanalyse anhand eines Datensatzes von Proband E (vgl. Abschnitt 6).

Ein Vorteil der vorgestellten modifizierten Diskriminanzanalyse liegt darin, dass sie hinsichtlich klassenspezifischer Kosten einer Fehlklassifikation über die Bewertungsfunktion $\psi(w)$ erweitert werden kann.

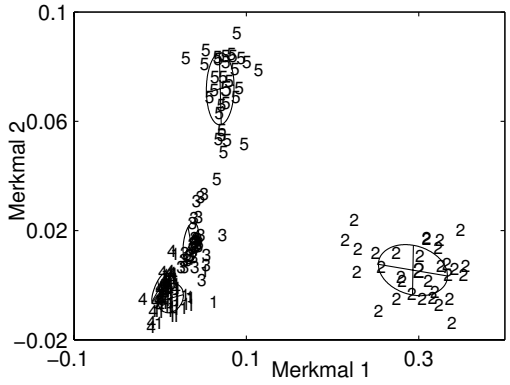


Bild 8: Transformierter Merkmalsraum mit $\mathbf{A}_{\text{opt}}$ nach (7)

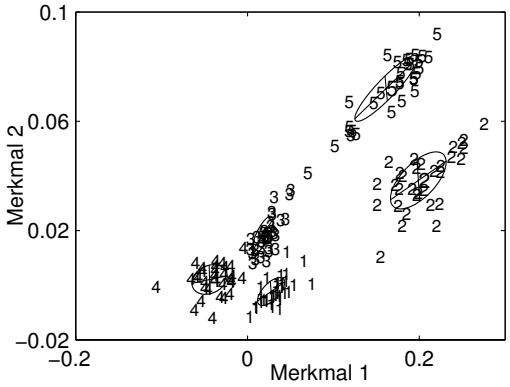


Bild 9: Transformierter Merkmalsraum mit $\mathbf{A}_{\text{opt}}$ nach (22)

5.2 Hierarchische Klassifikation

Ein weiteres Entwurfskonzept basiert auf einem hierarchisch aufgebauten Klassifikatormodell, in dem die Abbildung S_2 durch eine Sequenz von Teilabbildungen erstellt wird. Hierbei werden Merkmalsauswahl und Merkmalsaggregation mit dem Ziel durchgeführt, mindestens eine Klasse gut zu erkennen, um sie abtrennen zu können. Wenn sich eine Klasse im Sinne eines Kriteriums deutlich von den anderen unterscheidet, ist der Weg frei für die Konstruktion eines binären Klassifikators, der in die separierbare Klasse j und die Restklassen trennt. Die der Klasse j angehörenden n_j Datentupel werden danach aus dem Datenmaterial eliminiert. Im verbleibenden Datenmaterial wird wiederum nach einer Klasse gesucht, die sich abtrennen lässt. Dies wird solange fortgesetzt, bis alle Klassen separiert wurden oder bis das Kriterium signalisiert, dass eine weitere Klassenabtrennung eine starke Erhöhung der Fehlklassifikationsraten bedingen würde. In diesem Fall wird das in Abschnitt 5.1 beschriebene Verfahren für einen Klassifikatorentwurf für die verbleibenden Klassen verwendet. Ebenso besteht die Möglichkeit, die Anzahl der zu verwendenden Merkmale durch eine Merkmalsvorauswahl nach Abschnitt 4.2 zu beschränken.

Zunächst ist ein Maß zu finden, das die Separation zweier Wahrscheinlichkeitsdichten f_i, f_j beschreibt. Dabei ist einerseits der Abstand der Mittelpunkte zu betrachten, andererseits die Richtung der Streuellipsoide. Oft werden hierfür die Divergenz nach Kullback-Leibler [15]

$$d(f_i, f_j) = \int f_i \ln \frac{f_i}{f_j} + \int f_j \ln \frac{f_j}{f_i} \tag{23}$$

bzw. das Ähnlichkeitsmaß nach Bhattacharyya [16]

$$d(f_i, f_j) = -\int \ln \sqrt{f_i f_j} \tag{24}$$

verwendet. Alternativen hierzu finden sich in [17]. Für s-variate Normalverteilungsdichten f_i,f_j stellt Tabelle 1 diese Maße dar.

Da die Dreiecksungleichung für beide Maße nicht erfüllt ist, handelt es sich nicht um Abstandsmaße, sondern um Ähnlichkeitsmaße im Sinne einer Halbmetrik.

Maß	Berechnung
Kullback-Leibler-Divergenz	$d_{ij}^{KL} = (\boldsymbol{\mu}_i - \boldsymbol{\mu}_j)^T \frac{\boldsymbol{\Sigma}_i^{-1} + \boldsymbol{\Sigma}_j^{-1}}{2} (\boldsymbol{\mu}_i - \boldsymbol{\mu}_j) + \frac{1}{2}\mathrm{sp}(\boldsymbol{\Sigma}_i \boldsymbol{\Sigma}_j^{-1} + \boldsymbol{\Sigma}_i^{-1}\boldsymbol{\Sigma}_j - 2\mathbf{I}_s)$
Bhattacharyya	$d_{ij}^{B} = \frac{1}{8}(\boldsymbol{\mu}_i - \boldsymbol{\mu}_j)^T \left(\frac{\boldsymbol{\Sigma}_i + \boldsymbol{\Sigma}_j}{2}\right)^{-1} (\boldsymbol{\mu}_i - \boldsymbol{\mu}_j) + \frac{1}{2}\ln \frac{\det \frac{\boldsymbol{\Sigma}_i + \boldsymbol{\Sigma}_j}{2}}{\sqrt{\det(\boldsymbol{\Sigma}_i \boldsymbol{\Sigma}_j)}}$

Tabelle 1: Ähnlichkeitsmaße für normalverteilte Wahrscheinlichkeitsdichten

Grob gesprochen beschreibt die Kullback-Leibler-Divergenz das kumulierte Verhältnis von richtig zu falsch klassifizierten Realisierungen bezüglich der Klasse i und bezüglich der Klasse j. Dies kann zu unerwünschten Schlussfolgerungen aus dem Maß insbesondere bei Abweichungen von der angenommenen Verteilungsdichte führen. Das Aufspalten der Summe in (23) leistet Abhilfe

$$d(f_i, f_j) = \min\Big\{ \int f_i \ln \frac{f_i}{f_j}, \int f_j \ln \frac{f_j}{f_i} \Big\}. \tag{25}$$

Wird (25) für Normalverteilungen f_i und f_j bestimmt und werden Erwartungswert bzw. Kovarianzmatrix durch die betreffenden Schätzungen ersetzt, ergibt sich das folgende Separationsmaß

$$\begin{aligned} d_{ij}(\mathcal{I}) = \min\Big\{ & \frac{1}{2}(\overline{\mathbf{x}}_i - \overline{\mathbf{x}}_j)^T \hat{\boldsymbol{\Sigma}}_i^{-1} (\overline{\mathbf{x}}_i - \overline{\mathbf{x}}_j) + \frac{1}{2}\mathrm{sp}(\hat{\boldsymbol{\Sigma}}_i^{-1}\hat{\boldsymbol{\Sigma}}_j - \mathbf{I}_r), \\ & \frac{1}{2}(\overline{\mathbf{x}}_i - \overline{\mathbf{x}}_j)^T \hat{\boldsymbol{\Sigma}}_j^{-1} (\overline{\mathbf{x}}_i - \overline{\mathbf{x}}_j) + \frac{1}{2}\mathrm{sp}(\hat{\boldsymbol{\Sigma}}_j^{-1}\hat{\boldsymbol{\Sigma}}_i - \mathbf{I}_r) \Big\} \Big|_{\mathcal{I}}. \end{aligned} \tag{26}$$

In Formel (26) bezeichnet $\mathcal{I} = \{i_1, \ldots, i_r\}$ eine geordnete Indexmenge. Die Notation $|_{\mathcal{I}}$ meint, dass nur die in der Menge $\mathcal{I}$ spezifizierten Merkmale verwendet werden, um die betreffenden Vektoren und Matrizen zu bilden. Weiterhin steht r für die Kardinalität von $\mathcal{I}$.

Die Divergenzmatrix $\mathbf{D}(\mathcal{I})$ definiert nun, wie groß die Unterschiede der einzelnen Klassen gemäß (26) zueinander sind

$$\mathbf{D}(\mathcal{I}) = \begin{pmatrix} 0 & d_{12}(\mathcal{I}) & \ldots & d_{1k}(\mathcal{I}) \\ d_{21}(\mathcal{I}) & 0 & \ddots & \vdots \\ \vdots & \ddots & \ddots & d_{k-1,k}(\mathcal{I}) \\ d_{k1}(\mathcal{I}) & \ldots & d_{k,k-1}(\mathcal{I}) & 0 \end{pmatrix} \Bigg|_{\mathcal{C}}. \tag{27}$$

$\mathcal{C}$ ist die Indexmenge bereits abgespaltener Klassen. Die Notation $|_{\mathcal{C}}$ bedeutet, dass die in der Menge $\mathcal{C}$ enthaltenen Klassen nicht in die Berechnung einfließen. Zu Beginn ist $\mathcal{C} = \emptyset$.

Für jedes Merkmal kann aus einer jeden solchen Matrix die am stärksten separierte Klasse $j \notin \mathcal{C}$ und deren Abstand zu ihrem nächsten Nachbarn über

$$d_{\max}(\mathcal{I}) = \max_{\substack{1 \le j \le k \\ j \notin \mathcal{C}}} \; \min_{\substack{1 \le i \le k \\ i \neq j \\ i \notin \mathcal{C}}} d_{ij}(\mathcal{I}) \tag{28}$$

bestimmt werden. Die Größe von $d_{\max}(\mathcal{I})$ gibt einen Hinweis, ob die Indizes $\mathcal{I}$ Merkmale spezifizieren, die eine Klasse gut separieren. Es ist folglich dasjenige $\mathcal{I}$ bei

$$\mathbf{D}_{1,2} = \begin{pmatrix} 0.0 & 13.4 & 45.3 & 30.3 & 42.5 \\ 13.4 & 0.0 & 14.3 & 3.5 & 23.6 \\ 45.3 & 14.3 & 0.0 & 17.0 & 25.7 \\ 30.3 & 3.5 & 17.0 & 0.0 & 26.8 \\ 42.5 & 23.6 & 25.7 & 26.8 & 0.0 \end{pmatrix}$$

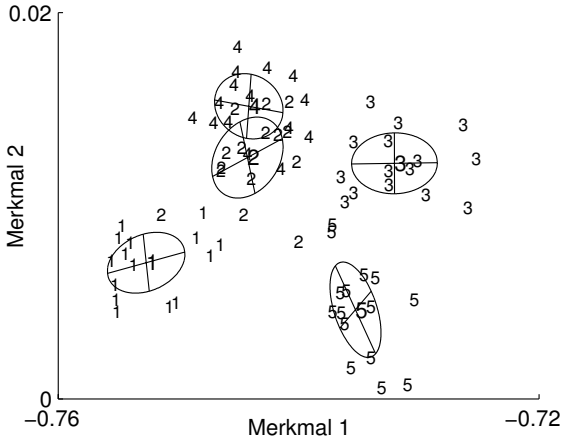


Bild 10: 5-Klassen-Beispiel mit Auswahl von zwei Merkmalen (Proband D, vgl. Abschnitt (6). Links: Divergenzmatrix, rechts: Klassifikationsverteilung mit Kovarianzellipsen

einer vorgegebenen Kardinalität r zu wählen, das $d_{\max}(\mathcal{I})$ über $\mathcal{I}$ maximiert. Bei s Merkmalen sind demzufolge $\binom{s}{r}$ Matrizen $\mathbf{D}(\mathcal{I})$ zu berechnen. Ein suboptimales Verfahren erweitert die Indexmenge $\mathcal{I}$ mit einem Erweiterungselement i_{r+1} sequentiell nach dem Prinzip

$$\mathcal{I}_{r+1} = \{\mathcal{I}_r, i_{r+1}\} \qquad \text{mit } i_{r+1} = \underset{\substack{1\le l\le s \\ l\notin\mathcal{I}}}{\operatorname{argmax}} \; d_{\max}(\{\mathcal{I}_r, l\}) \text{ und } \mathcal{I}_0 = \emptyset. \tag{29}$$

Hohe Kardinalitäten r bedingen hochdimensionale Merkmalsräume und somit eine große Anzahl zu schätzender Parameter beim Entwurf der Entscheidungsregel. Es ist sinnvoll, eine lineare Merkmalsaggregation mit der Transformationsmatrix $\mathbf{A}_{\text{hierch, opt}}$ durchzuführen, wenn die Kardinalität r eine vorgegebene Dimension s_d überschreitet. Die Transformationsmatrix ist dann so zu optimieren, dass die maximale Divergenz zweier Klassen maximiert wird:

$$\mathbf{A}_{\text{hierch, opt}} = \underset{\mathbf{A}_{\text{hierch}}}{\operatorname{argmax}} \; d_{\max}(\mathbf{A}_{\text{hierch}}, \mathcal{I}_{r+1}) \tag{30}$$

Als Startwerte für $\mathbf{A}_{\text{hierch}}$ eignen sich für $s_d < k$ die Verbindungsvektoren $\mathbf{a}_{\text{hierch},i}$ vom Mittelwert der Klasse mit maximaler Divergenz (gemäß (26)) zu den Mittelwerten der übrigen Klassen[6]. Es gilt $\mathbf{A}_{\text{hierch}} = (\mathbf{a}_{\text{hierch},1}, ..., \mathbf{a}_{\text{hierch},s_d})$. Für $s_d \geq k$ kann (7) als Startwert herangezogen werden. Ist $r \leq s_d$ wird $\mathbf{A}_{\text{hierch, opt}} = \mathbf{I}_{r,r}$ gesetzt.

Ob $\mathcal{I}_{r+1}$ und $\mathbf{A}_{\text{hierch, opt}}$ ausreichen, um eine Klasse zu separieren, wird mit dem Kriterium T_j geprüft, in das (21) einzusetzen ist. Für jede Klasse $j \notin \mathcal{C}$ beschreibt es die Güte der gefundenen Parameter

$$T_j = \sum_{\nu:y\ =j} \psi(p_\nu(\mathbf{A}_{\text{hierch,opt}})) \qquad \text{mit} \qquad \psi(w) = \begin{cases} 2-3w & \text{für } w < 0.5 \\ 1-w & \text{für } w \geq 0.5. \end{cases} \tag{31}$$

Klassen, deren Gütewert T_j eine vorgegebene Fehlertoleranz $T_{\max}$ unterschreiten, werden extrahiert, ihre Indizes der Menge $\mathcal{C}$ zugefügt. Anschließend wird der Algorithmus erneut durchlaufen. Sind alle zur Verfügung stehenden Merkmale bereits verwendet worden, so wird der Algorithmus durch eine Klassifikation der übrigen Klassen gemäß Abschnitt 5.1 abgebrochen.

Zwar ist der Algorithmus suboptimal, da

[6]Geordnet werden die Vektoren nach fallender Divergenz zur ausgewählten Klasse.

Gegeben: k, s_m, s_d, $T_{\max}$
$r = 0$: Anzahl ausgewählter Merkmale,
$\mathcal{C} = \emptyset$: Indexmenge abgespaltener Klassen
$\mathcal{I}_r = \emptyset$: Indexmenge verwendeter Merkmale

1. Wähle s_m Merkmale nach Abschnitt 4.2 aus
2. Solange card($\mathcal{C}$) $< k$
3. Für $l = 1$ bis s_m, $l \notin \mathcal{I}_r$ (übrige Merkmale)
4. Berechne Divergenzmatrix für Klassen $j \notin \mathcal{C}$ gemäß (27): $\mathbf{D}(\{\mathcal{I}_r, l\})$.
5. Berechne maximale Nächste-Nachbar-Divergenz gemäß (28): $d_{\max}(\{\mathcal{I}_r, l\})$.
6. Erweitere bestehenden Merkmalssatz gemäß (29): $\mathcal{I}_{r+1} = \{\mathcal{I}_r, i_{r+1}\}$.
7. Erhöhe Anzahl ausgewählter Merkmale: $r = r + 1$.
8. Wenn $r = s_m$, klassifiziere alle Klassen $j \notin \mathcal{C}$ gemäß Abschnitt (5.1). Ende.
9. Wenn $r > s_d$ (dann Merkmalsaggregation)
10. Startparameter festlegen: $\mathbf{A}_{\text{hierch}} = (\mathbf{a}_{\text{hierch},1}, ..., \mathbf{a}_{\text{hierch},s_d})$.
11. nach Gütekriterium (30) optimieren: $\mathbf{A}_{\text{hierch, opt}}$.
12. sonst $\mathbf{A}_{\text{hierch, opt}} = \mathbf{I}_{r,r}$.
13. Berechne Güte T_j der Klassifikation jeder Klasse $j \notin \mathcal{C}$ gemäß (31).
14. Extrahiere alle Klassen $j \notin \mathcal{C}$ mit $T_j < T_{\max}$: Setze $\mathcal{C} = \{\mathcal{C}, \{j | T_j < T_{\max}\}\}$.
15. Ende.

Bild 11: Algorithmus zum hierarchischen Klassifikatorentwurf

- Merkmale zu bisher gefundenen Merkmalskombinationen hinzugefügt und diese nicht neu berechnet werden und
- die Merkmalsauswahl erst nach der Reduktion der Dimension stattfinden darf.

Aus Aufwandsgründen wurde jedoch auf ein optimales Vorgehen verzichtet. Ebenso wie in Abschnitt 5.1 besteht bei diesem Verfahren die Möglichkeit, Klassifikationskosten zu integrieren. Die Anwendung des Verfahrens beschränkt sich dann auf die Schritte 7 bzw. 14 in Bild 11.

6 Ergebnisse

Als Datenmaterial dienen

- Schaltsignale mit Zeitversatz von vier Patienten (A-D) und
- Schaltsignale ohne Zeitversatz von zwei gesunden Probanden (E, F),

die mittels myoelektrischer Sensoren (Otto Bock, MYOBOCK, Abtastfrequenz $f_a =$ 100 Hz) aufgezeichnet wurden (Tabelle 2). Während bei den Patienten stets antagonistische Muskeln abgegriffen werden, kommen bei den Probanden beliebige Sensorplatzierungen über den aktivierten Muskeln zum Einsatz. Als Schaltsignale ohne Zeitversatz wurden bei den Probanden E und F einzelne Fingerbewegungen definiert.

Tabelle 3 zeigt die Ergebnisse einer 2-fachen, 5-fachen und 10-fachen Crossvalidierung des AML-Klassifikators über den Patientendaten basierend auf der Diskriminanz nach (7) (DI), der modifizierten Diskriminanz nach (22) (MD) und der hierarchischen Klassifikation nach Bild 11 (HK) und vergleicht diese mit den Ergebnissen

Person	A (Patient)	B (Patient)	C (Patient)	D (Patient)	E (Proband)	F (Proband)
Alter	34	31	34	62	26	27
Geschlecht	m	w	m	m	m	m
Amputation	rechter Unterarm	linker Unterarm	linker Oberarm	linker Oberarm	- (Proband)	- (Proband)
Zeitpunkt des Verlustes	congenital	congenital	kurz nach Geburt	im Alter von 20 Jahren	-	-
proth. Versorgung	Otto-Bock, 2DOF	Otto-Bock, 1DOF	keine	Otto-Bock, 3DOF	-	-
Dauer proth. Versorgung	viele Jahre	viele Jahre	-	seit 1985	-	-
abgegriffene Muskeln	Vorderarm: Extensoren + Flexoren	Vorderarm: Extensoren + Flexoren	Bizeps + Trizeps	Bizeps + Trizeps	Vorderarm	Vorderarm
Sensoren	2	2	2	2	4	2

Tabelle 2: Probandenbeschreibung (DOF: Freiheitsgrade)

der Fuzzy-Regelbasis (Fuzzy). Die Parameter der Verfahren sind der Tabelle zu entnehmen.

Sowohl die MD als auch die HK weisen bessere Ergebnisse als die DI und die Fuzzy-Regelbasis auf. Letztere sind in etwa vergleichbar.

Sind pro Klasse nur wenige Datentupel verfügbar, so ergibt die Fuzzy-Regelbasis gute Klassifikationsergebnisse (Person A, F, 2-fache-CV). Mit steigender Beispielanzahl sinkt die Güte im Vergleich zu den anderen Verfahren.

Hingegen liefert die DI sehr gute Ergebnisse, wenn die Klassenanzahl klein und die Beispielanzahl groß ist. Ursache hierfür ist die gute Schätzung von Kovarianzmatrizen und Mittelwerten.

Steigt die Klassenanzahl, überflügeln die modifizierten Verfahren die beiden anderen. Die MD ist hierbei geringfügig schlechter als die HK, allerdings sind zu deren Auswertung nur wenige Parameter (eine Transformationsmatrix und Parameter für einen Klassifikator) nötig. Dies macht eine Online-Implementierung einfach. Die HK zeigt große Vorteile bei der Abbildung des Klassifikationsproblems auf zwei Dimensionen und damit der Visualisierung der Klassifikation. Ebenso scheint er sehr robust bei niedrigen Beispielanzahlen und niedrigen Dimensionen zu sein. Er neigt allerdings bei größeren Merkmalsdimensionen (vgl. Person A, B, C, F , 2-fache-CV) bezogen auf die Anzahl der Klassen dazu, Daten auswendig zu lernen.

In früheren Arbeiten [18] wurde bereits belegt, dass neuronale Netze (Multi-Layer-Perceptronen bzw. Radial-Basis-Funktionen) über den vorliegenden Daten schlechtere Klassifikationsgüten liefern.

7 Zusammenfassung und Ausblick

Dieser Artikel stellt Methoden zur Verarbeitung myoelektrischer Signale für Handprothesen vor. Basierend auf der Trennung zwischen Griffartenauswahl durch Schaltsignale und nachfolgender Bewegungssteuerung wird ein neues Konzept vorgestellt, das anhand von Beispielen den Bewegungswunsch des Anwenders erlernt.

Person	A	B	C	D	E	F
Klassen	8	8	6	6	5	3
Beispiele	81	120	88	102	155	30
Merkmale	96	96	96	96	384	96
2-fache Crossvalidierung: 50% Lerndaten						
DI, 8→2	31.5 ± 6.6	27.0 ± 6.1	21.3 ± 4.8	34.6 ± 4.8	16.9 ± 5.6	31.1 ± 12.5
DI, 8→3	25.4 ± 7.8	12.5 ± 4.1	11.4 ± 4.0	20.2 ± 4.4	5.9 ± 2.3	38.5 ± 15.6
MD, 8→2	24.9 ± 6.4	13.0 ± 3.8	14.9 ± 5.0	20.6 ± 5.5	6.9 ± 2.8	28.7 ± 11.0
MD, 8→3	22.5 ± 6.0	9.7 ± 3.1	11.5 ± 4.1	17.6 ± 4.5	5.2 ± 1.8	28.5 ± 12.4
HK, 8→2,0.01	17.3 ± 4.7	6.9 ± 2.4	9.2 ± 3.9	16.6 ± 4.3	6.3 ± 2.5	4.1 ± 4.1
HK, 8→3,0.01	24.2 ± 5.9	9.1 ± 2.9	10.6 ± 3.9	15.4 ± 4.3	4.8 ± 1.9	5.4 ± 4.6
HK, 8→2,0.001	17.8 ± 5.2	6.3 ± 2.4	8.7 ± 3.3	16.3 ± 4.7	6.4 ± 2.8	4.6 ± 3.2
HK, 8→3,0.001	23.4 ± 6.1	7.6 ± 2.3	9.9 ± 3.7	15.9 ± 4.4	5.0 ± 2.4	4.4 ± 4.8
Fuzzy	21.9 ± 5.7	14.7 ± 3.5	19.2 ± 4.9	19.3 ± 4.7	11.2 ± 3.0	9.5 ± 3.5
5-fache Crossvalidierung: 80% Lerndaten						
DI, 8→2	22.4 ± 4.2	18.7 ± 3.4	18.2 ± 3.3	34.6 ± 3.3	15.5 ± 2.8	3.7 ± 2.5
DI, 8→3	12.1 ± 4.0	6.0 ± 1.7	5.1 ± 2.0	13.5 ± 2.9	5.6 ± 1.6	4.3 ± 3.2
MD, 8→2	15.7 ± 3.6	5.5 ± 1.9	8.3 ± 2.5	12.4 ± 3.2	4.3 ± 1.2	4.4 ± 3.5
MD, 8→3	10.4 ± 2.7	4.2 ± 1.7	4.9 ± 1.9	10.5 ± 2.1	3.0 ± 1.2	4.4 ± 3.7
HK, 8→2,0.01	9.4 ± 3.5	3.6 ± 1.1	4.5 ± 2.0	10.0 ± 2.0	6.3 ± 2.5	4.9 ± 2.8
HK, 8→3,0.01	10.5 ± 2.5	4.3 ± 1.4	4.1 ± 2.1	9.8 ± 2.0	4.8 ± 1.9	5.1 ± 2.5
HK, 8→2,0.001	11.8 ± 3.2	3.2 ± 1.1	4.1 ± 1.6	10.6 ± 2.5	6.4 ± 2.8	3.6 ± 2.7
HK, 8→3,0.001	8.5 ± 3.0	3.9 ± 1.2	4.9 ± 1.8	8.9 ± 1.6	5.0 ± 2.4	3.3 ± 2.4
Fuzzy	14.0 ± 3.1	10.6 ± 2.5	15.3 ± 3.2	16.5 ± 2.3	7.3 ± 1.9	4.4 ± 3.1
10-fache Crossvalidierung: 90% Lerndaten						
DI, 8→2	22.4 ± 3.6	15.4 ± 2.3	18.1 ± 3.0	34.3 ± 2.8	15.4 ± 2.3	1.5 ± 2.2
DI, 8→3	11.1 ± 2.9	4.3 ± 1.2	5.3 ± 1.6	11.6 ± 2.1	6.0 ± 1.2	1.1 ± 1.7
MD, 8→2	14.4 ± 2.4	4.1 ± 1.2	7.9 ± 2.6	12.0 ± 2.2	4.0 ± 1.0	1.8 ± 2.5
MD, 8→3	8.8 ± 2.9	3.1 ± 0.9	4.1 ± 1.6	10.0 ± 1.9	3.7 ± 0.8	2.1 ± 2.3
HK, 8→2,0.01	9.4 ± 2.4	2.9 ± 0.7	3.9 ± 1.5	10.4 ± 2.4	4.7 ± 1.5	5.7 ± 2.0
HK, 8→3,0.01	7.8 ± 2.4	3.9 ± 1.1	4.6 ± 1.9	9.2 ± 1.7	4.3 ± 1.2	5.5 ± 2.4
HK, 8→2,0.001	10.5 ± 2.5	3.3 ± 0.9	3.7 ± 1.3	9.8 ± 1.9	5.0 ± 1.5	4.6 ± 2.0
HK, 8→3,0.001	6.2 ± 2.1	3.7 ± 0.9	4.8 ± 1.6	9.2 ± 1.7	4.4 ± 1.7	4.7 ± 2.4
Fuzzy	13.0 ± 3.1	9.9 ± 2.1	16.0 ± 2.8	14.1 ± 2.2	6.0 ± 1.4	8.0 ± 3.4

Tabelle 3: Ergebnisse, 2-fache, 5-fache und 10-fache Crossvalidierung (CV), "$s_m \rightarrow s_d$": Auswahl von s_m Merkmalen, Diskriminanz auf s_d Merkmale, $T_{\max} \in \{0.01, 0.001\}$

Aus aufgezeichneten Schaltsignalen werden vollautomatisch Merkmale (z. B. Extrempunkte, Rauschintensität) und daraus Merkmalssätze extrahiert, die eine Abstraktion der einzelnen Schaltsignale darstellen. Eine Merkmalsauswahl, Merkmalsaggregation und approximative Maximum-Likelihood-Entscheidungsregel führt zur Erkennung der Bewegung.

Da die zunächst verwendeten Verfahren (statistische bzw. Fuzzy-Klassifikation) auf den vorliegenden Daten nur befriedigende Ergebnisse liefern, werden die Methoden erweitert. Dabei wird das Optimierungskriterium der Diskriminanzanalyse den vorliegenden Daten so angepasst, dass minimale Klassifikationsfehler entstehen. Weiterhin wird ein hierarchisches Klassifikationsschema vorgeschlagen. Hierfür ist ein Ähnlichkeitsmaß zur Separation der einzelnen Klassen entworfen worden. Auf diese Weise reduziert sich das hochdimensionale Mehrklassenproblem auf eine Reihe von niederdimensionalen Mehrklassenproblemen, bei denen jeweils eine oder mehrere Klassen abgespalten werden.

Eine Validierung findet anhand von sechs Patienten- und Probandendatensätzen statt. Die erweiterten Methoden zur Klassifikation liefern dabei besonders in schwierigen Fällen bessere Ergebnisse als das ursprüngliche Klassifikationsschema. Mit diesen Methoden ist es möglich, auf den Datensätzen Erkennungsraten zwischen 91-97% bei einer Crossvalidierung zu erhalten.

Kommende Aktivitäten befassen sich mit

- der Erweiterung der untersuchten Patientengruppe,
- der Untersuchung von Trainingseffekten,
- der Optimierung der Entwurfsalgorithmen,
- der Validierung von Online-Algorithmen,
- der Implementierung eines online-fähigen Systems sowie
- der Alltagserprobung der Steuerungen.

Danksagung
Die Autoren danken an dieser Stelle den Herrn Kraut, Voelkel und Milewski bzw. Schulz und Pylatiuk vom Forschungszentrum Karlsruhe für ihre Hilfe in der praktischen Umsetzung der Algorithmen bzw. die Möglichkeit, diese zu testen sowie Prof. Gerner und den Herren Rupp, Frühauf und Rebholz von der Orthopädischen Universitätsklinik Heidelberg für die klinische Unterstützung.

Literatur

[1] Kampas, P.: Myoelektroden - Optimal Eingesetzt. *Med. Orth. Techn.* 1 (2001), S. 21–27.

[2] N.N.: Technische Daten Otto Bock Hand. Techn. Ber., Otto Bock. 2000.

[3] Herberts, P.; Almstroem, C.; Caine, K.: Clinical application study of multifunctional prosthetic hands. *Journal of bone and joint surgery* 60-B (4) (1978), S. 552–560.

[4] Hudgins, B.; Parker, P.; Scott, R.: A new strategy for multifunction myoelectric control. *IEEE Transactions on Biomedical Engineering* 40 (1993), S. 82–94.

[5] Reischl, M.; Mikut, R.; Pylatiuk, C.; Schulz, S.: Control strategies for hand prostheses using myoelectric patterns. In: *Proc. 9th Zittau Fuzzy Colloquium*, S. 168–174. 2001.

[6] Mikut, R.; Peter, N.; Malberg, H.; Jäkel, J.; Gröll, L.; Bretthauer, G.; Abel, R.; Döderlein, L.; Rupp, R.; Schablowski, M.; Gerner, H.: Diagnoseunterstützung für die instrumentelle Ganganalyse (Projekt GANDI). *Bericht des Forschungszentrums Karlsruhe* FZKA 6613 (2001).

[7] Nishikawa, D.: *Studies on electromyogram to motion classifier*. Dissertation, Graduate school of engineering, Hokkaido University, Sapporo, Japan. Dec, 2001.

[8] Paulignan, Y.; Frak, V. G.; Toni, I.; Jeannerod, M.: Influence of object position and size on human prehension movements. *Experimental Brain Research* 114 (2) (1997), S. 226–234.

[9] Reischl, M.; Mikut, R.; Pylatiuk, C.; Schulz, S.: Erkennung von Bewegungsabsichten für myoelektrisch angesteuerte handprothesen. In: *Proceedings 11. Workshop Fuzzy Control Des GMA-FA 5.22* (Mikut, R.; Reischl, M., Hg.), S. 106–119. Karlsruhe: Forschungszentrum Karlsruhe, FZKA 6660. 2001.

[10] Reischl, M.; Mikut, R.; Pylatiuk, C.; Schulz, S.; Beck, S.; Bretthauer, G.: Steuerungs- und Signalverarbeitungskonzepte für eine multifunktionale Handprothese. *Automatisierungstechnik (at)* 50 (6) (2002), S. 279–286.

[11] Loose, T.; Jäkel, J.; Mikut, R.: Datenbasierte Generierung natürlichsprachlicher Erklärungstexte am Beispiel der Instrumentellen Ganganalyse. In: *Proc. 12. Workshop Fuzzy Systeme, Dortmund*, S. 43–57. Forschungszentrum Karlsruhe, FZKA 6767. 2002.

[12] Hubert, M.; Van Driessen, K.: Fast and robust discriminant analysis. *submitted to Computational Statistics and Data Analysis* (2002).

[13] Ahrens, H.; Läuter, J.: *Mehrdimensionale Varianzanalyse: Hypothesenprüfung, Dimensionserniedrigung, Diskrimination bei multivariaten Beobachtungen*. Berlin: Akademie-Verlag. 1974.

[14] Kraut, D.: *Entwicklung einer Testumgebung für mikrocontroller-basierte Steuerungen von myoelektrischen Handprothesen*. Forschungszentrum Karlsruhe. 2003.

[15] Kullback, S.; Leibler, R. A.: Information and sufficiency. *Ann. Math. Statist.* 22 (1951), S. 79–86.

[16] Bhattacharyya, A.: On a measure of divergence between two statistical populations defined by their probability distributions. *Bull, Calcutta Math Soc.* 99 (1943), S. 35.

[17] Ben-Bassat, M.: *Use of distance measures, information measures and error bounds for feature extraction*, Kap. 35, S. 773–791. North-Holland Publishing Company. 1982.

[18] Milewski, K.: *Verarbeitung myoelektrischer Signale mittels künsticher neuronaler Netze als Schnittstelle zwischen Mensch und Maschine*. Forschungszentrum Karlsruhe. 2003.

Ein hierarchisches Clusterverfahren basierend auf der Berechnung der Gravitation

Christian Kuhn

Technische Universität Ilmenau
Fakultät für Informatik und Automatisierung
Institut für Automatisierungs- und Systemtechnik
Tel.: 03677 894981
Fax: 03677 894981
E-Mail: christian.kuhn@tu-ilmenau.de

1 Einleitung

Auf vielen Gebieten der Informationsverarbeitung müssen Unmengen an Daten verarbeitet werden. Wünschenswert ist hier die Strukturierung der Daten nach bestimmten Kriterien, um die Datenflut besser handhaben zu können. Eine Möglichkeit der Informationsverdichtung bieten die Verfahren der automatischen Klassifikation, bei der die Daten anhand bestimmter Kriterien in *Klassen* eingeordnet werden. Jede Klasse läßt sich hierbei als eine Objektmenge verstehen, deren Elemente[1] eine (oder mehrere) ähnliche Ausprägungen charakteristischer Eigenschaften haben und sich hierdurch von den Elementen anderer Klassen abgrenzen, da die Elemente der anderen Klassen andere Ausprägungen dieser Eigenschaft besitzen.

1.1 Einteilung der Klassifikationsverfahren

1.1.1 Überwachte Klassifikation

Die Methoden der überwachten Klassifikation werden dazu benutzt, um aktuell vorherrschende Prozeßsituationen vorgegebenen Klassen zuzuordnen. Der Klassifikator ist der Algorithmus, der diese Entscheidung über die Zuordnung mit Hilfe von Vorwissen vornimmt, das ihm in einer offline Phase antrainiert worden ist. Zum Training oder auch zur Belehrung des Klassifikators wird eine Lernprobe verwendet, die charakteristische Beispiele nebst ihrer Klassenzuordnung enthalten. Durch diese Lernprobe wird der Klassifikator in die Lage versetzt, die Verhältnisse im Merkmalraum „einzuschätzen“ und neue Objekte, die ihm vorgeführt werden, anhand dieser Einschätzung zu klassifizieren.

[1] hier: Daten oder Objekte

Das korrekte Arbeiten des Klassifikators wird durch die Güte des Vorwissens bestimmt, d.h. die Lernprobe muß den Arbeitsbereich des Prozesses, dessen Prozeßzustände später klassifiziert werden sollen, genau widerspiegeln. Fehlerhaft vorgenommene Zuordnungen der Objekte in der Lernprobe können die Güte des Klassifikators später erheblich beeinträchtigen und zu Klassifikationsfehlern führen. Möglich sind solche Klassifikationsfehler beispielsweise durch die Adaption der zu beobachteten Systeme. Falls die Adaption nicht auch im Klassifikator berücksichtigt wird, basieren dessen Entscheidungen auf veraltetem, nicht mehr gültigen Wissen.

Den Klassifikator kann man sich allgemein als einen Beobachter des Merkmalraumes mit antrainiertem Wissen vorstellen.

1.1.2 Unüberwachte Klassifikation

Im Gegensatz zu den Methoden der überwachten Klassifikation ist für die Algorithmen der unüberwachten Klassifikation kein Vorwissen über die zu strukturierende Objektmenge erforderlich. Dieses ist oftmals auch gar nicht verfügbar, wenn beispielsweise zum Beispiel ein erster Eindruck über eine komplexe Datenmenge gewonnen werden soll oder eine grobe Strukturierung der Datenmenge vorgenommen werden muß. In diesen Fällen muß ein Algorithmus selbst in der Lage sein, mittels vorgegebener Kriterien eine Strukturierung der Datenmenge vorzunehmen und durch Zuweisung eines Klassenlabels eine sogenannte Partition zu erstellen, bei der alle Objekte einer Klasse aus dieser Partition angehören.

Die Überprüfung der Klassifizierung erfolgt mittels des vorgegebenen Kriteriums. In vielen Algorithmen wird der *Abstand* zwischen zwei Objekten als Ähnlichkeitsmaß herangezogen. Danach gelten zwei Objekte als ähnlich, wenn sie einen möglichst kleinen Abstand zueinander haben. Der Abstand zu den Objekten der anderen Klassen soll jedoch signifikant größer sein und damit auch die Unähnlichkeit zu den Objekten der anderen Klassen zunehmen.

Hierarchische Clusterverfahren führen noch eine Hierarchie zwischen den ermittelten Klassen ein, die mittels eines Dendrogramms wiedergegeben werden kann. Je nach Startpartition lassen sich aufsteigende und absteigende Clusterverfahren unterscheiden, die beide in der Bildung eines Dendrogramms resultieren. Zu Beginn werden alle Objekte einer Objektmenge ein und derselben Klasse zugeordnet. Mit zunehmender Differenzierung werden in dieser Grundklasse Subklassen sichtbar, deren Objekte sich von den Objekten der anderen Subklassen in bestimmter Weise unterscheiden. Setzt man die Betrachtung der Objekte noch differenzierter fort, so sind in den Subklassen wiederum Subklassen zu erkennen usw. Die Hierarchie läßt sich nicht weiter fortführen, wenn jedes Objekt bzw. zueinander

identische Objekte selbst ihre eigene Klasse bilden. Diese Hierarchie läßt sich auch als Entscheidungsbaum verwenden. Steht z.B. die Oberklasse eines Objektes fest, brauchen für die Objektklassifizierung nur noch die Subklassen betrachtet werden, die in dieser Oberklasse enthalten sind. Das Klassifikationsproblem läßt sich durch diese Maßnahme erheblich vereinfachen.

2 Bemerkungen zur Erkenntnistheorie

2.1 Zwei-Welten-Modell

Bei der Lösung des Klassifikationsproblems stehen sich zwei grundlegende Welten gegenüber: die *Realität*, in welcher der Prozeß abläuft, der schließlich durch den Klassifikator beobachtet werden soll und eine so bezeichnete *künstliche Welt*, in der die Untersuchungen der Meßsignale und die Klassifikation stattfinden, vgl. Abbildung 1.

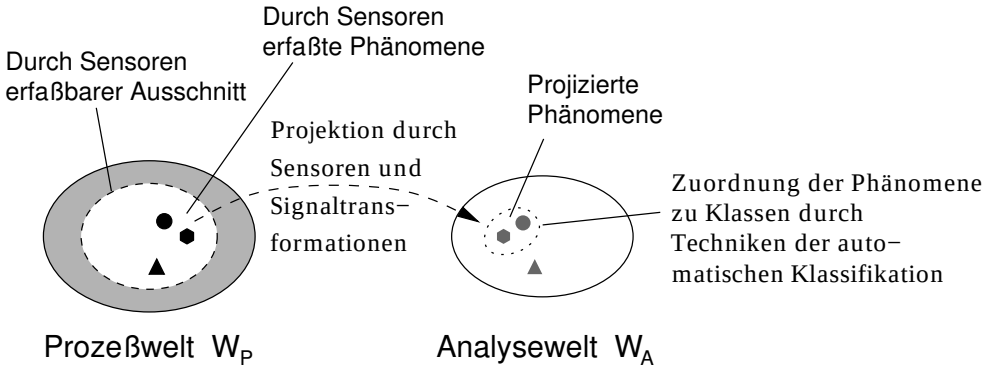


Abbildung 1: Darstellung des Zwei-Welten-Modells

Ähnlich wie die Sinnesorgane des Menschen dienen Sensoren der Abbildung von Prozeßsituationen in die künstliche Welt. Dabei muß das steuernde System bzw. das sich in der Prozeßwelt orientierende System nicht nur Wissen über einzelne Prozeßsituationen, sondern über einen ganzen Ausschnitt aus der Prozeßgeschichte besitzen. Sensoren liefern ebenso wie die Sinnesorgane des Menschen kein vollständiges Abbild ihrer Umwelt bzw. der Prozeßwelt. Vielmehr besitzen Sensoren einen begrenzten Arbeitsbereich und können auch nur einen begrenzten Ausschnitt der Prozeßwelt erfassen.

2.2 Noumene und Phänomene

2.2.1 Noumene

Dinge der Realität bzw. der Prozeßwelt, die keinen Zugang zur künstlichen Analysewelt finden, sollen hier als *Noumene* bezeichnet werden. Noumene sind Dinge der Realität, deren äußere Eigenschaften derart beschaffen sind, daß sie über die Sensoren bzw. Sinnesorgane nicht wahrgenommen werden können (weil die Eigenschaften beispielsweise außerhalb des Arbeitsbereiches der Sensoren liegen). Möglicherweise existieren sie gar nicht reell, sondern sind nur durch ihre Auswirkungen auf andere Dinge spürbar.

2.2.2 Phänomene

Unter *Phänomenen* sollen ganz allgemein Erscheinungen in einer Welt verstanden werden. Phänomene haben äußere Eigenschaften (*Schein*), über die sie durch die Sinnesorgane bzw. Sensoren erfaßt und in die künstliche Analysewelt projiziert werden. Phänomene haben jedoch auch innere Eigenschaften, die einem Beobachter auf den ersten Blick verborgen bleiben und die das *Wesen* des Phänomens ausmachen. Bezüglich der künstlichen Welt sind alle Phänomene die Folge eines kausalen Zusammenhanges, d.h. ein Phänomen, das in der künstlichen Welt erscheint, muß zwangsläufig ein korrespondierendes Phänomen in der Realität bzw. in der Prozeßwelt haben, das über die Sensoren wahrgenommen wurde.

Phänomene lassen sich durch eine Menge von Eigenschaften beschreiben. Die Menge der aller Eigenschaften eines Phänomens können einem Beobachter / einem sich orientierenden System nicht bekannt sein und sind zu dessen charakteristischer Beschreibung auch unerheblich. Über die Sensoren / Sinnesorgane können damit nur eine Untermenge der Eigenschaften von realen Phänomenen erfaßt werden.

2.3 Glauben und Wissen

Die Erkenntnistheorie beschäftigt sich unter anderem mit der Frage, in welchem Maße die Welt überhaupt erkennbar ist. Nach einer realistischen Weltanschauung existiert die Realität unabhängig vom menschlichen Bewußtsein. Der Zugang zu dieser Realität erfolgt über die Sinnesorgane, die einen Teil dieser Realität in das menschliche Bewußtsein abbilden. Da die Sensoren nur einen Ausschnitt aus der beobachteten Welt abbilden können, müßte eigentlich vielmehr vom *Glauben*, daß ein in der künstlichen Welt registriertes Objekt mit der in der Realität vermuteten

Situation übereinstimmt, gesprochen werden. Dieser Glauben kann jedoch auch falsch sein, was sich in Fehlschlüssen oder auch Fehlklassifikationen äußert.

In [1] wird das *Wissen* eines Beobachters x über einen Sachverhalt p wie folgt definiert:

x **weiß,** daß p, gdw.

- p der Fall ist
- x glaubt, daß p.

Um zu überprüfen, daß sich das Wissen von x auch tatsächlich auf wahren Sachverhalten stützt, ist ein weiterer, übergeordneter Beobachter notwendig, der Zugang zum Wissen des Beobachters x und zum durch x geglaubten Sachverhalt p hat und damit die Prozeßwelt als auch die künstliche Analysewelt überschauen muß. Ein solcher übergeordneter Beobachter ist beispielsweise der „Lehrer“ eines Klassifikators.

2.4 Objekte als Phänomene

Der Klassifikator spielt die Rolle des Beobachters der künstlichen Analysewelt. Hier beobachtet er die Objekte dieser künstlichen Welt und ordnet diese bestimmten Objektgruppen zu. Der interne Beobachter dieser Welt wird durch die Sensoren mit „von außen“ kommenden Objekten „versorgt“, die gleichzeitig die Schnittstelle zwischen realen Prozeßsituationen und den inneren zugehörigen Phänomenen darstellen. Bevor der Klassifikator nun überhaupt eine Aussage über die Objektgruppenzugehörigkeit machen kann, muß er zunächst ein elementares Klassifikationsproblem lösen: er muß ein Objekt von einem Nicht-Objekt im Merkmalraum unterscheiden. Dieses Grundwissen muß in der Struktur des Klassifikators bzw. in den ausgenutzten Gesetzmäßigkeiten verwurzelt und dem Klassifikator gewissermaßen „angeboren“ sein.

Statistische und Abstands-Klassifikationsverfahren lösen dieses Problem dadurch, daß leere Gebiete im Merkmalraum überhaupt nicht betrachtet werden und jede angegebene Raumposition implizit als Objekt gewertet wird, das mit einem externen diskreten Ereignis korrespondiert. Eine Wertung des externen Ereignisses ist hierdurch allerdings nicht möglich.

Auch Fuzzy-Klassifikationsverfahren werten angegebene Positionen automatisch als Objekte, wobei allerdings ein Zugehörigkeitswert μ erlaubt ist, der die Zugehörigkeit zu einer bestimmten Objektmenge angibt.

Ein phänomenologisches Objektverständnis geht davon aus, daß Objekte sowohl *äußere* als auch *innere* Eigenschaften besitzen. Die äußeren Eigenschaften – hier sei besonders die Position im Merkmalraum genannt – korrespondieren mit den Umständen des extern eingetretenen Ereignisses bzw. mit den charakteristischen Eigenschaften der zu erkennenden Prozeßsituation. Die inneren Eigenschaften ermöglichen es einem Objekt, vom internen Beobachter (Klassifikator) erkannt zu werden und durch ihn einer bestimmten Objektgruppe zugeordnet zu werden. Im einfachsten Fall lassen sich innere Eigenschaften durch *Zustände* repräsentieren. Damit ist es beispielsweise möglich, den gesamten Merkmalraum nach Objekten abzutasten. Das Vorhandensein von Objekten im Merkmalraum geht mit einer Zustandsänderung an diesen Stellen einher.

3 Struktur der künstlichen Welt

3.1 Beschreibung eines generischen Feldmodells

Auch wenn vom Entscheidungsverhalten eines Klassifikators eine gewisse Intelligenz erwartet wird, muß sich diese Intelligenz letztlich durch bestimmte Gesetzmäßigkeiten nachbilden lassen bzw. mit bestimmten Gesetzmäßigkeiten konvergieren. Im folgenden werden einige wichtige Gesetzmäßigkeiten beschrieben, die in der künstlichen Welt gültig sein sollen.

3.1.1 Affinität

Klassische Ähnlichkeitskriterien ziehen lediglich die Distanz zwischen zwei Objekten zur Bewertung der Ähnlichkeit heran. Für dieses Clusterverfahren soll ein Ähnlichkeitskriterium benutzt werden, in das sowohl äußere Merkmale (Position im Merkmalraum, Distanz) als auch innere Merkmale (Zustand) eingehen. Der Zustand eines Objektes läßt sich hier als eine Art „Beladung“ e interpretieren, mit der das Objekt beladen wurde [2, 3]. Der Wertebereich läßt auch negative Werte zu, um damit auch inverse Objekte in der künstlichen Welt modellieren zu können.

Die Beurteilung der äußeren Eigenschaften der Objekte erfordert ein Bezugs- bzw. Inertialsystem im Merkmalraum, an dem sich der innere Beobachter orientieren kann.

Zwischen zwei Objekten wird nun eine Kraft angenommen, die die Neigung zweier Objekte beschreibt, eine Verbindung miteinander einzugehen bzw. ein Cluster zu bilden. Diese Kraft wird als *Affinität* bezeichnet. Sie berechnet sich nach

$$\vec{F}_{1,2} = \frac{e_1 \cdot e_2}{\left(d\left(\boldsymbol{m}_1, \boldsymbol{m}_2\right)\right)^3} \cdot \vec{d}(\boldsymbol{m}_1, \boldsymbol{m}_2) \tag{1}$$

mit d als dem euklidischen Abstand zwischen beiden Objekten.

3.1.2 Feldstärke

Um ein Objekt herum breitet sich ein Affinitätsfeld mit der Feldstärke $\vec{A}$ kugelförmig aus. Sie läßt sich nach

$$\vec{A}_0 = \frac{e_1}{\left(d\left(\boldsymbol{m}_0, \boldsymbol{m}_1\right)\right)^3} \cdot \vec{d}(\boldsymbol{m}_0, \boldsymbol{m}_1) \tag{2}$$

berechnen. Neben der als Vektorfeld auftretenden Affinität existiert auch ein skalares Potentialfeld. Die Affinitätskraft ist so gerichtet, daß zwei Objekte mit gleichnamigen Beladungen voneinander abgestoßen werden. Diese Orientierung führt bei beweglichen Objekten zu einer Erhöhung der Entropie im System (entspricht dem zweiten Hauptsatz der Thermodynamik) und wirkt auf dieses gleichzeitig stabilitätserhaltend, da die Objekte bestrebt sind, ihre eigene Identität zu erhalten.

3.2 Ein Analyse-Raum-System

3.2.1 Der zelluläre Raum

Der Raum der künstlichen Welt entspricht dem n-dimensionalen Merkmalraum, in welchem sich die Objekte mit n Merkmalen befinden. Für numerische Untersuchungen des Merkmalraumes ist ein diskretisierter Raum sinnvoll. Dieser diskrete Raum besteht aus einer Vielzahl von orthogonalen, disjunkten Subräumen, die als *Hyxel* (Hyperspace Elements) bezeichnet werden sollen. Eine detaillierte Beschreibung des zellulären Raumes ist in [3] zu finden. Jedes in den Merkmalraum projizierte Objekt befindet sich damit in einem bestimmten Hyxel des Merkmalraumes.

Jedes Hyxel läßt sich ebenfalls durch die Zustandsgröße *Beladung* e charakterisieren. Ist die Beladung eines Hyxels größer als null, so läßt sich daraus schlußfolgern, das es Objekte enthält. Die Beladung eines Hyxels ist von der Zahl der enthaltenen Objekte abhängig und ist – einen fixen Wert für die Objektbeladung vorausgesetzt – proportional zur Objekthäufigkeit im Hyxel.

Die Untersuchung eines diskreten Raumes bietet den Vorteil der vereinfachten numerischen Untersuchung, da nicht mehr die einzelnen Objekte untersucht werden,

sondern stattdessen die Hyxel des zellulären Raumes. Dies läßt sich beispielsweise durch verschachtelte Schleifen recht einfach bewerkstelligen.

3.2.2 Ein Zwei-Raum-System

Versucht man, die Affinitätsfeldstärke an einer Position des Merkmalraumes zu berechnen, an der sich gerade ein Objekt befindet, erhält man den Wert $A \rightarrow \infty$. Um derartige Singularitäten zu vermeiden, werden zwei n-dimensionale Räume innerhalb eines $n+1$-dimensionalen Raumes betrachtet. Der Objektraum $\mathcal{O}$ nimmt lediglich die zu untersuchenden Objekte auf. Die Analyse der Feldverteilung findet dagegen im Analyseraum $\mathcal{A}$ statt, der um den Wert ζ auf der zusätzlichen $n+1$-ten Dimension gegenüber dem Objektraum verschoben wurde. Untersucht wird hierzu lediglich das n-dimensionale Vektorfeld, die $n+1$-te Komponente ist im n-dimensionalen Raum nicht spürbar. Der Analyseraum weist nun keine Singularitäten mehr auf, so daß ein glatter Feldverlauf im Analyseraum erwartet werden kann [3].

3.3 Attraktoren

3.3.1 Attraktoren als Ursache für stabiles Systemverhalten

Viele Systeme der Realität besitzen ein stabiles Systemverhalten. Regelkreise regeln den *Sollwert*, der durch die Führungsgröße vorgegeben ist, der menschliche Organismus besitzt eine stabile Körpertemperatur, Schwingkreise schwingen um einen Arbeitspunkt, Planeten kreisen um ihren Fixstern, Galaxien besitzen eine stabile Struktur, etc. Hinter all diesen Fällen lassen sich Attraktoren vermuten, die Ursache für dieses stabile Systemverhalten sind. Attraktoren sind oftmals nicht sichtbar. Meist lassen sich nur ihre Auswirkungen auf das Systemverhalten feststellen. Attraktoren sind damit *Noumene*.

Unüberwachte Klassifikationsverfahren stehen vor der Aufgabe, diese Attraktoren anhand der zur Verfügung stehenden Beobachtungen über das Systemverhalten im Merkmalraum zu rekonstruieren. Damit läßt sich das Clusterverfahren als *inverses* Problem formulieren: in der Realität bzw. der Prozeßwelt verursachen Attraktoren ein bestimmtes Systemverhalten. Dieses Systemverhalten wird (mittels Sensoren) beobachtet und in den Merkmalraum projiziert. Der innere Beobachter muß nun aus der Objektmenge den oder die ursprünglichen Attraktoren wieder rekonstruieren.

Ein innerer Beobachter „hat das Gefühl“, daß die Objekte eines Clusters durch einen unsichtbaren Attraktor über eine Art *Gravitation* zusammengehalten werden. Hier erscheint es sinnvoll, das in Abschnitt 3.1 vorgestellte Feldmodell für die Nutzung in einem Clusterverfahren näher zu untersuchen.

3.3.2 Attraktoren als Gravitationszentrum

Im folgenden soll untersucht werden, wie sich aus der Affinität ein Feld konstruieren läßt, welches den Zusammenhalt von Objekten eines Clusters beschreibt und damit Eigenschaften der Gravitation besitzt. Die Gravitation muß im Gegensatz zum Affinitätsfeld anziehend wirken. Sie wird im Analyseraum untersucht – aus diesem Grund muß bei ihrer Berechnung der Abstand ζ berücksichtigt werden – und berechnet sich im Punkt $\boldsymbol{m}_0$ nach

$$\vec{G}_0 \quad \blacktriangleleft\blacktriangleright \quad \frac{e_1}{\left(d_\zeta\left(\boldsymbol{m}_0, \boldsymbol{m}_1\right)\right)^3} \cdot \vec{d}\left(\boldsymbol{m}_0, \boldsymbol{m}_1\right) \tag{3}$$

$$= \quad -\frac{e_1}{\left(d_\zeta\left(\boldsymbol{m}_0, \boldsymbol{m}_1\right)\right)^3} \cdot \vec{d}\left(\boldsymbol{m}_0, \boldsymbol{m}_1\right). \tag{4}$$

Der Verlauf der Gravitation im Analyseraum ist für eine Objektmenge mit zwei Merkmalen, die aus zwei Objektclustern besteht, in Abbildung 2 dargestellt. Mit dem Merkmal m_3 wird lediglich die Verschiebung der beiden Räume um den Wert ζ beschrieben.

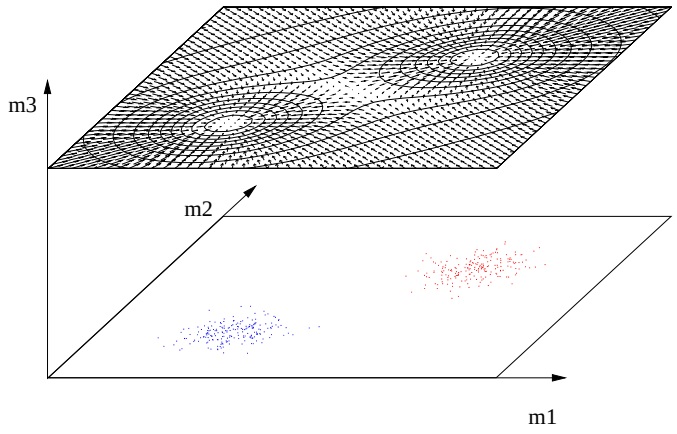


Abbildung 2: Verlauf der Gravitation bei zwei Objektclustern

Zu sehen ist ein Vektorfeld, dessen Feldlinien in zwei Punkten münden, die sich genau oberhalb der Objektcluster befinden. Die ebenfalls dargestellten Äquipotentiallinien deuten an diesen Punkten auf zwei Maxima des Potentials hin. Für

einen Beobachter im Analyseraum ergibt sich die merkwürdige Situation, daß die Gravitation im Gravitationszentrum selbst null wird, da er die maximal werdende $n+1$-te Feldkomponente nicht bemerkt. Hinter diesen Extremwerten verbergen sich zwei Attraktoren, die als Ursache für die Objekthäufungen angesehen werden können. Zwischen den Attraktoren existiert ein Gebiet, in dem die Gravitation ebenfalls null wird. Dieser Punkt stellt aber lediglich den Gleichgewichtspunkt zwischen den Attraktoren dar und äußert sich als Sattelpunkt im Feldverlauf.

Geometrisch können Attraktoren auch mit Hilfe der Spiegelmethode erklärt werden. Diese aus der Elektrotechnik bekannte Methode dient ursprünglich zur Lösung von Randwertaufgaben [2]. Jedes punktförmige n-dimensionale Objekt erzeugt eine kugelförmiges Affinitätsfeld. Durch Superposition der Teilfelder entsteht dann ein charakteristisches Gesamtfeld. Für die folgenden Betrachtungen soll in den Ursprung des $n+1$-ten Merkmals eine n-dimensionale Spiegelhyperebene gedacht werden, die zum Objektraum parallel verläuft. Die Anordnung der Räume ist in Abbildung 3 dargestellt.

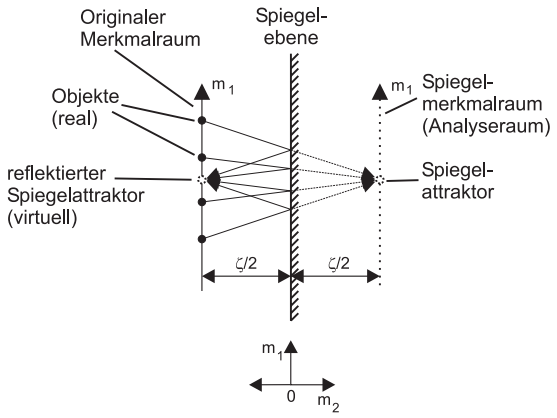


Abbildung 3: Entstehung virtueller Attraktoren im Objektraum

Wird der Objektraum aus dem Koordinatenursprung herausbewegt, spiegelt sich der Objektraum in der Spiegelebene. Die Beladungen der Objekte lassen sich im Spiegelraum zu einer Ersatzbeladung zusammenfassen, die im Gegensatz zu den Objekten ein umgekehrtes Vorzeichen trägt. Die Ersatzbeladung bekommt damit die Eigenschaft eines Attraktors und wirkt anziehend auf die Objekte des Objektclusters. An der Spiegelebene werden die Feldnormalen – hier als *Strahlen* symbolisiert – in den Objektraum zurückreflektiert, so daß im Objektraum ein virtueller Attraktor entsteht, der durch einen im Raum befindlichen Beobachter nicht wahrgenommen werden kann, jedoch zu einer Anziehung der Objekte führt und sich als Gravitationszentrum äußert. In Abbildung 3 sind diese Zusammenhänge für

den eindimensionalen Merkmalraum dargestellt. Sie sind jedoch auch prinzipiell auf den n-dimensionalen Raum anwendbar.

4 Beschreibung der Klassifikationsverfahren

4.1 Unüberwachte Klassifikation

Der Umstand, daß Attraktoren anziehend auf Objekte wirken, kann für ein Clusterverfahren ausgenutzt werden. Attraktoren lassen sich in diesem Fall als Clusterprototypen verwenden. Je größer der Abstand des Objektraumes von der Spiegelebene (und damit vom Analyseraum) ist, desto größer ist die Überlagerung der Affinitätsfelder der einzelnen Objekte. Es kommt zu einer Verschmelzung Attraktoren der einzelnen Objektcluster. Bei kleiner werdendem ζ nimmt die Überlagerung der Felder ab, und es kommt zu einer *Teilung* der Attraktoren, d.h. die zwei entstehenden Attraktoren wandern in die Zentren ihrer Objektmengen, und an der Stelle des vorherigen Gesamtattraktors entsteht ein Sattelpunkt. Diese Teilung des Attraktors in zwei Attraktoren ist als *Bifurkation* im Dendrogramm zu erkennen. Werden Sattelpunkte im Feldverlauf festgestellt, so kann damit auf zwei sich gegenüberstehende Attraktoren geschlossen werden, wobei der Sattelpunkt den Gleichgewichtspunkt zwischen den Attraktoren darstellt.[2]

Die Suche nach Attraktoren entspricht der Suche nach lokalen Extremwerten im Potentialverlauf. In einem zellulären Raum müssen hierfür die Potentiale der benachbarten Hyxel mit dem Potential des interessierenden Hyxels verglichen werden. Das Abtasten des gesamten Raumes ist durch die Anordnung der orthogonalen Hyxel relativ einfach.

Für ein hierarchisches Clusterverfahren ist es zweckmäßig, mit einem großen Abstand ζ zu beginnen. Der Potentialverlauf im Analyseraum wird berechnet und anschließend der Extremwert gesucht. Nach jeder erfolgreichen Suche wird der Abstand ζ verringert und die Suche von neuem gestartet. Der Algorithmus wird abgebrochen, wenn ein Minimalwert von ζ erreicht wird. Er bekommt damit folgende Gestalt:

1. Starte mit $\zeta = \zeta_{max}$
2. Berechne Potentialverlauf im Analyseraum

[2] Ähnlich einer Waage, dessen Hebel in der Mitte drehbar gelagert ist und an dessen Enden Gewichte hängen. Der Drehpunkt des Hebels kann mit dem Sattelpunkt zwischen den Attraktoren verglichen werden.

3. Suche nach Extremwerten im Potentialverlauf → Stellen der Extremwerte sind Positionen der Attraktoren
4. $\zeta = \zeta - \Delta\zeta$
5. $\zeta < \zeta_{min}$?
   (a) ja: Abbruch
   (b) nein: weiter bei 2

4.2 Überwachte Klassifikation

Mit Hilfe der Attraktoren ist eine Zuordnung von neuen Objekten zu bestehenden Objektclustern relativ einfach. Hier wird einfach die Affinität des neuen Objekts zu den Attraktoren ermittelt. Das Objekt wird dem Attraktor mit der größten Affinität zugeordnet, und es gilt

$$e = \text{argmax}\ _{i=1}^{C}\ |\vec{F}_i\,(\boldsymbol{m}_0)\,|. \tag{5}$$

5 Simulation und Anwendung

5.1 Ähnlichkeit zwischen zwei Signalreihen

Das Feldmodell kann zur Bestimmung der Ähnlichkeit eines Meßsignals zu einem vorgegebenen Referenzsignal benutzt werden. Ein Beispiel für derartige Signalverläufe ist in Abbildung 4 dargestellt. Neben dem Referenzsignal sind noch die Verläufe zweier Testsignale zu sehen. Die Testdaten stammen von einem Autohersteller und spiegeln simuliertes Bremsverhalten wider.

Die Signalverläufe umfassen jeweils 500 Abtastwerte. Ein Bremsvorgang kann somit durch 500 verschiedene Merkmale charakterisiert werden. Da das Feldmodell auch in höherdimensionalen Räumen gültig ist, wird das Referenzsignal als punktförmiges Objekt in einen 500-dimensionalen Analyseraum projiziert und übt hier die Funktion des Attraktors aus. Das Meßsignal wird als Objekt in den Objektraum gebracht. Hierzu ist es erforderlich, daß die Meßsignale die gleiche Anzahl von Abtastwerten besitzen müssen. Zum Attraktor läßt sich nun eine bestimmte Affinität feststellen, die vom Abstand der hochdimensionalen Punktobjekte voneinander abhängig ist.

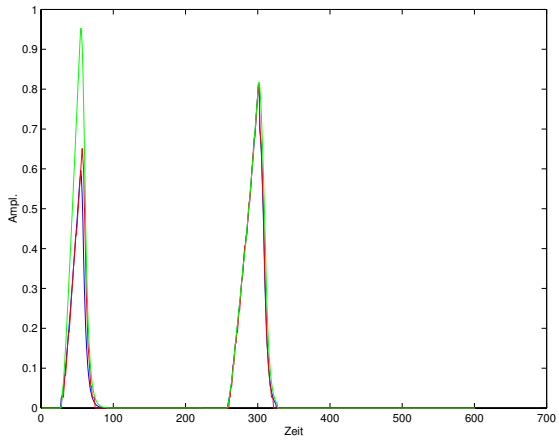


Abbildung 4: Verlauf des Referenzsignals

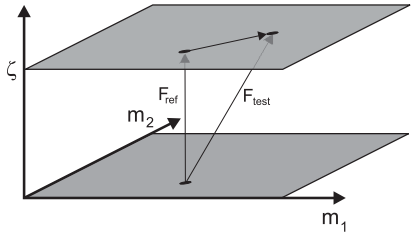


Abbildung 5: zur Ähnlichkeit zweier Punktobjekte

Der Wert ζ hat hier die Funktion, die *Schärfe* des Ähnlichkeitskriteriums festzulegen. Da die Affinität jedoch auch von ζ abhängig ist, muß die Affinität des Attraktors zum Testobjekt noch einmal mit der Affinität des Attraktors zu dem in den Objektraum gespiegelten Attraktor normiert werden, so daß man für die Ähnlichkeit zweier Signalverläufe x_{ref} und x_{test} als Ähnlichkeitskriterium s

$$s_{ref,\,test} = \frac{|\vec{F}_{ref,\,test}\,(\zeta)\,|}{|\vec{F}_{ref,\,ref}\,(\zeta)\,|} \tag{6}$$

erhält. Der geometrische Zusammenhang zwischen Attraktor und Testobjekt wird noch einmal in Abbildung 5 verdeutlicht.

Für die drei Testsignale ergaben sich bei einem Abstand $\zeta = 3$ die Ähnlichkeitswerte $s_{1,1} = 1$ (das Referenzsignal bezogen auf sich selbst), $s_{1,2} = 0.9743$ (das Referenzsignal bezogen auf Testsignal 1) und $s_{1,3} = 0.8$ (das Referenzsignal bezogen auf Testsignal 2).

5.2 Clusteranalyseverfahren

Zur Erprobung des Clusteranalyseverfahrens wurde ein unklassifizierter Datensatz bestehend aus 5 Objektclustern verwendet, vgl. Abbildung 6.

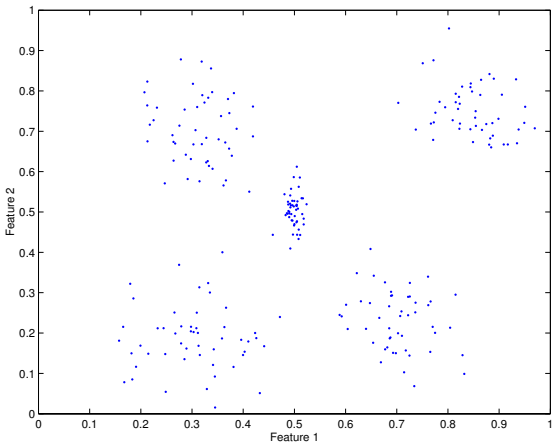


Abbildung 6: Unklassifizierter Datensatz

Anschließend wurden die Potentialmaps beginnend mit großen Werten für ζ berechnet, ζ wurde nach jeder Berechnung sukzessive verringert. In jeder Potentialmap wurde das Maximum bestimmt. Die Maxima ließen sich als Attraktoren für eine anschließende überwachte Klassifikation verwenden. Die beste Aussagekraft hatte die Potentialmap mit dem Abstand $\zeta = 100$, deren Maxima den fünf Clusterprototypen entsprechen. Der hier ermittelte Potentialverlauf ist in Abbildung 7 dargestellt.

Die anschließende Klassifikation der Objekte ergab Klassenzuordnungen, die den Objektgruppierungen nach Abbildung 6 entspricht.

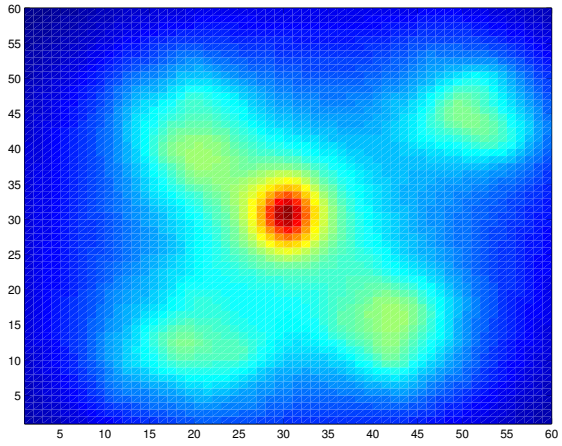


Abbildung 7: Potentialverlauf für $\zeta = 100$

Literatur

[1] Rechenauer, M: *Eine Einführung in die Erkenntnistheorie.* http://www.uni-konstanz.de/FuF/Philo/Philosophie/Mitarbeiter/spohn_books.shtml.

[2] Philippow, E.: *Grundlagen der Elektrotechnik.* Verlag, Technik, Berlin, 2000.

[3] Kuhn, Chr.: *Modelling and Analysis of Dynamical Systems and Processes with a Generic Field Model.* Proceedings of InTech /VJFuzzy '2002.

[4] Bocklisch, S. F.: *Prozeßanalyse mit unscharfen Verfahren.* Verlag Technik, Berlin, 1987.

[5] Höppner, F., F. Klawonn und R. Kruse: *Fuzzy-Clusteranalyse: Verfahren für die Bilderkennung, Klassifizierung und Datenanalyse.* Vieweg Verlag, Braunschweig, Wiesbaden, 1997.

Smoothed Grid Regression

Ulrich Priber
Fraunhofer-Institut für Werkzeugmaschinen und Umformtechnik
09126 Chemnitz, Reichenhainer Str. 88
Tel. +49 371 5397 435 / Fax +49 371 5397 488

29. September 2003

Kurzfassung

Zur praktischen Realisierung von nichtlinearen Zusammenhängen werden in der Praxis häufig einfache Kennfelder eingesetzt, die mittels einfacher linearer Interpolation zwischen den Stützwerten eines rechtwinkligen Rasters realisiert sind (speziell dafür entwickelte Hardware). Dabei ist die Anschaulichkeit und Interpretierbarkeit zumindest für 2-dimensionale Formen ein besonderer Vorteil (Akzeptanz). Die Methode der Smoothed Grid Regression (SGR) versucht aus gegebenem Datenmaterial das Kennfeld bei Vorgabe eines Rasters direkt zu berechnen. Speziell für sehr lückenhafte Daten stößt dies (nach der MKQ-Methode) auf Schwierigkeiten. Lösbarkeit und Stabilität sind dabei oft infrage gestellt. Statt der ansonsten durch Fuzzifizierung oder geeignete Modellannahmen (NN) erreichten Lösbarkeit wird hier die „Glattheit“ des Kennfeldes als zusätzliche Forderung hinzugenommen. Insofern ist der Vergleich mit diesen Technologien interessant. Es ergibt sich eine Modellierungsform, die sich auch auf höherdimensionale Problemstellungen ausdehnen läßt. Speziell für höherdimensionale Aufgabenstellungen wird ein verkürzter Ansatz eingeführt.

1 Vorbemerkung

Die modernen Formen Intelligenter Technologien sind im Grunde damit befaßt, für nichtlineare Abhängigkeiten von i.a. mehreren Einflußgrößen ein geeignetes Funktionsmodell aus einer geeignet erscheinenden Funktionenklasse aufzustellen und unter speziellen Bewertungskriterien zu optimieren. Dieser sehr allgemein formulierten Problemstellung lassen sich Fuzzy Control, Data Mining, Neuronale Netze in ihren zahlreichen Spielarten unterordnen. Natürlich sind auch Regressionsmodelle, Kennlinien und Kennfelder dieser Aufgabenstellung zuzuordnen.
Die Unterschiede werden deutlicher, wenn man die Informationen charakterisiert, die als Grundlage für den Modellentwurf dienen. In der Regel sind sie numerischer Natur, auch wenn bei Fuzzy Control die Unschärfe der Informationen mit in das Modell eingebracht wird. Einer Anzahl von Eingangsvektoren $x_i \in X$ stehen gemessene, simulierte oder erwünschte Ausgangswerte z_i gegenüber. Dies sei im folgenden als Datenbasis (DB) bezeichnet.
Für die Zielstellung der Modellierung sind eine Reihe von Fragestellungen hinsichtlich DB interessant:

- Gültigkeit/Fehlerhaftigkeit der DB
- Vollständigkeit der DB, d.h. wird der Definitionsbereich X ausreichend gestützt

- Widersprüche innerhalb von DB

Außerdem existieren (oft ohne sie direkt zu formulieren) Anforderungen an die Eigenschaften des Modells:

- Fehlerarme Reproduktion der DB
- Stetigkeit/Glattheit des Modells
- Beschränktheit der Funktionswerte z und der Gradienten
- Gutes Interpolationsverhalten
- Geringer Aufwand beim Modellentwurf
- Interpretierbarkeit, Visualisierung
- Möglichkeiten geeigneter Hardware-Realisierung

Mit dem Ziel, diese Anforderungen zu erfüllen, wurde ohne direkten Bezug auf die bekannten Intelligenten Technologien nach einer geeigneten Modellform gesucht. Der Ausgangspunkt dafür ist der für praktisch arbeitende Ingenieure typische Versuch, Abhängigkeiten zunächst durch eine Kennlinie (Polygonzug) zu erfassen. Für verallgemeinerte Kennlinien, Kennfelder und entsprechende höherdimensionale Modellformen wird versucht, die oben genannten Anforderungen befriedigend zu erfüllen. Dabei steht der Kompromiß zwischen guter Reproduktion der DB und den lokalen Glattheitseigenschaften des Modells im Vordergrund. Insbesondere die Betrachtung lokaler Glättungsbedingungen unterscheidet die Vorgehensweise vom üblichen Umgang mit Rasterkennfeldern [1].
Die schwer überschaubaren Zusammenhänge bei höherdimensionalen ($n \geq 3$) Betrachtungen erfordern Möglichkeiten ihrer Bewertung. Die RMS-Werte für Daten- und Glättungsfehler sind dazu allein nicht ausreichend. Da das Modell ohnehin ein (rechtwinkliges) Raster besitzt, liegt eine Visualisierung dieser Raster in Abhängigkeit von nicht darstellbaren Variablen als Wunsch sehr bald vor. Es wurden Formen gefunden, mit denen es möglich ist, mehrere verschiedenen Modelle gleichzeitig mit (oder ohne) Lerndaten zu visualisieren.

2 SGR-Modelle

2.1 Einordnung

Die betrachteten (vollständigen) SGR-Modelle sind eine besondere Form eines Kernfunktionsansatzes der Form

$$F_K(x) = \frac{\sum k_j K_j(x)}{\sum K_j(x)} \tag{1}$$

für beliebig dimensionale unabhängige Variable x. Die räumliche Lage, Anzahl und Form der Kerne K_j bestimmen die Eigenschaften des Modells F_K. Für SGR-Modelle wird ein rechtwinkliges (mehrdimensionales) Raster vorausgesetzt, wobei in jedem Rasterpunkt eine relativ einfache Kernfunktion definiert wird, deren Wirkungsbereich sich gerade bis zu den Nachbarrasterpunkten erstreckt. So sind im linearen Fall für ein x immer nur 2 , für ein Kennfeld 4 bzw. im n-dimensionalen Fall 2^n Kerne wirksam.
Diese Kernfunktionen realisieren die übliche lineare Interpolation zwischen den benachbarten Gitterpunkten. Die Besonderheit dieses Ansatzes besteht darin, daß die Modellparameter (je

Gitterpunkt genau ein Parameter) durch eine Regressionsaufgabe bestimmt werden können. Die Flexibilität ergibt sich nicht durch komplexe Kernfunktionen, sondern durch die Vielzahl der (einfachen) Komponenten. Dies entspricht auch dem für Neuronale Netze bekannten B-Spline-Ansatz [2], jedoch lediglich unter Verwendung B-Splines 2.Ordnung (lineare Interpolation).
Solche Modelle sind stetig, leicht realisierbar und gut interpretierbar, jedoch haben sie Unstetigkeiten an den Gitterpunkten und mit steigender Dimension eine explodierende Anzahl von Parametern.
Aus diesem Grunde wird zusätzlich ein verkürztes Modell (SGR/C) betrachtet, das (bei Einschränkungen an die Flexibilität) mit deutlich weniger Parametern auskommt.

2.2 SGR-Modell (2-dimensional)

Grundlage ist das Rasterfeld. In $x-$Richtung seien es die Rasterpunkte $(u_1, u_2, \ldots, u_{n_x})$ und in $y-$Richtung $(v_1, v_2, \ldots, v_{n_y})$ (strenge Monotonie natürlich vorausgesetzt). Der Ansatz für die Funktion $F_R(x, y)$ hat dann folgende Form:

$$\begin{aligned} F_R(x,y) &= \frac{(u_{k+1}-x)(v_{l+1}-y)r_{kl} + (u_{k+1}-x)(y-v_l)r_{k(l+1)}}{(u_{k+1}-u_k)(v_{l+1}-v_l)} \\ &\quad + \frac{(x-u_k)(v_{l+1}-y)r_{(k+1)l} + (x-u_k)(y-v_l)r_{(k+1)(l+1)}}{(u_{k+1}-u_k)(v_{l+1}-v_l)} \\ &\quad \text{für } u_k \le x \le u_{k+1} \text{ und } v_l \le y \le v_{l+1} \end{aligned} \tag{2}$$

Außerhalb des Rasterfeldes ist F_R nicht definiert, innerhalb dagegen stetig, wobei F_R achsenparallel betrachtet stückweise linear ist. Die $n_x n_y$ Elemente der Matrix $R = \{r_{kl}\}$ sind die Modellparameter und in diesen ist F_R auch linear.

$$F_R(x,y) = A^z(x,y)R^s \tag{3}$$

Mit R^s sei der Spaltenvektor aller Elemente von R bezeichnet. Der Zeilenvektor A^z ist zudem schwach besetzt, maximal 4 von Null verschiedene Elemente. Um diesen Ansatz als Kernfunktionsansatz zu interpretieren, betrachte man für jeden Rasterpunkt eine Kernfunktion (Element von $A(x,y)$), die jedoch nur in den angrenzenden Maschen des Rasters von Null verschieden ist. A ist also bekannt und lediglich durch das vorgegebene Raster und die Lage des Punktes (x, y) bestimmt. Es handelt sich um eine echte Interpolation, d.h. es gilt:

$$A(x,y) \ge 0 \qquad \sum_{k,l} A_{kl}(x,y) = 1 \tag{4}$$

(Für die Rasterpunkte (u_k, v_l) selbst gilt $A_{kl}(u_k, v_l) = 1$ bzw. $F_R(u_k, v_l) = r_{kl}$.)
Auch für den eindimensionalen Fall kann dieses Modell sinngemäß verwendet werden.

2.3 SGR-Modell (n-dimensional)

Dieses Modell läßt sich auf höhere Dimensionen direkt verallgemeinern. Mit den Rasterpunkten $(u_{j_1}, u_{j_2}, \ldots, u_{j_{n_j}})_{(j=1\ldots n)}$ und $x = (x_1, \ldots, x_n)$ ergibt sich:

$$F_R(x) = \frac{\sum_{k_1=0}^{1}\sum_{k_2=0}^{1}\cdots\sum_{k_n=0}^{1}\left[\prod_{j=1}^{n}{}^{(2k_j-1)}\left(x_j - u_{j_{k_j^*+(1-k_j)}}\right)\right] r_{(k_1^*+k_1)\ldots(k_n^*+k_n)}}{\prod_{j=1}^{n}(u_{j_{k_j^*+1}} - u_{jk_j^*})} \tag{5}$$

$$\text{für } u_{j_{k_j^*}} \leq x_j \leq u_{j_{k_j^*+1}} \text{ und } (j = 1, \ldots, n)$$

Auch hier gilt die Darstellung

$$F_R(x) = A^z(x) R^s \tag{6}$$

mit $n_p = \prod n_j$ und es ist A^z wiederum schwach besetzt ($\leq 2^n$).
Ohne auf die Berechnung der Modelle einzugehen, erkennt man sofort, daß die Parameteranzahl n_p mit steigender Dimension exponentiell steigt. Ein möglicher Weg, die Parameteranzahl zu reduzieren, besteht in der Betrachtung von Modellen verringerter Dimension und Einbeziehung der zunächst unberücksichtigten Dimensionen mittels Korrekturkennfeldern. Eine spezielle, in der Praxis gebräuchliche Form wird hier betrachtet.

2.4 SGR/C-Modell

Ohne Beschränkung der Allgemeinheit (gegebenenfalls Vertauschung der Reihenfolge) seien $(x_1, \ldots, x_b)$ die Basisdimensionen, mit deren Rasterstruktur eine (vollständige) SGR-Modellierung möglich sei. Es wird folgender Ansatz betrachtet:

$$F_K(x) = F_R^{(0)}(x_1 \ldots x_b) + K_{b+1}(x_{b+1}) F_R^{(b+1)}(x_1 \ldots x_b) + \ldots + K_n(x_n) F_R^{(n)}(x_1 \ldots x_b) \tag{7}$$

Hierbei sind die $F_R^{(0)}, F_R^{(b+1)}, \ldots, F_R^{(n)}$ b-dimensionale SGR-Modelle und $K_{b+1}, \ldots, K_n$ einfache stückweise lineare Kennlinien (eindimensionale SGR-Modelle). Allerdings müssen an die Kennlinien Normierungsbedingungen gestellt werden, um Eindeutigkeit der Modellparameter zu garantieren. Eine Möglichkeit dafür ist:

$$K_j(u_{j_1}) = 0 \quad \wedge \quad \max_{l=1}^{n_j} |K_j(u_{j_l})| = 1 \quad {}_{(j=b+1,\ldots,n)} \tag{8}$$

Es ergibt sich eine Gesamtparameterzahl von

$$n_p = (n - b + 1) \prod_{j=1}^{b} n_j + \sum_{j=b+1}^{n} n_j \tag{9}$$

Die deutlich geringere Parameteranzahl für steigende Dimensionen ist natürlich mit Einschränkungen der Modellflexibilität verbunden.

Zahlenbeispiel:

$$n = 4, (n_j) = (6, 6, 4, 3), b = 2 \quad \Rightarrow \quad \begin{array}{r} SGR\text{: } n_p = 432 \\ SGR/C\text{: } n_p = 79 \end{array}$$

Vergleichbar zu (6) hat das SGR/C-Modell folgende Form:

$$F_K(x) = A^z R_0^s + b_{b+1} q^{(b+1)}\ A^z R_{b+1}^s + \ldots + b_n q^{(n)}\ A^z R_n^s \tag{10}$$

Die Stützpunkte der Kennlinien K_j sind in den Parametervektoren $q^{(j)}$ zusammengefaßt, d.h. es gilt $q_i^{(j)} = K_j(u_{j_i})_{(j=b+1\ldots n;\ i=1\ldots n_j)}$.
Hierbei gilt $A = A(x_1 \ldots x_b)$ und $b_j = b_j(x_{b+j})_{(j=b+1\ldots n)}$. Dies sind also wiederum die (bekannten) lediglich rasterabhängigen Funktionen. Die Abhängigkeit von den tatsächlichen Parameter $R_{j\ (j=0,b+1\ldots n)}$ und $q_i^{(j)}\ {}_{(j=b+1\ldots n)}$ ist insgesamt betrachtet nicht mehr linear. Deshalb ist dieser Ansatz einer direkten regressiven Berechnung nicht zugänglich.

3 Modellbildung für SGR

3.1 Erweiterte Regressionsaufgabe

Generell ist in die Berechnung der Modelle und ihre Anwendung zu unterscheiden. Die Anwendung ist mit den obigen Definitionen hinlänglich beschrieben.
Um jedoch zu einem Modell zu kommen, wird die Zielstellung verfolgt, zu einen bekannten Datensatz ein Modell zu finden, das die DB möglichst gut reproduzieren kann. Es handelt sich also um eine klassische Regressionsaufgabe, zumindest dann, wenn das Modell in den Parametern linear ist. Für einen Datensatz $z_{(N,1)}$ und eine Argumentmatrix $X_{(N,n)}$ erhält man die Regressionsaufgabe:

$$z = A(X)p + \varepsilon_d \tag{11}$$

Diese ist aber nicht in jedem Fall eindeutig lösbar. Deshalb werden neben der obigen Forderung nach Übereinstimmung mit DB (Datenfehler $\sum \varepsilon_d^2 \to \min$) Zusatzforderungen einbezogen. Diese dienen einerseits zur Vermeidung von Verletzungen der Glattheitsforderungen und andererseits werden die insbesondere bei lückenhaft verteilten bzw. bei sehr wenigen Daten entstehenden Probleme der Unbestimmbarkeit von Parametern umgangen, die automatisch dann entstehen, wenn der Datenfehler als Modellierungskriterium von bestimmten Modellparametern unbeeinflußt bleibt (Datenlücken). Solche Bedingungen können auf verschiedene Art gestellt werden. Sie sollten sich jedoch möglichst als lineare Gleichungen der Modellparameter p formulieren lassen. Das Gewicht $\lambda > 0$, mit dem diese Bedingungen in die Modellberechnung eingehen, kann dabei im Vergleich zur Datenanpassung sehr gering sein.

$$\begin{array}{rcccl} z & = & Ap & + & \varepsilon_d \\ \lambda c & = & \lambda B p & + & \lambda \varepsilon_r \end{array} \tag{12}$$

Mit $\lambda \to 0$ wird das Minimum für den mittleren quadratischen Datenfehler erzielt und für $\lambda \to \infty$ wird für den mittleren quadratischen Glättungsfehler $\bar{\varepsilon}_r$ das Minimum ohne Rücksicht auf Datenfehler erreicht. (Mit der Bezeichnung „Glättungsfehler“ werden jegliche Formen der Verletzung von Zusatzforderungen in (12) zusammengefaßt, auch wenn sich um andere Regularisierungesvarianten handelt.) Beide Fehleranteile sind stark vom jeweiligen Datensatz abhängig. Das Gewicht λ wurde zunächst für alle Regularisierungsbedingungen einheitlich benutzt. Die Auswirkung von λ ist in Abb.(1,2) ersichtlich.
Wenn es dafür Anlaß gibt, kann jedoch für jede einzelne Regularisierungsbedingung ein spezielles λ benutzt werden, d.h.

$$\begin{array}{rcccl} z & = & Ap & + & \varepsilon_d \\ \lambda_1 c_1 & = & \lambda_1 B_{1.} p & + & \lambda_1 \varepsilon_{r1} \\ & \cdots & & & \\ \lambda_i c_i & = & \lambda_i B_{i.} p & + & \lambda_i \varepsilon_{ri} \\ & \cdots & & & \\ \lambda_M c_M & = & \lambda_M B_{M.} p & + & \lambda_M \varepsilon_{rM} \end{array} \tag{13}$$

Auch der theoretisch denkbare Fall, daß die Daten selbst unterschiedlich vertrauenswürdig

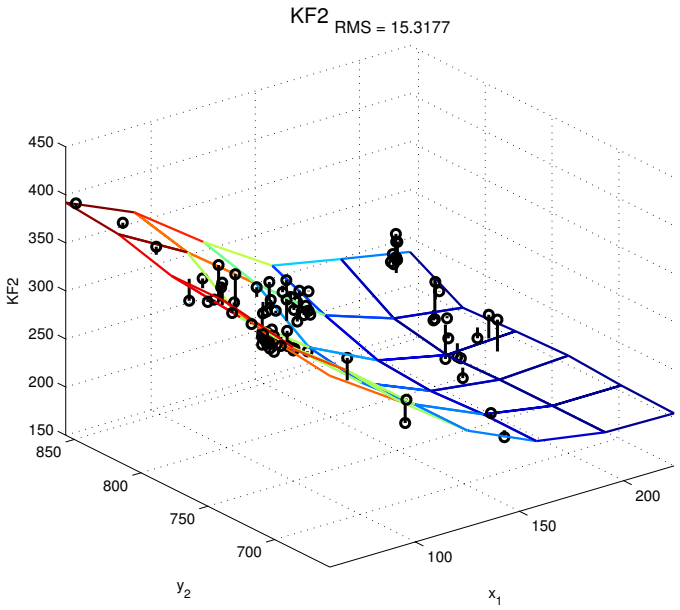


Abbildung 1: SGR-Modell (2-dim.) mit mittlerer Glättungsbedingung $\lambda = 1$

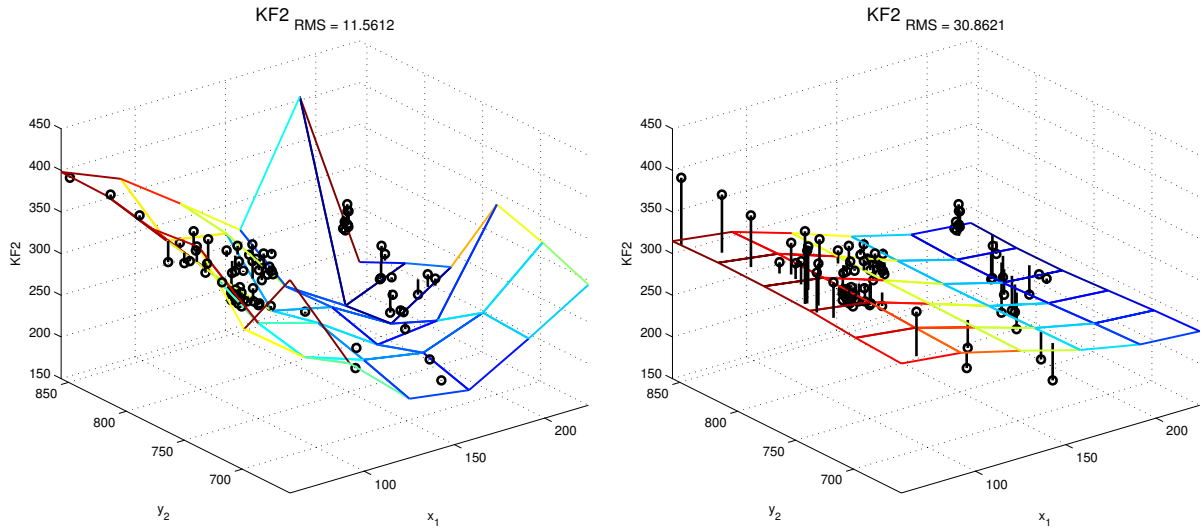


Abbildung 2: SGR-Modell (2-dim) mit minimaler bzw. maximaler Glättungsforderung

sind, kann durch Gewichtsfaktoren berücksichtigt werden:

$$
\begin{array}{rcrcl}
\omega_1 z & = & \omega_1 A_{1.} p & + & \omega_1 \varepsilon_{d1} \\
 & \cdots & & & \\
\omega_k z & = & \omega_k A_{i.} p & + & \omega_k \varepsilon_{dk} \\
 & \cdots & & & \\
\omega_N z & = & \omega_N A_{m.} p & + & \omega_N \varepsilon_{dN} \\
\lambda_1 c_1 & = & \lambda_1 B_{1.} p & + & \lambda_1 \varepsilon_{r1} \\
 & \cdots & & & \\
\lambda_i c_i & = & \lambda_i B_{i.} p & + & \lambda_i \varepsilon_{ri} \\
 & \cdots & & & \\
\lambda_M c_M & = & \lambda_M B_{M.} p & + & \lambda_M \varepsilon_{rM}
\end{array}
\tag{14}
$$

Natürlich sollte einer der Faktoren λ, ω fest (z.B. $\omega_1 = 1$) gesetzt werden.
Nach der Methode der kleinsten Quadrate wird (Lösbarkeit vorausgesetzt) der Parametersatz p bestimmt, für den

$$\sum_{i=1}^{N} \omega_i^2 \varepsilon_{di}^2 + \sum_{i=1}^{M} \lambda_i^2 \varepsilon_{ri}^2 \rightarrow min \tag{15}$$

erfüllt ist. Da mit ε_{dk} die tatsächlichen Datenfehler und mit ε_{ri} die Verletzungen der Zusatzforderungen bezeichnet wurden, ergibt sich hieraus die spezifische Gewichtung der Daten untereinander und gegenüber den Zusatzforderungen.
Zunächst wird jedoch von der vereinfachten Grundform (12) ausgegangen.

3.2 Globale Regularisierung

Wenn eine Modellform (quasi als Notbehelf) vorhanden ist, so kann diese für alle Modellparameter zur Regularisierung verwendet werden. Sei c_i der Ersatzwert für p_i, so kann mit $B = I$ das Problem regularisiert werden.
Auch Modellinformationen bzw. spezielle Forderungen, die nicht unbedingt im gesamten Raster und auch nicht unbedingt auf den Rasterpunkten formuliert werden können, können auf diese Weise berücksichtigt werden, wobei ihre Eignung zur Regularisierung, d.h. zur Erzielung der Lösbarkeit des Gleichungssystems (12) entweder geprüft bzw. durch weitere Bedingungen ergänzt werden sollte.

3.3 Lokale Regularisierung

Wenn keine Zusatzinformationen über das Modell vorliegen, kann die Regularisierung durch lokale Forderungen an die Glattheit des Modells erzeugt werden. Dazu wird jeder Gitterpunkt verglichen mit dem Wert, der sich aus den Gitterpunkten seiner Umgebung ergibt, wenn man aus diesen Punkten mit einem einfachen Regressionsmodell interpoliert. Die Größe der Umgebung und der Grad des lokalen Regressionsmodells bestimmen die damit erhobene Glattheitsforderung. Prinzipiell kann dies für jeden Parameter (Gitterpunkt) unterschiedlich formuliert werden, wie auch das Gewicht dieser Forderung generell für jeden Parameter anders sein kann. Dies ist sicher kaum im Vorfeld bestimmbar, jedoch sind mögliche Unterschiede für verschiedene Dimensionen und für Randpunkte des Gitters durchaus interessant.
Für diesen Fall gilt in der Forderung (12) $c = 0$ und die Matrix B hängt nur vom Gitter (und der gewählten Interpolationsvariante) ab, d.h. sie ist vom Datensatz unabhängig.
Es ist noch festzulegen, welche Punkte die Umgebung beschreiben und wie die Interpolation erfolgen soll (Polynomgrad {0,1,2}). Werden nur Nachbarn betrachtet, die sich nur in genau einer Dimension vom Bezugswert unterscheiden (also Nachbarn auf einer Gitterlinie sind), so wurde dieses Verfahren *axial* genannt. Wird quasi ein *Hyperquader* einbezogen, d.h. alle Punkte, die in allen Koordinaten einen festgelegten Abstand nicht übersteigen, so ergibt sich das *volum*-Verfahren. Die Anzahl der Gitterpunkte m, die (beidseitig) berücksichtigt wird, kann ebenfalls festgelegt werden, jedoch muß insgesamt gewährleistet sein, daß die lokale Regression möglich ist, d.h. ausreichend linear unabhängige Stützpunkte existieren. In Abb.3 werden links die Bestimmung des Glattheitsfehlers beim 1-dim Modell mit linearer Interpol. und einfacher Umgebung verdeutlicht und rechts die beiden Umgebungsvarianten für 2-dim. Modelle angegeben.
Die Möglichkeiten, für jeden Gitterpunkt die Glattheitsforderungen separat wählen zu können, erlauben z.B. in Abhängigkeit von der Besetztheit des Definitionsbereiches die Interpolationsordnung zu variieren.

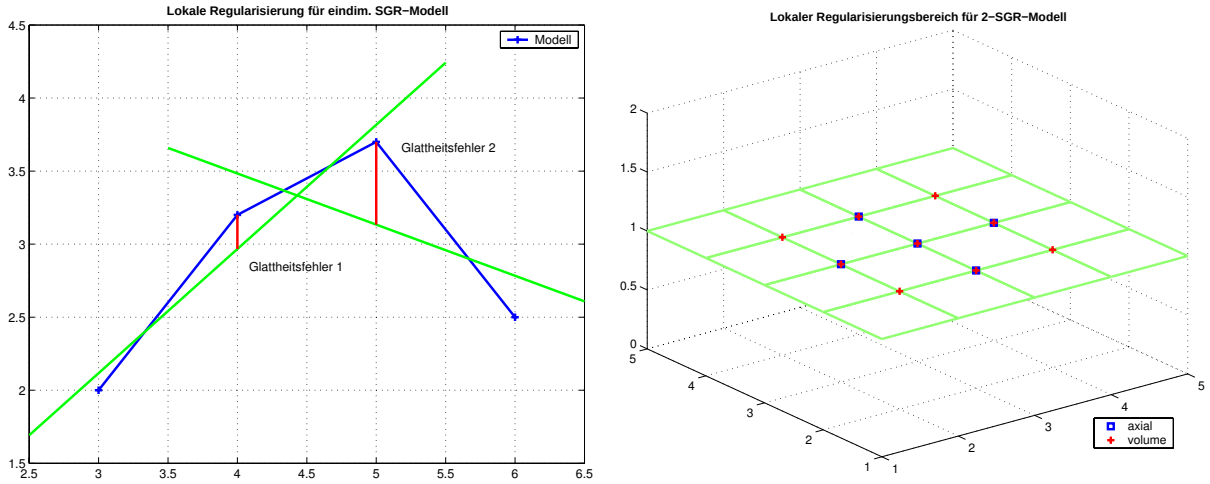


Abbildung 3: Prinzip der lokalen Regularisierung

Für die Randpunkte des Rasters sind ebenfalls besondere Vorkehrungen zu treffen.
Die Frage danach, welche Werte λ für die Regularisierung verwendet werden sollten, kann nicht problemunabhängig beantwortet werden. Hierzu können Fehlermaße (Datenfehler, Glattheitsmaße) und Visualisierungen herangezogen werden.

3.4 Modellbildung für SGR/C

Da das direkte Verfahren für diese Modellform (lineare Regressionsgleichungen wie bei SGR) nicht für alle Parameter R_*, q^* gleichzeitig möglich ist, wird ein 2-stufiges iteratives Verfahren zur Lösung von (10) vorgeschlagen. In einem ersten Schritt werden für die Kennlinien q Startannahmen (linear) gemacht und damit ein Parametersatz R bestimmt. Nun werden im zweiten Schritt diese Parameter R konstant angenommen. Damit erhält man eine neue Schätzung für q. Nach der Normierung entsprechend (8) steht eine erste Näherung zur Verfügung, die für eine weitere Berechnung von R verwendet werden kann. Beide Schritte entsprechen der Modellbildung beim SGR-Modell einschließlich der möglichen und notwendigen Regularisierungsbedingungen.
Die Konvergenz dieses Verfahren hängt sicher von der Gültigkeit der mit der Modellform verbundenen Forderungen an den zu beschreibenden nichtlinearen multivariaten Zusammenhang ab. In zahlreichen Beispielen konnte jedoch Konvergenz und gute Datenrepräsentation erzielt werden. Für einen 3-dim. Zusammenhang ist in Abb.4 die Modellierung mit SGR und SGR/C gegenübergestellt.
Der Erfolg dieser Modellbildung ist aber auch abhängig von der Auswahl der Dimensionen für das Basismodell (SGR) ab. Es handelt sich dabei um die Frage der optimalen Auswahl aus einer diskreten Menge, die bereits für die Merkmalsauswahl bei Klassifikationsproblemen untersucht wurde [3]. Eine Verallgemeinerung des methodischen Vorgehens gestattet die Anwendung auch für die Auswahl der Basisdimensionen beim SGR/C-Modell.

3.5 Iterative Verbesserung der Glattheit

Das Regressionsprinzip minimiert den mittleren quadratischen Fehler. Für Modelle mit vielen Parametern kann deshalb bei einzelnen Parametern (Gitterpunkten) eine erhebliche Spitze im Modell auftreten. Da der Regularisierungsparameter auch getrennt für jede Zeile, d.h.

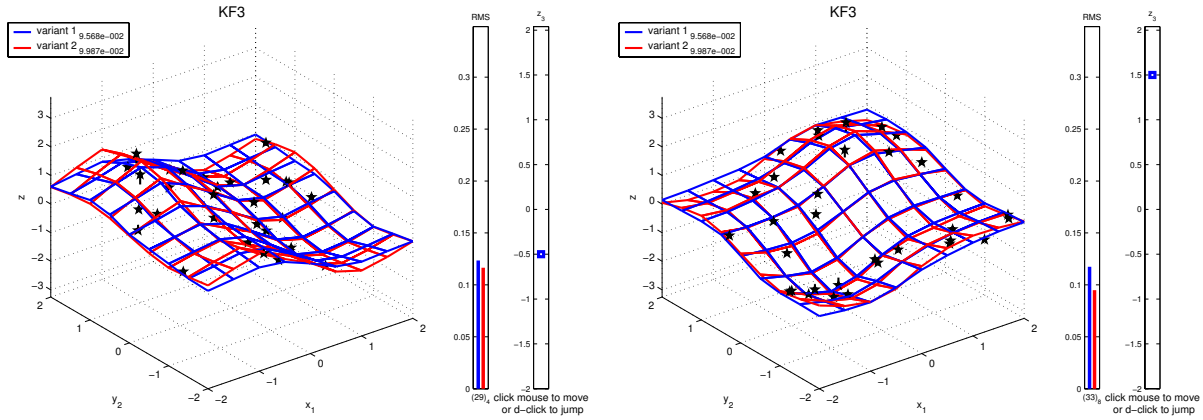


Abbildung 4: Beispiel für 3-dim SGR (Var. 1) und SGR/C(Var. 2) für 2 Werte von z

jeden Parameter anders gesetzt werden kann, wurde durch eine Erfassung der Restfehler (Glattheitsverletzung) für extreme Gitterpunkte der Regularisierungsparameter erhöht. Die Wiederholung der Rechnung ergibt einen leichten Anstieg der Datenfehler, der aber durch die Beseitigung der Spitzen gerechtfertigt werden kann. In Abb.5 ist der erzielte Effekt sichtbar. Auf diesem Weg wurde für SGR- und SGR/C-Modelle eine Modellverbesserung erzielt, die

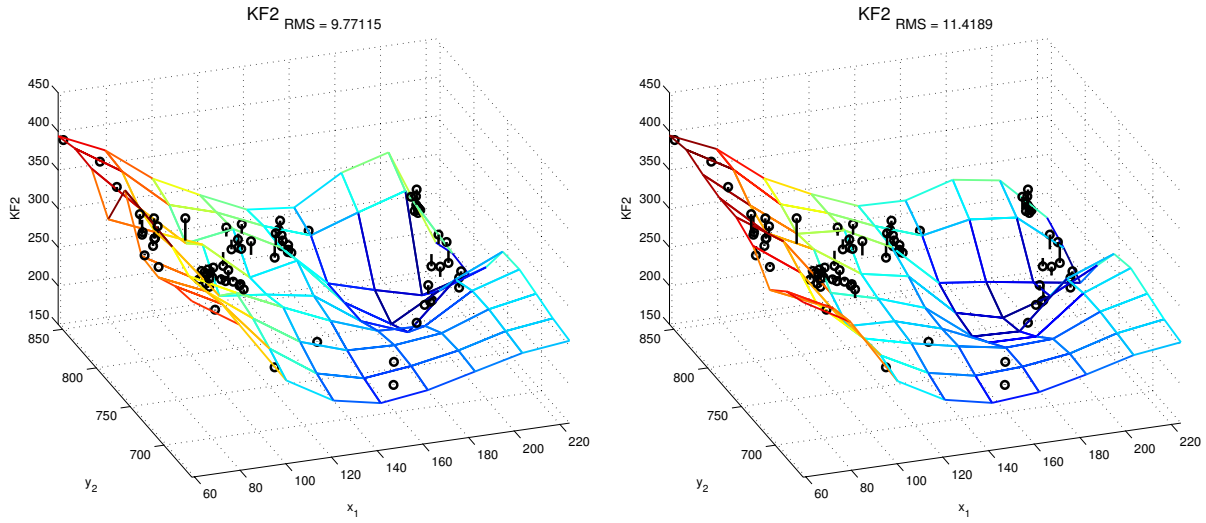


Abbildung 5: Beispiel für iterative Glattheitsverbesserung (links ohne, rechts mit Iteration)

durch einfache (gleichmäßige) Erhöhung des Regularisierungsparameters nicht möglich ist, da der Datenfehler erheblich ansteigen würde. Für SGR/C-Modelle wirkt sich die Überlagerung der iterativen Glattheitsverbesserung mit dem ohnehin iterativen Modellentwurf negativ auf die Konvergenz aus.

3.6 Variation der Interpolationsordnung

Die Untersuchungen haben ergeben, daß sich höhere Ordnungen bei der Regularisierung dann günstig auswirken, wenn die Daten den Raum ausfüllen. In unbesetzten Gebieten, insbesondere am Rand, ist die Interpolation 0-ter Ordnung, d.h. die Benutzung des Mittelwertes, meist günstiger.

3.7 Weitere Variationsmöglichkeiten

Für besondere Betrachtungen werden Modelle gesucht, die als Grenzfelder nur positive bzw. nur negative (möglichst kleine) Datenfehler liefern, aber dennoch „glatt“ sind.

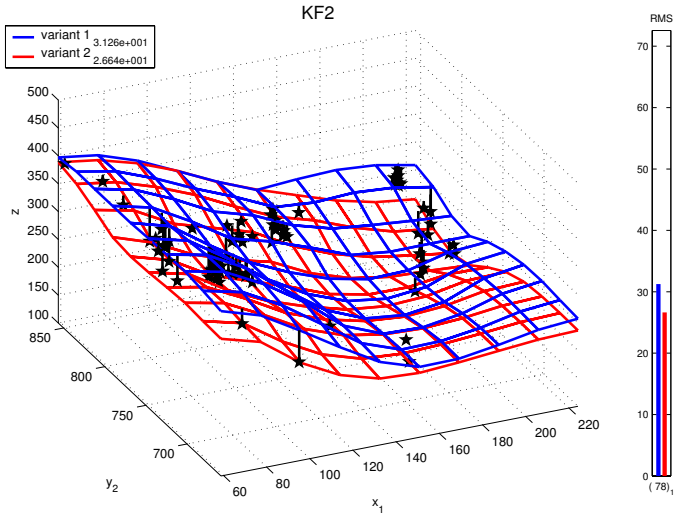


Abbildung 6: Beispiel für 3-dim SGR-Grenzfeldmodelle

4 Rastermodifikation

Die echte Verfeinerung der Rastereinteilung ist mit erheblichem Anwachsen der Parameteranzahl und damit des Realisierungsaufwandes verbunden. Dagegen ist die Veränderung der Gitterpositionen nicht mit Mehraufwand verbunden. Zur Lösung dieser Aufgabe wird folgender Weg vorgeschlagen:
Mit einem Ausgangsraster (gleichmäßig verteilt oder nach Vorgabe) wird ein SGR-Modell angepaßt. Auch ein anderes Modell (globale Regression o.ä.) kann dazu verwendet werden.
Die Neueinteilung der Gitterpunkte (gleicher Anzahl) erfolgt nun bei gleichen Randwerten so, daß die Summe der mit dem Startmodell gefundenen Restfehler in jedem Gitterintervall in etwa gleich ist. Dazu werden alle Datenpunkte einbezogen, die in der betrachteten Dimension in dem Gitterintervall liegen. Für den häufigen Fall von schlecht besetzten Definitionsbereichen entartet diese Aufgabenstellung. Durch vorgebbare maximale Zoomfaktoren für die Größenänderung der Intervalle (in beide Richtungen) läßt sich eine durchgängige Methode beschreiben. Das nachfolgend beschriebene Verfahren betrachtet jede Dimension getrennt voneinander und ist somit von deren Anzahl unabhängig:

Bezeichnung	Bedeutung	Formel/Bedingungen
$K+1$	Anzahl Punkte (K Intervalle)	$K \geq 2$
z_{in}	Zoomfaktor Verfeinerung	$z_{in} > 1$ ganzzahlig
z_{out}	Zoomfaktor Vergröberung	$z_{out} > 1$
N	Anzahl Feinintervalle	$N = Kz_{in}$
f_i	quadr. Fehlersumme im i-ten Feinintervall	
F	Gesamtfehlersumme	$F = \sum_{i=1}^{N} f_i$
b	Hilfsgröße zur Regularisierung	$\sum_{i=1}^{N}(f_i + b) = F + Nb$
n_{max}	Max. Anzahl von Feinintervallen innerhalb	$n_{max} = z_{in}z_{out}$
$\hat{f}$	max. Fehler in Feinintervall	$\hat{f} = \max_i f_i$
n_k	Index des letzten Feinintervalls im Intervall k	$n_K = N\,; n_0 = 0$

Gleichmäßigkeit der quadratischen Fehlersummen (auch für leere Intervalle):

$$\sum_{i=n_{k-1}+1}^{n_k} (f_i + b) \approx \frac{F + Nb}{K} = n_{max}b \tag{16}$$

Extremes Feinintervall darf Schwelle nicht überschreiten:

$$\hat{f} + b < n_{max}b \tag{17}$$

Aus den Forderungen (16,17) kann einerseits die Hilfsgröße b ermittelt und außerdem können Bedingungen für die Zoomfaktoren gewonnen werden:

$$b = \frac{F}{Kz_{in}(z_{out} - 1)}$$

Für den Fall $F < \hat{f}K$ erhält man die Einschränkung:

$$z_{out} \leq = 1 + \frac{(z_{in} - 1)F}{z_{in}(\hat{f}K - F)} \tag{18}$$

Auf dieser Basis ergibt (dimensionsweise) sich mit 16 eine Neuaufteilung der Gitterpunkte , die an Beispielen auch zu verbesserten Modellierungsergebnissen geführt hat (Abb.7).

5 Rastersubstruktur

Für den Entwurf einer mehrstufigen Struktur kann es verschiedene Gründe geben:

- Der zu modellierende Zusammenhang besitzt sehr unterschiedliche Dynamik, so daß man in Teilbereichen gern mit erhöhter Auflösung arbeiten möchte, ohne die generell exponentiell wachsende Parameteranzahl in Kauf nehmen zu müssen.
- Die Rechnerkapazität (Zeit und Speicher) steht einem (unstrukturierte) Modellentwurf entgegen. Eine Modellierung in getrennten Abschnitten könnte bei Lösung der Anschlußbedingungen Abhilfe schaffen.
- Um unterschiedliche Modellformen (verschiedene SGR-Varianten oder auch andere Modelle) zu kombinieren, sollte die Modellierung in Teilbereichen (SGR) möglich sein.

Gegenargument ist die damit verbundene Komplexität des Modells (Hardware-Realisierung) und die eventuellen Schwierigkeiten, die Anschlußbedingungen problemgerecht zu erfüllen.

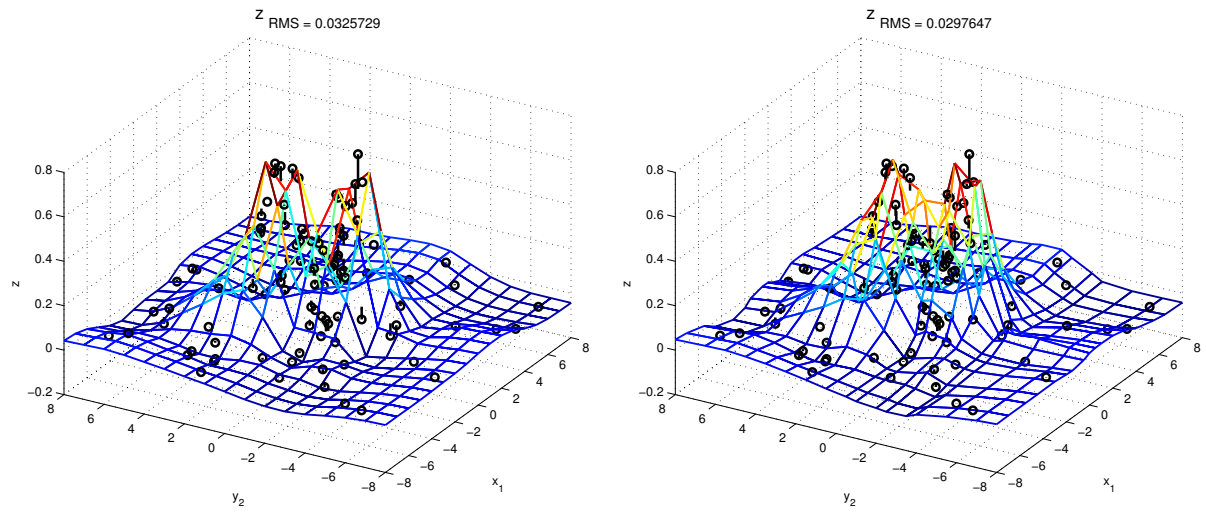


Abbildung 7: Beispiel für 2-dim SGR ohne (links) und mit (rechts) Rastermodifikation

5.1 Anschlußmodellierung ohne Rasteränderung

Wenn der Definitionsbereich (das gesamte Rasterfeld) in Grobsegmente lückenlos aufgeteilt wird, kann in jedem der Segmente eine getrennte Modellierung erfolgen. Da (als globale Vorgabe) bereits berechnete Werte auf dem Rand einbezogen werden können und deren „Durchgriff“ mittels Koppelfaktoren festgelegt werden kann, ist es von eben diesen Faktoren und auch von der Reihenfolge der Berechnung abhängig, inwieweit das Modellierungsergebnis einer in Parameteranzahl und Auflösung gleichwertigen unstrukturierten Modellierung nahe kommt. Das Resultat kann dann selbst als unstrukturiertes Modell benutzt werden. Wegen der progressiv mit der Modellgröße steigenden Rechenzeit für den Modellentwurf ist dieses Vorgehen u.U. von praktischem Interesse, obwohl es sich nur um eine additive Aufteilung aller Parameter (innere Randpunkte mehrfach) handelt.

5.2 Anschlußmodellierung mit Rasteränderung

Das oben beschriebene Vorgehen ermöglicht es auch, Teilbereiche des Rasterfeldes mit feinerem Raster zu modellieren. Um dies zu nutzen, sind natürlich Informationen über die Dynamik des Gesamtzusammenhangs in den Teilbereichen des Definitonsbereiches erforderlich. Die Untersuchung der Modellierungsfehler mit einem Startmodell kann dazu dienen. Für die Modellanwendung ist dann natürlich eine entsprechend strukturierte Modellform erforderlich. Die bereits genannten Probleme für die Anschlußbedingungen gelten in gleicher Weise.

5.3 Zusatzmodellierung

Wenn in kleinen Teilbereichen des Rasterfeldes die Dynamik des Zusammenhangs nicht erfaßt werden kann (Rastereinteilung zu grob), so ist es sinnvoll, in diesen Bereichen zusätzlich eine verfeinerte Modellierung durchzuführen. Das Startmodell wird dann innerhalb dieser Teile „außer Kraft“ gesetzt und liefert außerdem die erforderlichen Anschlußbedingungen. Die Feinheit des Globalmodells und die Anzahl und der Umfang der Feinmodelle sind Entscheidungen bei einem solchen Vorgehen, die problemabhängig die Vor- und Nachteile bestimmen. Die Abb.8 zeigt ein Beispiel, wie durch Zusatzmodellierung die Anzahl der Modellparameter von 1089 (generell feines Raster) auf 370 reduziert werden kann.

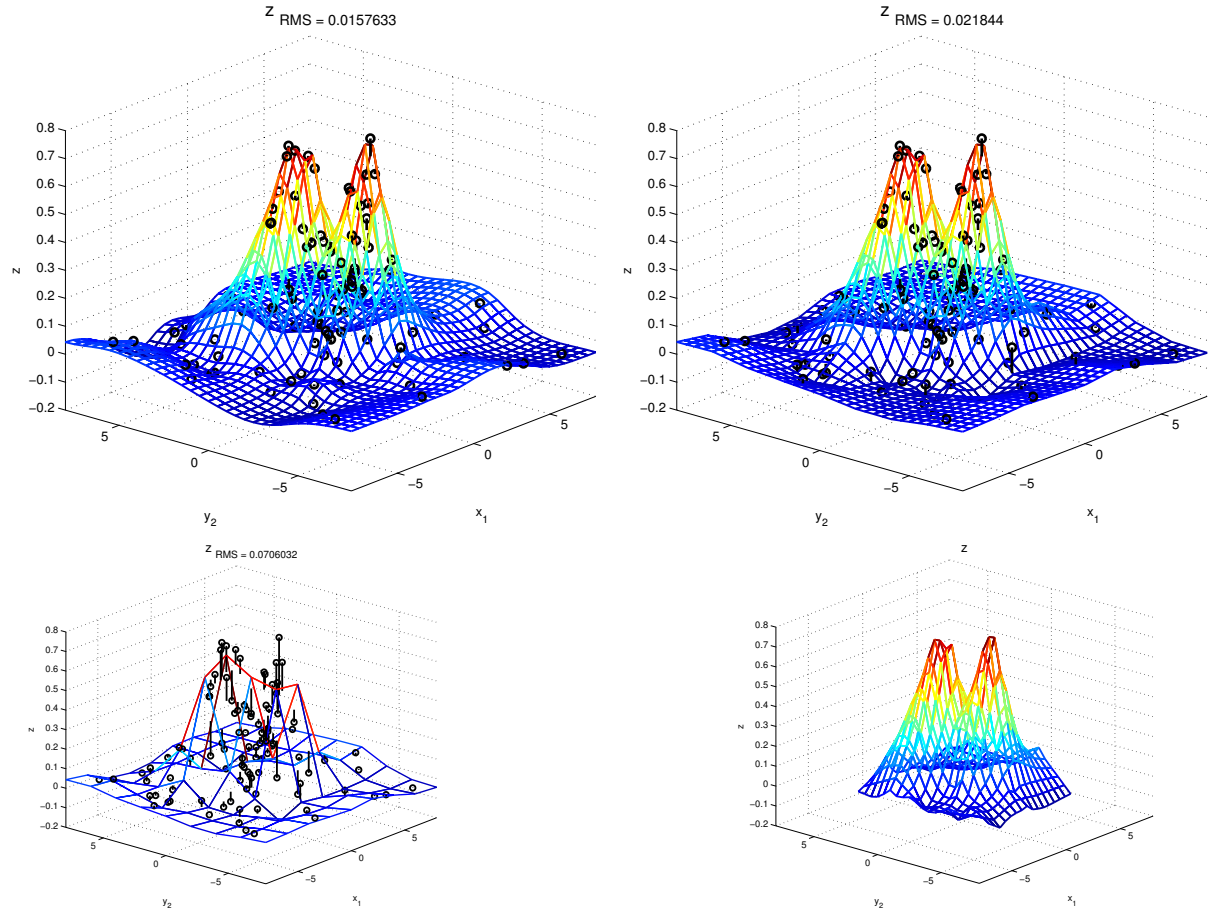


Abbildung 8: Bsp. Rastersubstruktur (oben: links Feinmodell, rechts struktur. Modell, unten: Grob- und Feinkomponente)

5.4 Strukturnotation

Die Verwendung von strukturierten Modellen sollte mit möglichst wenig Zusatzaufwand verbunden sein. Durch eine Modelliste (in Matlab z.B. als *cell array*) könnten folgende Eigenschaften realisiert werden:

- Unstrukturierte Modelle sind als Trivialfall enthalten.
- Unterschiedliche Modellformen (SGR, SGR/C, Regressionsmodelle) können miteinander kombiniert werden.
- Modellergänzungen leicht möglich.
- Priorität der Modellgültigkeit durch Reihenfolge (detaillierte Submodelle zuerst).
- Es werden nur die Modelle berechnet, die aktuell aktiv sind.

6 Zusammenfassung und Ausblick

Die vorgestellten Modellierungsformen wurden als eigenständige Formen der nichtlinearen (statischen) Beschreibung entwickelt. Ihre Anwendung als alternativer Zugang zur Lösung von

dafür geeigneten Aufgabenstellungen, die mit den Mitteln von Fuzzy Control bzw. Neuronalen Netzen bearbeitet werden, erscheint hoffnungsvoll, da aus der Sicht des Autors der Wunsch nach Glattheit trotz unterkritischer Datenbasen ein wesentliches Argument für den Einsatz Intelligenter Technologien ist. Der Entwurf von alternativen Problemlösungen für konkrete Aufgabenstellungen ist Zielstellung weiterer Untersuchungen.

Literatur

[1] Nelles,O.; Fink,A.: Tool zur Optimierung von Rasterkennfeldern. UC Berkeley, TU Darmstadt (Internet TU Darmstadt)

[2] Brown,M.; Harris,C.J.:
The B-spline Neurocontroller.
In: Rogers,E.(Hrsg.); Li,Y.(Hrsg.): Parallel Processing in a Control System Environment, Prentice-Hall, 1993, Kapitel 5

[3] Priber, U.:
Ein Verfahren zur Merkmalsreduktion für unscharfe Klassifikatoren.
Diss. A, TH Karl–Marx–Stadt(Chemnitz) 1989