

F. Hoffmann, E. Hüllermeier (Hrsg.)



Proceedings

19. Workshop Computational Intelligence

Dortmund,
2. - 4. Dezember 2009

Schriftenreihe des
Instituts für Angewandte Informatik / Automatisierungstechnik
am Karlsruher Institut für Technologie
Band 29

Eine Übersicht über alle bisher in dieser Schriftenreihe erschienenen Bände
finden Sie am Ende des Buchs.

Impressum

Karlsruher Institut für Technologie (KIT)
KIT Scientific Publishing
Straße am Forum 2
D-76131 Karlsruhe
www.uvka.de

KIT – Universität des Landes Baden-Württemberg und nationales
Forschungszentrum in der Helmholtz-Gemeinschaft



Diese Veröffentlichung ist im Internet unter folgender Creative Commons-Lizenz
publiziert: <http://creativecommons.org/licenses/by-nc-nd/3.0/de/>

KIT Scientific Publishing 2009
Print on Demand

ISSN: 1614-5267
ISBN: 978-3-86644-434-8

Vorwort

Dieser Tagungsband enthält die Beiträge des 19. Workshops „Computational Intelligence“ des Fachausschusses 5.14 der VDI/VDE-Gesellschaft für Mess- und Automatisierungstechnik (GMA) und der Fachgruppe „Fuzzy-Systeme und Soft-Computing“ der Gesellschaft für Informatik (GI), der vom 2. – 4. Dezember 2009 im Haus Bommerholz bei Dortmund stattfindet.

Die Schwerpunkte sind Methoden, Anwendungen und Tools für

- Fuzzy-Systeme,
- Künstliche Neuronale Netze,
- Evolutionäre Algorithmen und
- Data-Mining-Verfahren

sowie der Methodenvergleich anhand von industriellen und Benchmark-Problemen.

Die Mitglieder des Programmkomitees sind Prof. Dr. H.-G. Beyer (FH Vorarlberg), Dr. F. Hoffmann (Technische Universität Dortmund), Prof. Dr. E. Hüllermeier (Universität Marburg), Prof. Dr. A. Kistner (Universität Stuttgart), Prof. Dr. F. Klawonn (FH Braunschweig/Wolfenbüttel), Prof. Dr. A. Kroll (Universität Kassel), PD Dr. R. Mikut (Forschungszentrum Karlsruhe GmbH) und PD Dr. Th. Runkler (Siemens AG).

Die Ergebnisse werden von Teilnehmern aus Hochschulen, Forschungseinrichtungen und der Industrie in einer offenen Atmosphäre 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 erhalten Sie unter

<http://www.rst.e-technik.uni-dortmund.de/cms/de/Veranstaltungen/GMA-Fachausschuss/index.html>

Die Herausgeber bedanken sich an dieser Stelle bei allen Autoren und Rednern sowie bei den Mitgliedern des Programmkomitees, die maßgeblich an der Vorbereitung des Workshops beteiligt waren.

Frank Hoffmann und Eyke Hüllermeier

Inhaltsverzeichnis

O. Kramer (TU Dortmund): Nadaraya-Watson Surrogate Models in Stochastic Optimization	1
R. Alshut, J. Legradi, L. Yang, U. Strähle, R. Mikut, M. Reischl (Forschungszentrum Karlsruhe): Robust Identification of Coagulated Zebrafish Eggs using Image Processing and Classification Techniques	9
R. Senge, E. Hüllermeier (Universität Marburg): Learning Pattern Tree Classifiers Using a Co-Evolutionary Algorithm	22
R. Mikut, U. Liebel (Forschungszentrum Karlsruhe): Visualisierungstechniken zum Information Retrieval für Textdokumente in der Biologie	34
T. Haag, P. Reuß, M. Hanss (Universität Stuttgart): An Approach to the Identification of Uncertain Surrogate Models for Complex Systems	50
P. Gerland, H. Schulte, A. Kroll (Universität Kassel, Bosch Rexroth AG) Vergleichende Untersuchung nichtlinearer Beobachterkonzepte zur Fehlerdiagnose am Beispiel hydrostatischer Fahrtriebe	61
W. Jakob, A. Quinte, K.-U. Stucky, W. Süß (Forschungszentrum Karlsruhe): Schnelles Rescheduling von Gridjobs mit Heuristiken und dem Evolutionären Algorithmus GLEAM	75
B. Hartmann, O. Nelles (Universität Siegen): Advantages of Hierarchical versus Flat Model Structures for High-Dimensional Mappings	87
O. Flasch, T. Bartz-Beielstein, P. Koch, W. Konen (Fachhochschule Köln): Genetic Programming Applied to Predictive Control in Environmental Engineering	101
N. Rosemann, A. Buschermöhle, W. Brockmann (Universität Osnabrück): Beschleunigung der Selbstoptimierung durch Selbstsimulation	114
S. Schäfer, U. Lehmann, J. Krone, M. Schneider, H. Brenig (FH Südwestfalen): Konzeptionierung eines Green-Supercomputer-Clusters zur Berechnung von Künstlichen Neuronalen Netzen mit MATLAB	129

S. Turck, M. Schneider, U. Lehmann, J. Krone, Ch. Klimmek (FH Südwestfalen): KnowledgeBase-System aus CI-basierten Prognosemodellen zur Machbarkeitsprüfung von Blechumformprozessen	135
J. Braun, J. Krettek, F. Hoffmann, T. Bertram (TU Dortmund) Automatisierter Entwurf von Hydraulikventilreglern: Eine Evolutionsstrategie für Optimierungsprobleme mit variabler Parameterzahl	141
F. Svaricek, J. Gogolin, V. Härtel (Universität der Bundeswehr München): Adaptive Fuzzy-Regelung einer Gummispritzpresse	156
Ch. Kühnert, Th. Bernard (Fraunhofer Institut für Informations- und Datenverarbeitung): Optimization and Online-Monitoring in industrial batch processes using Data- Mining methods	170
S. Buhl, B. Neumann, M. Schneider, U. Lehmann, E. Eisenbarth, T. Weißbach (FH Südwestfalen): Automatische Segmentierung von mikroskopischen Zellbildern für die Biokompatibilitätsprüfung mit Hilfe eines Künstlich Neuronalen Netzes	181
Ch. Borgelt (European Center for Soft Computing, Mieres, Spanien) Graph Mining: An Overview	189
V. Gottfried, U. Lehmann, U. Klug, B. Neumann, M. Schneider, J. Brenig, S. Buhl (FH Südwestfalen): Simulation von Neuronalen Netzen für E-Learning	204
Ch. Arnold, B. Cuno (Hochschule Fulda): Entscheidungshilfe zur Umgebungsklimatisierung von Kulturgütern mit Methoden der Fuzzy-Theorie	209
K. Narayanan, F. Posada, F. Hoffmann, T. Bertram (TU Dortmund): Imitation Learning for Visual Robotic Behaviors	221
Th. Fober and E. Hüllermeier (Universität Marburg): Multiple Geometrical Alignments for the Analysis of Protein Active Sites	237
B. Naujoks, H. Trautmann, T. Wagner (TU Dortmund): OCD: Ein Online Konvergenzkriterium für die evolutionäre, mehrkriterielle Optimierung	256

Nadaraya-Watson Surrogate Models in Stochastic Optimization

Oliver Kramer

Technische Universität Dortmund, Lehrstuhl XI

Otto-Hahn-Str. 14

Tel.: (0231) 755-7708

Fax: (0231) 755-7740

E-Mail: oliver.kramer@tu-dortmund.de

Abstract

Objective function surrogate models reduce the number of fitness function evaluations in time consuming optimization scenarios. From previously evaluated solutions a regression method can be trained that builds the surrogate model to replace the original fitness function. This paper investigates the use of the Nadaraya-Watson estimator as objective function surrogate. The Nadaraya-Watson estimator is based on kernel regression and kernel density estimation. It is used during optimization with the covariance matrix self-adaptation evolution strategy, the self-adaptive variant of the CMA-ES. An experimental analysis evaluates the meta-model assisted optimization process on a set of test functions and investigates model sizes and the balance between objective function evaluations on the real function and on the surrogate.

1 Introduction

Meta-modeling is a well known technique and has successfully been applied in engineering and time consuming optimization tasks in the past [1, 2, 3, 4]. Various regression methods can be used as function surrogates. This work investigates the use of the Nadaraya-Watson estimator [5] as objective function surrogate as a first step towards a line of research on kernel regression based meta-modeling techniques. The Nadaraya-Watson kernel regression estimator is based on two concepts: kernel regression and kernel density estimation. It is a non-parametric approach, i.e. no training is necessary, but the data itself represents the model. To apply kernel regression to stochastic optimization various questions have to be answered: what is a proper choice for the complexity of the regression model specified by bandwidth h , how many data samples are necessary to estimate an accurate objective function surrogate and how many fitness function evaluations can be replaced by meta-model evaluation in order to improve the optimization process. As stochastic optimizer we use the covariance matrix self-adaptation evolution strategy (CMSA-ES) that has been introduced by Beyer [6] as a self-adaptive variant of the covariance matrix adaptation evolution strategy (CMA-ES).

Meta-modeling of the objective function is a known technique to reduce the number of function evaluations. The idea of meta-modeling is the following: a regression technique predicts the fitness values based on previous function evaluations. Basically, two approaches can be distinguished in meta-model based stochastic optimization: first, alternately using the real objective function in one generation and the surrogate model in the following. Second, surrogate model and original objective function are used simultaneously in one generation, but in a certain balance. The whole bandwidth of methods for

regression can be applied as meta-models. Giannakoglou *et al.* [7] have applied radial basis function networks as meta-models in evolutionary optimization. Emmerich *et al.* [1] use Kriging and with a local error calculation that estimates the meta-model reliability using an evolution strategy with small populations. Ulmer *et al.* [2] use Gaussian processes as surrogate models in evolution strategies based optimization. The approach contains a preselection criterion that controls the tradeoff between exploitation and exploration using the probabilistic interpretation of the Gaussian process model. Kern *et al.* [3] use locally weighted regression as meta-model for optimization with the CMA-ES. As the latter is an elitist approach they propose a model-assisted ranking procedure. Their experiments show a significant speedup, in particular with full quadratic local models. A recent overview over meta-models is given by Forrester *et al.* [4] who review radial basis functions, Kriging, and support vector regression in particular for optimization tasks in aerospace design.

Section 2 shortly repeats the basis of the CMSA-ES, while Section 3 introduces the Nadaraya-Watson estimator. An analysis of the Nadaraya-Watson estimator as objective function surrogate in combination with the CMSA-ES is given in Section 4, where in particular the balance between objective function and surrogate as well as adequate model sizes are analyzed.

2 Covariance Matrix Self-Adaptation

The CMSA-ES has recently been proposed by Beyer *et al.* [6]. It is a variant of the CMA-ES by Hansen and Ostermeier [8, 9]. Although the CMA-ES turned out to be the state-of-the-art evolutionary optimization algorithm for many years, the original algorithm suffers from free parameters with little theoretical guidance [6]. The CMSA-ES is based on a self-adaptive step control of these parameters: After initialization, λ candidate solutions are produced. First, with the help of the global self-adaptive step size $\vec{\sigma} = (\sigma_1, \dots, \sigma_N)$ each individual c_i gets a log-normal mutated size:

$$\sigma_i = \langle \sigma \rangle \cdot e^{\tau \mathcal{N}_i(0,1)} \quad (1)$$

depending on a global step size $\langle \sigma \rangle$ that is the average of the step sizes from the μ best solutions of the previous generations¹. Then, correlated random directions \mathbf{s}_i are generated with the help of the covariance matrix \mathbf{C} by multiplication of its Cholesky decomposition $\sqrt{\mathbf{C}}$ with the standard normal vector $\mathcal{N}_i(\mathbf{0}, \mathbf{1})$:

$$\mathbf{s}_i = \sqrt{\mathbf{C}} \mathcal{N}_i(\mathbf{0}, \mathbf{1}) \quad (2)$$

This random direction is scaled in length with the help of the self-adaptive step size σ_i :

$$\mathbf{z}_i = \sigma_i \mathbf{s}_i \quad (3)$$

The resulting vector \mathbf{z}_i is added to the global parent \mathbf{y} :

$$\mathbf{y}_i = \mathbf{y} + \mathbf{z}_i \quad (4)$$

Finally, quality f_i of solution \mathbf{y}_i is evaluated. When λ offspring solutions have been generated, the μ best solutions are selected and their components \mathbf{z}_i and σ_i are recombined.

¹denoted by $\langle \cdot \rangle$

Beyer proposes to apply global recombination, i.e. the arithmetic mean of each parameter is calculated. The outer product \mathbf{ss}^T of the search directions is computed for each of the μ parental solutions and the matrices are averaged afterwards. Important is the final covariance matrix update step:

$$\mathbf{C} = \left(1 - \frac{1}{\tau_c}\right) \mathbf{C} + \frac{1}{\tau_c} \langle \mathbf{ss}^T \rangle \quad (5)$$

Parameter τ_c balances between the last covariance matrix \mathbf{C} and the outer product of the search direction of the μ best solutions. Beyer recommends:

$$\tau_c = \frac{N(N+1)}{2\mu}. \quad (6)$$

These steps are iterated until a termination condition is satisfied. The CMSA-ES combines the self-adaptive step size control with a simultaneous update of the covariance matrix. Initially, the covariance matrix \mathbf{C} is chosen as the identity matrix $\mathbf{C} = \mathbf{I}$. The learning parameter τ defines the mutation strength of the step sizes σ_i . According to Beyer [6] the covariance matrix adapts to "spherical conditions". For the *Sphere* function the optimal learning parameter is $\tau = \frac{1}{\sqrt{2N}}$ [10]. Smaller values increase the approximation speed at the beginning of the search. The following section introduces the Nadaraya-Watson estimator as regression technique.

3 The Nadaraya-Watson Estimator

The Nadaraya-Watson estimator is based on two concepts: kernel regression and kernel density estimation. Let $\mathcal{D} = \{x_1, \dots, x_N\}$ be a data set, i.e. independent and identically distributed samples of a random variable. We seek for the density function $p(x)$ of the data distribution. Distribution estimators can be classified into parametric and non-parametric approaches. Kernel density estimation is a non-parametric approach. It is based on a kernel density function $p(x)$. The kernel density approximation of the probability density function is:

$$p(x) = \frac{1}{Nh} \sum_i^N K\left(\frac{x - x_i}{h}\right) \quad (7)$$

with kernel K and the so-called bandwidth parameter h . Kernel regression is the second ingredient of the Nadaraya-Watson estimator. The problem of finding a statistical model for matching input data $\mathbf{x} \in \mathbb{R}^q$ to output data $\mathbf{y} \in \mathbb{R}^d$ can be expressed as a regression function. Input and output data may be stochastically disturbed and hence, can be expressed as realizations of random variables – X for input and Y for output data. For the sake of simplicity we assume the variables x and y to be univariate. Let $p(x, y) = (X = x, Y = y)$ be a given joint distribution. The optimal mapping $f(\cdot)^*$ is also known as regression function and can be expressed in terms of the joint and marginal distributions $p(x, y)$ and $p(x)$:

$$f^*(x) = \int yp(y|x)dy = \int y \frac{p(x, y)}{p(x)} dy. \quad (8)$$

We can express the joint distribution $p(x, y)$ with its kernel density estimate:

$$\hat{p}(x, y) = \frac{1}{N} \sum_{i=1}^N K_{h_x}(x - x_i) K_{h_y}(y - y_i) \quad (9)$$

with two kernel density functions K_{h_j} and bandwidth h_j for $j = x, y$ that do not have to be equal. This marginal density $\hat{p}(x)$ and the joint density $\hat{p}(x, y)$ inserted into Equation 7 yield:

$$\hat{f}(x) = \sum_i \frac{K_{h_x}(x - x_i)}{\sum_j K_{h_x}(x - x_j)} \int y K_{h_y}(y - y_i) dy \quad (10)$$

and the Nadaraya-Watson estimator [5]:

$$\hat{f}(x; h) = \sum_i y_i \frac{K_{h_x}(x - x_i)}{\sum_j K_{h_x}(x - x_j)} \quad (11)$$

with the bandwidth $h = h_x$. Bandwidth h_y is not relevant anymore. Of course, in praxis we want to apply the Nadaraya-Watson estimator to multivariate input data, i.e. for $q \geq 1$, and maybe also output data, $d \geq 1$, i.e. in multi-objective optimization. For this sake the smoothing parameter h has to be introduced to each dimension. It can be expressed with a diagonal matrix $\mathbf{H} = \text{diag}(h_1, h_2, \dots, h_d)$. With this bandwidth matrix we can express the multivariate Gaussian kernel function:

$$K_{\mathbf{H}}(\mathbf{x}) = \frac{1}{(2\pi)^{q/2} \det(\mathbf{H})} \exp\left(-\frac{1}{2} \|\mathbf{H}^{-1} \mathbf{x}\|^2\right). \quad (12)$$

With the scalars $K_{\mathbf{H}}(\cdot)$ the Nadaraya-Watson estimator can be written by weighting the contribution of a data sample \mathbf{x}_i to the estimate with its relative kernel density:

$$\mathbf{f}(\mathbf{x}; \mathbf{H}) = \sum_i y_i \frac{K_{\mathbf{H}}(\mathbf{x} - \mathbf{x}_i)}{\sum_j K_{\mathbf{H}}(\mathbf{x} - \mathbf{x}_j)}. \quad (13)$$

With this multivariate extension we can use the Nadaraya-Watson estimator as objective function surrogate in single or multi-objective optimization. The Nadaraya-Watson estimator is integrated into the CMSA-ES as follows: $\alpha\%$ of the λ offspring solutions are evaluated with the meta-model in each iteration, the rest on the original objective function. In the next Section 4.2 we investigate influence of α on the optimization process.

4 CMSA-ES and Nadaraya-Watson Surrogate Models

Now, we apply the Nadaraya-Watson estimator as objective function surrogate in stochastic optimization. Surrogate functions estimate the real objective function and are used to evaluate solutions to save original fitness function evaluations. The questions arise how to parameterize the regression model, i.e. how to choose the kernel bandwidth h that controls the smoothness of the function, what the proper balance between original and surrogate evaluations is, and how many points are necessary to build the model.

4.1 Balance Between Objective Function and Surrogate Model

Our first analysis concerns the balance between the evaluation of the objective function and the surrogate model. In each iteration a fraction of $\alpha\%$ of the λ solutions are evaluated on the surrogate model instead of on the real objective function. Which is the proper fraction α , can we observe a loss in performance if too many solutions are evaluated on

the meta-model? We tested a (25,100)-CMSA-ES on the two functions² *Doublesum* and *Rastrigin* with $N = 5$ dimensions. We use a model size of $m = 30$ and the recommended standard settings for τ as well as a constant bandwidth of $h = 0.1$. Table 1 shows the median number of fitness function evaluations until the optimum is found with accuracy $\theta = 10^{-20}$ of 25 optimization runs using the meta-model for 10%, 25%, 35% and 50% of all evaluations.

	<i>no mm</i>	10	25	35	50
f_{Dou}	5,406	4,978	4,928	4,891	4,576
f_{Ras}	12,119	11,338	10,620	10,718	10,918

Table 1: Analysis of balance between Nadaraya-Watson surrogate and true objective function evaluations

The results show that the usage of the Nadaraya-Watson meta-model leads to a significant acceleration of the optimization process, that has also been confirmed by a Wilcoxon rank-sum test. For the unimodal function *Doublesum* the speedup increases with the fraction of surrogate evaluations. On *Rastrigin* the largest speedup can be observed if one fourth of the solutions are evaluated on the meta-model. As further tests confirm that one fourth is a good compromise for balancing we use this setting in the following.

4.2 Model Size

On the one hand the accuracy of the meta-model depends on the number of evaluated data pairs that are put into the regression model. On the other hand the efficiency of the non-parametric approach deteriorates with the number of data samples. Thus, we seek for a comparably low number of solutions that we have to put into the regression model in order to achieve satisfying results. A model size m means that the last m evaluated solutions are used for the prediction process. Again, we tested a (25,100)-CMSA-ES on the two functions *Doublesum* and *Rastrigin* with $N = 5$. Table 2 shows the median number of fitness function evaluations we achieved with various model sizes ($m = 10, 30, 50, 100$) in comparison to optimization without the meta-model (*no mm*).

	<i>no mm</i>	10	30	50	100
f_{Dou}	5,406	5,005	4,928	4,928	4,774
f_{Ras}	12,119	11,328	10,620	10,114	10,468

Table 2: Dependency of optimization process on number m of last evaluated solutions that are used for prediction

On problem *Doublesum* the best results have been achieved with a model size of $m = 100$ while the setting $m = 50$ achieved the best results on *Rastrigin*.

4.3 Comparison

In this section we conduct a broader experimental analysis and comparison of the CMSA-ES and Nadaraya-Watson optimization scheme on well known test functions from litera-

²Problem definitions and initial settings can be found in Section 4.3.

ture. Table 3 states the test problems we refer to in our analysis, the starting interval \mathbf{s}_{init} , the initial step sizes σ_{init} , and the termination condition f_{stop} .

name	function	\mathbf{s}_{init}	σ_{init}	f_{stop}
<i>Sphere</i>	$f_{\text{Sp}}(\mathbf{s}) = \sum_{i=1}^N x_i^2$	$[-10, 10]^N$	1.0	10^{-20}
<i>Doublesum</i>	$f_{\text{Dou}}(\mathbf{s}) = \sum_{i=1}^N \left(\sum_{j=1}^i (x_j) \right)^2$	$[-10, 10]^N$	1.0	10^{-20}
<i>Rosenbrock</i>	$f_{\text{Ros}}(\mathbf{s}) = \sum_{i=1}^{N-1} \left((100(x_i^2 - x_{i+1})^2 + (x_i - 1)^2) \right)$	$[-10, 10]^N$	1.0	10^{-20}
<i>Rastrigin</i>	$f_{\text{Ras}}(\mathbf{s}) = \sum_{i=1}^N (x_i^2 - 10 \cos(2\pi x_i) + 10)$	$[-10, 10]^N$	1.0	10^{-20}
<i>Griewank</i>	$f_{\text{Gri}}(\mathbf{s}) = \sum_{i=1}^N \frac{x_i^2}{4000} - \prod_{i=1}^N \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$	$[-10, 10]^N$	1.0	10^{-20}

Table 3: Survey of test functions, initial conditions, and termination criterion.

As further analyses have shown that a model size of $m = 30$ leads acceptable results, we use this setting in the following as well as a fraction of $\alpha = 25\%$ of surrogate evaluations that have been determined in Section 4.1. Table 4 shows the empirical results on the test problems with $N = 5$ dimensions. Population model and bandwidth h are stated, initial settings and termination condition are chosen like in the previous experiments. The figures show the best, mean, median and worst numbers of fitness function evaluations of 25 experiments until the optimum is found. Parameter *win* shows the fraction of evaluations the meta-model based optimization (lower line) needs in comparison to the solely

	CMSA-ES	h	<i>best</i>	<i>mean</i>	<i>dev</i>	<i>median</i>	<i>worst</i>	<i>win</i>
f_{Sp}	(25,100)	-	5,508	5,671.2	116.29	5,712	5,814	1.00
		0.1	5,005	5,164.0	104.97	5,236	5,236	0.92
f_{Dou}	(25,100)	-	5,304	5,406.0	72.12	5,406	5,508	1.00
		0.1	4,543	4,886.8	202.53	4,928	5,082	0.91
f_{Ros}	(25,100)	-	14,994	17,625.6	1,624.65	17,646	18,972	1.00
		0.001	13,013	14,419.4	1,399.28	14,245	16,632	0.8
f_{Ras}	(50,200)	-	10,705	11,876.6	676.02	12,119	12,321	1.00
		0.1	10,089	10,660.4	467.37	10,620	11,353	0.88
f_{Gri}	(25,100)	-	4,284	4,488.0	124.92	4,488	4,590	1.00
		0.01	3,798	4,132.2	225.47	4,235	4,312	0.94

Table 4: Experimental comparison of the meta-model based optimization on the test suite with $N = 5$ dimensions

CMSA-ES (upper line) based on the median. We observe an improvement of up to 20 percent when using the meta-model. Amazingly, the performance win on the functions *Rosenbrock* and *Rastrigin* is higher than on the simple unimodal functions *Sphere* and *Doublesum*.

The experimental results on our test suite for $N = 10$ dimensions, see Table 5, confirms the picture. Also here we can observe significant accelerations. Our first experimental results in higher dimensions, i.e. $N = 20$, on the problems *Sphere*, *Doublesum* and *Griewank* show that the performance win decreases in comparison to the CMSA-ES without meta-model. The reason is probably a worse prediction accuracy in higher dimensions. As a next step we will control the bandwidth h online during the optimization process. For this sake we will apply leave-one-out cross-validation (LOOCV) to adjust

	CMSA-ES	h	$best$	$mean$	dev	$median$	$worst$	win
f_{Sp}	(25,100)	-	10,098	10,322.4	196.20	10,302	10,608	1.00
		0.1	9,342	9,553.0	189.14	9,548	9,779	0.92
f_{Dou}	(25,100)	-	10,302	10,587.6	309.38	10,404	11,016	1.00
		0.1	9,188	9,676.2	333.41	9,625	10,087	0.92
f_{Ros}	(100,400)	-	160,398	178,488	17,779.12	179,694	204,618	1.00
		0.001	146,653	168,529	12,671.32	171,937	178,723	0.95
f_{Ras}	(100,400)	-	31,356	33,044.4	1,375.05	32,562	33,044.4	1.00
		0.01	30,185	31,899.2	1,171.00	31,668	33,176	0.97
f_{Gri}	(25,100)	-	7,956	8,302.8	234.82	8,262	8,568	1.00
		0.01	7,186	7,617.6	294.69	7,700	7,931	0.93

Table 5: Experimental comparison of the meta-model based optimization on the test suite with $N = 10$ dimensions

	CMSA-ES	h	$best$	$mean$	dev	$median$	$worst$	win
f_{Sp}	(25,100)	-	17,340	17,829.6	410.54	17,850	18,462	1.00
		0.1	16,170	16,852.6	646.19	16,734	17,864	0.93
f_{Dou}	(25,100)	-	19,176.0	19,869.6	654.70	19,890	20,706	1.00
		0.1	17,710	19,054.8	1077.88	19,096	20,559	0.96
f_{Gri}	(25,100)	-	14,076	14,280.0	239.21	14,178	14,586	1.00
		0.01	13,090	13,469.6	345.61	13,346	14,014	0.94

Table 6: Experimental comparison of the meta-model based optimization on the test suite with $N = 10$ dimensions

the model. LOOCV chooses the bandwidth that results in the least reconstruction error of data sample x if we do not consider this sample for reconstruction, but all other data samples, in particular the neighbors of x . The LOOCV error that has to be minimized becomes:

$$E(X) = \sum_{i=1, i \neq j}^N \left\| y_i - \sum_j \frac{K(x_i, x_j)}{\sum_k K(x_j, x_k)} y_j \right\|^2 \quad (14)$$

In a first experiment we try to find an optimal setting h^* , we test $k = 15$ random settings for h' based on the last bandwidth h using Bäck's continuous interval log-normal mutation operator that restrict the random numbers to an interval $[a, b]$:

$$h' = (a - b) \cdot \left(1 + \frac{1 - h}{h} \cdot e^{\tau \cdot \mathcal{N}(0,1)} \right)^{-1} + a \quad (15)$$

with $\tau = 1.0$. A first analysis on *Rastrigin* with $a = 0$ and $b = 20$ shows significant saving of fitness evaluations. With the same setting of the previous experiments, but a LOOCV step in each iteration, we achieved a fastest optimization process with only 9,978 fitness function evaluations and a median of 10,291 evaluations. The bandwidth decreases in the course of evaluation to values between $h = 10^{-7}$ to $h = 10^{-9}$. This encouraging result motivates more investigations into the online regularization of the bandwidth.

5 Summary and Outlook

Nadaraya-Watson based objective function surrogates improve the optimization using covariance matrix self-adaptation. But the performance win decreases with increasing number of dimensions. First experiments with online regularization of the regression model, i.e. control of bandwidth h by LOOCV during the optimization process shows encouraging results. We will analyze various regularization techniques in the future. Furthermore, we will enhance the Nadaraya-Watson model to quadratic and local polynomial regression with the hope to increase the accuracy of the fitness forecast. Meta-models are not restricted to single-objective optimization. Hence, we plan to apply the meta-models to multi-objective problems as the multivariate Nadaraya-Watson variant is able to make predictions in \mathbb{R}^d

References

- [1] Emmerich, M.; Giotis, A.; Ödemir, M.; Bäck, T.; Giannakoglou, K.: Metamodel-Assisted Evolution Strategies. In: *PPSN*, S. 361–370. 2002.
- [2] Ulmer, H.; Streichert, F.; Zell, A.: Optimization by Gaussian Processes Assisted Evolution Strategies. S. 434–442. Heidelberg, Germany: Springer Verlag. 2003.
- [3] Kern, S.; Hansen, N.; Koumoutsakos, P.: Local Meta-models for Optimization Using Evolution Strategies. In: *PPSN*, S. 939–948. 2006.
- [4] Forrester, A.; Keane, A.: Recent Advances in Surrogate-Based Optimization. *Progress in Aerospace Sciences* (2009).
- [5] Hastie, T.; Tibshirani, R.; Friedman, J.: *The Elements of Statistical Learning*. Berlin: Springer. 2009.
- [6] Beyer, H.-G.; Sendhoff, B.: Covariance Matrix Adaptation Revisited - The CMSA Evolution Strategy -. In: *Proceedings of the 10th Conference on Parallel Problem Solving from Nature - PPSN X*, S. 123–132. 2008.
- [7] Giannakoglou, K.; Giotis, A.; Karakasis: Low-cost Genetic Optimization Based on Inexact Pre-evaluations and the Sensitivity Analysis of Design Parameters. *Inverse Problems in Engineering* 9 (2001), S. 389–412.
- [8] Ostermeier, A.; Gawelczyk, A.; Hansen, N.: A Derandomized Approach to Self-Adaptation of Evolution Strategies. *Evolutionary Computation* 2 (1994) 4, S. 369–380.
- [9] Hansen, N.: The CMA Evolution Strategy: A Tutorial. Technische Universität Berlin, ETH Zürich. 2005.
- [10] Beyer, H.-G.; Schwefel, H.-P.: Evolution strategies - A Comprehensive Introduction. *Natural Computing* 1 (2002), S. 3–52.

Robust Identification of Coagulated Zebrafish Eggs using Image Processing and Classification Techniques

**Rüdiger Alshut¹, Jessica Legradi², Lixin Yang², Uwe Strähle²,
Ralf Mikut¹, Markus Reischl¹**

¹ Forschungszentrum Karlsruhe GmbH, Institut für Angewandte Informatik,

² Forschungszentrum Karlsruhe GmbH, Institut für Toxikologie und Genetik

E-Mail: ruediger.alshut@kit.edu

1 Introduction

Fish embryo toxicity tests have become a common approach for determining and comparing the acute toxicity of thousands of compounds. The zebrafish has several features making it an interesting model organism for these assays, for instance, its morphological and physiological similarity to mammals, its fertility, its transparency, and its ex utero development [1]. More importantly, standard protocols for conducting the fish embryo test have already been defined [2–5] that ensure reproducibility and comparability [6].

Although zebrafish embryos are the best developed and most promising model for high-throughput in vivo screening of intact organ systems [6], several challenges remain. First, the zebrafish model must attain a high level of reliability in order to become widely accepted as a standard model for biological testing [1, 7]. Second, there is a need for standardizing and fully automating the assays since manual inspection of tens of thousands of images is not feasible for high-throughput analysis [8]. One of the most challenging tasks in the automation of the whole fish embryo test is the evaluation of the data produced by a screening microscope. Currently, this step is performed by hand due to several reasons. The screening is commonly performed 24 or 48 hours after fertilization, and at this time, the embryo is already able to move within the chorion. Therefore, neither the chorion has a well defined position in the image nor is the alignment of the embryo within the chorion fixed. The acquired image can show the embryo in different orientations, for example from dorsal, lateral, or ventral. Hence, the embryo is manually dechorionated and its orientation is fixed prior to acquiring the images with the microscope [9–12]. This approach is time-consuming and an automation is difficult.

Here, we present the first fully automatic approach for evaluating the zebrafish embryos without dechorionating or aligning them. Our method is able to distinguish, for example, between healthy embryos and coagulated eggs. The presented method can be used in high-throughput assays for determining dose-response relationships from which common values of interest such as LC50 (lethal concentration 50), LOEL (lowest observed effect level) and NOEL (no observed effect level) can easily be calculated.

2 Material and Methods

2.1 Overview

The identification of coagulated zebrafish eggs can be divided into four basic steps: image acquisition, preprocessing, feature extraction, and classification. Each of these processing steps will be described in the following sections.

2.2 Image Acquisition and Datasets

The rapid development of the embryos make it necessary to acquire the images in a small time frame. In order to obtain comparable results, the age difference between zebrafish, measured in *hours post fertilization* [hpf], should not exceed approximately three hours. We used automated microscopes equipped with a robot arm for loading 96-well round-bottom microtiter plates. For this analysis, four of these were prepared by separating each embryo in a single well using a pipet. Two plates were exposed to two different substances known to cause toxic effects in zebrafish embryo [6, 13]. The other two plates were controls without treatment. Table 1 shows the used toxicant and dose.

Name	Symbol	Concentration [mg/l]	Concentration [mM]
3,4-Dichloroaniline	34DCA	15	75
p-Tert-Butylphenol	PTBP	1.2	0.08

Table 1: Used toxicants and doses

The embryos were exposed to the toxicants at an age of 24-34 hpf and the images were taken at approximately 34 hpf. Imaging of the 96-well plates was carried out on a "Scan^R" high content screening microscope [14] (Olympus Biosystems) with a Hamilton Micro-Lab SWAP plate gripper, a 2.5x objective and an Olympus Biosystems DB-1 (1344x1024 pixels) CCD camera. In an initial step, the central focal plane of the embryo was detected once by hand. This position was used as reference for all wells of a plate. The other imaging parameters like e.g. exposure time were also detected by hand and then fixed for one plate. As the embryos are not flat and the depth of view of every microscope is limited, it is impossible to acquire an image where the whole embryo appears sharp. To still be able to capture every part of the embryo, the focal plane is shifted 8 times as shown in Fig. 1. This results in so called stack images, which are merged into one single very sharp image in an additional step. An example of stack images of a zebrafish embryo is shown in Fig. 2. After acquiring all images, the first of the two datasets consisted of 384 embryos and 8 stacks for each embryo resulting in 3072 images to be processed. An overview of the dataset and the details of all captured images that were used for this analysis is given in Table 2.

2.3 Preprocessing

Errors can occur during the preparation of the plates due to faulty well loading, for example with more than one embryo, with no embryo, or with dirt. Fig. 3 shows examples

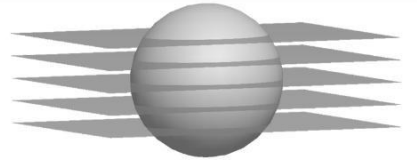


Figure 1: Stacked focal planes

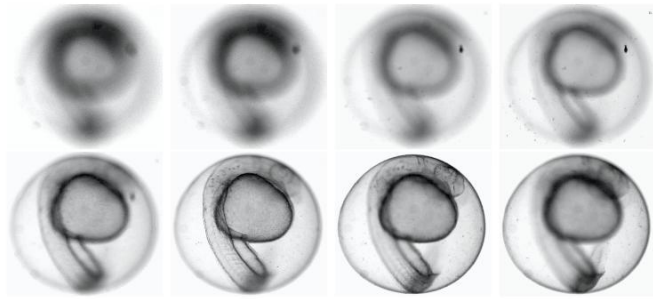


Figure 2: Cropped stack sequence of a zebrafish embryo

Dataset Number	Class	Examples	Z-Stacks	Resolution $M \times N$ [Pixel]	Image Size [cm]	Color Depth [bit]
DS1	alive	187	8	1344 x 1024	47.4 x 36.1	16
DS2	dead	190	8	1344 x 1024	47.4 x 36.1	16
DS4	erroneous	7	8	1344 x 1024	47.4 x 36.1	16

Table 2: Image dataset details

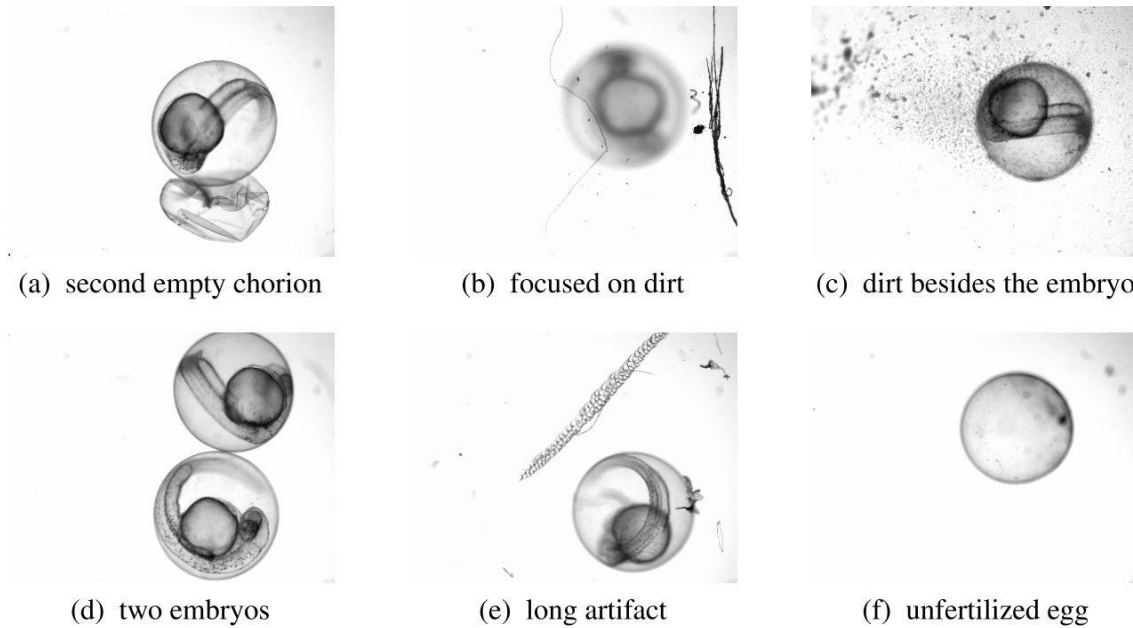


Figure 3: Erroneous images

of errors that may appear in the final image. To filter the worst errors, a manual quality check was made to label images with errors.

In order to optimize the contrast over the available range of intensity levels, the original intensity values are scaled such that a certain percent of pixels is at the highest and lowest intensity value. Assuming that a is an intensity value in the image $I(u, v)$, $u \in [1, \dots, M]$, $v \in [1, \dots, N]$, the enhanced pixel intensity value is then cal-

culated by:

$$f(a) = \begin{cases} a_{\min} & \text{for } a \leq \hat{a}_{\text{low}} \\ a_{\min} + (a - \hat{a}_{\text{low}}) \cdot \frac{a_{\max} - a_{\min}}{\hat{a}_{\text{high}} - \hat{a}_{\text{low}}} & \text{for } \hat{a}_{\text{low}} < a < \hat{a}_{\text{high}} \\ a_{\max} & \text{for } a \geq \hat{a}_{\text{high}} \end{cases} \quad (1)$$

with parameters a_{\min} , a_{\max} depending on the color depth of the images (e.g. $a_{\min} = 0$, $a_{\max} = 255$ for 8 bit).

The thresholds are calculated over all pixels using

$$\begin{aligned} \hat{a}_{\text{low}} &= \min\{i \mid H(i) \geq M \cdot N \cdot s_{\text{low}}\} \\ \hat{a}_{\text{high}} &= \max\{i \mid H(i) \leq M \cdot N \cdot (1 - s_{\text{high}})\} \end{aligned} \quad (2)$$

where $M \cdot N$ is the number of pixels in the source image I and $H(i)$ is the histogram of the image with $i_{\max} + 1$ intervals. For s_{low} and s_{high} , a value of 2% was chosen and i_{\max} was set to 255.

In the following step, the open source software *CombineZP*¹ was applied for merging the image stacks for receiving an extended focus. For this analysis, the images were saved by the microscope as 16 bit TIFF (Tagged Image File) using Lempel-Ziv-Welch (LZW) compression [15]. These images were enhanced as described above. Then, the images had to be converted to 8 bit TIFF grayscale images in order to be able to read them with *CombineZP*. Since it was not possible to read compressed TIFF images with Matlab 5.3, it was also necessary to save the images in an uncompressed format.

2.4 Filter Chain

After enhancing and converting the images, the chorion needs to be distinguished from the background and the artifacts in the image. The following steps calculate a mask for each image that deletes everything but the chorion with the embryo. First, the algorithm distinguishes the chorion from artifacts and exposure errors. Then, position and dimensions of the chorion are determined and uninteresting areas are cropped out of the image. The whole filtering process is shown in Fig. 4c and will now be described in detail.

Starting point of the filter is the preprocessed image (see Fig. 4a for a living and Fig. 4b for a dead embryo). In the next step (see Fig. 4c, Step 2), a threshold filter is applied that uses the ISO-Data method [17] for calculating an image-specific threshold value for separating the embryo from the background. The next step is to identify all black pixels surrounded by white pixels and color them white. The aim of this procedure is to fill the "black holes" inside the chorion (see Fig. 4c, Step 5) [18]. Unfortunately, the border of the chorion is not always recognized as an uninterrupted line. In images having a low contrast, the border is often interrupted and the chorion is only partly recognized as shown in Fig. 4e. In these cases, a previous morphological closing is performed (see Fig. 4c, Step 3 and 4), which bridges these little gaps in most instances (see [18] for more details) and increases the robustness. In order to eliminate artifacts from the image and recognize the chorion, all pixels belonging to one of the different compact areas $b = 1, \dots, B$ containing white pixels are labelled in a blob image with $B(u, v) = b$ and their size is measured (see

¹see <http://www.hadleyweb.pwp.blueyonder.co.uk>

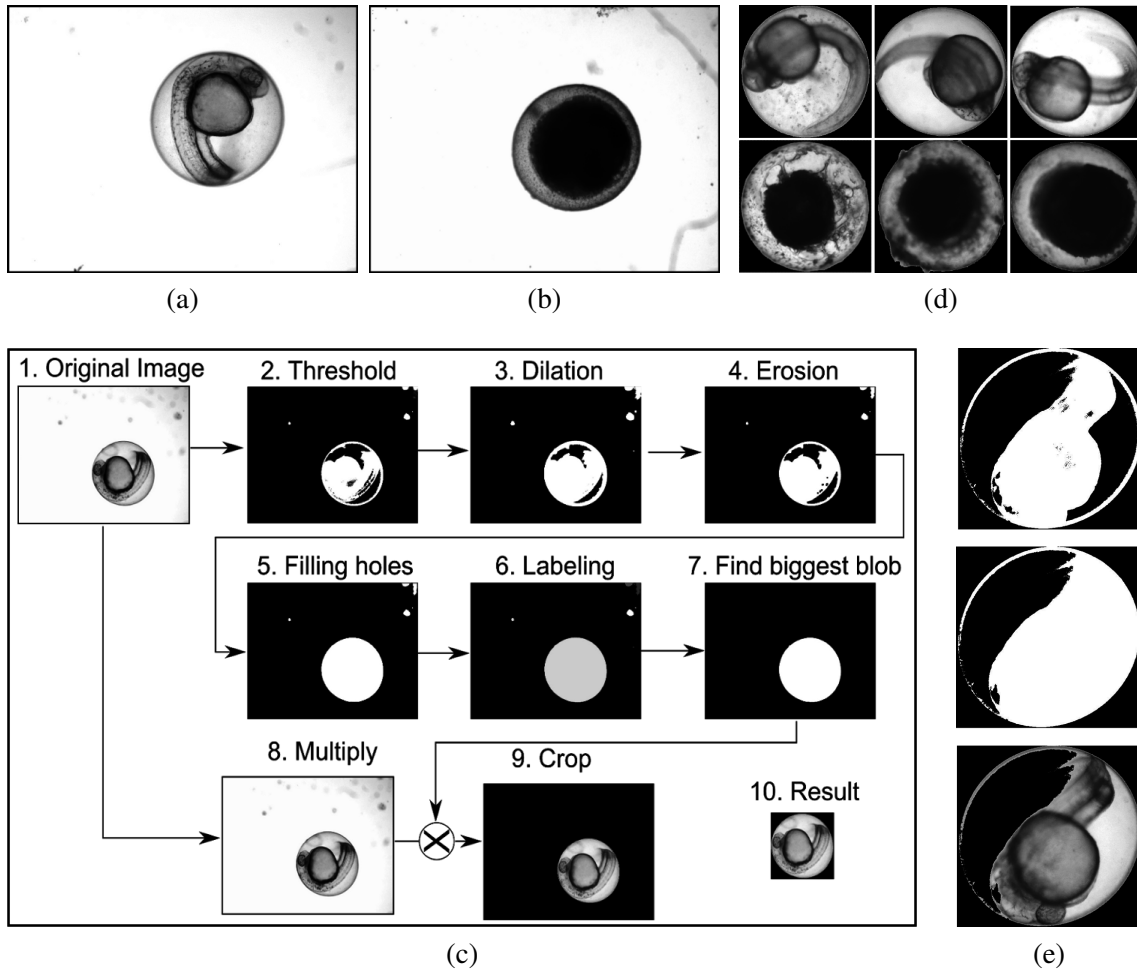


Figure 4: Microscopic images of living (Fig. 4a) and dead (Fig. 4b) embryos. The filtering process is shown in Fig. 4c [16]. Examples for cropped images are given in Fig. 4d. In cases where the images show low contrast the chorion is not completely detected as shown in Fig. 4e

Fig. 4c, Step 6). In Step 7 of Fig. 4c, only the object (blob) b_{max} with the highest number of pixels is retained. A mask can be calculated out of the filtered image of Step 7 in Fig. 4c according to the following equation:

$$I'(u, v) = \begin{cases} 1 & \text{if } B(u, v) = b_{max} \\ 0 & \text{else.} \end{cases} \quad (3)$$

The resulting $I'(u, v)$ is a mask containing only black pixels except at the position where the chorion is located. Multiplying this mask with the source image results in a new image containing merely the chorion and black pixels (see Fig. 4c, Step 8). The Step 9 shown in Fig. 4c deletes the black pixels and stores the result (see Fig. 4c, Step 10) in a new file with a smaller image. This is done by calculating the center-of-mass of the blob

$$x_s = \frac{\sum_{u=1}^M \sum_{v=1}^N u \cdot I'(u, v)}{\sum_{u=1}^M \sum_{v=1}^N I'(u, v)}, \quad y_s = \frac{\sum_{u=1}^M \sum_{v=1}^N v \cdot I'(u, v)}{\sum_{u=1}^M \sum_{v=1}^N I'(u, v)}, \quad (4)$$

and determining the extension of the chorion by assuming that it is a round blob

$$r = \sqrt{\frac{\sum_{u=1}^M \sum_{v=1}^N I'(u, v)}{\pi}}. \quad (5)$$

Examples of cropped images for both living and dead embryos are given in Fig. 4d.

2.5 Feature extraction

Once the chorion is isolated from the image, robust features for calculating and describing the morphology of dead and living embryos need to be extracted. Typical examples of dead embryos are coagulated eggs. Like mentioned before, the orientation of the embryo is unknown. Since dorsally, laterally, or ventrally oriented embryos appear very different in an image, calculated features need to be independent from the embryo's orientation. One approach is to acquire three-dimensional data but this is still too much time-consuming to be done in high-throughput analyses. Another important requirement of extracted features is that they need to be independent from the alignment of the embryo. Coagulated eggs show a dark area in the chorion in contrast to a developing embryo. Although the alignment of the embryo is not known, the mentioned dark point has no orientation and is always set in the middle of the egg. Thus the aim of all features is to distinguish a developed embryo from this dark point.

For the feature extraction, only the relatively small cropped images $I(u, v), u \in [1, \dots, u_{max}], v \in [1, \dots, v_{max}]$, are used. The first calculated feature is a value determined out of the image histogram. Comparing the histogram of a coagulated egg with the one of a living embryo results in a peak at low intensity values (see Fig. 5b). The center-of-mass of the histogram is calculated by

$$x_1 = \frac{\sum_{i=0}^{i_{max}} i \cdot H(i)}{\sum_{i=0}^{i_{max}} H(i)}. \quad (6)$$

The calculated value is significantly lower for coagulated eggs than for living embryos as indicated by the arrow in Fig. 5b. For the example in Fig. 5, the value of the coagulated chorion is 94 and for the living one 164.

As second feature, the mean value over all pixels is calculated and divided by the total number of non-black pixels:

$$x_2 = \frac{\sum_{u=1}^{u_{max}} \sum_{v=1}^{v_{max}} I(u, v)}{u_{max} \cdot v_{max} - \text{card}\{(u, v) \mid I(u, v) = 0\}}. \quad (7)$$

Black pixels with a value of 0 are usually background pixels and thus not included in the calculation.

The third and fourth features take advantage of the previous knowledge that the "dark point" is located in the middle of the chorion. In Fig. 5c, the curve of intensity levels

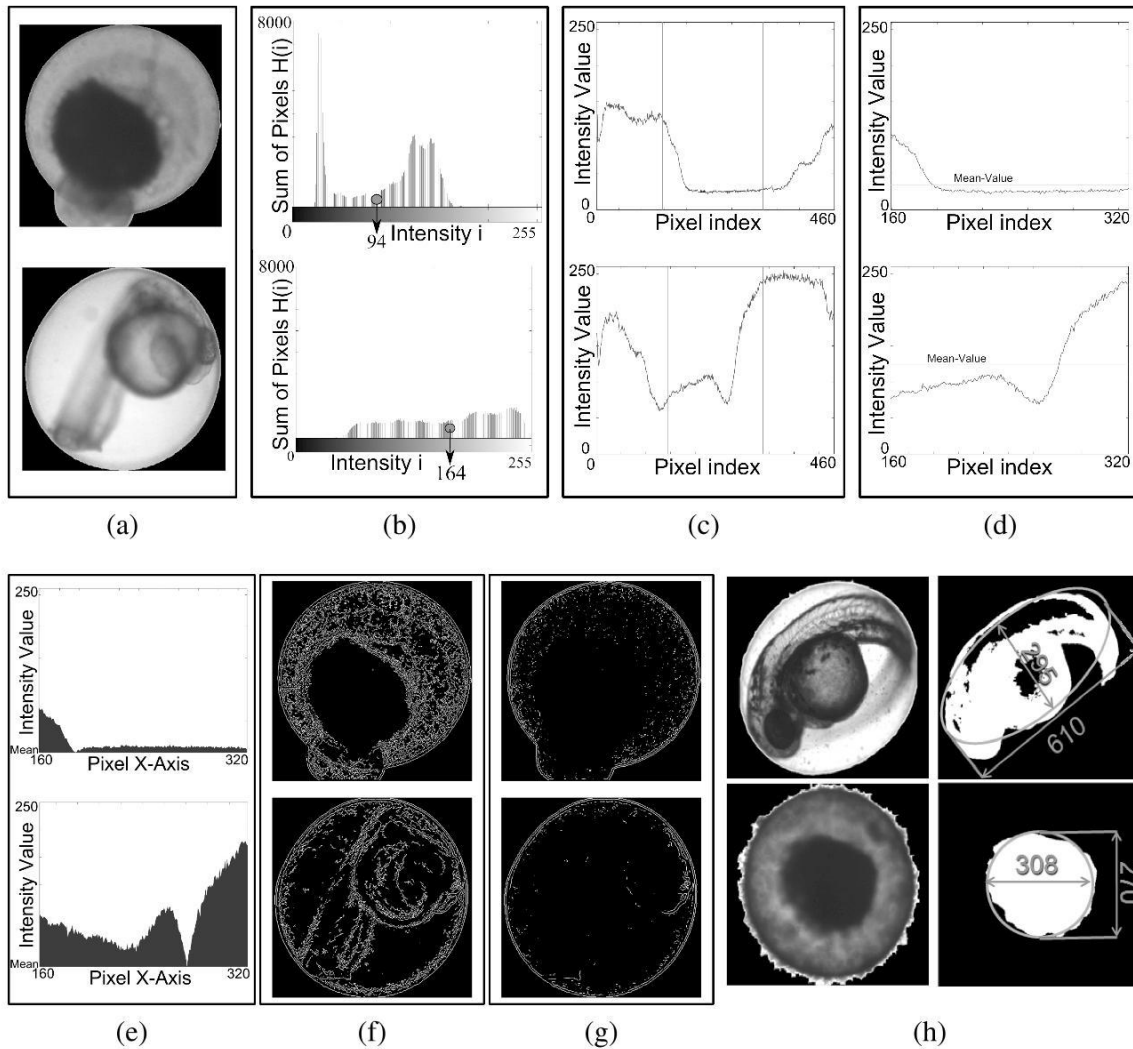


Figure 5: Two example images for alive and dead embryos (Fig. 5a) and illustration of the implemented features $x_1 - x_7$ (Fig. 5b-5h)

coming from the central row of pixels is plotted for the coagulated (above) and the healthy (below) chorion. From this plot, two features can be calculated that are related only to the center of the curve with the dark point marked by the two vertical lines in the plot. Therefore, the first 30% of the curve and the last 30% are neglected. From the remaining interval, the mean value is taken as the third feature (see the labeled line in Fig. 5d). The mean intensity value for dead embryos is lower due to the dark pixels in the "dark point". The feature value is calculated by

$$x_3 = \frac{\sum_{j=n_1}^{n_2} I(j, \text{round}(\frac{v_{max}}{2}))}{n_2 - n_1 + 1} \quad (8)$$

with $n_1 = \text{round}(0.3 \cdot u_{max})$ and $n_2 = \text{round}(0.7 \cdot u_{max})$. Indices are round off integer values using the operator $\text{round}(\cdot)$.

Before calculating the fourth feature, the curve is corrected by the mean value and the absolute values are calculated such as:

$$\delta(u) = \left| I\left(\text{round}(u, \frac{v_{max}}{2})\right) - x_3 \right|. \quad (9)$$

Then, the fourth feature is given as sum of the absolute values. This sum is the dark-colored area in Fig. 5e and represents the oscillation of the intensity values:

$$x_4 = \sum_{j=n_1}^{n_2} \delta(j). \quad (10)$$

For dead embryos, the values do not vary inside the "dark point". Therefore, the sum calculated from images of dead embryos will be lower than the sum of living ones (see Fig. 5e).

The fifth and sixth feature are binary images created using two different edge filter algorithms. The first one uses the Canny edge algorithm [19] and the second one uses the LoG edge algorithm [20]. The resulting binary images represent the contour of the embryos. For the feature values, the sum of all white pixels is counted as a measure of the length of the edges assuming that the dark point shows less contour and therefore less edges than a living embryo:

$$x_5 = \text{card}\{(u, v) \mid I_{\text{Canny}}(u, v) = 1\}, \quad (11)$$

$$x_6 = \text{card}\{(u, v) \mid I_{\text{LoG}}(u, v) = 1\}. \quad (12)$$

Examples for the binary images for the two edge algorithms applied to living and dead embryos are given in Fig. 5f and Fig. 5g, respectively.

The seventh feature measures the sphericity after applying the ISO-Data algorithm. The threshold image shows for coagulated eggs an almost round point, whereas developed embryos exhibit oval bodies (Fig. 5h). To express this characteristic, the aspect ratio is calculated from the largest and smallest axis length of the oval best fitting on the threshold image. First, the required central moments $(\mu_{11}, \mu_{22}, \mu_{12})$ of the threshold filtered image are determined and then major and minor axis lengths are calculated (see [21] for further details):

$$\text{Major / Minor Axis} = 2 \cdot \sqrt{2} \cdot \sqrt{\mu_{11} + \mu_{22} \pm \sqrt{(\mu_{11} - \mu_{22})^2 + 4 \cdot \mu_{12}^2}}. \quad (13)$$

Feature number seven is then calculated as the difference between axis lengths:

$$x_7 = \text{Major Axis} - \text{Minor Axis}. \quad (14)$$

For the round point, the blob is almost a circle, thus the feature x_7 is small. In the example in Fig. 5h the coagulated chorion has a feature value of 38, whereas the living embryo has a value of 315.

The last feature is a validation check and helps to sort out the labeled erroneous images automatically. It measures the size of the image found by the filter chain by calculating

$$x_8 = u_{max} \cdot v_{max} \quad (15)$$

with u_{max}, v_{max} the indices of the cropped image. Because the chorion is always approximately 1mm^2 in area, all cropped images differing significantly in size will contain artifacts or other errors.

2.6 Classification

All the calculated features permit distinguishing between the classes "coagulated" and "alive". All N_{img} -tuples of the s -dimensional feature space \mathbf{x} can be combined to a feature matrix \mathbf{X} for a selected set of features $\mathcal{I}_k = \{s_1, \dots, s_k\}$:

$$\mathbf{X}_{\mathcal{I}_k} = \begin{pmatrix} x_{1,s_1} & x_{1,s_2} & \dots & x_{1,s_k} \\ x_{2,s_1} & x_{2,s_2} & \dots & x_{2,s_k} \\ \vdots & \vdots & & \vdots \\ x_{N_{img},s_1} & x_{N_{img},s_2} & \dots & x_{N_{img},s_k} \end{pmatrix} = \begin{pmatrix} \mathbf{x}_{\mathcal{I}_k}^T [1] \\ \mathbf{x}_{\mathcal{I}_k}^T [2] \\ \vdots \\ \mathbf{x}_{\mathcal{I}_k}^T [N_{img}] \end{pmatrix}. \quad (16)$$

Based on manually labeled data, a classifier algorithm is trained and the resulting decision rule can be applied to unknown data. Provided that the unknown data is comparable to the training data, a successful classification can be performed. For the classification a Bayes classifier is used [22].

The classification was performed using the open source Matlab-Toolbox Gait-CAD [23]. For reasons of data illustration, best interpretability, and classifier robustness, a reduction to the two most powerful features is recommended ($k = 2$). An ANOVA (analysis of variances) and a MANOVA (multivariate analysis of variances) were performed for identifying the best features [24, 25] by the evaluation measure

$$Q_{\mathcal{I}_k, (M)ANOVA} = 1 - \frac{\det(\mathbf{W}(\mathbf{X}_{\mathcal{I}_k}))}{\det(\mathbf{T}(\mathbf{X}_{\mathcal{I}_k}))} \quad (17)$$

with

$$\mathbf{T}(\mathbf{X}_{\mathcal{I}_k}) = N_{img} \cdot \mathbf{S}(\mathbf{X}_{\mathcal{I}_k}), \quad \mathbf{W}(\mathbf{X}_{\mathcal{I}_k}) = \sum_{c=1}^C N_c \cdot \mathbf{S}_c(\mathbf{X}_{\mathcal{I}_k}), \quad (18)$$

with $\det(\cdot)$ being the determinant, \mathbf{T} the total variance matrix, \mathbf{W} the variance matrix within classes, $\mathbf{X}_{\mathcal{I}_k}$ the matrix with the selected feature values from (16), C the number of classes (here with $C = 2$ resp. $C = 3$), N_{img} the total number of preprocessed images, N_c the number of images in class c , \mathbf{S} the estimated covariance matrix for all images and \mathbf{S}_c the estimated covariance matrix for the images in class c . If the index set of the selected features \mathcal{I}_k consists of only one feature the MANOVA leads to ANOVA. For a perfect combination of features the value of $Q_{\mathcal{I}_k, (M)ANOVA}$ is 1, and 0 if there is no information within the model.

After determining the two best features, the classification is performed utilizing a Bayes classifier. The decision \hat{y} is made for the benefit of the class c with the biggest estimated *a posteriori* probability:

$$\hat{y} = \underset{c}{\operatorname{argmax}} (\hat{p}(y = c | \mathbf{x}_{\mathcal{I}_k})) = \underset{c}{\operatorname{argmax}} \left(\frac{\hat{P}(y = c) \cdot \hat{p}(\mathbf{x}_{\mathcal{I}_k} | y = c)}{\sum_{j=1}^C \hat{P}(y = j) \cdot \hat{p}(\mathbf{x}_{\mathcal{I}_k} | y = j)} \right) \quad (19)$$

$$\hat{p}(\mathbf{x} | y = c) = \frac{1}{(2\pi)^{\frac{k}{2}} \sqrt{\det(\mathbf{S}_c(\mathbf{X}_{\mathcal{I}_k}))}} \cdot e^{(-\frac{1}{2} \cdot (\mathbf{x}_{\mathcal{I}_k} - \boldsymbol{\mu}_c)^T \cdot \mathbf{S}_c^{-1}(\mathbf{X}_{\mathcal{I}_k}) \cdot (\mathbf{x}_{\mathcal{I}_k} - \boldsymbol{\mu}_c))}. \quad (20)$$

Here, $\hat{P}(y = c)$ is the estimated *a priori* probability of each class and $\hat{p}(\cdot)$ means the multidimensional probability density functions under assumption of normal distributions. $\boldsymbol{\mu}_c$ is a vector of the estimated mean values of the k selected features for the c -th class.

3 Results

To obtain the training data, the pictures are labeled manually as either coagulated or alive. From the four 96-well plates, 384 images were captured. Seven erroneous images were sorted out due to faulty load of the zebrafish embryos, empty wells, or wells containing more than one embryo. The remaining 377 pictures contained 189 living and 188 dead embryos and were utilized for training the classification algorithm. The resulting decision rule was applied on the same data. Then the performance of the classifier was determined using 5-fold cross-validation.

The result of ANOVA for the two classes "coagulated" and "alive" is shown in Table 3.

Position	Feature	Name	ANOVA value
1	x_3	Mean intensity value within the chorion center	0.928
2	x_2	Mean intensity value for $I > 0$	0.873
3	x_1	Center of mass of histogram	0.873
4	x_4	Fluctuation of intensity values within the chorion center	0.660
5	x_7	Sphericity value of the threshold image	0.061
6	x_6	Quantity of zeros after applying the edge filter "LoG"	0.007
7	x_5	Quantity of zeros after applying the edge filter "Canny"	0.004
8	x_8	Chorion size	0.000

Table 3: Result of the ANOVA over all 8 features - Two-class case

The ANOVA shows the highest relevance for the feature x_3 that measures the "Mean intensity value within the chorion center". The feature with the best performance in combination with the feature determined by the ANOVA is "Center of mass of histogram" (x_1) and was determined using MANOVA (see Table 4).

Position	Feature	Name	MANOVA value
1	x_3	Mean intensity value within the chorion center	— (0.928)
2	x_1	Center of mass of histogram	0.939
3	x_2	Mean intensity value for $I > 0$	0.939
4	x_4	Fluctuation of intensity values within the chorion center	0.937
5	x_7	Sphericity value of the threshold image	0.934
6	x_6	Quantity of zeros after applying the edge filter "LoG"	0.929
7	x_5	Quantity of zeros after applying the edge filter "Canny"	0.929
8	x_8	Chorion size	0.928

Table 4: Result of the MANOVA for the best pair for classification - Two-class case, value in parentheses: ANOVA value for x_3

The result of the classification for the two classes "coagulated" and "alive" is shown in Fig. 6a. Out of 377 examples, two could not be classified corresponding to an error of 0.5%. The misclassified embryos are shown in Fig. 6c and were both labeled as alive. In the 5-fold cross validation, the error is $0.6 \pm 0.1\%$. The confusion matrix is given in Table 5.

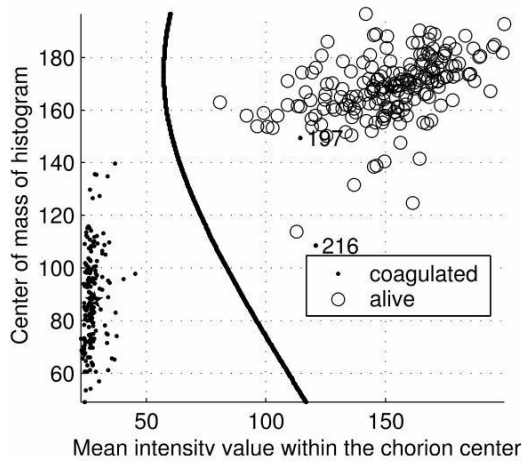
The erroneous images can be identified well by the chorion size. The result of the MANOVA for the three classes is shown in Table 7. The classification result for three

$$\begin{pmatrix} 187 & 0 \\ 2 & 188 \end{pmatrix}$$

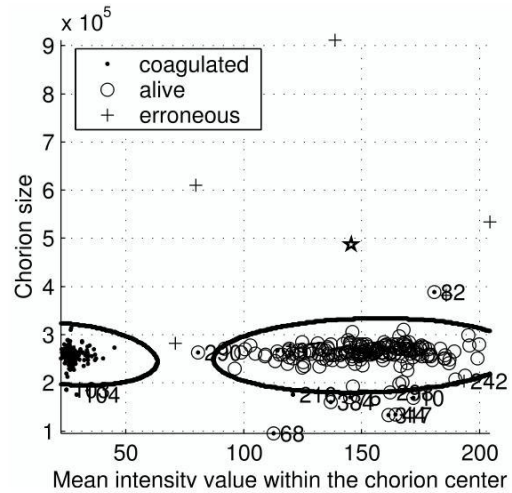
Table 5: Confusion matrix for the classification of "coagulated" and "alive" (2 classes)

$$\begin{pmatrix} 186 & 1 & 4 \\ 0 & 177 & 10 \\ 0 & 1 & 5 \end{pmatrix}$$

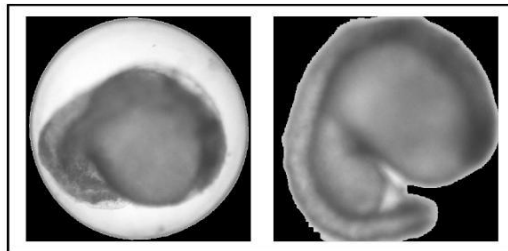
Table 6: Confusion matrix for the classification of "coagulated", "alive" and "erroneous" (3 classes)



(a) Classification results for the two-class case. The two numbered points are incorrectly classified.



(b) Classification results for the three-class case. The class "erroneous" is seen as an envelope around the data points.



(c) Misclassified images from Fig. 6a. DS 216 (right) and DS 197 (left).

Figure 6: Classification results and misclassifications

classes ("coagulated", "alive", "erroneous") shows an error of 3.6%. Using 5-fold cross-validation, the average error of ten runs is $4.3 \pm 0.7\%$. The classification result is visualized in Fig. 6b. The confusion matrix is given in Table 6.

4 Discussion and Conclusion

Both classifications show a very good result confirmed using 5-fold cross-validation. The two misclassified examples in the two-class case are not readily classifiable manually (see Fig. 6c). Since the algorithm was developed for detecting coagulated eggs, the classification of these two cases is still correct because they are not coagulated. Because of its

Position	Feature	Name	ANOVA value
1	x_3	Mean intensity value within the chorion center	— (0.919)
2	x_8	Chorion size	0.948
3	x_4	Fluctuation of intensity values within the chorion center	0.935
4	x_5	Quantity of zeros after applying the edge filter "Canny"	0.933
5	x_1	Center of mass of histogram	0.926
6	x_2	Mean intensity value for $I > 0$	0.926
7	x_7	Sphericity value of the threshold image	0.925
8	x_6	Quantity of zeros after applying the edge filter "LoG"	0.920

Table 7: Result of the MANOVA for the best feature pair for classification - including erroneous images

robustness and easiness to calculate features, the presented method allows for performing high-throughput screens. The automated evaluation of screening images is thus preferable compared to manual inspection because of the speed, the shown robustness and repeatability of the process.

The extracted features have shown to be very powerful for classification. Additionally, the size of the cropped image has proved useful for rejecting erroneous images. Despite the fact that the classification result including the erroneous images is worse (classification error of 4%), the errors are with one exception, all done by classifying "coagulated" or "alive" images to the "erroneous" class. This means that all erroneous images are rejected and will not affect further examinations. In a high-throughput screen, it is generally better to exclude an image than misclassifying it.

Our new algorithm is capable of examining toxicological assays without manual intervention even with certain faulty images. This provides the possibility of performing high-throughput analyses for determining dose-response relationships. The new features presented in this paper allow for correctly distinguishing between "alive" and "coagulated" zebrafish embryos.

Further work is required for automatically detecting more than the two endpoints "coagulated" or "alive" in a high-throughput analysis, like e.g. a detached tail or the heart beat.

5 Acknowledgement

We would like to thank Urban Liebel for his support with the microscope system and Markus Wahl for assistance with the experiments. We also thank Andreas Schlicker and Dominique Verreault for comments that greatly improved the manuscript.

6 References

- [1] Zon, L. I.; Peterson, R. T.: In vivo drug discovery in the zebrafish. *Nature Reviews Drug Discovery* 4(1) (2005), pp. 35–44.
- [2] Braunbeck, T.; Böttcher, M.; Hollert, H.; Kosmehl, T.; Lammer, E.; Leist, E.; Rudolf, M.; Seitz, N.: Towards an alternative for the acute fish LC50 test in chemical assessment: the fish embryo toxicity test goes multi-species – an update. *ALTEX - Alternativen zu Tierexperimenten* 22(2) (2005), pp. 87–102.

- [3] DIN: *DIN 38 415-T6, German standard methods for the examination of water, waste water and sludge – Subanimal testing – Part 6: Determination of the non-acute-poisonous effect of waste water to fish eggs by dilution limits*. German Standardization Organization, Beuth Vertrieb GmbH, Berlin. 2001.
- [4] OECD: *Fish embryo toxicity (FET) test. Draft OECD guideline for the testing of chemicals*. 2006.
- [5] OECD: *Background Paper on Fish Embryo Toxicity Assay*. 2006.
- [6] Nagel, R.: DarT: The embryo test with the Zebrafish (*Danio rerio*) – a general model in ecotoxicology and toxicology. *ALTEX - Alternativen zu Tierexperimenten* 19 Suppl 1 (2002), pp. 38–48.
- [7] Patton, E.; Zon, L.: The art and design of genetic screens: zebrafish. *Nature Reviews Genetics* 2(12) (2001), pp. 956–966.
- [8] Carpenter, A. E.: Image-based chemical screening. *Nature Chemical Biology* 3(8) (2007), p. 461.
- [9] Liu, T.; Lu, J.; Wang, Y.; Campbell, W.; Huang, L.; Zhu, J.; Xia, W.; Wong, S.: Computerized image analysis for quantitative neuronal phenotyping in zebrafish. *Journal of Neuroscience Methods* 153 (2006), pp. 190–202.
- [10] Lu, J.; Liu, T.; Nie, J.; Ding, J.; Zhu, J.; Yang, J.; Guo, L.; Xia, W.; Wong, S. T.: Automated quantitation of zebrafish somites in high-throughput screens. In: *Proc., IEEE/NLM Life Science Systems and Applications Workshop*, pp. 1–2. 2006.
- [11] Sabaliauskas, N.; Foutz, C.; Mest, J.; Budgeon, L.; Sidor, A.; Gershenson, J.; Joshi, S.; Cheng, K.: High-throughput zebrafish histology. *Methods* 39 (2006), pp. 246–254.
- [12] Tran, T.; Sneed, B.; Haider, J.; Blavo, D.; White, A.; Aiyejorun, T.; Baranowski, T.; Rubinstein, A.; Doan, T.; Dingleline, R.; et al.: Automated, quantitative screening assay for antiangiogenic compounds using transgenic zebrafish. *Cancer Research* 67(23) (2007), pp. 11386–92.
- [13] Lammer, E.; Carr, G.; Wendler, K.; Rawlings, J.; Belanger, S.; Braunbeck, T.: Is the fish embryo toxicity test (FET) with the zebrafish (*Danio rerio*) a potential alternative for the fish acute toxicity test? *Comparative Biochemistry and Physiology Part C: Toxicology & Pharmacology* 149 (2009), pp. 196–209.
- [14] Liebel, U.; Starkuviene, V.; Erfle, H.; Simpson, J.; Poustka, A.; Wiemann, S.; Pepperkok, R.: A microscope-based screening platform for large-scale functional protein analysis in intact cells. *FEBS Letters* 554 (2003) 3, pp. 394–398.
- [15] Welch, T. A.: A Technique for High-Performance Data Compression. *Computer* 17(6) (1984), pp. 8–19.
- [16] Alshut, R.: *Entwicklung eines Bildverarbeitungsalgorithmus zur automatisierten Evaluierung eines Hochdurchsatzexperimentes mit toxikologisch behandelten Modellorganismen*. Master's thesis, Universität Karlsruhe (TH). 2008.
- [17] Ridler, T. W.; Calvard, S.: Picture thresholding using an iterative selection method. *IEEE Transactions on Systems, Man and Cybernetics* 8 (1978), pp. 630–632.
- [18] Soille, P.: *Morphological image analysis: principles and applications*. Springer-Verlag New York, Inc. Secaucus, NJ, USA. 2003.
- [19] Canny, J.: Finding Edges and Lines in Images. Tech. rep., Massachusetts Institute of Technology. 1983.
- [20] Huertas, A.; Medioni, G.: Detection of intensity changes with subpixel accuracy using Laplacian-Gaussian Masks. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 8(5) (1986), pp. 651–664.
- [21] Haralick, R. M.; Shapiro, L. G.: *Computer and robot vision*. Addison-Wesley Longman Publishing Co., Inc. Boston, MA, USA. 1992.
- [22] Berger, J. O.: *Statistical Decision Theory and Bayesian Analysis*. Springer, 2nd ed. 1993.
- [23] Mikut, R.; Burmeister, O.; Braun, S.; Reischl, M.: The Open Source Matlab Toolbox Gait-CAD and its Application to Bioelectric Signal Processing. In: *Proc., DGBMT-Workshop Biosignalverarbeitung, Potsdam*, pp. 109–111. 2008.
- [24] Mikut, R.: *Data Mining in der Medizin und Medizintechnik*. Universitätsverlag Karlsruhe. 2008.
- [25] Tatsuoka, M. M.: *Multivariate Analysis*. New York: Macmillan. 1988.

Learning Pattern Tree Classifiers Using a Co-Evolutionary Algorithm

Robin Senge and Eyke Hüllermeier

Department of Mathematics and Computer Science
University of Marburg, Germany
{senge, eyke}@informatik.uni-marburg.de

Abstract

Pattern tree induction has recently been introduced as a novel method for classification. Roughly speaking, a pattern tree is a hierarchical, tree-like structure, whose inner nodes are marked with generalized (fuzzy) logical operators, and a pattern tree classifier consists of one such tree per class. Since a pattern tree can thus be considered as a kind of logical characterization of a class, the approach is very appealing from an interpretation point of view. Yet, as will be argued in this paper, the method that has originally been proposed for learning pattern trees is not optimal and offers scope for improvement. To overcome its disadvantages, we propose a new method which is based on the use of co-evolutionary algorithms. Experimentally, it will be shown that our approach is indeed able to outperform the original learning method in terms of predictive accuracy.

1 Introduction

Pattern tree induction has recently been introduced as a novel method for classification by Huang, Gedeon and Nikraves [1]. Roughly speaking, a pattern tree is a hierarchical, tree-like structure, whose inner nodes are marked with generalized (fuzzy) logical operators, and whose leaf nodes are assigned to input attributes. A node takes the values of its descendants as input, applies the respective operator, and submits the output to its predecessor. Thus, a pattern tree implements a recursive mapping producing outputs in $[0, 1]$. A pattern tree classifier consists of a set of pattern trees, one for each class. A query instance to be classified is submitted to each tree, and a prediction is made in favor of the class whose tree produces the highest output.

Pattern trees are interesting for several reasons, especially from an interpretation point of view. In fact, each tree can be considered as a kind of logical description of a class. Alternatively, in the context of *preference learning* [2], a tree can be seen as a utility function: Each class corresponds to a choice alternative, and the one with the highest utility is selected.

Even though first experiments seem to suggest that pattern tree classifiers perform reasonably well in terms of predictive accuracy, the learning algorithm originally proposed in [1] is arguably not optimal. In particular, as will be explained in more detail later on, it translates training examples into examples for each tree in a questionable way. Besides, it expands trees by using a simple greedy strategy and, therefore, is susceptible to local optima. In this paper, we therefore propose an alternative method for training pattern tree classifiers which is based on the idea of optimization via co-evolution.

The rest of the paper is structured as follows. Section 2 gives a short introduction to pattern tree induction. Section 3 is devoted to our novel method of co-evolutionary pattern tree learning. Experimental results are presented in Section 4. The paper ends with some concluding remarks and an outlook on future work in Section 5.

2 Pattern Trees

In this section, we briefly describe pattern trees and the original learning algorithm; for technical details, we refer to [1]. Subsequently, we discuss some deficiencies of this method and motivate an alternative approach.

2.1 Tree Structure and Components

We proceed from the common setting of supervised learning and assume an attribute-value representation of instances, which means that an instance is a vector

$$\mathbf{x} \in \mathcal{X} = \mathcal{X}_1 \times \mathcal{X}_2 \times \dots \times \mathcal{X}_m ,$$

where \mathcal{X}_i is the domain of the i -th attribute A_i . Each domain \mathcal{X}_i is discretized by means of a fuzzy partition, that is, a set of fuzzy subsets $F_{i,j}$ of \mathcal{X}_i such that $\sum_j F_{i,j}(x) > 0$ for all $x \in \mathcal{X}_i$ (recall that a fuzzy set $F_{i,j}$ is an $\mathcal{X}_i \rightarrow [0, 1]$ mapping). The $F_{i,j}$ are often associated with linguistic labels (such as “small” or “large”) and then also referred to as *fuzzy terms*. Each instance is associated with a class label

$$y \in \mathcal{Y} = \{y_1, y_2, \dots, y_k\} .$$

A training example is a tuple $(\mathbf{x}, y) \in \mathcal{X} \times \mathcal{Y}$.

Unlike decision trees, which assume an input at the root node and output a class prediction at each leaf, pattern trees process information in the reverse direction. The input of a pattern tree is entered at the leaf nodes. More specifically, a leaf node is labeled by an attribute A_i and a fuzzy subset F of the corresponding domain \mathcal{X}_i . Given an instance $\mathbf{x} = (x_1, \dots, x_m) \in \mathcal{X}$ as an input, the node produces $F(x_i)$ as an output, that is, the degree to which x_i is in F . This degree of membership is then propagated to the parent node.

Internal nodes are labeled by generalized logical or arithmetic operators, including

- t-norms and t-conorms [3],
- weighted and ordered weighted average [7, 9].

A t-norm is a generalized conjunction, namely a monotone, associative and commutative $[0, 1]^2 \rightarrow [0, 1]$ mapping with neutral element 1 and absorbing element 0. Likewise, a t-conorm is a generalized disjunction, namely a monotone, associative and commutative $[0, 1]^2 \rightarrow [0, 1]$ mapping with neutral element 0 and absorbing element 1. Some examples of t-norms and t-conorms are shown in Table 1.

Name	T-Norm	Code
MIN	$\min\{a, b\}$	MIN
Algebraic	ab	ALG
Lukasiewicz	$\max\{a + b - 1, 0\}$	LUK
EINSTEIN	$\frac{ab}{2-(a+b-ab)}$	EIN
Name	T-CoNorm	Code
MAX	$\max\{a, b\}$	MAX
Algebraic	$a + b - ab$	COALG
Lukasiewicz	$\min\{a + b, 1\}$	COLUK
EINSTEIN	$\frac{a+b}{1+ab}$	COEIN

Table 1: Fuzzy Operators

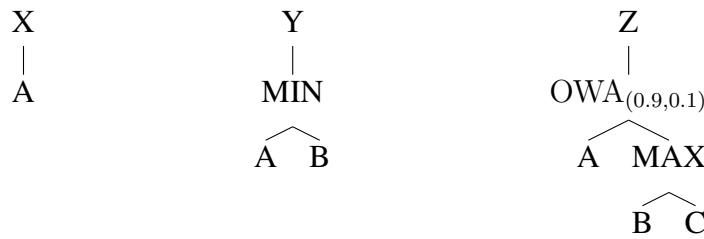


Figure 1: Pattern Tree Examples

An ordered weighted average (OWA) combination of k values $v_1 \dots v_k$ is defined by

$$\text{OWA}_w(v_1 \dots v_k) \stackrel{\text{df}}{=} \sum_{i=1}^k w_i \cdot v_{\tau(i)}, \quad (1)$$

where τ is a permutation of $\{1 \dots k\}$ such that $v_{\tau(1)} \leq v_{\tau(2)} \leq \dots \leq v_{\tau(k)}$ and $w = (w_1 \dots w_k)$ is a weight vector satisfying $w_i \geq 0$ for $i = 1 \dots k$ and $\sum_{i=1}^k w_i = 1$. Thus, just like the normal weighted average (WA), an OWA operator is parameterized by a set of weights.

Note that for $k = 2$, (1) is simply a convex combination of the minimum and the maximum. In fact, the minimum and the maximum operator are obtained, respectively, as the two extreme cases of (1): $w_1 = 1$ yields $\text{OWA}_w(v_1 \dots v_k) = v_{\tau(1)} = \min(v_1 \dots v_k)$ and $w_k = 1$ gives $\text{OWA}_w(v_1 \dots v_k) = v_{\tau(k)} = \max(v_1 \dots v_k)$. Therefore, the class of OWA operators nicely “fills the gap” between the largest conjunctive combination, namely the minimum t-norm, and the smallest disjunctive combination, namely the maximum t-conorm.

The result of an evaluation of an internal node is again propagated to its parent and so forth. The output produced by a tree is the output of its root node. Fig. 1 shows some examples.

A pattern tree classifier consists of a set of pattern trees pt_i , $i = 1, 2, \dots, k$, one for each class. Given a new instance \mathbf{x} to be classified, a prediction is made in favor of the class whose tree produces the highest score:

$$\hat{y} = \arg \max_{y_i \in \mathcal{Y}} pt_i(\mathbf{x}) \quad (2)$$

2.2 Pattern Tree Induction

Following [1], pattern trees are build one by one independently of each other. For each class, the induction method carries out the following steps:

1. Initialize with primitive pattern trees
2. Filter candidates by evaluation of their similarity to the class
3. Check stopping criterion
4. Recombine candidates using a set of fuzzy operators
5. Loop at step 2

During initialization, primitive pattern trees are created. They consist of only one leaf node. The first set of candidate trees is built by creating one such primitive pattern tree for each fuzzy term of each attribute.

In the second step, the “similarity” of each candidate tree with its respective class y_i is evaluated. Roughly speaking, this is done by comparing the subset A of examples of that class with the fuzzy subset B of instances as predicted by the tree: If (\mathbf{x}, y) is a training example, then $A(\mathbf{x}) = 1$ if $y = y_i$ and $A(\mathbf{x}) = 0$ if $y \neq y_i$. Likewise, $B(\mathbf{x})$ is the output of the tree produced for the input \mathbf{x} . A and B are similar if high values $A(\mathbf{x})$ come along with high values $B(\mathbf{x})$ and vice versa. This can be quantified in terms of different measures of similarity, such as the generalized Jaccard coefficient

$$\frac{|A \cap B|}{|A \cup B|} = \sum_{\mathbf{x}_i} \frac{\min(A(\mathbf{x}_i), B(\mathbf{x}_i))}{\max(A(\mathbf{x}_i), B(\mathbf{x}_i))} .$$

Only candidate trees with a high degree of similarity are considered within the next steps. In step 4, every possible combination of two candidate trees, sticked together by one of the allowed fuzzy operators, is created (i.e., by taking the operator as a root and the two candidate trees as subtrees). All these candidates are used for the next iteration starting at step 2. The algorithm stops if a stopping criterion is satisfied, for example if the candidate trees reach a certain depth.

2.3 Discussion

The learning procedure as outlined above can be criticized for several reasons, notably the following. First, the learning algorithm implements a kind of greedy search in the hypothesis space. Since this space is extremely complex, it is likely to get stuck in local optima. Clearly, the complexity of the search space and the highly non-linear nature of the models prevents from the use of search algorithms which guarantee optimality. Yet, there is hope that better solutions can be found at the cost of an increased though still acceptable search effort. As mentioned previously, we shall resort to search methods from the field of evolutionary optimization.

Second, one may argue that the learning problem is made more difficult than necessary. In fact, as described above, the learning algorithm seeks to find, for each class y_i , a pattern tree that delivers outputs close to 1 for instances x from this class and outputs close to 0 for instances from other classes. This property is indeed a *sufficient* criterion for correct classification, but actually not a *necessary* one. Indeed, according to (2), a prediction is made by combining the outputs of all pattern trees using the $\arg \max$ operator. Therefore, a prediction is correct as soon as the true class receives the highest score. This does not mean, of course, that the score must be close to 1, while all other scores are close to 0. Trying to comply with this much stronger property will presumably lead to models that are more complex than necessary.

As an illustration, suppose that all classes are correctly characterized by simple linear functions (i.e., the trees have depth 2 and a WA operator as a root node). Combined with $\arg \max$, these functions will always produce the correct prediction, even though the outputs will not always be close to 0 and 1, respectively. Instead, more complex, non-linear models will be needed to produce these type of predictions.

To avoid making the learning problem more difficult than necessarily, we shall propose an alternative formalization in the next section.

3 Co-Evolutionary Pattern Tree Induction

3.1 Co-Evolutionary Algorithms

Evolutionary algorithms (EA) are population-based stochastic search methods which seek to optimize a solution by mimicking the process of biological evolution. They can be applied in a quite universal way and have been used in a wide spectrum of application domains.

To apply evolutionary algorithms to complex problems more efficiently, a modularization technique, referred to as *co-evolution*, has recently been proposed [5, 4]. The general idea of co-evolution is to evolve the sub-components of a (structured) solution, also referred to as *species*, in different sub-populations. To assure that the sub-components can be assembled into a globally optimal solution, the fitness of an individual in a species is evaluated by its ability to participate in a cooperative team consisting of one representative per species. The global fitness function used in this context is also referred to as the *shared domain model*.

Determining the fitness of individuals of a certain species can simply be achieved by the evaluation of collaborations formed with representatives from each of the other species. Representatives of a species can be individuals of a certain fitness, or even the whole population. Essential for the validity of an individual's fitness, which can also be seen as a measure of the individual's contribution to the overall solution, is the selection of representatives of the other species.

3.2 Pattern Tree Induction

In our concrete application of pattern tree induction, an individual is a single pattern tree. We evolve one species per class and denote by $I_i^{(t)}$ the t -th generation of the i -th species.

A hypothesis h consists of exactly one individual for each species, that is, one pattern tree for each class. As a fitness criterion (shared domain model), we use the classification accuracy of a hypothesis on the training data.

The evolutionary process comprises the following steps:

1. Initialize each species
2. Evaluate collective fitness
3. Check termination condition
4. Reproduce each species
5. Mutate each species
6. Continue at 2

3.2.1 Step 1 - Initialization

To obtain a first generation of pattern trees, random pattern trees of size 1 or 3 are created. Here, the size of a tree is defined as the number of nodes in the tree, including both, internal and leaf nodes.

3.2.2 Step 2 - Evaluation

After a new generation has been created, the fitness of all individuals of each species must be calculated. This is done by building every possible classifier, that is, every combination

$$h_{j_1, \dots, j_k}^{(t)} \stackrel{\text{df}}{=} (pt_{1, j_1}^{(t)}, pt_{2, j_2}^{(t)}, \dots, pt_{k, j_k}^{(t)}) \in I_1^{(t)} \times I_2^{(t)} \times \dots \times I_k^{(t)}$$

of pattern trees with k denoting the number of species (classes) and pt_{i, j_i} being the j_i -th pattern tree of the i -th species. The number of possible combinations is m^k , with m the size of each population (we evolve each species using the same population size).

As mentioned above, a hypothesis h is evaluated by its accuracy on the training data, $\text{acc}(h)$. Moreover, the evaluation (fitness) of a single pattern tree $pt_{j_i}^{(t)}$ is given by the best hypothesis in which it has participated:

$$F \left(pt_{j_i}^{(t)} \right) \stackrel{\text{df}}{=} \max_{j_1, \dots, j_{i-1}, j_{i+1}, \dots, j_k} \text{acc} \left(h_{j_1, \dots, j_k}^{(t)} \right) .$$

3.2.3 Step 3 - Termination

The proposed Co-EA terminates if one of the following conditions hold. First, the iteration stops after a maximum number of iterations. Second, the Co-EA also stops if no significant improvement can be achieved during a certain number of iterations.

Therefore, two thresholds δ and d have been introduced to track the accuracy improvement of the most accurate (best) hypothesis h_{best}^t of each generation. If the condition

$$acc(h_{best}^t) + \delta \geq acc(h_{best}^{t+l})$$

holds for all $l \in \{1, \dots, d\}$, iteration stops.

3.2.4 Step 4 - Reproduction

Reproduction in terms of EAs means the creation of a new generation. Therefore, individuals of the current (parent) generation are selected at random, with a probability proportional to their fitness, to form a set of parents.

To make sure that the best solution found so far is not lost, elitist selection is applied, which means that the best hypothesis of each generation is directly transferred to the new generation.

Given two individuals $pt^{(t)}$ and $pt'^{(t)}$, a cross-over operator on trees is used for reproduction, that is, to create their children $pt^{(t+1)}$ and $pt'^{(t+1)}$. To this end, one node within each tree is chosen at random and the corresponding subtrees are interchanged. Fig. 2 illustrates this operation by means of an example.

3.2.5 Step 5 - Mutation

To guarantee a proper level of diversity, a set of randomly chosen individuals are mutated after reproduction. The probability of an individual being chosen for mutation is uniform over the population and determined by the predefined mutation rate. If an individual has been chosen for mutation, one of three different mutation operators is applied. The selection of the mutation operator is again equally random.

Mutate Leaf: Randomly selects a leaf node and replaces it by a randomly chosen other leaf node. This comes down to replacing a fuzzy term F_{ij} of an attribute A_i by another term F_{lj} of another attribute A_l .

Mutate Operator: Randomly selects an internal node and replaces it with a randomly chosen different one. Since an internal node represents a fuzzy operator, this mutation changes the type of aggregation of its children. In the experiments presented in Section 4, we used the t-norms and t-conorms already presented in Table 1 and, moreover, the WA and OWA operators with weight vectors $(w_1, w_2) \in \{(0, 1), (0.2, 0.8), \dots, (1, 0)\}$.

Mutate Tree: Randomly selects a subtree and replaces it by a new, randomly created tree. This operator somehow combines the first and the second one.

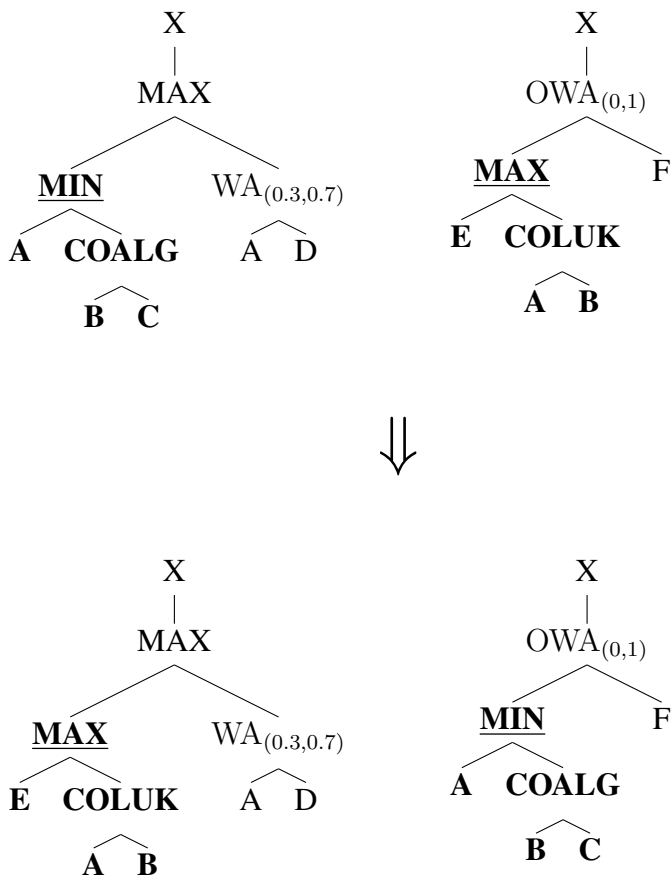


Figure 2: Cross-Over Operator

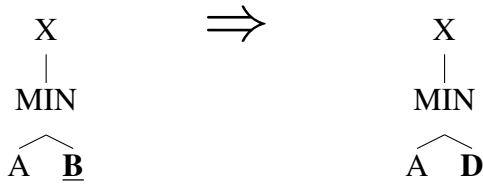


Figure 3: Example Mutate Leaf

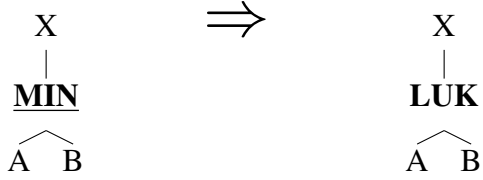


Figure 4: Example Mutate Operator

4 Experiments

We have compared the original pattern tree algorithm (PT) with our co-evolutionary variant (CPT) on a number of benchmark data sets; see Table 2 for a summary of the data sets and their properties (number of classes (#C), number of instances (#I), number of numerical (#num) and nominal attributes (#nom)).

All attributes have been fuzzified in the following way. For nominal attributes, each possible value v was encoded as a single fuzzy set $Term_v$:

$$Term_v(x) = \begin{cases} 1 & x = v \\ 0 & otherwise \end{cases}$$

For a numeric attribute with min and max being the minimum and the maximum value in the training data, two fuzzy terms Low and $High$ have been created. The corresponding membership functions are defined as follows:

$$Low(x) = \begin{cases} 1 & x < min \\ 0 & x > max \\ 1 - \frac{x-min}{max-min} & otherwise \end{cases}$$

$$High(x) = \begin{cases} 1 & x > max \\ 0 & x < min \\ \frac{x-min}{max-min} & otherwise \end{cases}$$

These fuzzy sets allow for modeling two types of influence of an attribute on the class membership, namely a positive and a negative one. (Note that all operators appearing at inner nodes of a pattern tree are monotone increasing in their arguments.)

Both PT and CPT were implemented under the WEKA Machine Learning Framework [8]. We used the following parametrization: Mutation rate 0.3, population size between 5 and 20 (depending on the number of classes), accuracy improvement thresholds $\delta = 0.001$ and $d = 1000$, maximum number of overall iterations 5000.

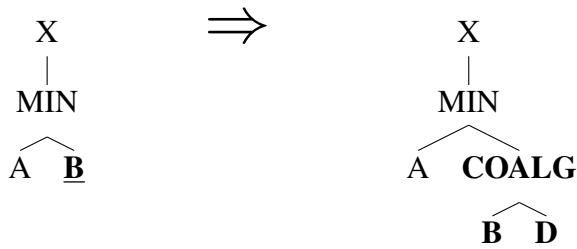


Figure 5: Example Mutate Tree

Dataset	#C	#I	#num	#nom
Australian	2	690	6	9
Authorship	4	841	69	1
Blood	2	748	4	1
Cancer	2	683	9	1
CMC	3	1473	2	8
Credit	2	690	6	10
German	2	1000	7	14
Haberman	2	306	3	1
HallOfFame	3	1320	15	2
Heart	2	270	7	7
Ionosphere	2	351	34	1
Iris	3	150	4	1
Vehicle	4	846	18	1
Wine	3	178	13	1

Table 2: Data sets and their properties

Table 3 shows the results of a 10-fold cross validation. As a non-competitor, we also included the WEKA implementation of the well-known C4.5 classifier [6], just to convey an idea about the absolute performance of pattern tree induction in comparison to state-of-the-art methods. As can be seen from the results, in a direct comparison, CPT is superior and outperforms PT most of the time. In terms of a simple sign test applied to the win/loss statistic, the superiority of CPT is indeed significant at a 10% level. Yet, the differences in classification accuracy are often rather small, and on average, they are actually smaller than expected. Thus, in summary, the experiments show that there is indeed scope for improving the simple greedy strategy underlying PT. On the other hand, they also show that, in light of its purely heuristic nature, this strategy performs comparatively well. Moreover, one has to consider that the increase in predictive accuracy achieved by CPT comes at the price of a significantly higher runtime. In fact, it is well known that evolutionary optimization is rather expensive from a computational point of view.

5 Conclusions and Future Work

In this paper, we have developed an alternative method for learning pattern tree classifiers. This work was mainly motivated by two alleged disadvantages of the original learning method: First, it is based on a simple greedy and, therefore, myopic search strategy, which is likely to get stuck in local optima. Second, it seems to solve a problem which is actually more difficult than necessary.

data set	PT	CPT	C4.5
Australian	85.2174	85.7971	86.0870
Authorship	97.5030	96.6706	93.6980
Blood	77.0053	78.8770	77.8075
Cancer	96.1933	97.0717	96.0469
CMC	52.4779	53.4284	52.1385
Credit	85.2174	85.9420	85.9420
German	72.4000	72.6000	70.7000
Haberman	73.2026	75.4902	71.8954
HallOfFame	92.3485	92.5758	92.8788
Heart	81.4815	81.8519	80.0000
Ionosphere	89.4587	90.8832	91.4530
Iris	96.6667	94.0000	96.0000
Vehicle	61.1111	61.9385	72.5768
Wine	96.0674	93.8202	93.8202

Table 3: Average accuracy of PT and CPT in a 10-fold cross validation study. The best result among these two is marked in bold font. Additionally, results are shown for C4.5.

To overcome these disadvantages, we have employed a co-evolutionary algorithm as a more sophisticated search method. Moreover, instead of maximizing a similarity function for each class separately, we take the interdependency of the individual pattern trees into account and seek to maximize classification accuracy directly.

The experimental results are in a sense ambivalent. On the one hand, they show that our approach is indeed able to improve predictive accuracy, albeit at the cost of an increased runtime. Thus, it confirms our presumption that the original learning algorithm is not optimal. On the other hand, the gains are not as high as we expected. This, of course, can have different reasons. First, one cannot exclude that, despite its obvious shortcomings, the original learning method does produce good models. In this case, it would be interesting to find out why it actually works. Moreover, it is of course possible that our approach is still far from optimal, and that better methods can be developed.

These issues will be addressed in future work. Besides, we plan to apply pattern trees to learning problems beyond simple classification, such as label ranking [2], for which they seem to be especially appealing. Finally, we are interested in analyzing the robustness of pattern tree induction, that is, the question of how sensitive the topology and parametrization of the learned trees is toward variations of the data. This question is especially important from an interpretation point of view.

References

- [1] Z. Huang, T.D. Gedeon, and M. Nikravesh. Pattern tree induction: A new machine learning method. *IEEE Transactions on Fuzzy Systems*, 16(4):958–970, 2008.
- [2] E. Hüllermeier, J. Fürnkranz, W. Cheng, and K. Brinker. Label ranking by learning pairwise preferences. *Artificial Intelligence*, 172:1897–1917, 2008.
- [3] EP. Klement, R. Mesiar, and E. Pap. *Triangular Norms*. Kluwer Academic Publishers, 2002.

- [4] Jason Morrison and Franz Oppacher. A general model of co-evolution for genetic algorithms. In *In Proceedings of the International Conference on Artificial Neural Networks and Genetic Algorithms (ICANNGA '99)*, (Portoroz, pages 262–268. Springer-Verlag, 1999.
- [5] M.A. Potter and K.A. De Jong. Cooperative coevolution: An architecture for evolving coadapted subcomponents. *Evolutionary Computation*, 8:1–29, 2000.
- [6] J.R. Quinlan. *C4.5: Programs for Machine Learning*. Morgan Kaufmann, San Mateo, CA, 1993.
- [7] B. Schweizer and A. Sklar. *Probabilistic Metric Spaces*. North-Holland, New York, 1983.
- [8] IH. Witten and E. Frank. *Data Mining: Practical machine learning tools and techniques*. Morgan Kaufmann, San Francisco, 2 edition, 2005.
- [9] R.R. Yager. On ordered weighted averaging aggregation operators in multicriteria decision making. 18(1):183–190, 1988.

Visualisierungstechniken zum Information Retrieval für Textdokumente in der Biologie

Ralf Mikut¹, Urban Liebel²

¹Forschungszentrum Karlsruhe GmbH, Institut für Angewandte Informatik,

²Forschungszentrum Karlsruhe GmbH, Institut für Toxikologie und Genetik,

E-Mail: ralf.mikut@kit.edu, urban.liebel@kit.edu

1 Motivation

Der aktuelle wissenschaftliche Stand in der Biologie ist in einer Vielzahl von Zeitschriftenartikeln (z.T. als Abstract oder Volltext verfügbar), Webseiten, Bildern und Datensatzbeschreibungen verstreut. Dabei kommen täglich neue Beiträge dazu, so dass die Verfolgung dieses Standes bereits bei kleinen Fachgebieten einen großen Aufwand erfordert.

Ein Beispiel hier ist das aktuelle Fachgebiet von Toxizitätsuntersuchungen mit Hilfe von Modellorganismen wie Zebrafischen. Die Suche nach den Begriffen "zebrafish" und "toxicity" mit der Metasuchmaschine Harvester (<http://harvester.fzk.de>) am 18. Juni 2009 ergab 104.000 Einträge bei Google, 75.800 Einträge bei Bing, 107.000 Einträge bei Clusty, 2.711 Einträge in der auf wissenschaftliche Seiten spezialisierten Suchmaschine Sciencenet (<http://sciencenet.fzk.de/>), 674 Einträge in Abstracts wissenschaftlicher Veröffentlichungen auf Medline (<http://medline.de>) und 16.200 Einträge auf <http://scholar.google.com>.

Eine rein schlagwortbasierte Suche stößt hier auf die bekannten Probleme:

1. Die Qualität der Suchergebnisse hängt stark von einer geschickten Komposition der Suchbegriffe ab. Oft werden viele irrelevante Dokumente gefunden, die die jeweiligen Schlagwörter enthalten. Hingegen werden relevante Dokumente selbst bei geringfügig abweichenden Begriffen (z.B. Synonymen) nicht gefunden. Semantische Suchanfragen (z.B. "Finde Dokumente, die Methoden der Bioinformatik zur Modellierung der Toxizität bei Zebrafischen einsetzen") sind nicht möglich.
2. Die Relevanzkriterien für das Sortieren der gefundenen Dokumente sind ein Kompromiss zwischen den Bedürfnissen aller Nutzer und des Suchmaschinenbetreibers (z.B. Werberelevanz usw.). Individuelle Kriterien wie z.B. die ausschließliche Suche nach wissenschaftlichen oder "biologischen" Dokumenten können bei allgemeinen Suchmaschinen nicht eingestellt werden, weil die Relevanzkriterien nicht offengelegt sind. Folglich erscheinen viele für einen bestimmten Nutzer relevante Dokumente erst in hinteren Positionen der sortierten Liste.
3. Die Präsentation der Ergebnisse erfolgt in Form langer Listen mit Verlinkungen zu den gefundenen Dokumenten. Die Listen sind unübersichtlich und die Links gerade bei Volltexten wissenschaftlicher Veröffentlichungen nur teilweise zugänglich. Die Suche nach ähnlichen Dokumenten ist bestenfalls für Dokumente möglich, die in der gleichen Datenbank enthalten sind. Somit ist ein Vergleich eigener nicht in der Datenbank enthaltener Dokumente gegen Dokumente in der Datenbank nicht möglich, weil die

Formate für Dokumente und die Kriterien für Ähnlichkeit nicht offengelegt sind. Die schnelle Identifikation von Dokumenten, die seit der letzten Suche mit der gleichen Suchanfrage neu hinzu gekommen sind, wird nicht oder nur bedingt (z.B. in Form von Erscheinungsjahren in *scholar.google.com*) unterstützt. Somit fällt das Auffinden von Neuerscheinungen schwer.

4. Eine Antwort auf inhaltliche Fragen wie *Haben Erkrankte mit Tumor Typ B eine erhöhte Expression von Gen A?* und eine Wissensextraktion in Form von Regeln *Wenn ein Patient Tumor Typ B hat, ist Gen A immer stärker exprimiert* ist ein ungelöstes Problem.

Ein Weg zur Lösung des ersten und zweiten Problems ist die Suche nach semantischen Informationen in einem Dokument anstelle einer reinen Schlagwortsuche. Dazu gibt es zwei wesentliche Wege:

Ein erster Weg ist das manuelle Ergänzen maschinenlesbarer semantischer Informationen für Webdokumente (siehe z.B. Ansätze zum sogenannten Semantic Web [1] oder zu semantischen Tags in Wikipedia [2]). Allerdings erfordert diese Vorgehensweise einen großen manuellen Aufwand bei der Erstellung der Dokumente. Bislang unterstützen nur wenige Dokumente diesen Standard.

Ein zweiter Weg basiert auf einer vollautomatischen Kategorisierung von Dokumenten mit Hilfe einer geeigneten Ontologie. Hierbei werden Dokumente meist durch Vektoren repräsentiert, die die Häufigkeit der Wörter im Dokument enthalten (Vektorraum-Modell bzw. "Bag-of-words"-Modell [3]). Alternative Modelle verwenden n -Gramme (n -Gramm: Gruppe von n aufeinander folgenden Zeichen) anstelle von Wörtern [4, 5]. Die Ähnlichkeit zweier Dokumente oder zwischen Dokument und Kategorien wird aus den Ähnlichkeiten der Wort- oder n -Gramm-Häufigkeiten bestimmt. Dazu werden u.a. Support-Vektor-Maschinen [6], Clustering [7,8] oder Selbstorganisierende Karten [9] verwendet. Alle genannten Ansätze haben Probleme bei der Bewertung unterschiedlicher Wörter mit ähnlicher Bedeutung. Zur Lösung dieses Problems wird seit einigen Jahren Wikipedia als Quelle für kategorisierte Dokumente mit verwandtem Vokabular eingesetzt. Außerdem dient Wikipedia zunehmend als Thesaurus, Quelle für Ontologien und für mehrsprachige Suchen (siehe [10] für eine umfassende Übersicht).

Die Dimension der benötigten Häufigkeitsvektoren ist sehr groß (alle vorkommenden Wörter oder n -Gramme). Zudem ändert sich die Größe, wenn neue Wörter hinzukommen. Deswegen werden Methoden zur Dimensionsreduzierung genutzt (z.B. Singulärwertzerlegungen, Cluster mit ähnlichen Wörtern oder zufällige Projektionen der Wörter [11] bzw. n -Gramme [4,5,12] mit Hash-Tabellen). Bei Hash-Tabellen wird jedes Wort bzw. n -Gramm mit einer Transformation auf eine eindeutige Adresse abgebildet. Wenn die Anzahl der möglichen Adressen gegenüber der Anzahl der möglichen Wörter groß ist, reduziert sich die Anzahl der Kollisionen (verschiedene Wörter mit der gleichen Hash-Adresse) auf ein akzeptables Maß. Hash-Tabellen sparen Speicherplatz, generieren Dokumentrepräsentationen gleicher Länge und erleichtern den schnellen Zugriff in großen Datenbanken. Eine weitere Strategie ist die Entwicklung kleiner Hash-Tabellen, bei denen korrelierte Wörter und n -Gramme auf das gleiche Hash-Element abgebildet werden. Dazu eignen sich die Hauptkomponentenanalyse oder Multidimensional Scaling [13]. In [14] wird eine Hierarchie von Hash-Tabellen für ganze Dokumente eingesetzt, die in einem 20 oder 30 Bit langen Binärcode mit ähnlichen Hash-Codes für ähnliche Dokumente resultiert.

Die Option zur Suche nach Kategorien eröffnet wohl das größte Potenzial für semantische Suchanfragen. Ein Beispiel ist die Suche nach medizinischen Publikationen auf der Webseite www.gpubmed.org (Transinsight GmbH, Dresden, Germany). Hier wurden vorhandene Ontologien (GO: Gene Ontology, MeSH: Medical Subject Headings, UniProt: Universal Protein Resource) zum Kategorisieren von Dokumenten verwendet. Die Datenbank wird von der U.S. National Library of Medicine und dem National Institutes of Health (<http://www.ncbi.nlm.nih.gov/pubmed/>) bereitgestellt und umfasst Abstracts von Zeitschriftenartikeln. Die gefundenen Dokumente können nach Kategorien, Autoren, Orten, Schlagwörtern, Erscheinungsjahren usw. eingeteilt werden, wobei zusätzliche Statistiken und Auswertungen (wie z.B. Kooperationsnetze von Autoren usw.) zur Verfügung gestellt werden [15, 16]. Eine ähnliche Variante mit etwas reduziertem Leistungsumfang, aber ebenfalls mit Ontologien, existiert auch für Webseiten (GoWeb).

Zur Lösung des dritten Problems bietet sich eine semantisch orientierte Visualisierung einer größeren Sammlung von Dokumenten anstelle einer Auflistung an. Hierzu wurden unter anderem Graphen mit semantisch ähnlichen Dokumenten in unterschiedlichen Knoten [17], zweidimensionale Karten mit unterlegten Schlagwörtern zur Orientierung [18] oder Scatterplots mit farbcodierten Kategorien mit einer stochastischen Einbettung von Nachbarn [14] vorgeschlagen.

Das vierte Problem ist nur langfristig lösbar, weil es die höchsten Ansprüche an die semantische Analyse der Dokumente stellt. Erste Ansätze sind domänenspezifisch, z.B. zur Extraktion von Genen, Protein-Protein-Interaktionen und Protein-Funktionen in der molekularen Biologie (siehe [19, 20] für eine Übersicht). Zu diesen Themen finden regelmäßig Wettbewerbe wie BioCreAtIve [21, 22] statt, um die vorgeschlagenen Verfahren zu vergleichen.

Der Aufbau freier Suchmaschinen (wie z.B. YaCy, <http://yacy.net/>) mit einer offenen Indizierung eröffnet mittelfristig Perspektiven, Suchtechniken zu verbessern und insbesondere die Ergebnispräsentation stärker auf Biologen als Nutzergruppe zuzuschneiden. Allerdings sind bislang zwei Kernprobleme nicht gelöst:

- Es fehlt ein generell akzeptiertes Format zur Repräsentation von Dokumenten. Dieses Format sollte eine Kategorisierung ermöglichen, offen, recheneffizient, speicherplatzsparend, sprachunabhängig, plattformunabhängig, nachträglich und vollautomatisch berechenbar für existierende Dokumente sowie einsetzbar für beliebige Textdokumente inkl. Webseiten, Datensatzbeschreibungen, Volltexte und Abstracts von Artikeln (auch für nicht frei verfügbare Volltexte aus kommerziellen Datenbanken) sein.
- Es fehlt eine generell akzeptierte, einfach verständliche Technik zur Visualisierung der Semantik eines Dokuments.

Das Ziel dieses Beitrags besteht darin, neue Ansätze zur Lösung beider Probleme vorzuschlagen und diese Ansätze in ein Konzept einzubetten, das viele der oben diskutierten bereits bekannten Strategieelemente enthält (Abschnitt 2). Das Konzept basiert auf einem zweidimensionalen Hash-Ansatz, der Dokumente einheitlich als Bilder repräsentiert (Abschnitt 2.1). Als Kandidat für eine umfassende und regelmäßig aktualisierte Ontologie werden Wikipedia-Kategorien vorgeschlagen (Abschnitt 2.2). Anschließend werden Algorithmen zur Berechnung von repräsentativen Bildern für Kategorien (Abschnitt 2.3) und passende Maße für die Ähnlichkeit zwischen Dokumenten und Kategorien (Abschnitt 2.4) präsentiert, die als Basis für eine semantische Visualisierung dienen. Abschließend werden Implementierungsaspekte diskutiert (Abschnitt 2.5). Die vorgeschlagenen Methoden werden dann auf biologische Dokumente angewendet (Abschnitt 3).

2 Methoden

2.1 Einheitliche Repräsentation von Dokumenten als Hash-Bilder

In einem ersten Schritt werden alle Wörter ($n = 1, \dots, N_w$) eines Dokuments D_k extrahiert. Dazu werden bestimmte Zeichen als Wortgrenzen interpretiert (Leerzeichen, Punkte, Frage- und Ausrufezeichen, Punkte, Kommas, Semikolons, Klammern, Zeilenende). In einem Wort werden nur die Zeichen "a-z" (ASCII 97-122), "0-9" (48-57), "-" (45) und "_" (95) akzeptiert. ASCII Zeichen für Großbuchstaben "A-Z" werden in Kleinbuchstaben "a-z" umgewandelt. Alle anderen Zeichen (darunter die deutschen Umlaute) werden durch '_' (95) ersetzt. Diese Ersetzung erleichtert den Umgang mit verschiedenen Codierungen für ASCII (z.B. Codierung 25, 188 oder 255 für 'ß'). Somit wird das Wort "Zebra-bärbling" in "Zebra_b_rbling" abgebildet. Lange Wörter werden auf eine Länge von L_{max} Zeichen begrenzt (hier: $L_{max} = 50$). Wörter mit nur einem Zeichen werden ignoriert.

Die Häufigkeiten der Wörter in einem Dokument werden in einer Hash-Tabelle F mit der Dimension $H \times H$ (hier: $H = 512$) abgelegt. Für jedes Wort in einem Dokument wird $F_{c_1[n], c_2[n]}$ um Eins inkrementiert. Die Hash-Adresse des Worts $c_d[n]$ wird durch eine Multiplikation des ASCII Codes $w_l[n]$ für das l . Zeichen im n . Wort (Wortlänge: L Zeichen) mit einem oder mehreren ($d = 1, \dots, D$, hier $D = 2$) Hash-Vektoren h_d berechnet:

$$c_d[n] = 1 + \text{rem} \left(\sum_{l=1}^L w_l[n] \cdot h_{ld}, H \right). \quad (1)$$

Der Operator rem berechnet den Rest einer Ganzzahldivision. Die verwendeten Hash-Parameter h_{ld} wurden mit einem Zufallsgenerator ermittelt und sind in Tabelle 1 angegeben. Eine einheitliche Verwendung dieser Parameter sorgt für eine einheitliche Repräsentation der Dokumente.

Beispielsweise wird das Wort "Cell" zunächst in Kleinschreibung umgewandelt: "cell". Der resultierende ASCII-Code lautet $w = (99, 101, 108, 108)$. Die Abbildung in die Hash-Tabelle mit $H = 512$, $D = 2$ ergibt

$$\begin{aligned} c_1 &= 1 + \text{rem}(486 \cdot 99 + 311 \cdot 101 + 456 \cdot 108 + 234 \cdot 108, 512) = 446 \\ c_2 &= 1 + \text{rem}(118 \cdot 99 + 249 \cdot 101 + 390 \cdot 108 + 9 \cdot 108, 512) = 52. \end{aligned}$$

Für jedes Auftreten von "cell" oder "Cell" wird der Wert von $F_{446,52}$ um Eins erhöht. Die gewählte Hash-Tabelle besteht aus $H^2 = 262144$ Elementen. Diese Größe reduziert das Risiko zufälliger Kollisionen, allerdings ist mit einer bestimmten Anzahl von Kollisionen zu rechnen (Beispiel: Hash-Element $F_{51,383}$ für die Wörter "brain" und "Canada").

Diese Hash-Strategie ist auf textbasierte Dokumente in beliebigen Sprachen und Dateiformaten (z.B. nur Text, extrahierter Text aus PDF- oder Word-Dokumenten, HTML usw.) anwendbar. Sogenannte Stoppwörter (häufige Wörter wie Artikel, verbleibende Steuerelemente wie HTML-Tags) werden hier noch nicht unterdrückt. Diese Aufgabe wird später durch die Ähnlichkeitsmaße in Abschnitt 2.4 mit gelöst. Eine Extraktion des Wortstamms durch das Abschneiden von Endungen wie "-en" (deutsch) bzw. "-ing", "-s" (englisch, siehe z.B. [23]) ist zwar prinzipiell möglich, wird aber wegen der angestrebten Sprachunabhängigkeit und der Reduzierung der Auswirkungen von Hash-Kollisionen bewusst nicht durchgeführt.

Position eines Zeichens l	Hash-Parameter h_{l1}	Hash-Parameter h_{l2}	Position eines Zeichens l	Hash-Parameter h_{l1}	Hash-Parameter h_{l2}
1	486	118	26	99	349
2	311	249	27	155	277
3	456	390	28	77	357
4	234	9	29	194	440
5	421	228	30	437	304
6	315	405	31	254	461
7	472	378	32	421	330
8	90	208	33	419	338
9	479	469	34	175	148
10	210	458	35	175	273
11	30	181	36	372	158
12	416	5	37	429	291
13	71	104	38	190	360
14	102	309	39	280	228
15	139	102	40	356	318
16	8	382	41	407	490
17	228	477	42	268	451
18	239	214	43	89	502
19	433	269	44	139	129
20	104	344	45	448	378
21	429	10	46	70	6
22	349	194	47	458	102
23	426	257	48	153	339
24	363	220	49	146	240
25	156	97	50	33	506

Tabelle 1: Parameter für die Berechnung der Hash-Werte in (1)

In einem nächsten Schritt wird für jedes Dokument aus der Hash-Tabelle der Häufigkeiten F ein korrespondierendes Bild generiert, das als Grauwert-Matrix I abgespeichert wird. Dazu wird jede Hash-Adresse als Pixel-Code interpretiert. Die Helligkeit des Pixels der Position (x, y) wird mittels

$$I_{xy} = \text{ceil} \left(P_{max} \cdot \frac{\log(1 + F_{xy})}{\max_{x,y} \log(1 + F_{xy})} \right) \quad (2)$$

berechnet. Der Operator ceil rundet den Wert auf, um für alle vorkommenden Wörter einen Mindestwert von Eins zu garantieren. Der Logarithmus wurde gewählt, um die starken Unterschiede der Worthäufigkeiten zwischen 1 und einigen Tausend geeignet abzubilden. Der maximale Grauwert wird für jedes Dokument auf einen einheitlichen Wert P_{max} (hier: $P_{max} = 255$ als Codierung für weiß) gesetzt. Daraus ergibt sich eine 8-Bit-Codierung, siehe Beispiel in Abschnitt 3.1. Wenn für die korrespondierende Worthäufigkeit $F_{xy} = 0$ gilt, erhält das Pixel einen Grauwert $I_{xy} = 0$ und erscheint schwarz. Die Anzahl der nicht schwarzen Pixel ist eine untere Abschätzung für die Anzahl verschiedener Wörter in einem Dokument, eventuelle Unterschiede resultieren aus Hash-Kollisionen.

Die resultierenden Bilder werden im weitverbreiteten Portable Network Graphics (PNG) Format abgespeichert. PNG ist ein Bitmap-Format mit einer verlustfreien Kompression. Bei den verwendeten 8-Bit-Grauwerten sind die Bilder relativ speicherplatzspa-

rend. Beispielsweise wird ein Verzeichnis mit kurzen Abstracts (Größe der Abstracts: 1.30 ± 0.47 kByte) auf PNG-Bilder der Größe 0.70 ± 0.10 kByte abgebildet. Ein Verzeichnis mit PDF-Dokumenten (Größe 1.04 ± 2.52 MByte) wird zunächst in extrahierte Textdokumente der Größe 75.8 ± 143.6 kByte und dann in PNG-Bilder der Größe 4.92 ± 3.40 kByte konvertiert. Das größte Dokument (ein komplettes Buch mit 258 Seiten, PDF-Größe 7.97 MByte) hat als PNG-Bild eine Größe von 30 kByte.

Eine Rückübersetzung von Pixeln in Wörter ist für die Analyse von dominanten Wörtern in Bildern interessant. Für diese Aufgabe wird bei der Umwandlung von Dokumenten in Bilder ein Wörterbuch mit korrespondierenden Wörtern zu Pixel-Adressen angelegt. Das Wörterbuch hat allerdings einige Mehrfachzuordnungen durch die angesprochenen Hash-Kollisionen.

2.2 Definition von Kategorien

Kategorien werden im Folgenden über eine Menge an zugehörigen Bildern definiert, die Dokumente oder Subkategorien repräsentieren. Dazu bieten sich zwei Varianten an:

- Erstellung von *nutzerspezifischen* Kategorien (z.B. auf Basis einer Verzeichnisstruktur mit enthaltenen Dokumenten, wobei die Verzeichnisse als Kategorien dienen),
- Verwendung von *standardisierten* Kategorien (z.B. auf Basis der Kategorien von Wikipedia, siehe [24]).

Wikipedia enthält eine große Sammlung kategorisierter Dokumente in verschiedenen Sprachen und deckt einen beträchtlichen Teil des vorhandenen Wissens in der Welt ab. Wikipedia wird ständig aktualisiert und erweitert, was eine regelmäßige Aktualisierung der Kategorien ermöglicht (z.B. in Form jährlicher Aktualisierungen).

Die hier verwendete Kopie der englischen Wikipedia vom November 2008 umfasst ca. 2600000 Dokumente und 384544 als gerichteter Graph organisierte Kategorien. Auf der obersten Ebene existieren 25 Kategorien (engl. *main topic classifications*): *mathematics, people, science, music, law, history, geography, culture, agriculture, politics, social sciences, nature, technology, education, computing, health, business, belief, humanities, chronology, visual arts, crafts, environment, arts, language*. Der Graph ist nicht streng hierarchisch, z.B. ist *applied sciences* sowohl auf der obersten Ebene als auch als Kind von *science* auf der 2. Ebene vertreten. In solchen Fällen wurde die tiefere Ebene verwendet. Die Kategorien ändern sich von Zeit zu Zeit. Beispielsweise existierten im August 2009 nur noch 22 Kategorien auf der obersten Ebene (neu: *culture*, auf untere Ebenen verschoben: *music, social sciences, visual arts, crafts*). Die aktuellste Version ist unter http://en.wikipedia.org/wiki/Kategorie:Main_topic_classifications verfügbar.

Für die folgenden biologischen Anwendungen wurde ein Kategorie-Baum mit 346 Kategorien eingesetzt (Basis: November 2008). Er enthält alle 25 Kategorien der obersten Ebene und ausgewählte Subkategorien bis zur 5. Ebene. Die 2. Ebene umfasst 21 Subkategorien von *science* (z.B. *applied science, computational science, cybernetics*), die 3. Ebene 21 Subkategorien von *applied science* (z.B. *management*) und 5 Subkategorien von *natural science* (z.B. *biology*). Auf Ebene 4 gibt es 38 Subkategorien von *biology* (z.B. *genetics*) und auf Ebene 5 236 weitere Subkategorien der Ebene 4 (z.B. *viral life cycle*). Einige allgemeine Kategorien wurden ignoriert, z.B. Kategorien mit Strings wie "Articles with", "All pages needing" usw. Für die Ebene 0 wird eine zusätzliche Basiskategorie *total* eingeführt, die alle Kategorien der obersten Ebene zusammenfasst.

2.3 Repräsentation von Kategorien als Hash-Bilder

Im Rahmen des vorgeschlagenen Konzepts werden Kategorien ebenfalls als Bild repräsentiert. Die Kategorie-Bilder werden durch ein gewichtetes Mittel aller zugehörigen Bilder für Dokumente und Subkategorien ermittelt:

$$\mathbf{I}(\mathbb{C}_c) = \text{round} \left(\frac{\sum_{k \text{ mit } \mathbf{I}_k \in \mathbb{C}_c} w_k \cdot \mathbf{I}_k}{\sum_{k \text{ mit } \mathbf{I}_k \in \mathbb{C}_c} w_k} \right). \quad (3)$$

$\mathbf{I}_k \in \mathbb{C}_c$ bedeutet, dass das Bild \mathbf{I}_k zur Kategorie \mathbb{C}_c gehört. \mathbf{I}_k repräsentiert entweder ein zugehöriges Textdokument D_k (z.B. *cell_biology.txt* für die Kategorie *cell biology*) oder eine Subkategorie \mathbb{C}_k (z.B. *cell signaling* als Subkategorie von *cell biology*). Der Operator *round* rundet alle Pixelwerte. Anstelle des Rundungsoperators, der u.U. seltene Pixel mit Werten zwischen Null und Eins auf Null abrundet, kann auch der immer aufrundende Operator *ceil* verwendet werden, der seltene Pixel erhält.

w_k ist ein Wichtungsfaktor für das k . Bild. Solche Wichtungsfaktoren bevorzugen Bilder mit mehr Information, z.B. auf der Basis längerer Dokumente oder mit vielen Subkategorien. Ein guter Indikator ist der kleinste Grauwert mit einem Wert größer Null in einem Bild $P_{min}(\mathbf{I}_k)$. Er fällt mit der Länge des Dokuments (einmal vorkommende Wörter mit kleinen relativen Häufigkeiten) und mit der Anzahl von Dokumenten in einer Subkategorie (seltene Wörter, die nur in einer Subkategorie vorkommen) durch Anwenden von (3). Deshalb wird der folgende Wichtungsfaktor genutzt:

$$w_k = 1/P_{min}(\mathbf{I}_k). \quad (4)$$

Dennoch enthalten einige Kategorien nur wenige Dokumente und Subkategorien. Die resultierenden Bilder haben dann relativ große Grauwerte, was bei der späteren Ähnlichkeitsberechnung zu unerwünschten Präferenzen zu diesen schlecht abgedeckten Kategorien führen kann. Deshalb werden nach der Kategorie-Berechnung die Bilder mit Grauwerten $\{0, P_{min}, \dots, P_{max}\}$ auf Grauwerte von $\{0, 1, \dots, P_{max}\}$ normalisiert (hier: $P_{max} = 255$). Das erfolgt durch die Operation:

$$\mathbf{I}_{norm} = \text{round} \left(\frac{P_{max}}{(P_{max})^{\alpha_{norm}}} \cdot \mathbf{I}^{\alpha_{norm}} \right) \text{ mit } \alpha_{norm} = \frac{\log(P_{max})}{\log(P_{max}) - \log(P_{min})}. \quad (5)$$

2.4 Ähnlichkeit zwischen Dokumenten und Kategorien

Das nächste Ziel ist die Bestimmung paarweiser Ähnlichkeiten zwischen *Bildern*, die einzelne Dokumente und Kategorien repräsentieren. Die Hauptanwendung ist die Kategorisierung von Dokumenten.

Klassische Ähnlichkeitsmaße wie Worthäufigkeit in einem Dokument vs. Inverse Häufigkeit des Vorkommens von Wörtern in allen Dokumenten TF-IDF (engl. term frequency - inverse document frequency) [7] sind nicht direkt nutzbar, weil in (2) bereits logarithmische Verhältnisse relativer Häufigkeiten berechnet werden und die benötigten Werte für TF folglich nicht mehr verfügbar sind. Außerdem steht die Anzahl der Dokumente in der Datenbank, in denen ein Wort wenigstens einmal vorkommt (IDF), nicht zur Verfügung. Deshalb werden beide Maße durch die verwandten Maße der Grauwerte in einem

Dokument bzw. einer Kategorie (statt TF) und den inversen Grauwert für das Bild der Basiskategorie $I(\mathbb{C}_0)$ (statt IDF) ersetzt. Dieses Bild stellt einen mittleren Grauwert für alle Kategorien dar und hängt stets von der Sprache und den existierenden Kategorien ab. Daraus entsteht eine Wichtungsmatrix \mathbf{W} mit Elementen

$$W_{xy} = \frac{1}{1 + I_{xy}(\mathbb{C}_0)}, \quad (6)$$

die seltene Wörter mit niedrigen Grauwerten bevorzugt. Alternative Wichtungsstrategien (z.B. Verhältnisse zwischen verschiedenen Kategorien zum Identifizieren charakteristischer Pixel) wurden bislang nicht umfassend untersucht.

Ein Maß für die Ähnlichkeit zwischen zwei Bildern sind hohe Grauwerte für gleiche Pixel. Dabei sind nur die Pixel bedeutsam, die heller als im Bild der Basiskategorie und die in der Basiskategorie möglichst selten sind. Diese beiden Eigenschaften werden durch Subtrahieren des Bildes der Basiskategorie und Wichten mit (6) umgesetzt:

$$I_{xy}^* = \max(0, \text{round}((I_{xy, Norm} - I_{xy}(\mathbb{C}_0)) \cdot W_{xy})). \quad (7)$$

Die Operation wird für jedes Dokument- und jedes Kategorie-Bild einmal ausgeführt, wobei die mit (5) normalisierten Bilder verwendet werden. Dabei ist zu beachten, dass wegen des Abspeicherns als PNG wieder ganzzahlige Grauwerte entstehen. Der Rundungsoperator setzt eine relativ große Anzahl von Pixeln mit kleinen Grauwerten auf einen Grauwert Null, sofern sie im Bild der Basiskategorie keinen Wert von Null haben, was zu $w_{ik} < 1$ führt. Der resultierende Effekt wirkt wie eine Merkmalsreduktion.

Wenn dieser Effekt nicht erwünscht ist (z.B. bei sehr kurzen Dokumenten, bei denen auch einzelne Wörter wichtig sind), kann anstelle von (7) für die Dokument-Bilder auf die Normierung mit (5) verzichtet werden:

$$I_{xy}^* = \max(0, \text{round}((I_{xy} - I_{xy}(\mathbb{C}_0)) \cdot W_{xy})). \quad (8)$$

Außerdem beeinflussen zufällige Schwankungen der Worthäufigkeiten und Hash-Kollisionen die Ähnlichkeiten. Einen erheblichen Beitrag zu Hash-Kollisionen liefert die relativ große Anzahl seltener Wörter in einem Dokument (z.B. Namen, Orte, E-Mail-Adressen, spezielle mathematische Symbole, Wortfragmente durch PDF zu Text-Umwandlungen, nicht erkannte Hypertext-Tags usw.). Das Risiko ist besonders dann hoch, wenn eine Kategorie bezogen auf die Hash-Größe viele nicht schwarze Pixel aufweist. Das gesamte Risiko wird mit einem Maß für zufällige Pixel-Ähnlichkeiten abgeschätzt:

$$S_{rand,c} = \alpha_{Heur,abs} + \frac{\alpha_{Heur,rel}}{H^2} \sum_{x=1}^H \sum_{y=1}^H I_{xy}^*(\mathbb{C}_c). \quad (9)$$

Der Wert $\alpha_{Heur,abs}$ muss an zufällige Schwankungen der Worthäufigkeiten angepasst werden. Der zweite Term schätzt den Beitrag der Hash-Kollisionen durch den Parameter $\alpha_{Heur,rel}$, die Hash-Größe H und die Summe der Grauwerte in einem Kategorie-Bild. Im Folgenden werden für alle Kategorien die einheitlichen Einstellungen $\alpha_{Heur,abs} = 0.01, \alpha_{Heur,rel} = 5$ verwendet.

Das Maß für zufällige Pixel-Ähnlichkeiten in $S_{rand,c}$ in (9) muss nur einmal pro Kategorie berechnet werden. Es schätzt den Beitrag *pro Pixel* in einem Dokument ab, folglich wird

das korrespondierende Maß für zufällige Beiträge *pro Dokument* noch mit der Anzahl der nicht schwarzen Pixel pro Dokument $N_{Pix}(D_k)$ multipliziert:

$$S_{rand,kc} = \alpha_{min} + S_{rand,c} \cdot N_{Pix}(D_k). \quad (10)$$

Der Wert α_{min} (hier: $\alpha_{min} = 2$) trägt dazu bei, Probleme mit extrem kurzen Dokumenten zu vermeiden. Im nächsten Schritt misst

$$S_{cooc,kc} = \sum_{x=1}^H \sum_{y=1}^H \min(I_{xy}^*(D_k), I_{xy}^*(\mathbb{C}_c), S_{rand,kc}) \quad (11)$$

die Pixel-Ähnlichkeit zwischen den Bildern des Dokuments D_k und der Kategorie \mathbb{C}_c . Alle Pixel, die in beiden Bildern nicht schwarz sind, tragen zur Ähnlichkeit bei. Der Minimum-Operator wählt den niedrigeren Grauwert beider Bilder. Der Term $S_{rand,kc}$ beschränkt den Beitrag eines Pixels auf den Wert der zufälligen Pixel-Ähnlichkeit des Dokuments, um eine Dominanz durch einzelne extrem helle Pixel zu vermeiden, die u.U. durch Hash-Kollisionen entstehen. Die Berechnung von (11) erfordert nur Integer- und Extrema-Operationen und kann auf spezielle Grafikhardware ausgelagert werden.

Aus der Pixel-Ähnlichkeit in (11) und der zufälligen Pixel-Ähnlichkeit des Dokuments in (10) berechnet sich nun die Ähnlichkeit zwischen Dokument und Kategorie:

$$S_{kc} = \max\left(0, \frac{S_{cooc,kc}}{S_{rand,kc}} - 1\right). \quad (12)$$

Ein Wert von Null bedeutet keine Ähnlichkeit, steigende Werte zeigen eine zunehmende Ähnlichkeit bezüglich des verwendeten Vokabulars an. Die Genauigkeit verbessert sich mit zunehmender Dokumentlänge und mit der Spezifik des Vokabulars einer Kategorie. Verwandte Kategorien führen zu vergleichbaren Ähnlichkeiten der Dokumente.

Eine scharfe Kategorisierung entscheidet zugunsten der ähnlichsten Kategorie:

$$\hat{c}_{opt}(\mathbf{I}_k) = \operatorname{argmax}_c S_{kc}. \quad (13)$$

Für eine hierarchische Kategorisierung eines Dokuments muss die Strategie nur leicht modifiziert werden:

1. Für jede Kategorie \mathbb{C}_c werden gezielt die charakteristischen Pixel gesucht, die häufiger als in ihren direkten Oberkategorien und deren Oberkategorien sind. Dazu wird das Bild der Basiskategorie $\mathbf{I}(\mathbb{C}_0)$ in (6) und (7) durch ein spezifisches Bild $\mathbf{I}(\mathbb{C}_{0c})$ für jede Kategorie \mathbb{C}_c ersetzt. Dieses Bild entsteht durch eine Maximum-Verknüpfung aller Bilder von direkten oder indirekten Oberkategorien mit

$$I_{xy}(\mathbb{C}_{0c}) = \max_{i \in \operatorname{Par}(c)} I_{xy}(\mathbb{C}_i). \quad (14)$$

Der Menge $\operatorname{Par}(c)$ umfasst alle Oberkategorien der c -ten Kategorie auf den höheren Hierarchieebenen, inkl. der Basiskategorie auf der nullten Ebene. Falls nur eine Ebene existiert, gilt folglich $\mathbf{I}(\mathbb{C}_{c0}) = \mathbf{I}(\mathbb{C}_0)$.

2. Start mit der höchsten Kategorie-Ebene (Ebenen-Zähler auf $L_{hierarchy} = 1$ setzen).
3. Ähnlichkeiten für alle Kategorien der aktuellen Ebene $L_{hierarchy}$ berechnen und beste Kategorie mit (13) bestimmen.

4. Alle Subkategorien (mit $L_{hierarchy} + 1$) für die beste Kategorie und alle weiteren Kategorien aus Schritt 3 mit vielversprechenden Ähnlichkeiten

$$\frac{S_{cooc,kc}}{S_{rand,kc}} > S_{thres} \quad (15)$$

bestimmen (z.B. das hier gewählt wenig selektive Niveau $S_{thres} = 0.5$, um nicht vorzeitig möglicherweise interessante Kategorien auf tieferen Ebenen auszuschließen).

5. Stopp wenn in Schritt 4 keine Subkategorien mehr gefunden werden, sonst Ebenen-Zähler erhöhen ($L_{hierarchy} = L_{hierarchy} + 1$) und mit Schritt 3 fortsetzen.

Diese Strategie liefert auf jeder Ebene die beste Kategorie. Zudem kann die Zahl der Ähnlichkeitsberechnungen reduziert werden, weil nur Ähnlichkeiten für vielversprechende Subkategorien ausgewertet werden. Bei einer Anzahl von C nicht hierarchisch angeordneten Kategorien müssen hingegen C Ähnlichkeiten bestimmt werden.

2.5 Implementierungsaspekte

Alle Algorithmen wurden in MATLAB (The MathWorks, Inc.) implementiert und in ein Text-Mining-Erweiterungspaket der MATLAB-Toolbox Gait-CAD [25] integriert. Gait-CAD kann unter <http://sourceforge.net/projects/gait-cad/> frei bezogen werden, die vorläufige Version des Erweiterungspakets ist per Mailanfrage an die Autoren erhältlich. Die Konvertierung von PDF in Textdateien wurde mit der Open-Source-Toolbox PDFTEXT (Version 3.02, Glyph & Cog, LLC) durchgeführt.

Die beschriebenen Algorithmen eignen sich gut für eine Parallelisierung und eine datenorientierte Handhabung. Das ist eine notwendige Voraussetzung für die Beherrschung großer Datenbanken mit Millionen Dokumenten wie Wikipedia und biologische Suchmaschinen.

3 Experimentelle Ergebnisse

3.1 Illustratives Beispiel

Im folgenden illustrativen Beispiel wird die Hash-Repräsentation gemäß Abschnitt 2.1 und die Kategorisierung für ein Dokument beschrieben (Titel und Abstract von [26]):

Zebrafish embryos as models for embryotoxic and teratological effects of chemicals: The experimental virtues of the zebrafish embryo such as small size, development outside of the mother, cheap maintenance of the adult made the zebrafish an excellent model for phenotypic genetic and more recently also chemical screens. The availability of a genome sequence and several thousand mutants and transgenic lines together with gene arrays and a broad spectrum of techniques to manipulate gene functions add further to the experimental strength of this model. Pioneering studies suggest that chemicals can have in many cases very similar toxicological and teratological effects in zebrafish embryos and humans. In certain areas such as cardiotoxicity, the zebrafish appears to outplay the traditional rodent models of toxicity testing. Several pilot projects used zebrafish embryos to identify new chemical entities with specific biological functions. In combination with

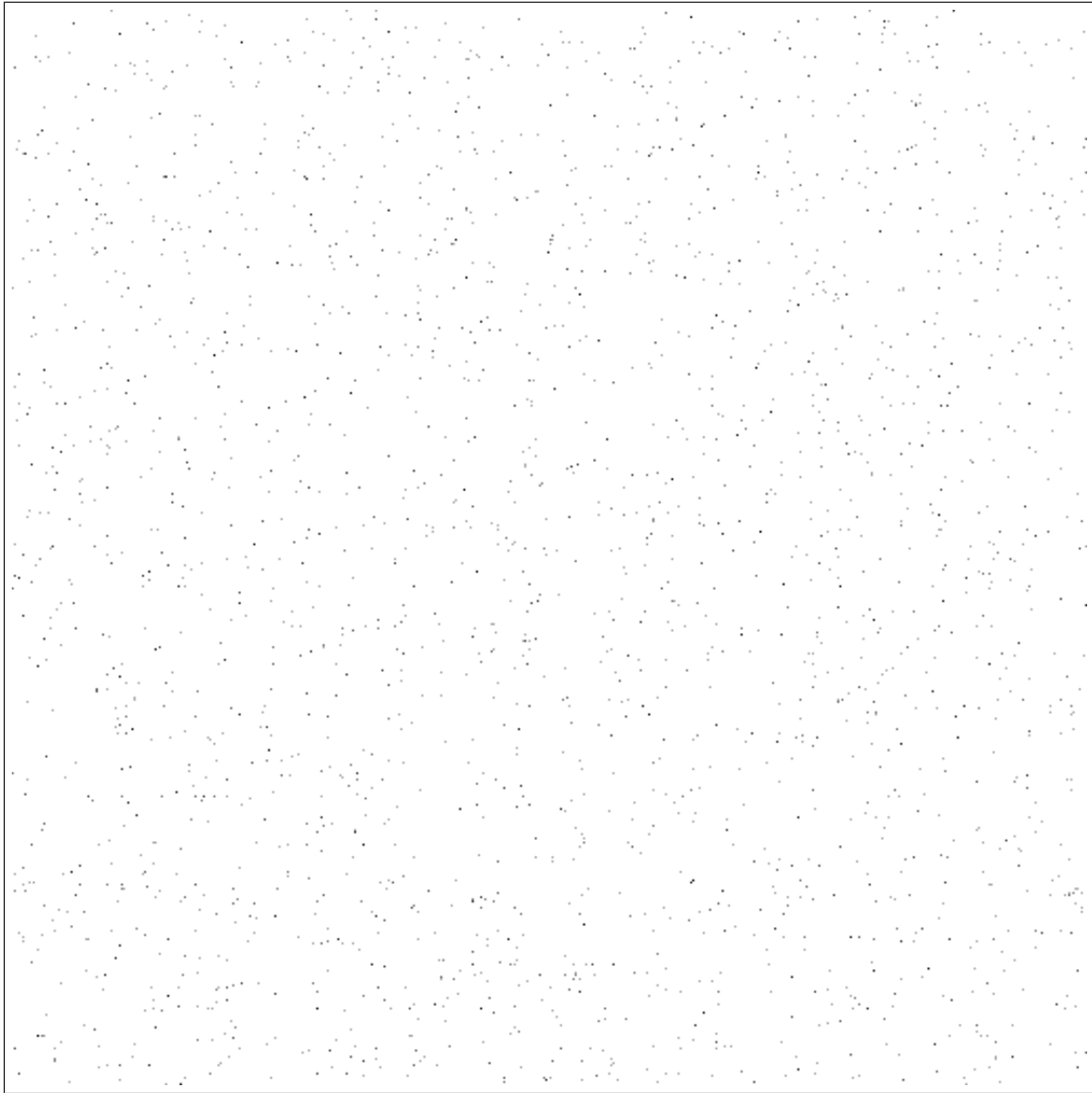


Bild 1: Resultierendes PNG-Bild für den Volltext von [26]. Die Koordinate $(x = 1, y = 1)$ befindet sich oben links, die Koordinate $(x = 512, y = 512)$ unten rechts. Zur Verbesserung der Sichtbarkeit in der Druckversion wurden die Grauwerte invertiert und mit einer Gammakorrektur bearbeitet ($\gamma = 0.5$). Wegen dieser Invertierung codieren besonders dunkle Pixel häufige Wörter.

the establishment of transgenic sensor lines and the further development of existing and new automated imaging systems, the zebrafish embryos could therefore be used as cost-effective and ethically acceptable animal models for drug screening as well as toxicity testing.

Der Abstract enthält 104 verschiedene Wörter. Das häufigste Wort ist "the" (12x, wird auf Grauwert 255 abgebildet), gefolgt von "and" und "of" (10x, Grauwert 239) sowie "zebrafish" (7x, Grauwert 207), siehe Tabelle 2. Alle einmal vorkommenden Wörter werden auf einen Grauwert von 69 abgebildet. Die unerwünschten allgemeinen Wörter werden durch Subtrahieren und Wichten mit dem Bild der Wikipedia-Basiskategorie gemäß (8) unterdrückt. Die Ergebnisse für den Volltext des gleichen Artikels sind ähnlich (Tabelle 2). Das PNG-Bild für den Volltext ist in Bild 1 dargestellt.

Die jeweils fünf besten Wikipedia-Kategorien für Abstract und Volltext in Tabelle 3 zei-

	Originalbild	Verarbeitetes Bild mit (8)
Abstract	104 nicht schwarze Pixel mittlerer Grauwert 88 1. x=121 y=127 Gray=255 the 2. x=489 y= 13 Gray=239 and 3. x=165 y= 97 Gray=239 of 4. x=185 y=140 Gray=207 zebrafish 5. x=476 y=146 Gray=194 as	83 nicht schwarze Pixel mittlerer Grauwert 50 1. x=185 y=140 Gray=207 zebrafish 2. x=136 y=271 Gray=161 embryos 3. x= 80 y= 74 Gray=110 chemicals 4. x=128 y=217 Gray=110 gene 5. x=120 y=347 Gray=110 transgenic
Volltext	2238 nicht schwarze Pixel mittlerer Grauwert 46 1. x=121 y=127 Gray=255 the 2. x=165 y= 97 Gray=251 of 3. x=249 y=357 Gray=232 in 4. x=489 y= 13 Gray=225 and 5. x=185 y=140 Gray=220 zebrafish	2121 nicht schwarze Pixel mittlerer Grauwert 34 1. x=185 y=140 Gray=220 zebrafish 2. x=136 y=271 Gray=176 embryos 3. x=128 y=217 Gray=159 gene 4. x=415 y=325 Gray=157 genes 5. x=305 y= 69 Gray=151 hpf ... 7. x= 80 y= 74 Gray=137 chemicals 15. x=120 y=347 Gray=122 transgenic

Tabelle 2: Pixel-Statistik für den Abstract und Volltext von [26]: Anzahl nicht schwarzer Pixel im Dokument, mittlerer Grauwert der nicht schwarzen Pixel, x-, y-Koordinaten und Grauwert der fünf hellsten Pixel sowie Liste der korrespondierenden Wörter aus der Rückübersetzung. Alle Statistiken sind jeweils für das originale PNG-Bild aus Abschnitt 2.1 und das verarbeitete Bild nach Subtraktion des Bildes der Basiskategorie mit nachfolgender Wichtung gemäß (8) angegeben. Die Abkürzung "hpf" bedeutet Stunden nach der Befruchtung (engl. hours post fertilization).

gen eine klare Präferenz zu biologischen Kategorien und eine große Ähnlichkeit der Kategorisierung. Die Differenzen zwischen Abstract und Volltext resultieren aus den Nuancen der Worthäufigkeit. Einerseits sind Wörter für Modellorganismen wie "zebrafish", "rodent", "animal", "model" und "models" im Abstract überproportional vertreten, wodurch die entsprechende Kategorie *model organisms* besonders stark ist. Andererseits ist charakteristisches biologisches Vokabular im Volltext häufiger.

Unter Nutzung der hierarchischen Kategorien wird der Abstract den Kategorien *science* (Ebene 1), *scientific method* (Ebene 2), *pharmaceutical sciences* (Ebene 3), *biotechnology* (Ebene 4) und *model organisms* (Ebene 5) zugeordnet. Die entsprechenden Einteilungen für den Volltext lauten *health* (Ebene 1), *scientific method* (Ebene 2), *pharmaceutical sciences* (Ebene 3), *biotechnology* (Ebene 4) und *biochemistry methods* (Ebene 5). Diese Einteilung zeigt, dass auf den unteren Ebenen nicht nur Subkategorien der besten oberen Ebene ausgewählt werden. Beispielsweise ist *scientific method* keine Subkategorie von *health*. Der vorgeschlagene Algorithmus stellt sicher, dass solche Subkategorien auch gefunden werden, indem er mittels (15) die Subkategorien aller aussichtsreichen Kategorien untersucht.

3.2 Datensatz mit Abstracts von *gopubmed.org*

Im nächsten Experiment wurde ein Datensatz mit 1000 Abstracts von *gopubmed.org* zusammengestellt. Dieser Datensatz enthält die aktuellsten 100 Abstracts für die folgenden zehn Schlagwörter, die als nutzerspezifische Kategorien verwendet werden: *antimicrobial peptides*, *biofilm formation*, *bioinformatics*, *cell signalling*, *confocal microscopy*, *data*

Kategorie	Abstract		Volltext	
	Rang	Ähnlichkeit S_{kc}	Rang	Ähnlichkeit S_{kc}
model organisms	1	4.38	12	1.41
biotechnology	2	2.96	4	2.11
scientific method	3	2.56	3	2.21
science	4	2.39	19	1.28
pharmaceutical sciences	5	2.38	1	2.43
genetics	10	2.07	5	1.91
biology	31	0.85	2	2.38

Tabelle 3: Ähnlichkeiten und Rangordnung der besten Wikipedia-Kategorien für den Abstract und Volltext von [26]

mining, high throughput microscopy, neuroprostheses, neurotoxins und *zebrafish toxicity*. Der Datensatz wurde in einen Lern- und einen Testdatensatz mit jeweils 50 Dokumenten pro Kategorie geteilt. 11 Artikel werden mehrfach für verschiedene Kategorien gefunden.

In einem ersten Versuch wurden aus den Abstracts des Trainingsdatensatzes repräsentative Bilder für die zehn nutzerspezifischen Kategorien gemäß Abschnitt 2.1 und 2.2 gelernt. Für die Abstracts des Testdatensatzes wurde das jeweils ähnlichste Kategorien-Bild bestimmt. Das Ergebnis war eine Klassifikationsgüte von 75.6 %, wobei es Zuordnungsprobleme zwischen den Kategorien *data mining* vs. *bioinformatics* sowie *confocal microscopy* vs. viele andere Kategorien wie *biofilm formation, cell signalling, und high throughput microscopy* gab. Die Ursache sind semantische Ähnlichkeiten zwischen ähnlichen Methoden sowie Methoden zur Lösung bestimmter Probleme wie konfokale Mikroskopie als Methode zur Analyse von Zellsignalen und der Entstehung von Biofilmen.

In einem zweiten Versuch wurden die Ähnlichkeiten zu ausgewählten Wikipedia-Kategorien in zweidimensionalen Scatterplots visualisiert (Bild 2). Das linke Teilbild zeigt die Ähnlichkeiten zu den Wikipedia-Kategorien *bacteriology* und *optics*, das rechte zu den Kategorien *model organisms* und *bioinformatics*. Auffällig sind insbesondere die hohen Werte für die Wikipedia-Kategorie *microscopy* für viele Dokumente der nutzerspezifischen Kategorie *confocal microscopy* sowie die hohen Werte für die Wikipedia-Kategorie *bioinformatics* für die nutzerspezifischen Kategorien *bioinformatics* und *data mining*. In beide Bilder wurden beispielhaft die Positionen des in Abschnitt 3.1 umfassend diskutierten Abstracts und Volltextes von [26] sowie des Abstracts von [27] eingezeichnet.

Interdisziplinäre Arbeiten können durch hohe Ähnlichkeiten zu mehreren semantisch verschiedenen Kategorien gefunden werden. Beispielsweise untersucht [27] Interaktionen zwischen sieben verschiedenen Spezies in Biofilmen mit konfokalen Mikroskopen. Das Dokument ist in zwei nutzerspezifischen Kategorien (*biofilm formation* und *confocal microscopy*) enthalten. Der Abstract zeigt relevante Ähnlichkeiten zu den Kategorien *bacteriology* und *optics* (als geeignete Oberkategorie für Mikroskopie) (siehe Bild 2). Außerdem kann aus den Ähnlichkeiten S_{kc} quantitativ abgeschätzt werden, dass hier beide Themen etwa in gleichem Umfang behandelt werden. Viele andere Dokumente im Datensatz haben hingegen nur Ähnlichkeit zu *optics* oder *bacteriology*. Für eine Suchanfrage nach allen bakteriologischen Untersuchungen mit mikroskopischen Methoden können in Frage kommende Dokumente mit der Maus bereichsweise selektiert und später weiter ausgewertet werden. Solche grafischen Navigationsmethoden eröffnen somit einen intuitiveren Zugang zu semantischen Suchanfragen.

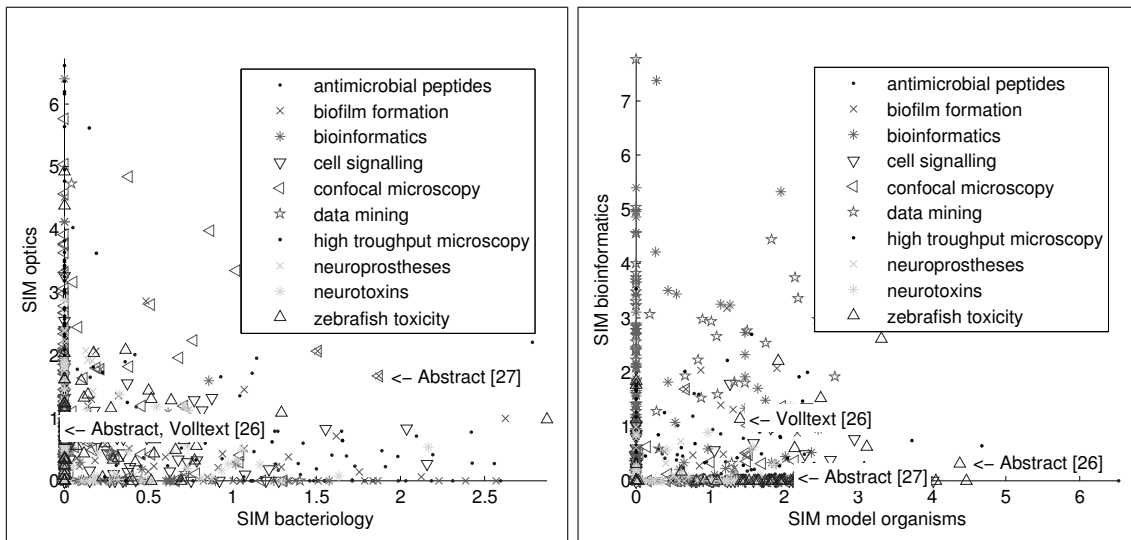


Bild 2: Zweidimensionale Scatterplots der Ähnlichkeiten S_{kc} (SIM) zu verschiedenen Wikipedia-Kategorien (jeweils dargestellt für alle Dokumente des Lern- und Testdatensatz sowie zusätzlich für den Volltext von [26]). Die zehn nutzerspezifischen Kategorien sind farb- und symbolcodiert.

4 Diskussion und Zusammenfassung

Die vorgestellte Methode umfasst einen ersten Schritt in Richtung einer verbesserten Repräsentation und Visualisierung von Dokumenten zur inhaltsbasierten Suche nach textbasierten Dokumenten in einer großen Datenbasis (engl. Information Retrieval). Die Hauptidee besteht in der *einheitlichen* Repräsentation von Dokumenten in einem zweidimensionalen Hash-Index, der Häufigkeiten von Wörtern in einem Dokument mit einer stark reduzierten 8-Bit-Auflösung als Bild codiert. Somit werden alle enthaltenen Wörter unabhängig von der Sprache und der Bekanntheit des Worts auf genau ein Hash-Element abgebildet. Die Hash-Darstellung nimmt einen bestimmten Anteil von Mehrfachzuordnungen von Wörtern zu Pixeln sowie eine reduzierte Genauigkeit bei der Repräsentation der Worthäufigkeiten in Kauf. Diese Darstellung hat den großen Vorteil, kompatibel zum gängigen PNG-Format für Rastergrafiken mit verlustfreier Bildkompression zu sein. Alle Textdokumente und daraus berechnete zusammenfassende Formate für Kategorien werden somit speichersparend als Grauwert-Bild abgelegt, was eine einfache Weiterverarbeitung ermöglicht (u.a. auch auf spezialisierten Hardwareumgebungen wie Grafikkarten).

Solche Bilder sind für Volltexte von wissenschaftlichen Verlagen mit restringiertem Zugang geeignet, weil der Volltext des Dokuments in die Ähnlichkeitsbetrachtung einbezogen werden kann, aber das PNG-Bild keine vollständige Rekonstruktion des Inhalts ermöglicht. Ein weiterer Vorteil bietet sich beim Einsatz für Suchmaschinen. Eine Suchmaschine kann einem Client auf der Basis einer Anfrage eine mittelgroße Menge an passenden PNG-Bildern zurücksenden. Die weitere Verarbeitung (z.B. eine nutzerspezifische Visualisierung oder eine Ähnlichkeitsberechnung zu einem Dokument) erfolgt erst beim Client.

Ein weiterer Beitrag umfasst das Grundkonzept, graduelle Zugehörigkeiten von Dokumenten zu Kategorien berechnen und grafisch darzustellen. Die Zuordnung von Dokumenten zu Kategorien erfolgt wahlweise standardisiert durch externe Quellen (z.B. Wikipedia) oder nutzerspezifisch aus kategorisierten Sammlungen von Dokumenten. Aus den

so ermittelten Kategorien werden ebenfalls Grauwert-Bilder im PNG-Format berechnet. Damit reduziert sich die Berechnung der graduellen Zugehörigkeiten eines neuen Dokuments auf einen Bildvergleich zwischen einem Dokument und verschiedenen Kategorien. Die Angabe gradueller Zugehörigkeiten erleichtert insbesondere das Auffinden von Dokumenten mit interdisziplinären Arbeiten, bei denen mehrere Teilgebiete einfließen. Somit kann eine Suche über eine logische Verknüpfung von Mindestzugehörigkeiten zu verschiedenen Kategorien erfolgen.

Die so entstandenen graduellen Zugehörigkeiten können auf verschiedene Art und Weise visualisiert werden. Dokumente können als Scatterplot der graduellen Zugehörigkeiten von ausgewählten Kategorien angezeigt werden.

Ein weiteres Potenzial liegt in einer Kategorisierung fremdsprachiger Dokumente (engl. CLIR: Cross-language information retrieval, vgl. Ansätze z.B. in [28, 29]). Mit einer 1:1-Übersetzung von Pixeln einer Sprache in korrespondierende Pixel einer anderen Sprache unter Nutzung eines Wörterbuchs gelingt sicherlich keine hochwertige Übersetzung, aber möglicherweise eine grobe Abschätzung des Inhalts. Hierzu muss zunächst die Sprache eines Dokuments ermittelt werden und danach eine 1:1-Übersetzungstabelle von Pixeln nach Pixeln angewendet werden. Die Übersetzungstabelle entsteht aus der Zuordnung der Adresse für das dominierende Wort für ein Pixel in der ursprünglichen Sprache des Dokuments (z.B. Zelle - $F_{385,194}$) in die Adresse für das wichtigste korrespondierende Wort in der Zielsprache (z.B. cell - $F_{446,52}$). Somit werden Grauwerte $F_{385,194}$ eines deutschen Dokuments in Grauwerte $F_{446,52}$ eines englischen Dokuments konvertiert. Somit lässt sich eine gerichtete Übersetzungstabelle in extrem kompakter Form in zwei PNG-Bildern mit einer Bittiefe von jeweils $\log_2(P_{max})$ codieren (ein Bild für die neue x-Adresse, ein Bild für die neue y-Adresse).

Danksagung: Die Autoren danken Björn Kindler für die Bereitstellung und Aufbereitung der Wikipedia-Kopie.

Literatur

- [1] Berners-Lee, T.; Hendler, J.; Lassila, O.; et al.: The Semantic Web. *Scientific American* 284(5) (2001), S. 28–37.
- [2] Völkel, M.; Kröttsch, M.; Vrandečić, D.; Haller, H.; Studer, R.: Semantic Wikipedia. In: *Proc., 15th International Conference on World Wide Web*, S. 585–594. 2006.
- [3] Salton, G.; Wong, A.; Yang, C.: A Vector Space Model for Automatic Indexing. *Communications of ACM* 18(11) (1975), S. 613–620.
- [4] Damashek, M.: Gauging Similarity with n-Grams: Language-Independent Categorization of Text. *Science* 267 (1995) 5199, S. 843–848.
- [5] Liu, Y.; Dantzig, P.; Sachs, M.; Corey, J.; Hinnebusch, M.; Damashek, M.; Cohen, J.: Visualizing Document Classification: A Search Aid for the Digital Library. *Journal of the American Society for Information Science* 51(3) (2000) 3, S. 216–227.
- [6] Wang, P.; Domeniconi, C.: Building Semantic Kernels for Text Classification using Wikipedia. In: *Proc., 4th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining*, S. 713–721. ACM New York, NY, USA. 2008.
- [7] Salton, G.: *Automatic Text Processing: The Transformation Analysis and Retrieval of Information by Computer*. Addison-Wesley. 1989.
- [8] Dhillon, I.; Modha, D.: Concept Decompositions for Large Sparse Text Data using Clustering. *Machine Learning* 42(1-2) (2001), S. 143–175.
- [9] Lagus, K.; Kaski, S.; Kohonen, T.: Mining Massive Document Collections by the WEBSOM Method. *Information Sciences* 163(1-3) (2004), S. 135–156.

- [10] Medelyan, O.; Milne, D.; Legg, C.; Witten, I.: Mining Meaning from Wikipedia. *International Journal of Human-Computer Studies* 67(9) (2009), S. 716–754.
- [11] Kaski, S.: Dimensionality Reduction by Random Mapping: Fast Similarity Computation for Clustering. In: *Proc., IEEE World Congress on Computational Intelligence*. 1998.
- [12] Cohen, J.: Recursive Hashing Functions for n-Grams. *ACM Transactions on Information Systems (TOIS)* 15 (1997) 3, S. 291–320.
- [13] Stein, B.: Principles of Hash-based Text Retrieval. In: *Proc., 30th ACM SIGIR Conf. on Research and Development in Information Retrieval*, S. 527–534. 2007.
- [14] Salakhutdinov, R.; Hinton, G.: Semantic Hashing. *International Journal of Approximate Reasoning* 50(7) (2009), S. 969–978.
- [15] Doms, A.; Schroeder, M.: GoPubMed: Exploring PubMed with the Gene Ontology. *Nucleic Acids Research* 33 (2005) Web Server Issue, S. W783.
- [16] Dietze, H.; Alexopoulou, D.; Alvers, M.; Barrio-Alvers, L.; Andreopoulos, B.; Doms, A.; Hakenberg, J.; Monnich, J.; Plake, C.; Reischuck, A.; et al.: GoPubMed: Exploring PubMed with Ontological Background Knowledge. In: *Bioinformatics for Systems Biology*, S. 385. Humana Press Inc. 2009.
- [17] Chen, C.: Visualising Semantic Spaces and Author Co-citation Networks in Digital Libraries. *Information Processing and Management* 35(3) (1999) 3, S. 401–420.
- [18] Fortuna, B.; Grobelnik, M.; Mladenić, D.: Visualization of Text Document Corpus. *Informatica* 29 (2005), S. 497–502.
- [19] Krallinger, M.; Erhardt, R.; Valencia, A.: Text-Mining Approaches in Molecular Biology and Biomedicine. *Drug Discovery Today* 10(6) (2005), S. 439–445.
- [20] Jensen, L.; Saric, J.; Bork, P.; et al.: Literature Mining for the Biologist: From Information Retrieval to Biological Discovery. *Nature Reviews Genetics* 7(2) (2006) 2, S. 119.
- [21] Hirschman, L.; Yeh, A.; Blaschke, C.; Valencia, A.: Overview of BioCreAtIvE: Critical Assessment of Information Extraction for Biology. *BMC Bioinformatics* 6 (2005) Suppl 1, S. S1.
- [22] Krallinger, M.; Morgan, A.; Smith, L.; Leitner, F.; Tanabe, L.; Wilbur, J.; Hirschman, L.; Valencia, A.: Evaluation of Text-Mining Systems for Biology: Overview of the Second BioCreAtIvE Community Challenge. *Genome Biology* 9 (2008) Suppl 2, S. S1.
- [23] Porter, M.: An Algorithm for Suffix Stripping. *Program* 14(3) (1980), S. 130–137.
- [24] Krötzsch, M.; Vrandečić, D.; Völkel, M.: Wikipedia and the Semantic Web - The Missing Links. In: *Proc., WIKIMANIA*. 2005.
- [25] Burmeister, O.; Reischl, M.; Bretthauer, G.; Mikut, R.: Data-Mining-Analysen mit der MATLAB-Toolbox Gait-CAD. *at - Automatisierungstechnik* 56(7) (2008), S. 381–389.
- [26] Yang, L.; Ho, N.; Alshut, R.; Legradi, J.; Weiss, C.; Reischl, M.; Mikut, R.; Liebel, U.; Müller, F.; Strähle, U.: Zebrafish Embryos as Models for Embryotoxic and Teratological Effects of Chemicals. *Reproductive Toxicology* 28 (2009), S. 245–253.
- [27] Bandara, H.; Yau, J.; Watt, R.; Jin, L.; Samaranyake, L.: *Escherichia coli* and its Lipopolysaccharide Modulate *in vitro* *Candida* Biofilm Formation. *Journal of Medical Microbiology* (2009). In press.
- [28] Levow, G.; Oard, D.; Resnik, P.: Dictionary-based Techniques for Cross-language Information Retrieval. *Information Processing and Management* 41(3) (2005) 3, S. 523–547.
- [29] Potthast, M.; Stein, B.; Anderka, M.: A Wikipedia-Based Multilingual Retrieval Model. In: *Proc., Advances in Information Retrieval: 30th European Conference on IR Research, ECIR*, S. 522. 2008.

An Approach to the Identification of Uncertain Surrogate Models for Complex Systems

Thomas Haag, Pascal Reuß, Michael Hanss

Institut für Angewandte und Experimentelle Mechanik,
Universität Stuttgart,
Pfaffenwaldring 9, 70550 Stuttgart
E-Mail: {haag, hanss}@iam.uni-stuttgart.de

Abstract

For the mathematical representation of systems with epistemic uncertainties, arising, for example, from a lack of knowledge or simplifications in the modeling procedure, models with fuzzy-valued parameters prove to be a suitable and promising approach. In practice, however, the determination of these parameters turns out to be a non-trivial problem, and the identification procedure to appropriately update these parameters on the basis of measured data requires the solution of an inverse problem. Against this background, an inverse method for the computation of the fuzzy-valued parameters of a model with epistemic uncertainties is presented. This method stands out due to the fact that it only uses feedforward simulations of the model, based on the transformation method of fuzzy arithmetic, along with the measurement signal. Thereby, no additional expensive model evaluations are needed, as the inverse approach is entirely based on the feedforward results of the transformation method. Unlike common approaches of parameter identification and model updating where the discrepancy between the simulation output and the measured data is minimized through a best fit of crisp parameters, the presented approach accounts for the existence of epistemic uncertainties, which are dealt with by a possibilistic approach and represented by fuzzy parameters. Random scatter in the measurement, acting as an aleatory uncertainty, is neglected in the approach as it is empirically very small in comparison with the epistemic uncertainties. The general problem in inverse computations consisting of the existence of a unique solution is addressed through the computation of a feasibility index which is also based on the output array of the transformation method only. Applied to the problem of identifying the fuzzy-valued model parameters of a mechanical system with epistemic uncertainties from its measured frequency response function, this new approach allows a restriction of the identification process to some special range of frequency where its feasibility can be guaranteed. Finally, the practicability of the method is demonstrated by approximating a 2D macro-slip friction model by two decoupled 1D macro-slip friction models.

1 Introduction

A well-known problem in the numerical simulation of real-world systems is the question of how detailed the structure of the model has to be chosen in order to appropriately represent reality. If the structure of the model does not exhibit any simplifications of the reality, crisp-valued model parameters are adequate to describe the real system and a conventional model-updating procedure can be used for their identification. Due to computational limitations, however, it is often necessary to neglect certain phenomena in the modeling phase, such as high frequency dynamics or nonlinearities. This leads to simplified and, from a computational perspective, less

expensive models, but obviously also to specific modeling errors when crisp parameters are used as the optimal ones. In order to cover these modeling errors, simplified models with uncertain parameters can be used, providing a conservative prediction of the real system behavior. Based on the fact that the inherent uncertainty is a consequence of model simplifications, it can be classified as epistemic [1]. As probability theory may not be appropriate to effectively represent epistemic uncertainties [2], the alternative strategy of quantifying epistemic uncertainties by fuzzy numbers [3, 4] is pursued in this paper. The propagation of the uncertainty through the system, i.e. the evaluation of the model with fuzzy-valued parameters, is performed by the use of fuzzy arithmetic on the basis of the transformation method [4, 5].

As an extension of the transformation method of fuzzy arithmetic, this paper proposes a method to estimate the uncertain parameters of a simplified model on the basis of the output of an advanced model, or based on a single measurement signal. The proposed method uses inverse computations that are based on the data of the transformation method. This implies that an inversion of the model equations is not needed, which enables the method to be used with existing software, e.g. commercial finite element codes. The general difficulty of inverse problems, namely the question of existence of a unique solution, is addressed in a very practical way, by the limitation of the identification procedure to regions where reliable solutions are available. As highly overdetermined inverse problems are considered here, the presented approach proves to be reasonable and well indicated, representing a significant advancement of the algorithm in [6].

In Section 2, the theory of the inverse fuzzy arithmetical concept is described, which is applied to a practical mechanical example system in Section 4. The exemplary mechanical system considered in this paper is a friction-coupled one-mass-oscillator with two DOFs, which is described in detail in Section 3. The results of the inverse uncertainty computations are presented in Section 4 where two uncertain damping parameters prove to be capable of approximating the error that results from model simplifications in a conservative way.

2 Inverse Fuzzy Arithmetical Concept

All fuzzy-valued computations in this paper are performed by the use of the transformation method, a detailed description of which can be found in [5, 7]. The fundamental idea of the presented identification method is to identify the uncertain parameters of a simplified model by using the simulation output of a sophisticated and more advanced model or by using measurement values of a real system.

This section is organized according to the steps that are needed for the identification: First, the notation and the procedure for the feedforward computation are explained. Second, the construction of the fuzzy-valued output quantities from the reference output is clarified. And third, the updated input parameters are computed, using the inverse fuzzy arithmetic.

2.1 Evaluation of fuzzy-parameterized models:

In general, a fuzzy-parameterized model consists of three key components:

1. A set of n independent fuzzy-valued model parameters \tilde{p}_i with the membership functions $\mu_{\tilde{p}_i}(x_i)$, $i = 1, 2, \dots, n$.
2. The model itself, which can be interpreted as a set of N generally nonlinear functions f_r , $r = 1, 2, \dots, N$, that perform some operations on the fuzzy input variables \tilde{p}_i , $i = 1, 2, \dots, n$.

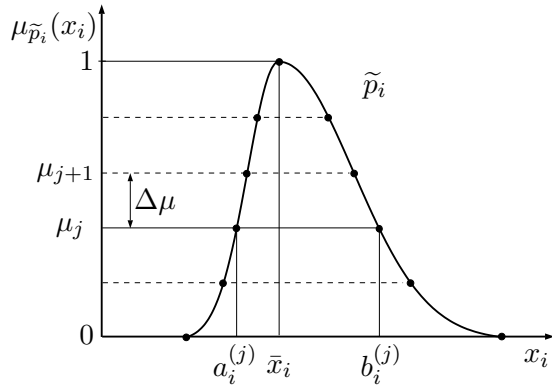


Figure 1: Implementation of the i th uncertain parameter as a fuzzy number \tilde{p}_i decomposed into intervals (α -cuts).

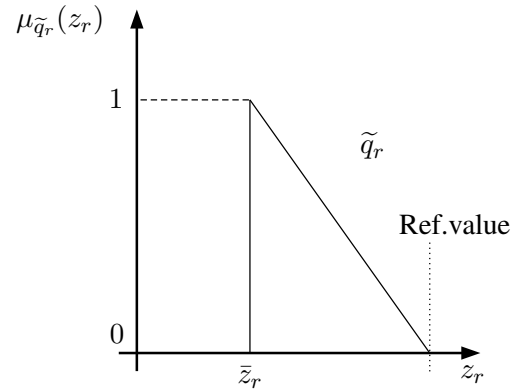


Figure 2: Definition of the output fuzzy number \tilde{q}_r based on a reference value.

3. A set of N fuzzy-valued output parameters \tilde{q}_r with the membership functions $\mu_{\tilde{q}_r}(z_r)$, $r = 1, 2, \dots, N$, that are obtained as the result of the functions f_r .

Thus, a fuzzy-parameterized model can in general be expressed by a system of equations of the form

$$\begin{aligned}
 \tilde{q}_1 &= f_1(\tilde{p}_1, \tilde{p}_2, \dots, \tilde{p}_n) \\
 &\vdots \\
 \tilde{q}_N &= f_N(\tilde{p}_1, \tilde{p}_2, \dots, \tilde{p}_n).
 \end{aligned} \tag{1}$$

As a pre-condition for the application of inverse fuzzy arithmetic, the invertibility of the system, i.e. its unique solution for the uncertain model parameters \tilde{p}_i , $i = 1, 2, \dots, n$, has to be guaranteed. This issue is addressed in detail in the following paragraphs. Additionally, only those models shall be considered in this paper where the output variables $\tilde{q}_1, \tilde{q}_2, \dots, \tilde{q}_N$ are strictly monotonic with respect to each of the model parameters $\tilde{p}_1, \tilde{p}_2, \dots, \tilde{p}_n$. This allows the uncertain model to be simulated and analyzed by simply applying the transformation method in its reduced form, omitting also the recursive procedure in the final retransformation step [7, 8].

2.2 Definition of fuzzy outputs:

To identify the uncertain input parameters of the simplified model that is capable of representing a more sophisticated model in a somehow coarse way, a set of fuzzy-valued outputs \tilde{q}_r needs to be defined from the crisp outputs of the sophisticated model. The nominal values \bar{q}_r of the fuzzy values \tilde{q}_r must correspond to the nominal output of the sophisticated model. The worst case deviation from this nominal value is given by the output of the sophisticated model. Consequently, one value of the bounds of the support of \tilde{q}_r is set to the measurement value while the other one is set to the nominal value. For the other μ -levels, a linear interpolation between $\mu = 0$ and $\mu = 1$ is chosen. This procedure leads to a single-sided, triangular fuzzy number, as visualized in Fig. 2.

2.3 Inverse fuzzy arithmetic:

With regard to the structure of fuzzy-parameterized models as defined in Eq. 1, the main problem of inverse fuzzy arithmetic consists in the identification of the uncertain model parameters $\tilde{p}_1, \tilde{p}_2, \dots, \tilde{p}_n$ on the basis of given values for the output

variables $\tilde{q}_1, \tilde{q}_2, \dots, \tilde{q}_N$. In the case $N < n$ the identification problem is under-determined, whereas its solution requires the application of an optimization procedure for $N > n$. In the present paper, the case $N = Sn$ with $S \in \mathbb{N}$, $S > 1$, shall be considered, where the number N of available output variables is much larger than the number n of uncertain model parameters. The factor S is the number of times for which the fully determined inverse computation ($N^* = n$) has to be performed. For the example of a frequency dependent mechanical system, the factor S corresponds to the number of frequency steps that are to be considered.

For one specific value of s , $s = 1, 2, \dots, S$, the condition $N^* = n$ holds and the number of available outputs is equal to the number of parameters that are to be identified. At first view, the solution to the inverse fuzzy arithmetical problem with $N^* = n$ appears to be rather straightforward and easy to achieve – at least for linear systems – by evaluating the inverted model equations, solved for the parameters $\tilde{p}_1(s), \tilde{p}_2(s), \dots, \tilde{p}_n(s)$, using the transformation method with $\tilde{q}_1(s), \tilde{q}_2(s), \dots, \tilde{q}_n(s)$ as input variables. This way of proceeding, however, clearly fails and leads to an enormous overestimation of the uncertainty of the model parameters $\tilde{p}_1(s), \tilde{p}_2(s), \dots, \tilde{p}_n(s)$ (see example in [5], Chapter 5.2). The reason for this failure must be seen in the fundamental pre-condition of the transformation method which requires its fuzzy-valued input parameters to be strictly independent, i.e. to independently initiate the overall uncertainty in the system by uncertain parameters of different origin. This condition, of course, can never be fulfilled for the use of inverted model equations since the input parameters $\tilde{q}_r(s)$, $r = 1, 2, \dots, n$, of this procedure do all feature a functional dependency on the model parameters $\tilde{p}_i(s)$, $i = 1, 2, \dots, n$ (see Eq. 1).

To still successfully solve the inverse fuzzy arithmetical problem, the following scheme can be applied, consisting of an appropriate combination of the simulation and the analysis part of the transformation method:

1. Determination of the nominal values $\tilde{x}_1(s), \tilde{x}_2(s), \dots, \tilde{x}_n(s)$:

Owing to Eq. 1, the nominal values $\bar{x}_i(s)$ of the real model parameters $\tilde{p}_i(s)$, $i = 1, 2, \dots, n$, and the nominal values $\bar{z}_r(s)$ of the output variables $\tilde{q}_r(s)$, $r = 1, 2, \dots, n$, are related by the system of equations

$$\begin{aligned} \bar{z}_1(s) &= f_1(\bar{x}_1(s), \bar{x}_2(s), \dots, \bar{x}_n(s)) \\ &\vdots = \vdots \\ \bar{z}_n(s) &= f_n(\bar{x}_1(s), \bar{x}_2(s), \dots, \bar{x}_n(s)) . \end{aligned} \quad (2)$$

Starting from the n given values $\bar{z}_r(s)$ in the inverse problem, the n nominal values $\tilde{x}_i(s)$ of the unknown fuzzy-valued model parameters $\tilde{p}_i(s)$, $i = 1, 2, \dots, n$ can be determined either by analytically solving the equations (2) for $\bar{x}_i(s)$, $i = 1, 2, \dots, n$, as one can easily do for linear systems, or by numerically solving the system of equations using a certain iteration procedure.

2. Computation of the gain factors:

For the determination of the single-sided gain factors $\eta_{ri+}^{(j)}(s)$ and $\eta_{ri-}^{(j)}(s)$, which play an important role in the inverse fuzzy arithmetical concept (see Eq. 5), the model has to be simulated for some initially assumptive uncertain parameters $\tilde{p}_i^*(s)$, $i = 1, 2, \dots, n$, using the transformation method of fuzzy arithmetic. The nominal values of $\tilde{p}_i^*(s)$ have to be set equal to the just computed values $\tilde{x}_i(s)$, $i = 1, 2, \dots, n$, and the assumed uncertainty should be set to a large enough value, so that the expected real range of uncertainty in $\tilde{p}_i(s)$ is preferably covered.

3. Assembly of the uncertain parameters $\check{\tilde{p}}_1(s), \check{\tilde{p}}_2(s), \dots, \check{\tilde{p}}_n(s)$:

Recalling the representation of a fuzzy number in its decomposed form (Fig. 1), the lower and upper bounds of the intervals of the fuzzy parameters $\check{\tilde{p}}_i(s)$ at the $(m + 1)$ levels of membership μ_j shall be defined as $\check{a}_i^{(j)}(s)$ and $\check{b}_i^{(j)}(s)$, and the bounds of the given output values $\check{\tilde{q}}_r(s)$ as $\check{c}_r^{(j)}(s)$ and $\check{d}_r^{(j)}(s)$. The interval bounds $\check{a}_i^{(j)}(s)$ and $\check{b}_i^{(j)}(s)$, which finally provide the membership functions of the unknown model parameters $\check{\tilde{p}}(s)$, $i = 1, 2, \dots, n$, can then be determined through

$$\begin{bmatrix} \check{a}_1^{(j)}(s) - \check{\tilde{x}}_1(s) \\ \check{b}_1^{(j)}(s) - \check{\tilde{x}}_1(s) \\ \check{a}_2^{(j)}(s) - \check{\tilde{x}}_2(s) \\ \check{b}_2^{(j)}(s) - \check{\tilde{x}}_2(s) \\ \vdots \\ \check{a}_n^{(j)}(s) - \check{\tilde{x}}_n(s) \\ \check{b}_n^{(j)}(s) - \check{\tilde{x}}_n(s) \end{bmatrix} = H^{(j)-1}(s) \begin{bmatrix} \check{c}_1^{(j)}(s) - \check{\tilde{z}}_1(s) \\ \check{d}_1^{(j)}(s) - \check{\tilde{z}}_1(s) \\ \check{c}_2^{(j)}(s) - \check{\tilde{z}}_2(s) \\ \check{d}_2^{(j)}(s) - \check{\tilde{z}}_2(s) \\ \vdots \\ \check{c}_n^{(j)}(s) - \check{\tilde{z}}_n(s) \\ \check{d}_n^{(j)}(s) - \check{\tilde{z}}_n(s) \end{bmatrix} \quad (3)$$

with

$$H^{(j)}(s) = \begin{bmatrix} H_{11}^{(j)}(s) & | & H_{12}^{(j)}(s) & | & \dots & | & H_{1n}^{(j)}(s) \\ \hline H_{21}^{(j)}(s) & | & H_{22}^{(j)}(s) & | & \dots & | & H_{2n}^{(j)}(s) \\ \hline \vdots & & \vdots & & \vdots & & \vdots \\ \hline H_{n1}^{(j)}(s) & | & H_{n2}^{(j)}(s) & | & \dots & | & H_{nm}^{(j)}(s) \end{bmatrix} \quad (4)$$

and

$$H_{ri}^{(j)}(s) = \frac{1}{2} \begin{bmatrix} \eta_{ri-}^{(j)}(s)(1 + \text{sgn}(\eta_{ri-}^{(j)}(s))) & \eta_{ri+}^{(j)}(s)(1 - \text{sgn}(\eta_{ri+}^{(j)}(s))) \\ \eta_{ri-}^{(j)}(s)(1 - \text{sgn}(\eta_{ri-}^{(j)}(s))) & \eta_{ri+}^{(j)}(s)(1 + \text{sgn}(\eta_{ri+}^{(j)}(s))) \end{bmatrix}$$

$$i, r = 1, 2, \dots, n, \quad j = 0, 1, \dots, m - 1.$$

The values $\check{a}_i^{(m)}(s) = \check{b}_i^{(m)}(s)$, $i = 0, 1, \dots, n$, are already determined by the nominal values $\check{\tilde{x}}_i(s)$.

The special case where the number of outputs is equal to the number of parameters can be solved with the previously explained scheme. To extent the methodology to the overdetermined case where $N = S n$, two further steps are necessary. First, the inverse problem is solved for each s , for which a reliable solution can be expected. Second, one set of uncertain parameters needs to be extracted from the solution for all $s = 1, 2, \dots, S$.

In order to extract those inverse problems where a reliable solution can be expected, the invertibility of the matrix $H^{(j)}(s)$ and the magnitude of its elements are considered as a feasibility criterion. The invertibility is assessed numerically, and based on the numerical result, the solution of Eq. 3 is obtained. Depending on the magnitude of the elements of $H^{(j)}(s)$, numerical inaccuracies can reach significant levels, and it is even possible that pseudo-solutions are obtained where a symbolic computation would clearly reveal infeasibility. For this reason, those solutions are excluded

from any further computation where the values of all the absolute measure of influence $\kappa_i(s)$, $i = 1, 2, \dots, n$, remain below a certain threshold. According to [5], the absolute measures of influence $\kappa_i(s)$ are defined by

$$\kappa_i(s) = \frac{\sum_{j=1}^{m-1} \mu_j \left| \eta_i^{(j)}(s) \left(\tilde{a}_i^{(j)}(s) + \tilde{b}_i^{(j)}(s) \right) \right|}{2 \sum_{j=1}^{m-1} \mu_j}. \quad (5)$$

For the reasons explained previously, the solution for one specific s is rated as reliable if for all output quantities

$$\kappa_i(s) > \epsilon > 0 \quad \forall i = 1, 2, \dots, n. \quad (6)$$

So far, for each s with a reliable solution, one set of fuzzy parameters $\tilde{p}_i(s)$, given by the nested intervals $[\tilde{a}_i^{(j)}(s), \tilde{b}_i^{(j)}(s)]$, $i = 1, 2, \dots, n$, $j = 0, 1, \dots, m$, has been identified. In order to extract the one set of uncertain input parameters \tilde{p}_i that is representative for all values of s , for each level of membership μ_j the maximum uncertainty over all values of s is used.

$$\begin{aligned} \check{a}_i^{(j)} &= \min_s \left(\tilde{a}_i^{(j)}(s), \tilde{a}_i^{(j+1)} \right), \quad i = 1, 2, \dots, n; \quad j = 0, 1, \dots, m-1 \\ \check{b}_i^{(j)} &= \max_s \left(\tilde{b}_i^{(j)}(s), \tilde{b}_i^{(j+1)} \right) \end{aligned} \quad (7)$$

Equation 7 ensures the convexity of the resulting fuzzy parameters, which is a crucial property for further computations, especially for the re-simulation of the identified system.

To verify the identified model parameters $\check{p}_1, \check{p}_2, \dots, \check{p}_n$, the model equations (1) can be re-simulated by means of the transformation method, using $\check{p}_1, \check{p}_2, \dots, \check{p}_n$ as the fuzzy input parameters. The degree of conformity of the so-obtained output fuzzy numbers $\check{q}_1, \check{q}_2, \dots, \check{q}_n$ with the original output values $\tilde{q}_1, \tilde{q}_2, \dots, \tilde{q}_n$ can serve as a measure of the quality of the identification.

3 Application to a friction model

This section describes the mechanical example system and computational methods that are used to demonstrate the capabilities of the previously presented uncertainty identification algorithm. The example is derived from the damping description of a cantilever beam with an attached friction damper (Fig. 3). In [9], the discretized continuous systems is modeled by different substructures using the finite-element method.

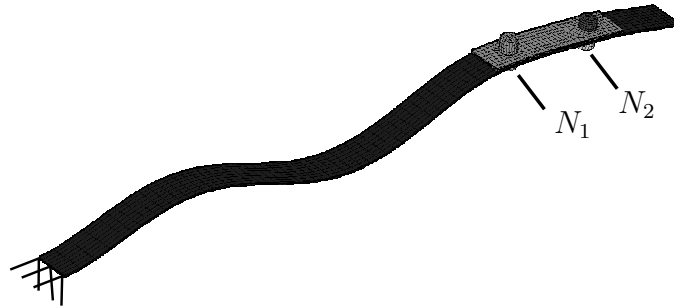


Figure 3: Beam with 600 friction elements and the normal forces N_1 and N_2 .

Damping occurs due to the friction between the surfaces of the beam and the damper. Two bolts are used to generate the normal forces N_1 and N_2 which act vertically and lead to the contact pressure in the friction surface. Friction is modeled by 600 node-to-node friction models in the contact surface. The Harmonic Balance Method (HBM) is used to accomplish the computation of the Frequency Response Function (FRF) of a forced vibration analysis of the nonlinear system. Friction for a two-dimensional motion is modeled by a 2D macro-slip model [10]. The dynamic behavior of one friction element of the beam is exemplarily demonstrated by a two-degree-of-freedom oscillator that is constrained via a 2D macro-slip friction element.

Oscillator with 2-D macro-slip friction model:

The dynamics of the 2-DOF oscillator, as it is presented in Fig. 4, is governed by the equation

$$\mathbf{M}\ddot{\mathbf{x}} + \mathbf{D}\dot{\mathbf{x}} + \mathbf{K}\mathbf{x} + \mathbf{f}(\mathbf{x}, \dot{\mathbf{x}}) = \mathbf{f}_{\text{exc}} \quad , \quad \mathbf{x} = \begin{bmatrix} x \\ y \end{bmatrix} , \quad (8)$$

with the linear mass, stiffness and damping matrices

$$\mathbf{M} = \begin{bmatrix} m & 0 \\ 0 & m \end{bmatrix} , \quad \mathbf{K} = \begin{bmatrix} k_x & 0 \\ 0 & k_y \end{bmatrix} , \quad \mathbf{D} = \begin{bmatrix} c_x & 0 \\ 0 & c_y \end{bmatrix} , \quad (9)$$

and the 2D-friction force $\mathbf{f}(\mathbf{x}, \dot{\mathbf{x}})$ as well as the external excitation

$$\mathbf{f}_{\text{exc}} = \begin{bmatrix} 100 \text{ N} \\ (30 + 30i) \text{ N} \end{bmatrix} e^{i\omega t} . \quad (10)$$

All nominal simulations were performed with the parameters given in Table 1, where μ and c_t are the parameters of the friction model. The 2-D macro-slip friction model which is used in this paper as well as the mechanical parameters are adopted from [10].

Parameter	m	c_x	c_y	k_x	k_y	c_t	μ	N
Value	1	4	4	$4 \cdot 10^4$	$4 \cdot 10^4$	$15 \cdot 10^3$	0.8	400
Unit	kg	Ns/m	Ns/m	N/m	N/m	N/m	—	N

Table 1: Parameters of the 2D-Oscillator.

The axial friction forces in a two dimensional motion are coupled and therefore a fully analytical solution by HBM is not achievable. For this reason, the system can only be simulated numerically, which proves to be computationally very expensive, because of the complicated friction modeling. In contrast to this advanced friction model, a simplified model consisting of two one-dimensional friction elements is used instead in order to evaluate computations analytically (Fig. 5). This approach reduces the computational cost drastically, but leads to an error in friction modeling. The objective of the algorithm proposed in this paper is to quantify the uncertainty related to the two 1-D macro-slip friction models, such that a conservative approximation of the 2-D macro-slip friction model can be provided.

4 Uncertainty Identification

This section describes the proposed procedure to identify the uncertain parameters of a simplified model based on the outputs of a more advanced model. For the sake of

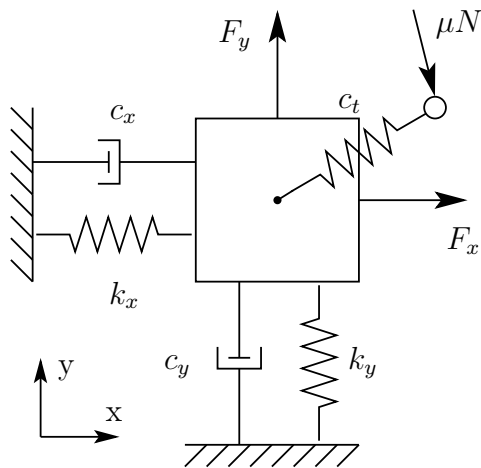


Figure 4: Oscillator with 2D-Friction.

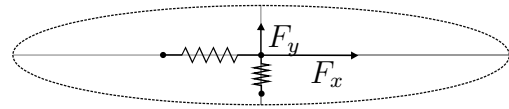


Figure 5: Approximation of the 2D-friction by two 1-D friction elements.

simplicity, the identification procedure is carried out with two uncertain parameters and accordingly with two outputs, namely the frequency response functions in x - and y -direction. For the simplified model, friction is modeled with two 1-D macro-slip friction elements, whereas the advanced model uses a 2-D macro-slip friction model. Thus, the simplification consists of neglecting the coupling of the two DOFs within the friction element.

It turns out that the simplified model is well capable of describing the overall dynamic behavior of the advanced model if the two damper constants c_x, c_y are considered uncertain. As a first step of their identification, a feedforward computation of the simplified model with some assumed uncertain damping parameters is needed. These are chosen as triangular fuzzy numbers with a nominal value of 4 Ns/m and left and right hand deviation of 4Ns/m. In Fig. 6, the resulting uncertain FRFs are shown (thin lines) together with the output of the advanced model (thick line) and a conventionally achieved best fit of the simplified model (dotted line). For the y -direction, the simplified model seems to be bounded at about -3.2 dB. This phenomenon can be explained by a massive increase in friction damping for large vibration amplitudes. By increasing the damping parameter c_y in the y -direction, such large vibration amplitudes are not reached at all, which is why the previously described phenomenon cannot be observed in this case.

As the invertibility of a system cannot be guaranteed in general, a special criterion is needed to determine this property. In a postprocessing step, the transformation method of fuzzy arithmetic provides measures of influence κ_i that quantify the influence of the input parameters' uncertainty on the uncertainty of the outputs. For the present example, the values of κ_i are shown in Fig. 7, plotted against frequency. They reveal that a coupling of the two DOFs only occurs in the frequency range between about 36 Hz and 38.5 Hz. In this frequency range, large vibration amplitudes lead to sliding and thus to additional damping, and the damper in the x -direction has a significant influence on the motion in the y -direction, and vice versa.

As the criterion for the invertibility of the system, it turns out that the inversion leads to reasonable results if all κ_i are larger than a certain threshold ϵ . The restriction of the identification process to frequencies where the previous condition holds leads to the frequency dependent fuzzy numbers shown in Fig. 8. Due to the large uncertainty of the identified parameters and the nonlinear behavior of the system, an iterative solution becomes necessary. Tab. 2 documents the convergence of the iterative solution and shows that adequate convergence is practically reached after two

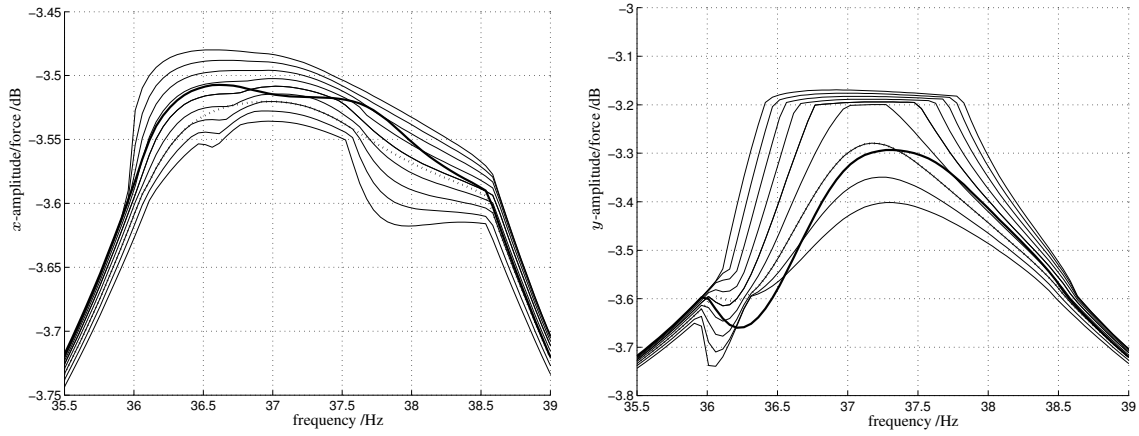


Figure 6: Fuzzy-valued response functions in x - and y -direction for the simplified model.

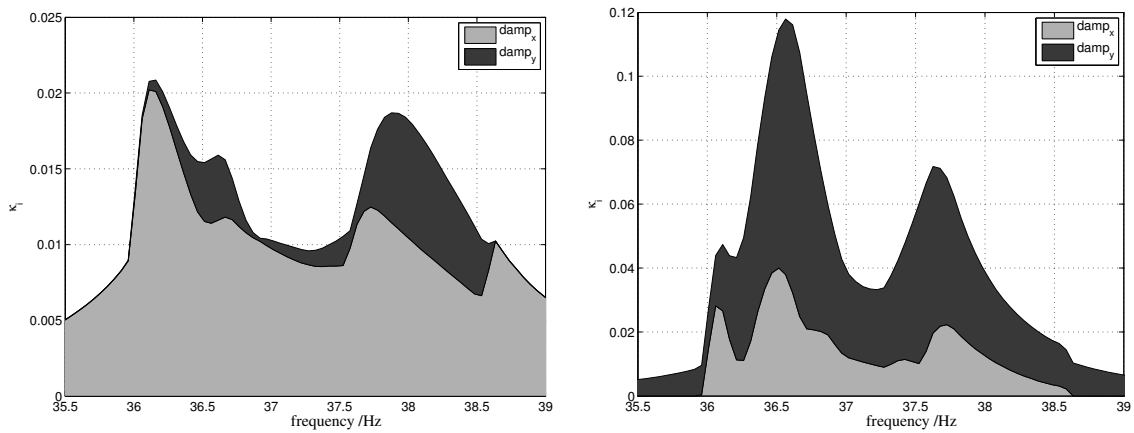


Figure 7: Absolute measures of influence κ_i .

iterations. By determining the maximum uncertainty for each level of membership over frequency and by ensuring the convexity of the fuzzy number, the fuzzy-valued damping parameters \tilde{p}_1 and \tilde{p}_2 plotted on the right hand side of each contour plot in Fig. 8, can be extracted.

Iteration	1	2	3	4	5	6
\tilde{p}_1	4/4	1.3/1.35	2.9/1.32	2.85/1.6	2.75/1.56	2.8/1.55
\tilde{p}_2	4/4	0.3/7.1	0.15/5.5	0.1/5.35	0.1/5.5	0.1/5.5

Table 2: Worst-case deviations of the identified parameters \tilde{p}_i from the nominal values $\bar{p}_i = 4 \text{ Ns/m}$ (left hand deviation / right hand deviation) for each iteration.

In order to prove that the uncertain damping parameters \tilde{p}_1 and \tilde{p}_2 that were identified are reasonable, the system is re-simulated with these values. The results of this re-simulation are shown in Fig. 9 and prove that the overall objective of the identification is reached, namely the conservative approximation of an advanced model by a simplified model with uncertain parameters. In comparison to the simulation results in Fig. 6, which are obtained for some initially assumed uncertain input parameters, the newly obtained simulation results for the identified fuzzy parameters (Fig. 9)

lead to a much tighter approximation of the reference outputs. The uncertainty in the predicted output is reduced by up to 50% in the x-direction. In the y-direction, the uncertainty is reduced by approximately 20% for most frequencies, but where this is necessary in order to achieve the goal of a conservative approximation, it is increased. For both directions, the reference outputs are bounded for all frequencies by the uncertainty of the outputs of the simplified model, i.e. the advanced model is adequately represented by the simplified model with updated fuzzy-valued parameters. A qualitative assessment of the output plots in Fig. 9 leads to the conclusion that a further reduction of the input uncertainties can hardly be expected as the outputs of the advanced model almost reach the worst case bounds of the simplified model for at least one frequency.

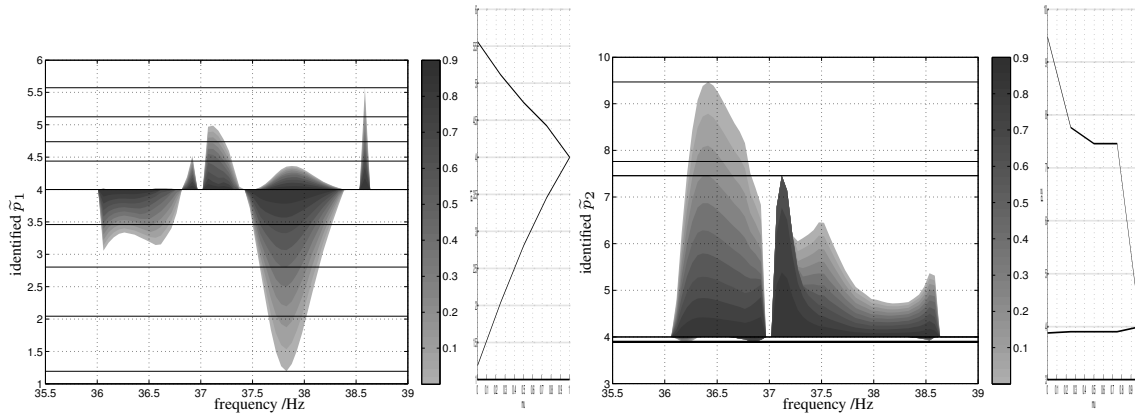


Figure 8: Identified input parameters \tilde{p}_1 and \tilde{p}_2 .

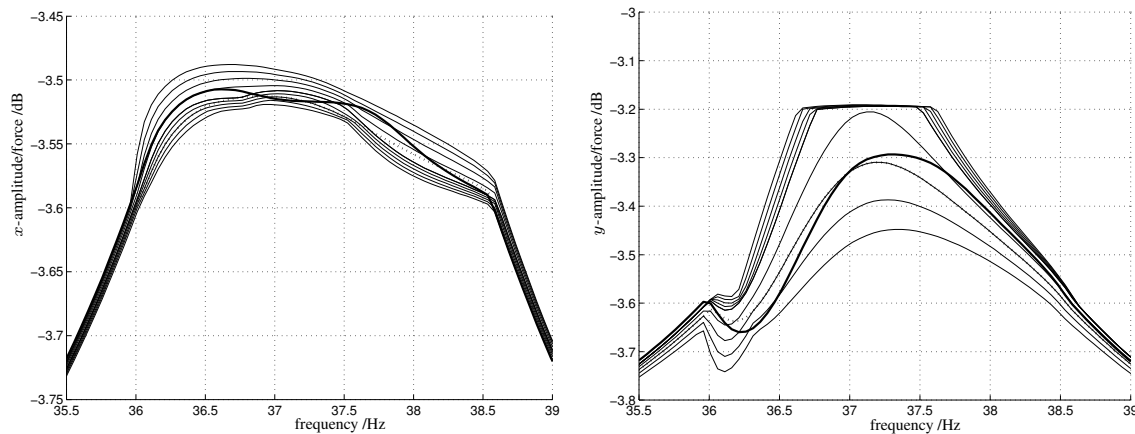


Figure 9: Fuzzy-valued response function using identified parameters.

5 Conclusions

A systematic procedure has been proposed that allows the identification of uncertain parameters of a model based on the output of another, more sophisticated and advanced model or based on a measurement signal. The general problem of inverse computations, namely its feasibility, is addressed through the restriction of the identification process to a valid frequency range which is determined by the computation of frequency dependent influence measures as a feasibility criterion. The fuzzy-valued

parameters of the simplified model are finally extracted through a conservative approximation of the resulting frequency-dependent fuzzy numbers. Future work will focus on the identification of a larger number of parameters and a less conservative extraction of the identified parameters.

References

- [1] Oberkampf, W. L.: Model Validation under Both Aleatory and Epistemic Uncertainty. In: *Proc. of NATO AVT-147 Symposium on Computational Uncertainty in Military Vehicle Design*. Athens, Greece. 2007.
- [2] Hemez, F. M.; Booker, J. M.; Langenbrunner, J. R.: Answering The Question of Sufficiency: How Much Uncertainty is Enough? In: *Proc. of The 1st International Conference on Uncertainty in Structural Dynamics – USD 2007* (Manson, G.; Worden, K., ed.), pp. 23–48. Sheffield, UK. 2007.
- [3] Zadeh, L. A.: Fuzzy sets. *Information and Control* 8 (1965), pp. 338–353.
- [4] Kaufmann, A.; Gupta, M. M.: *Introduction to Fuzzy Arithmetic*. New York: Van Nostrand Reinhold. 1991.
- [5] Hanss, M.: *Applied Fuzzy Arithmetic – An Introduction with Engineering Applications*. Berlin: Springer. 2005.
- [6] Hanss, M.: An approach to inverse fuzzy arithmetic. In: *Proc. of the 22nd International Conference of the North American Fuzzy Information Processing Society – NAFIPS 2003*, pp. 474–479. Chicago, IL, USA. 2003.
- [7] Hanss, M.: The transformation method for the simulation and analysis of systems with uncertain parameters. *Fuzzy Sets and Systems* 130 (2002) 3, pp. 277–289.
- [8] Hanss, M.: Simulation and analysis of fuzzy-parameterized models with the extended transformation method. In: *Proc. of the 22nd International Conference of the North American Fuzzy Information Processing Society – NAFIPS 2003*, pp. 462–467. Chicago, IL, USA. 2003.
- [9] Becker, J.; Gaul, L.: CMS methods for efficient damping prediction for structures with friction. In: *Proc. of the IMAC-XXVI*. Orlando, Florida, USA. 2008.
- [10] Sanliturk, K. Y.; Ewins, D. J.: Modelling two-dimensional friction contact and its application using harmonic balance method. *Journal of Sound and Vibration* 193 (1996) 2, pp. 511–523.

Vergleichende Untersuchung nichtlinearer Beobachterkonzepte zur Fehlerdiagnose am Beispiel hydrostatischer Fahrtriebe

Patrick Gerland¹, Horst Schulte², Andreas Kroll¹

¹Universität Kassel, FB Maschinenbau, FG Mess- und Regelungstechnik

²Bosch Rexroth AG, Hydraulics

E-Mail: patrick.gerland@mrt.uni-kassel.de

In mobilen Arbeitsmaschinen werden zur Verbesserung der Arbeitsleistung und Steigerung der Energieeffizienz zunehmend druck- und leistungsgeregelte hydrostatische Fahrtriebe eingesetzt. Eine zuverlässige und sichere Funktionsweise solcher Systeme erfordert eine kontinuierliche Überwachung der integrierten Sensoren und Stellgeräte, da z.B. der Ausfall von Drucksensoren ein enormes Gefährdungspotential für den Menschen darstellt.

Den Stand der Technik der Sensorüberwachung bilden allgemein kennfeldbasierte Verfahren zur Grenzwertüberwachung und zur Plausibilisierung der Messwerte. Damit ist eine zuverlässige Überwachung von Drucksensoren, die hochdynamische Änderungen im hydrostatischen Lastkreis erfassen müssen, nicht im ausreichenden Maße realisierbar. Abhilfe bieten beobachterbasierte Fehlerdiagnosearchitekturen. Da es sich bei einem hydrostatischen Fahrtrieb um ein stark nichtlineares System handelt, erfordert dies den Einsatz von nichtlinearen Beobachterkonzepten.

In dieser Arbeit werden strukturell stark unterschiedliche Beobachteransätze miteinander verglichen. Im Einzelnen werden dazu ein Takagi-Sugeno (TS) Fuzzy Beobachter und ein Sliding-Mode (SM) Beobachter entworfen. Anhand von Simulationen und experimentellen Daten wird demonstriert, dass der TS Fuzzy Beobachter im Gegensatz zum SM Beobachter in der Lage ist, das stark nichtlineare Prozessverhalten über den gesamten Arbeitsbereich zu beschreiben. Weiterhin wird gezeigt, dass der TS Fuzzy Beobachter im Vergleich zum SM Beobachter eine geringere Robustheit gegenüber begrenzten Modellparameterfehlern aufweist.

1 Einleitung

Modellbasierte Diagnosekonzepte (z.B. [1], [2], [3]) bieten gegenüber signalbasierten den Vorteil der Generalisierungsfähigkeit und Wiederverwendbarkeit. Für den Anwender heißt das, er kann zur adäquaten Beschreibung der Dynamik seiner Komponenten und Systeme physikalische Modelle benutzen, die allein durch Anpassung der physikalischen Parameter (Nenngröße, Fahrzeuggewicht etc.) die Dynamikunterschiede zwischen einzelnen Systemvarianten und Kundenapplikationen widerspiegeln. Eine Methode der modellbasierten Diagnose beruht auf der Verwendung von Zustandsbeobachtern. Hierzu werden die Zustände des Prozesses mit Hilfe von mathematischen Modellen berechnet. Zur Kompensation von unterschiedlichen Anfangsbedingungen und Modellunsicherheiten wird der Ausgangsfehler in die Zustandsberechnung zurückgekoppelt.

In [4] werden verschiedene modellbasierte Diagnoseverfahren wie Paritätsraumverfahren, Ausgangs- und Zustandsbeobachter am Beispiel einer Simulation eines Aktuators zur automatischen Kupplungsbetätigung bewertet. Hierzu wurden Kriterien wie Realisierungsaufwand, Einsatzfähigkeit, Mindestanforderungen an das Systemverständnis und

bisherige Praxiserfahrung herangezogen. Die untersuchten Verfahren, basierend auf physikalischen Modellen, verwenden ausschließlich lineare reduzierte Modelle. In dieser Arbeit werden erstmals zwei nichtlineare Beobachterkonzepte an einem real existierenden hydrostatischen Fahrantrieb miteinander verglichen. Hierbei handelt es sich um ein stark nichtlineares System, das in sehr weiten Arbeitsbereichen betrieben wird. Der Schwerpunkt dieser Arbeit liegt hierbei erstens auf der nachvollziehbaren Darstellung des Entwurfs ausgehend von dem in [5], [6] erstellten physikalischen Modell hydrostatischer Antriebe und zweitens auf dem Vergleich der dynamischen Rekonstruktionsfehler in linearen und nichtlinearen Betriebsphasen sowie dem Vergleich der Robustheitseigenschaften bei Modellparameterfehlern. Die Untersuchungen werden sowohl an Simulationsläufen als auch an realen Messungen durchgeführt.

Im zweiten Abschnitt werden zunächst die grundlegenden Beziehungen der zwei zu vergleichenden Beobachterkonzepte vorgestellt. Das heißt, die Struktur und die notwendigen Schritte des Entwurfs werden ausführlich dargestellt. Im dritten Abschnitt werden anschließend die allgemein gültigen Beobachterkonzepte zur Fehlerdiagnose in hydrostatischen Fahrantrieben eingesetzt. Im Rahmen dieser Problemstellung werden zur sogenannten Symptombildung und -erkennung Residuen aus dem Vergleich der berechneten und beobachteten Zustände bestimmt. Die Ursachenermittlung steht dabei nicht im Fokus dieser Arbeit. Der eigentliche Beobachtervergleich erfolgt im vierten Abschnitt anhand von Simulationsläufen und experimentellen Untersuchungen.

2 Grundlagen der Beobachterkonzepte

2.1 Struktur und Entwurf reduzierter TS Fuzzy Beobachter

Für ein nichtlineares dynamisches System, das sich durch das MIMO TS Fuzzy Modell

$$\begin{aligned}\dot{\mathbf{x}}(t) &= \sum_{i=1}^r h_i(z_1(t), z_2(t)) \mathbf{A}_i \mathbf{x}(t) + \sum_{i=1}^r h_i(z_1(t), z_2(t)) \mathbf{B}_i \mathbf{u}(t) \\ \mathbf{y}(t) &= \mathbf{C} \mathbf{x}(t)\end{aligned}\quad (1)$$

mit $\mathbf{x} \in \mathbb{R}^n$, $\mathbf{y} \in \mathbb{R}^p$, $\mathbf{u} \in \mathbb{R}^m$ sowie den Matrizen \mathbf{A}_i , \mathbf{B}_i und \mathbf{C} passender Dimensionen beschreiben lässt, kann ein Fuzzy Beobachter zur Rekonstruktion des vollständigen Zustandsvektors entworfen werden. In der hier betrachteten Anwendung wird der Umstand ausgenutzt, dass lediglich ein Systemzustand, der Differenzdruck Δp , geschätzt werden soll. Infolgedessen lässt sich die Ordnung des Beobachters durch Ausnutzen der in den Messgrößen enthaltenen Informationen über den Zustandsvektor reduzieren. Dazu wird der Zustandsvektor in zwei Teile zerlegt: Dies sind der Vektor der direkt messbaren Variablen $\mathbf{x}_a \in \mathbb{R}^p$ und der zu schätzenden Zustandsvariablen $\mathbf{x}_b \in \mathbb{R}^{(n-p)}$. Die Matrizen jedes linearen Teilmodells $i = 1, \dots, r$ werden entsprechend zerlegt:

$$\begin{bmatrix} \dot{\mathbf{x}}_a(t) \\ \dot{\mathbf{x}}_b(t) \end{bmatrix} = \begin{bmatrix} \mathbf{A}_{i_{aa}} & \mathbf{A}_{i_{ab}} \\ \mathbf{A}_{i_{ba}} & \mathbf{A}_{i_{bb}} \end{bmatrix} \begin{bmatrix} \mathbf{x}_a(t) \\ \mathbf{x}_b(t) \end{bmatrix} + \begin{bmatrix} \mathbf{B}_{i_a} \\ \mathbf{B}_{i_b} \end{bmatrix} \mathbf{u}(t)\quad (2)$$

$$\mathbf{y}(t) = \begin{bmatrix} \mathbf{E} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{x}_a(t) \\ \mathbf{x}_b(t) \end{bmatrix},\quad (3)$$

mit \mathbf{E} als Einheitsmatrix.

Darauf basierend lässt sich die bekannte Struktur des reduzierten Beobachters für ein linear zeitinvariantes System [7] angeben mit

$$\dot{\hat{\mathbf{x}}}_c(t) = [\mathbf{A}_{i_{bb}} - \mathbf{L}_i \mathbf{A}_{i_{ab}}] \hat{\mathbf{x}}_b(t) + [\mathbf{A}_{i_{ba}} - \mathbf{L}_i \mathbf{A}_{i_{aa}}] \mathbf{y}(t) + [\mathbf{B}_{i_b} - \mathbf{L}_i \mathbf{B}_{i_a}] \mathbf{u}(t) \quad (4)$$

$$\hat{\mathbf{x}}_b(t) = \mathbf{x}_c + \mathbf{L}_i \mathbf{y}(t) , \quad (5)$$

worin \mathbf{L}_i , $i = 1, \dots, r$ gewöhnliche Rückführmatrizen linearer Beobachter darstellen. Anmerkung: Der Entwurf des reduzierten Beobachters erfordert typischerweise die zeitliche Ableitung des Messvektors \mathbf{y} , was jedoch unerwünscht ist. Um dieser Schwierigkeit zu begegnen, wird daher der neue Zustand \mathbf{x}_c definiert.

Die resultierende Beobachterdynamik ergibt sich aus der gewichteten Summe der einzelnen linearen Beobachter (4)

$$\dot{\hat{\mathbf{x}}}_c(t) = \sum_{i=1}^r h_i(\mathbf{z}(t)) [\mathbf{A}_{i_{bb}} - \mathbf{L}_i \mathbf{A}_{i_{ab}}] \hat{\mathbf{x}}_b(t) + \sum_{i=1}^r h_i(\mathbf{z}(t)) [\mathbf{A}_{i_{ba}} - \mathbf{L}_i \mathbf{A}_{i_{aa}}] \mathbf{y}(t) \quad (6)$$

$$+ \sum_{i=1}^r h_i(\mathbf{z}(t)) [\mathbf{B}_{i_b} - \mathbf{L}_i \mathbf{B}_{i_a}] \mathbf{u}(t)$$

$$\hat{\mathbf{x}}_b(t) = \mathbf{x}_c(t) + \sum_{i=1}^r h_i(\mathbf{z}(t)) \mathbf{L}_i \mathbf{y}(t) . \quad (7)$$

Die Gewichtungsfunktionen $h_i(\mathbf{z})$ mit $\mathbf{z} = [\tilde{\alpha}_M, \omega_P]^T$ in (6) sowie der Anzahl der linearen Modelle ($r = 4$) sind identisch zu denen des Streckenmodells (1).

Für den folgenden TS Fuzzy Beobachterentwurf wird angenommen, dass das System lokal beobachtbar ist. Dies bedeutet, dass alle Matrizenpaare $(\mathbf{A}_{i_{bb}}, \mathbf{A}_{i_{ab}})$, $i = 1, \dots, r$ das Beobachtbarkeitskriterium von Kalman erfüllen. Für den Beobachtungsfehler

$$\mathbf{e}_b(t) = \mathbf{x}_b(t) - \hat{\mathbf{x}}_b(t) \quad (8)$$

ergibt sich die Fehlerdynamik durch die Subtraktion des Ausdrucks

$$\dot{\hat{\mathbf{x}}}_b(t) = \sum_{i=1}^r h_i(\mathbf{z}(t)) [\mathbf{A}_{i_{bb}} \mathbf{x}_b(t) + \mathbf{A}_{i_{ba}} \mathbf{x}_a(t) + \mathbf{B}_{i_b} \mathbf{u}(t)] \quad (9)$$

von (6) mit (7) zu

$$\dot{\mathbf{e}}_b(t) = \sum_{i=1}^r \sum_{j=1}^r h_i(\mathbf{z}(t)) h_j(\mathbf{z}(t)) [\mathbf{A}_{i_{bb}} - \mathbf{L}_i \mathbf{A}_{j_{ab}}] \mathbf{e}_b(t) . \quad (10)$$

Im Falle einer stabilen Fehlerdynamik (10) nähert sich die Zustandsschätzung asymptotisch dem wahren Systemzustand an, d. h. der Zustandsbeobachtungsfehler klingt asymptotisch ab. Die Stabilität der Fehlerdynamik kann durch das folgende Theorem nachgewiesen werden:

Theorem 1: Der Fuzzy Beobachter (6), (7) ist global asymptotisch stabil, falls eine positiv definite Matrix $\mathbf{P} > \mathbf{0}$ existiert, so dass

$$\begin{aligned} \mathbf{A}_{i_{bb}}^T \mathbf{P} + \mathbf{P} \mathbf{A}_{i_{bb}} - \mathbf{A}_{j_{ab}}^T \mathbf{N}_i^T - \mathbf{N}_i \mathbf{A}_{j_{ab}} + 4\alpha \mathbf{P} + \mathbf{A}_{j_{bb}}^T \mathbf{P} + \mathbf{P} \mathbf{A}_{j_{bb}} \\ - \mathbf{A}_{i_{ab}}^T \mathbf{N}_j^T - \mathbf{N}_j \mathbf{A}_{i_{ab}} < \mathbf{0}, \end{aligned} \quad (11)$$

$$\mathbf{A}_{i_{bb}}^T \mathbf{P} + \mathbf{P} \mathbf{A}_{i_{bb}} - \mathbf{A}_{i_{ab}}^T \mathbf{N}_i^T - \mathbf{N}_i \mathbf{A}_{i_{ab}} + 2\alpha \mathbf{P} < \mathbf{0}$$

für alle $i, j = 1, \dots, r$ und $\mathbf{N}_i = \mathbf{P} \mathbf{L}_i$ gilt.

Die Matrizen \mathbf{P} und \mathbf{N}_i können unter Anwendung konvexer Optimierungstechniken numerisch berechnet werden, falls die linearen Matrixungleichungen (11) für eine vorgegebene Abklingrate $\alpha \in \mathbb{R}^+$ ein mögliches Ergebnis liefern. Die $i = 1, \dots, r$ Beobachterrückführmatrizen ergeben sich dann zu $\mathbf{L}_i = \mathbf{P}^{-1} \mathbf{N}_i$. Der Beweis dieses Theorems folgt direkt aus dem Beweis für das Theorem des Beobachters vollständiger Ordnung [8].

2.2 Struktur und Entwurf eines Sliding-Mode-Beobachters

Edwards und Spurgeon schlagen in [9], [10] eine Sliding-Mode Beobachterstruktur sowie ein Entwurfsverfahren vor, welches dem von Walcott und Žak [11] ähnelt, aber das Problem der notwendigen symbolischen Manipulation in [11] umgeht. Dieses Entwurfsverfahren beinhaltet die Berechnung von Beobachterverstärkungsmatrizen, die sowohl das Erreichen der Gleitfläche (sliding surface) in endlicher Zeit als auch den aktuellen Gleitzustand bei begrenzten Unsicherheiten garantiert. Während dieses Gleitens beträgt der Fehler zwischen System- und Beobachterzuständen $\mathbf{e}(t) = \mathbf{x}(t) - \hat{\mathbf{x}}(t)$ nahezu Null. Dieses Entwurfsverfahren wird im Folgenden beschrieben.

Der Entwurf des diskontinuierlichen Sliding-Mode Beobachters geht aus von dem unsicheren dynamischen System

$$\begin{aligned} \dot{\mathbf{x}}(t) &= \mathbf{A} \mathbf{x}(t) + \mathbf{B} \mathbf{u}(t) + \mathbf{D} \boldsymbol{\xi}(t, \mathbf{x}, \mathbf{u}) \\ \mathbf{y}(t) &= \mathbf{C} \mathbf{x}(t), \end{aligned} \quad (12)$$

mit $\mathbf{A} \in \mathbb{R}^{n \times n}$, $\mathbf{B} \in \mathbb{R}^{n \times m}$, $\mathbf{C} \in \mathbb{R}^{p \times n}$, $\mathbf{D} \in \mathbb{R}^{n \times q}$ und $n > p \geq q$. \mathbf{D} ist dabei die Störungsverteilungsmatrix. Es sei angenommen, dass die Matrizen \mathbf{B} , \mathbf{C} und \mathbf{D} jeweils den vollen Rang besitzen und das Matrizenpaar (\mathbf{A}, \mathbf{C}) beobachtbar ist. Weiterhin sei die Funktion $\boldsymbol{\xi} : \mathbb{R}_+ \times \mathbb{R}^n \times \mathbb{R}^m \rightarrow \mathbb{R}^q$, welche die Unsicherheiten und Nichtlinearitäten des Systems beinhaltet, unbekannt aber begrenzt, so dass

$$\|\boldsymbol{\xi}(t, \mathbf{x}, \mathbf{u})\| \leq r_1 \|\mathbf{u}\| + \alpha(t, \mathbf{y}) \quad (13)$$

gilt, wobei r_1 ein bekannter Skalar ist, $\alpha : \mathbb{R}_+ \times \mathbb{R}^q \rightarrow \mathbb{R}_+$ eine bekannte Funktion und $\|\cdot\|$ die euklidische Vektornorm bezeichnet. Die Anwendung der Lineartransformation $\mathbf{z} = (\mathbf{z}_1^T \mathbf{z}_2^T)^T = \mathbf{T}_0 \mathbf{x}$ mit $\mathbf{T}_0 \in \mathbb{R}^{n \times n}$ auf (12) liefert die kanonische Form des Systems:

$$\begin{aligned} \dot{\mathbf{z}}_1(t) &= \mathbf{A}_{11} \mathbf{z}_1(t) + \mathbf{A}_{12} \mathbf{z}_2(t) + \mathbf{B}_1 \mathbf{u}(t) \\ \dot{\mathbf{z}}_2(t) &= \mathbf{A}_{21} \mathbf{z}_1(t) + \mathbf{A}_{22} \mathbf{z}_2(t) + \mathbf{B}_2 \mathbf{u}(t) + \mathbf{D}_2 \boldsymbol{\xi}(t, \mathbf{z}, \mathbf{u}) \\ \mathbf{y}(t) &= \mathbf{z}_2(t). \end{aligned} \quad (14)$$

Darin sind $\mathbf{z}_1 \in \mathbb{R}^{n-p}$, $\mathbf{z}_2 \in \mathbb{R}^p$. Die Matrix \mathbf{A}_{11} wird als stabil angenommen, d. h. die Eigenwerte sind negativ. Notwendige Bedingungen für die Existenz der kanonischen Form (14) sind (vgl. Proposition 6.2 in [10]):

- $\text{rang}(\mathbf{C}\mathbf{D}) = q$
- Alle invarianten Nullstellen von $(\mathbf{A}, \mathbf{D}, \mathbf{C})$ müssen in \mathbb{C}_- liegen.

Im Folgenden sei ein Beobachter der Form

$$\begin{aligned}\dot{\hat{\mathbf{z}}}_1(t) &= \mathbf{A}_{11} \hat{\mathbf{z}}_1(t) + \mathbf{A}_{12} \hat{\mathbf{z}}_2(t) + \mathbf{B}_1 \mathbf{u}(t) + \mathbf{A}_{12} \mathbf{e}_y(t) \\ \dot{\hat{\mathbf{z}}}_2(t) &= \mathbf{A}_{21} \hat{\mathbf{z}}_1(t) + \mathbf{A}_{22} \hat{\mathbf{z}}_2(t) + \mathbf{B}_2 \mathbf{u}(t) + [\mathbf{A}_{22} - \mathbf{A}_{22}^s] \mathbf{e}_y(t) - \boldsymbol{\nu}\end{aligned}\quad (15)$$

mit der stabilen Entwurfsmatrix $\mathbf{A}_{22}^s \in \mathbb{R}^{p \times p}$ angenommen, worin $\mathbf{e}_y(t) = \mathbf{y}(t) - \hat{\mathbf{y}}(t)$ den Ausgangsfehler beschreibt. Weiterhin tritt in (15) der diskontinuierliche Schaltvektor $\boldsymbol{\nu}$ auf:

$$\boldsymbol{\nu} = \begin{cases} -\rho(t, \mathbf{y}, \mathbf{u}) \|\mathbf{D}_2\| \frac{\mathbf{P}_2 \mathbf{e}_y(t)}{\|\mathbf{P}_2 \mathbf{e}_y(t)\|} & , \text{ falls } \mathbf{e}_y(t) \neq \mathbf{0} \\ \mathbf{0} & , \text{ sonst} \end{cases} \quad (16)$$

Darin ist $\rho : \mathbb{R}_+ \times \mathbb{R}^q \times \mathbb{R}^m \rightarrow \mathbb{R}_+$ ein Skalar und erfüllt die Bedingung

$$\rho(t, \mathbf{y}, \mathbf{u}) \geq r_1 \|\mathbf{u}\| + \alpha(t, \mathbf{y}) + \gamma_0 \quad (17)$$

Weiterhin sind γ_0 ein positiver Skalar und $\mathbf{P}_2 \in \mathbb{R}^{p \times p}$ ist die symmetrische, positiv definite Lösung der Lyapunovgleichung für \mathbf{A}_{22}^s (vgl. [10]). Mit den Zustandsbeobachtungsfehlern $\mathbf{e}_1(t) = \mathbf{z}_1(t) - \hat{\mathbf{z}}_1(t)$ und $\mathbf{e}_2(t) = \mathbf{z}_2(t) - \hat{\mathbf{z}}_2(t) = \mathbf{e}_y(t)$ ergeben sich aus (14) und (15) die folgenden Fehlerdifferentialgleichungen:

$$\begin{aligned}\dot{\mathbf{e}}_1(t) &= \mathbf{A}_{11} \mathbf{e}_1(t) \\ \dot{\mathbf{e}}_y(t) &= \mathbf{A}_{21} \mathbf{e}_1(t) + \mathbf{A}_{22}^s \mathbf{e}_y(t) + \boldsymbol{\nu} + \mathbf{D}_2 \boldsymbol{\xi}\end{aligned}\quad (18)$$

Theorem 2: Das nichtlineare Fehlerdifferentialgleichungssystem (18) ist quadratisch stabil und eine Gleitbewegung (Sliding-Motion) findet statt, so dass $\mathbf{e}_y(t) = \mathbf{0}$ in endlicher Zeit erreicht wird. Daraus folgt, dass das System (15) ein robuster Sliding-Mode-Beobachter für das System (14) ist.

Der vollständige Beweis dieses Theorems ist in [10] angegeben.

Der Beobachter kann in nicht transformierter Form (d. h. in den ursprünglichen Koordinaten) angegeben werden:

$$\dot{\hat{\mathbf{x}}}(t) = \mathbf{A} \hat{\mathbf{x}}(t) + \mathbf{B} \mathbf{u}(t) + \mathbf{G}_l \mathbf{e}_y(t) - \mathbf{G}_n \boldsymbol{\nu} \quad (19)$$

$$\hat{\mathbf{y}}(t) = \mathbf{C} \hat{\mathbf{x}}(t) \quad (20)$$

mit $\hat{\mathbf{x}}(t) \in \mathbb{R}^n$, $\hat{\mathbf{u}}(t) \in \mathbb{R}^m$, $\hat{\mathbf{y}}(t) \in \mathbb{R}^p$ und den linearen und nichtlinearen Verstärkungsmatizen

$$\mathbf{G}_l = \mathbf{T}_0^{-1} \begin{bmatrix} \mathbf{A}_{12} \\ \mathbf{A}_{22} - \mathbf{A}_{22}^s \end{bmatrix} \quad \text{und} \quad \mathbf{G}_n = \mathbf{T}_0^{-1} \begin{bmatrix} \mathbf{0} \\ \mathbf{I}_p \end{bmatrix} \quad (21)$$

Die \mathbf{G}_l -Matrix ist eine gewöhnliche Luenberger-Rückführmatrix, um $\mathbf{A} - \mathbf{G}_l \mathbf{C}$ zu stabilisieren und das Einschwingverhalten des Beobachters zu dimensionieren. Durch eine geeignete Wahl von $\rho(t, \mathbf{y}, \mathbf{u})$ in (16) kann eine ideale Gleitbewegung auf der Schaltfläche $S_0 = \{\mathbf{e}(t) \in \mathbb{R}^n : \mathbf{C} \mathbf{e}(t) = \mathbf{e}_y(t) = \mathbf{0}\}$ einsetzen. Für die Wahl von $\rho = 0$ entfällt der nichtlineare Schaltterm und der Beobachter verhält sich wie ein gewöhnlicher Luenberger-Beobachter, mit zunehmendem ρ verstärkt sich der zusätzliche diskontinuierliche Schaltterm $\boldsymbol{\nu}$.

3 Anwendung der Beobachterkonzepte zur Fehlerüberwachung

Der Fokus dieser Arbeit liegt in der Untersuchung von zwei strukturell verschiedenen Beobachteransätzen zur Drucksensorüberwachung in hydrostatischen Fahrdrives. Dabei wird zunächst nur die Symptomerkennung und nicht die Ursachenermittlung betrachtet. Diese Einschränkung ermöglicht eine gewisse Flexibilität während der Beobachterentwürfe. Dies ist notwendig, um die zuvor dargestellten Beobachteransätze bestmöglich miteinander vergleichen zu können. Dieser Abschnitt ist folgendermaßen gegliedert: Zunächst wird ein nichtlineares physikalisches Modell zur Beschreibung der Dynamik hydrostatischer Fahrdrives angegeben. Basierend auf diesem Modell wird eine äquivalente TS Modellstruktur der Form (1) bestimmt. Abschließend werden unter Verwendung dieses TS Modells die einzelnen Beobachter entworfen.

3.1 Nichtlineares Streckenmodell hydrostatischer Antriebe

Die Dynamik des in [5], [6] beschriebenen hydrostatischen Fahrdrives wird durch das nichtlineare Zustandsraummodell vierter Ordnung

$$\begin{aligned}
 \dot{x}_1(t) &= -\frac{1}{T_{u_P}} x_1(t) + \frac{k_P}{T_{u_P}} u_1(t) \\
 \dot{x}_2(t) &= -\frac{1}{T_{u_M}} x_2(t) + \frac{k_M}{T_{u_M}} u_2(t) \\
 \dot{x}_3(t) &= \frac{10}{C_H} \left(\tilde{V}_{\max_P} x_1(t) \omega_P(t) - \tilde{V}_{\max_M} x_2(t) x_4(t) - k_{\text{leak}} x_3(t) \right) \\
 \dot{x}_4(t) &= \frac{1}{J_v} i_g^2 i_a^2 \eta_g \eta_{mh} \tilde{V}_{\max_P} 10^{-4} x_2(t) x_3(t) - \tilde{d}_{vc} i_a^2 x_4(t) - M_{L_w}(t) i_g i_a
 \end{aligned} \tag{22}$$

mit den skalierten Verdrängungsvolumen

$$\tilde{V}_{\max_P} := \frac{V_{\max_P}}{2\pi}, \quad \tilde{V}_{\max_M} := \frac{V_{\max_M}}{2\pi}, \tag{23}$$

dem Zustandsvektor

$$\mathbf{x}(t) = [x_1(t), x_2(t), x_3(t), x_4(t)]^T := [\tilde{\alpha}_P(t), \tilde{\alpha}_M(t), \Delta p(t), \omega_M(t)]^T \tag{24}$$

und dem Eingangsvektor

$$\mathbf{u}(t) = [u_1(t), u_2(t)]^T := [u_P(t), u_M(t)]^T \tag{25}$$

repräsentiert. Die verwendeten Symbole sind in der Tabelle 1 näher beschrieben.

Das Differenzialgleichungssystem (22) läßt sich durch Herauslösen der Nichtlinearitäten und Konzentration in die Zugehörigkeitsfunktionen h_i in das (innerhalb der Grenzen der Sektornichtlinearitäten) äquivalente Takagi-Sugeno Fuzzy System überführen

$$\begin{aligned}
 \dot{\mathbf{x}}(t) &= \sum_{i=1}^r h_i(z_1(t), z_2(t)) \mathbf{A}_i \mathbf{x}(t) + \mathbf{B} \mathbf{u}(t) \\
 \mathbf{y}(t) &= \mathbf{C} \mathbf{x}(t)
 \end{aligned} \tag{26}$$

Tabelle 1: Variablen und Parameter des hydrostatischen Fahrtriebs

Symbol	Beschreibung	Wert	Einheit
x_1	Schwenkwinkel Hydropumpe, normiert ($x_1 = \tilde{\alpha}_P$)	$\in \{-1; 1\}$	-
x_2	Schwenkwinkel Hydromotor, normiert ($x_1 = \tilde{\alpha}_M$)	$\in \{0; 1\}$	-
x_3	Differenzdruck ($x_3 = \Delta p$)	[-450 ; 450]	bar
x_4	Drehzahl Hydromotor ($x_4 = \omega_M$)	[-340 ; 340]	rad/s
u_1	Stellsignal Hydropumpe, normiert ($u_1 = u_P$)	$\in \{-1; 1\}$	-
u_2	Stellsignal Hydromotor, normiert ($u_2 = u_M$)	$\in \{0; 1\}$	-
T_{u_P}	Zeitkonstante Hydropumpe	0,13	s
T_{u_M}	Zeitkonstante Hydromotor	0,22	s
k_P	Statische Verstärkung Hydropumpenverstellung	241,67	-
k_M	Statische Verstärkung Hydromotorverstellung	283,33	-
k_{leak}	Leckflusskoeffizient	0,14	mm ³ /sbar
C_H	Hydraulische Kapazität	1840,8	mm ⁵ /N
\tilde{V}_{max_P}	max. Verdrängungsvolumen Hydropumpe	145	cm ³
\tilde{V}_{max_M}	max. Verdrängungsvolumen Hydromotor	170	cm ³
ω_P	Drehzahl Hydropumpe	[105 ; 262]	rad/s
J_v	Massenträgheitsmoment	16512	Nms ²
i_g	Getriebeübersetzung	-6,12	-
i_a	Achsübersetzung	-23,3	-
η_g	mechanischer Getriebewirkungsgrad	0,98	-
η_{mh}	hydromechanischer Wirkungsgrad Hydromotor	0,697	-
\tilde{d}_{vc}	viskoser Dämpfungskoeffizient	0,33	Nms
M_{L_w}	externes Lastmoment an Antriebsachse	-	Nm

mit $r = 4$, $z_1 := x_2$, $z_2 := \omega_P$, den Matrizen

$$\mathbf{A}_i = \begin{bmatrix} -\frac{1}{T_{u_P}} & 0 & 0 & 0 \\ 0 & -\frac{1}{T_{u_M}} & 0 & 0 \\ \frac{10\tilde{V}_{max_P} *_2}{C_H} & 0 & \frac{-10k_{leak}}{C_H} & \frac{-10\tilde{V}_{max_M} *_1}{C_H} \\ 0 & 0 & \gamma *_1 & -\frac{\tilde{d}_{vc}}{J_v} i_a^2 \end{bmatrix}, \mathbf{B} = \begin{bmatrix} \frac{k_P}{T_{u_P}} & 0 \\ 0 & \frac{k_M}{T_{u_M}} \\ 0 & 0 \\ 0 & 0 \end{bmatrix}, \mathbf{C} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (27)$$

mit $*_1 \in [\underline{x}_2, \bar{x}_2]$, $*_2 \in [\underline{\omega}_P, \bar{\omega}_P]$ und $\gamma = \frac{1}{J_v} i_g^2 i_a^2 \eta_g \eta_{mh} \tilde{V}_{max_P} 10^{-4}$.

Die Zugehörigkeitsfunktionen

$$\begin{aligned} h_1(z_1, z_2) &= w_{11}(z_1) \cdot w_{21}(z_2), \\ h_2(z_1, z_2) &= w_{12}(z_1) \cdot w_{21}(z_2), \\ h_3(z_1, z_2) &= w_{11}(z_1) \cdot w_{22}(z_2), \\ h_4(z_1, z_2) &= w_{12}(z_1) \cdot w_{22}(z_2) \end{aligned} \quad (28)$$

in (26) werden aus dem Produkt der Sektorfunktionen

$$w_{j1}(z_j) = \frac{\bar{z}_j - z_j}{\bar{z}_j - \underline{z}_j}, \quad w_{j2}(z_j) = \frac{z_j - \underline{z}_j}{\bar{z}_j - \underline{z}_j} \quad (29)$$

berechnet wobei $\underline{z}_1 := \underline{x}_2$, $\bar{z}_1 := \bar{x}_2$, $\underline{z}_2 := \underline{\omega}_P$ und $\bar{z}_2 := \bar{\omega}_P$. Die notwendigen Transformationsschritte sind ausführlich in [6] beschrieben.

3.2 Ergebnisse zum TS Beobachterentwurf

Mit dem TS Modell des hydrostatischen Antriebs (26), den zugehörigen Antriebsstrangparametern aus Tabelle 1 und der gewählten Abklingrate $\alpha = 2,0$ erhält man für die linearen Matrixgleichungen (11) die mittels des LMI-Gleichungslösers der Matlab Robust Control Toolbox [12] numerisch berechnete Lösung:

$$\mathbf{P} = 2,6282 \cdot 10^{-2}$$

$$\begin{aligned} \mathbf{N}_1 &= \begin{bmatrix} 0 & 0 & -9,0124 \cdot 10^{-5} \end{bmatrix}, & \mathbf{N}_2 &= \begin{bmatrix} 0 & 0 & -2,1937 \cdot 10^{-6} \end{bmatrix}, \\ \mathbf{N}_3 &= \begin{bmatrix} 0 & 0 & -4,5062 \cdot 10^{-5} \end{bmatrix}, & \mathbf{N}_4 &= \begin{bmatrix} 0 & 0 & +5,7926 \cdot 10^{-6} \end{bmatrix} \end{aligned}$$

Die Beobachterrückführungsmatrizen aus (6), (7) ergeben sich dann zu

$$\begin{aligned} \mathbf{L}_1 &= \begin{bmatrix} 0 & 0 & -3,4291 \cdot 10^{-3} \end{bmatrix}, & \mathbf{L}_2 &= \begin{bmatrix} 0 & 0 & -8,3468 \cdot 10^{-5} \end{bmatrix}, \\ \mathbf{L}_3 &= \begin{bmatrix} 0 & 0 & -1,7145 \cdot 10^{-3} \end{bmatrix}, & \mathbf{L}_4 &= \begin{bmatrix} 0 & 0 & +2,2040 \cdot 10^{-4} \end{bmatrix}. \end{aligned}$$

3.3 Ergebnisse zum Sliding-Mode Beobachterentwurf

Für den Beobachterentwurf wurden die Funktionen `obsform` und `disobs.m` der SMC Matlab Control Toolbox von Edwards und Spurgeon [13] verwendet. Das für den Beobachterentwurf erforderliche lineare Streckenmodell wurde in dieser Arbeit direkt aus dem TS Modell (26) als eingefrorenes System mit fest gewählten z -Koordinaten $z_1 = 1$ und $z_2 = 273,58$ rad/s berechnet (d. h. hierzu wurde keine Linearisierung mittels Taylorreihenentwicklung um diesen Arbeitspunkt durchgeführt). Diese explizite Wahl der z -Koordinaten entspricht einem Arbeitspunkt, bei dem der Hydromotor voll ausgeschwenkt ist und die Hydropumpe mit einer konstanten Drehzahl von 273,58 rad/s betrieben wird. Hierfür ergeben sich die folgenden Systemmatrizen:

$$\begin{aligned} \mathbf{A} &= \begin{bmatrix} 7,6923 & 0 & 0 & 0 \\ 0 & -4,5454 & 0 & 0 \\ 34298 & 0 & -12,676 & -146,98 \\ 0 & 0 & -2,1965 & -0,1047 \end{bmatrix}, \\ \mathbf{B} &= \begin{bmatrix} 7,6923 & 0 \\ 0 & 4,5454 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}, & \mathbf{C} &= \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \end{aligned}$$

Um eine ähnliche Fehlerabklingdynamik der zu vergleichenden Beobachter zu erhalten, wurde der Pol des reduzierten TS Beobachters ($p_{TS} = -8,4825$) auch für die Pole des Sliding-Mode Beobachters verwendet ($p_{SM} = -8,4825 \cdot [1; 1; 1; 1]$). Für diese Beobach-

terpolvorgabe erhält man die folgenden Matrizen:

$$\mathbf{A}_{22}^s = -8,4825 \cdot \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad \mathbf{P}_2 = 0,0589 \cdot \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix},$$

$$\mathbf{G}_l = \begin{bmatrix} 0,7902 & 0 & 0 & 0 \\ 0 & 3,9371 & 0 & 0 \\ 34298 & 0 & -4,1933 & -146,98 \\ 0 & 0 & 2,1905 & 8,3779 \end{bmatrix}, \quad \mathbf{G}_n = 7,6923 \cdot \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

4 Vergleich der Beobachterkonzepte

In diesem Abschnitt werden die zuvor entworfenen Beobachter auf ihre Eignung zur Differenzdruckschätzung in hydrostatischen Fahrtrieben untersucht. Für diese Aufgabe werden im Folgenden Simulationen des nichtlinearen physikalischen Modells (22) sowie auf einem Testgelände mit einem Serienfahrzeug aufgenommene Messungen herangezogen.

Um die Vergleichbarkeit der verschiedenen Beobachter zu gewährleisten, wurden die Beobachterpole identisch gewählt. Zur Überprüfung wurden in einem ersten Versuch das Einschwingverhalten der Beobachter anhand einer Simulation analysiert. Dabei wurden die Beobachter mit den Ein- und Ausgangsdaten des nichtlinearen physikalischen Simulationsmodells (22) betrieben. Die Anfangswerte des Streckenmodells betragen $\mathbf{x}(0) = [0; 1; 5; 0]^T$ und die des Beobachters $\hat{\mathbf{x}}(0) = [0; 0; 0; 0]^T$. Weiterhin wurde der diskontinuierliche Schaltvektor ν durch die Wahl des Designparameters $\rho = 0$ auf Null gesetzt. Hierdurch verhält sich der SM Beobachter wie ein gewöhnlicher, linearer Luenberger-Beobachter. Die Simulation wurde ohne Parameterunsicherheiten durchgeführt. Wie anhand des Simulationsergebnisses in Bild 1 zu erkennen ist, klingen die Beobachtungsfehler des reduzierten TS Beobachters (—) und des SM Beobachters mit den Polen $\mathbf{p}_{SM} = -8,48 \cdot [1; 1; 1; 1]$ (—·) bei unterschiedlichen Anfangswerten nahezu gleich schnell ab. Zum Zeitpunkt $t = 0,5s$ sind die Beobachtungsfehler beider Beobachter annähernd verschwunden. Zum Vergleich ist in dem Bild zusätzlich ein SM Beobachter mit den Polen $\mathbf{p}_{SM} = -3 \cdot [1; 1; 1; 1]$ (—) abgebildet. Wie zu erwarten, klingt dessen Beobachtungsfehler deutlich langsamer ab.

In einer weiteren Untersuchung wurden drei verschiedene Beschleunigungs- und Verzögerungsmanöver, wie zuvor ohne Parameterunsicherheiten simuliert. Dabei wurde die Fahrzeugbeschleunigung bzw. -verzögerung durch das Verstellen der Hydropumpe (ansteigendes u_P bedeutet Beschleunigung) bei konstantem Schluckvolumen des Hydromotors ($u_M = \text{konst.}$) eingeleitet. Mit dieser Einschränkung, auch bekannt als Primärverstellung, verhält sich das Modell des Hydrostaten bei konstanter Dieseldrehzahl bzw. Pumpendrehzahl ω_P rein linear. Die Anfangswerte des Streckenmodells und der Beobachter wurden dabei identisch gewählt ($\mathbf{x}(0) = \hat{\mathbf{x}}(0) = [0; 0; 0; 0]^T$). Beide Beobachter liefern sehr gute Differenzdruckschätzungen (vgl. Bild 2).

In den Bildern 3 und 4 sind die Ergebnisse derselben Simulation, jedoch mit begrenzter Parameterunsicherheit, abgebildet. Hierzu wurde der Leckflußkoeffizient k_{leak0} der

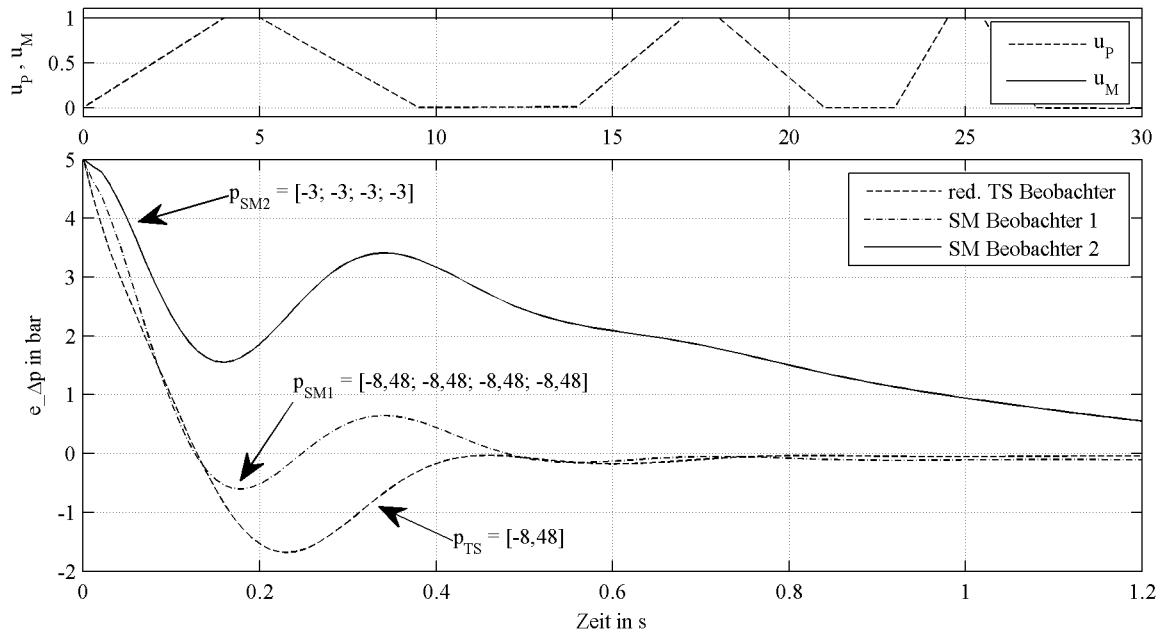


Bild 1: Vergleich des Einschwingverhaltens von reduziertem TS Beobachter mit den SM Beobachtern 1 und 2 (unterschiedliche Beobachterpole, $\rho = 0$)

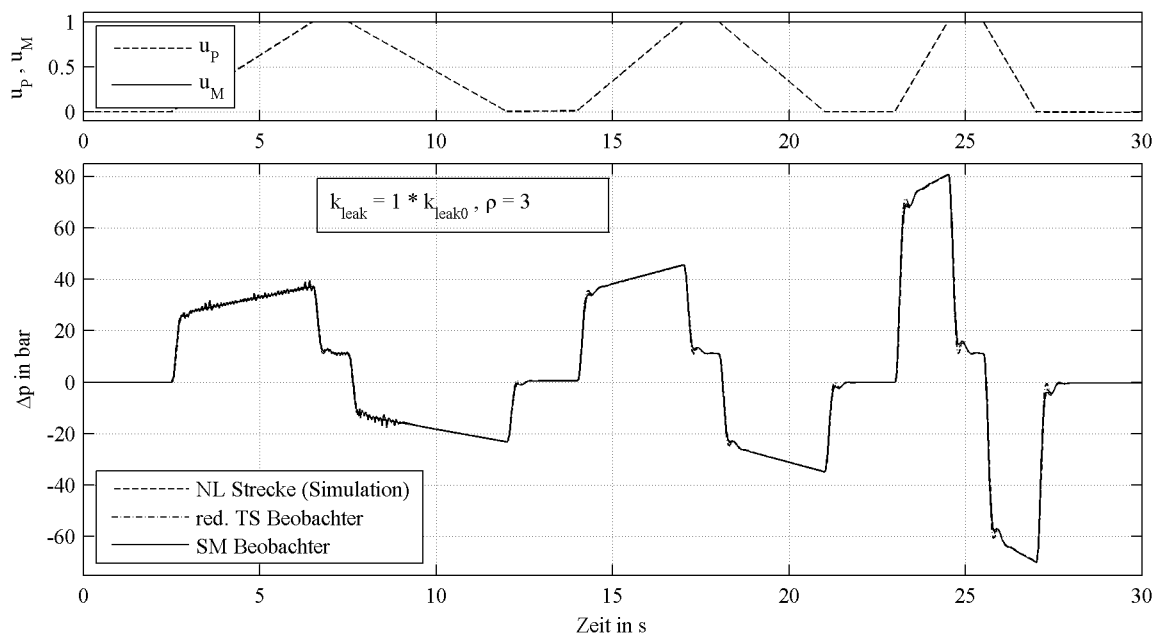


Bild 2: Vergleich des simulierten Differenzdrucks der nichtlinearen Strecke (22) mit den Schätzungen des reduzierten TS Beobachters und des SM Beobachters für eine rein lineare Betriebsphase (ohne Parameterunsicherheit).

Strecke um 15% gegenüber dem der Beobachter reduziert. Weiterhin wurde der Designparameter ρ des SM Beobachters in Bild 3 zu $\rho = 0$ (keine Robustheit gegenüber Parameterfehlern) und in Bild 4 zu $\rho = 10$ gewählt, wodurch der nichtlineare Schaltterm des SM Beobachters aktiviert wird. Dies führte dazu, dass in Bild 3 beide Beobachter aufgrund der Modellunsicherheit nur fehlerhafte Schätzwerte lieferten. Andererseits ist eine Robustheitssteigerung des SM Beobachters durch den zusätzlichen nichtlinearen Schalt-

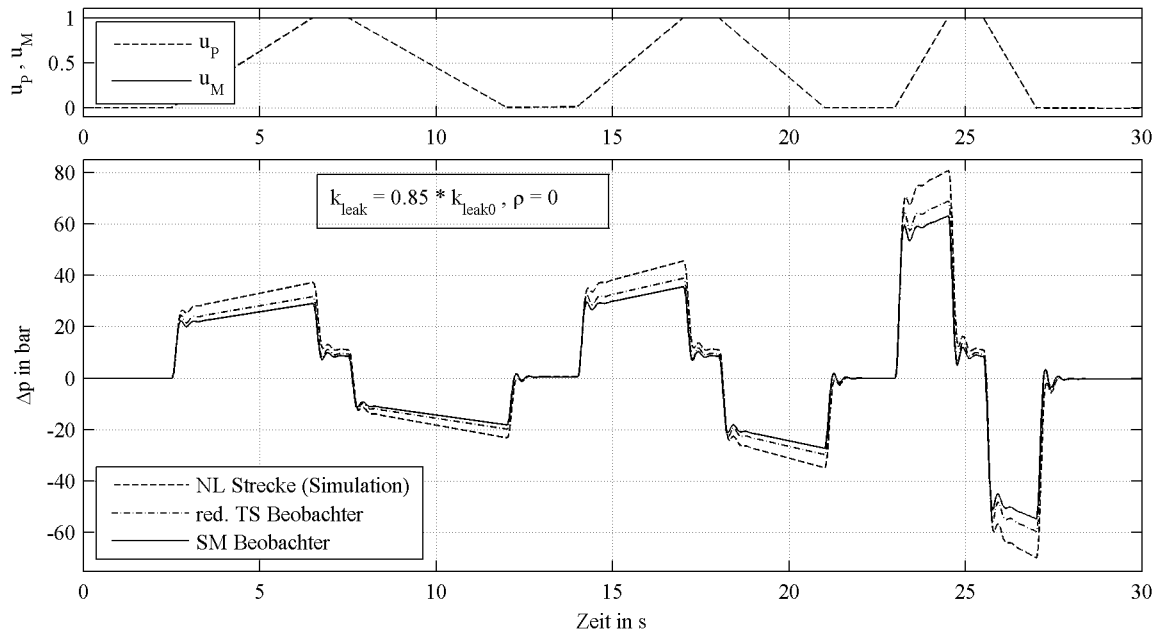


Bild 3: Vergleich des simulierten Differenzdrucks der nichtlinearen Strecke (22) mit den Schätzungen des reduzierten TS Beobachters bzw. des rein linearen ($\rho = 0$) SM Beobachters für eine lineare Betriebsphase (mit Parameterunsicherheit; 15% Fehler auf k_{leak}).

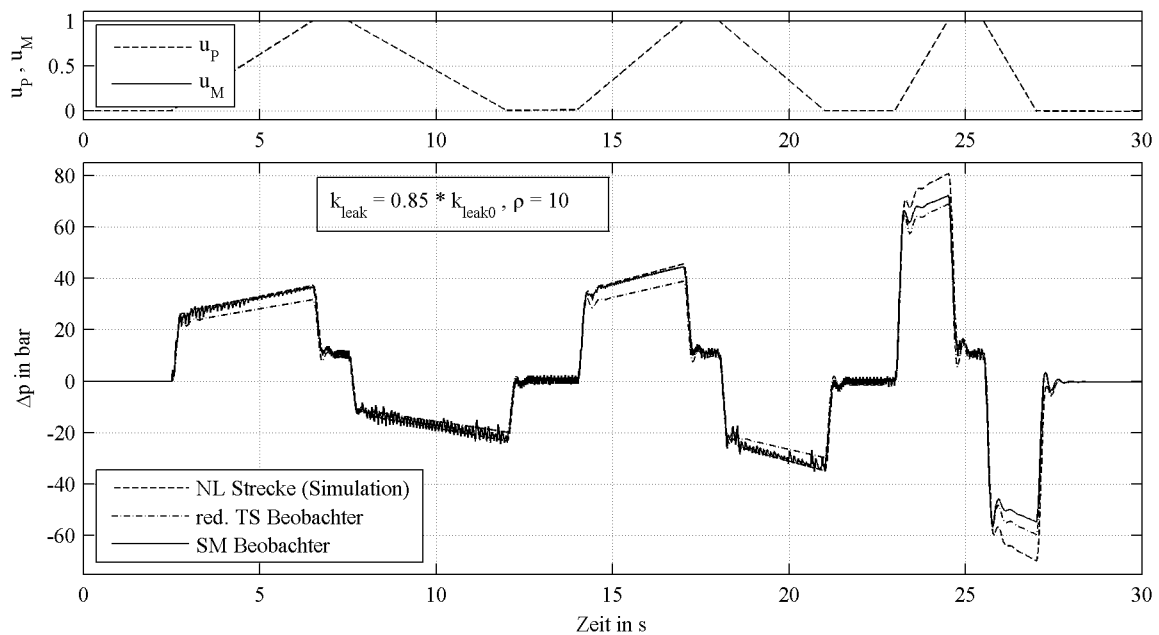


Bild 4: Vergleich des simulierten Differenzdrucks der nichtlinearen Strecke (22) mit den Schätzungen des reduzierten TS Beobachters bzw. des robusten SM Beobachters ($\rho = 10$) für eine lineare Betriebsphase (mit Parameterunsicherheit; 15% Fehler auf k_{leak}).

term in Bild 4 zu erkennen.

Das Bild 5 zeigt das Simulationsergebnis für den Fall, dass zusätzlich zur Hydropumpe auch der Hydromotor verstellt wird. Diese Simulation wurde wieder ohne Parameterunsicherheiten durchgeführt. Wie aus dem Bild hervorgeht, ist eine Differenzdruckschätzung aufgrund des stark nichtlinearen Systemverhaltens wegen der zusätzlichen Hydromotor-

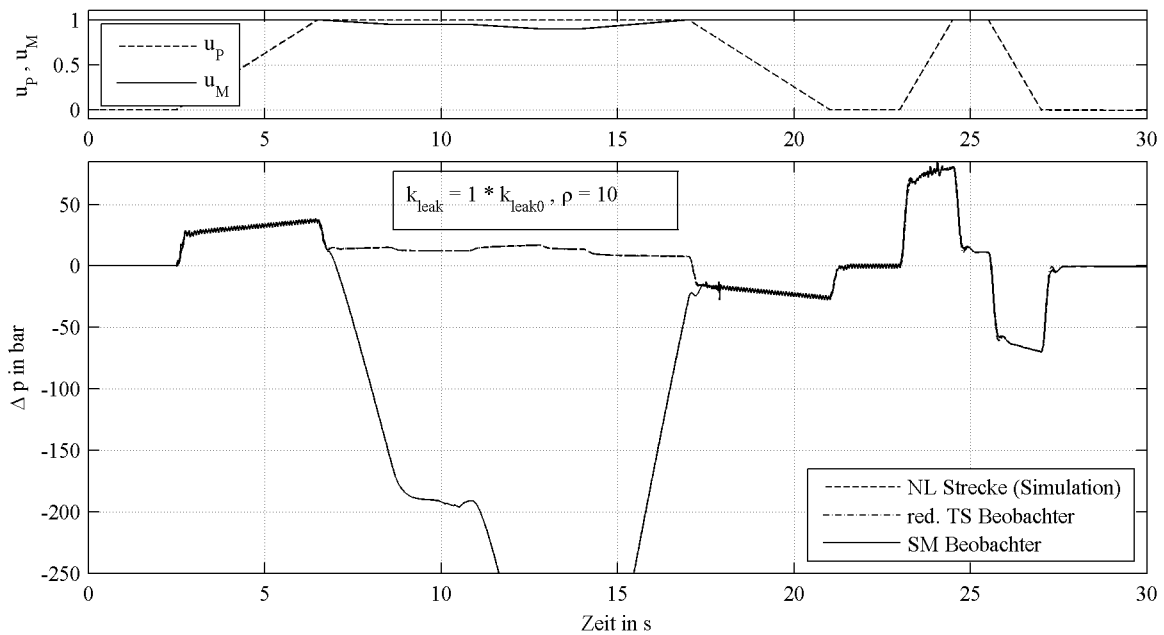


Bild 5: Vergleich des simulierten Differenzdrucks der nichtlinearen Strecke (22) mit den Schätzungen des reduzierten TS Beobachters bzw. des robusten SM Beobachters ($\rho = 10$) für den vollständigen Betriebsbereich (ohne Parameterunsicherheiten).

verstellung nur mit dem reduzierten TS Beobachter möglich. Der SM Beobachter liefert ein temporär sehr fehlerhaftes Schätzergebnis und ist somit zur Differenzdruckschätzung in hydrostatischen Fahrtrieben ungeeignet. In einer abschließenden Untersuchung wurde der reduzierte TS Beobachter mit den Messwerten einer realen Testfahrt validiert. Das Ergebnis dieser Untersuchung ist in Bild 6 abgebildet. Auch hierbei ist eine für die Praxis ausreichende Übereinstimmung der gemessenen und der geschätzten Differenzdrücke feststellbar.

Aus den Simulationsstudien lassen sich nun die folgenden Ergebnisse ableiten:

- Im rein linearen Betriebsbereich und ohne Parameterunsicherheiten liefern beide Beobachter sehr gute Differenzdruckschätzungen.
- Im rein linearen Betriebsbereich und bei existierendem Parameterfehler kann man aufgrund der Robustheitseigenschaft des SM Beobachters deutlich bessere Differenzdruckschätzungen erzielen als mit dem TS Beobachter.
- Aufgrund des extrem nichtlinearen Systemverhaltens über den vollständigen Arbeitsbereich des Hydrostaten ist der SM Beobachter in diesem Fall ungeeignet. Im Gegensatz dazu liefert der TS Beobachter sehr gute Schätzergebnisse.

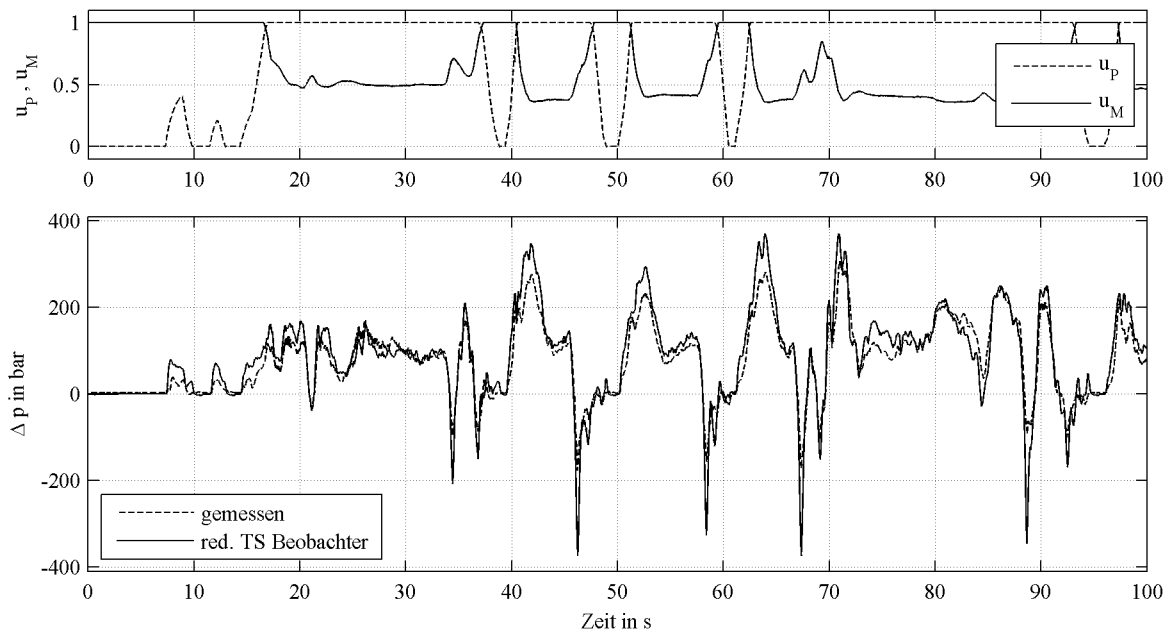


Bild 6: Validierung des mittels reduziertem TS Beobachter geschätzten Differenzdrucks anhand einer realen Messung.

5 Zusammenfassung und Ausblick

In dieser Arbeit wurden die Ergebnisse einer vergleichenden Untersuchung der Einsatzmöglichkeit zweier nichtlinearer Beobachterkonzepte zur kontinuierlichen Überwachung des dynamischen Druckaufbaus in nichtlinearen hydrostatischen Fahrtrieben präsentiert. Im Einzelnen wurden ein reduzierter Takagi-Sugeno (TS) Fuzzy Beobachter und ein Sliding-Mode (SM) Beobachter betrachtet. Anhand experimenteller Ergebnisse wurden die jeweiligen Charakteristika, aber auch die Grenzen der einzelnen Verfahren aufgezeigt und miteinander verglichen. Es wurde demonstriert, dass der TS Fuzzy Beobachter im Gegensatz zum SM Beobachter in der Lage ist, das stark nichtlineare Prozessverhalten über den vollständigen Arbeitsbereich zu beschreiben. Andererseits weist der TS Fuzzy Beobachter im Gegensatz zum SM Beobachter eine geringere Robustheit gegenüber begrenzten Modellfehlern auf.

Aktuelle Forschungsaktivitäten befassen sich mit der Integration der Methoden zu TS-SM Beobachtern als Weiterentwicklung des Ansatzes von [14].

Literatur

- [1] M. Blanke, M. Kinneart, J. Lunze, M. Staroswiecki: *Diagnosis and Fault-Tolerant Control*. Heidelberg: Springer, 2003.
- [2] J. Chen, R. J. Patton: *Robust model-based fault diagnosis for dynamic systems*. Dordrecht: Kluwer Academic Publishers, 1999.
- [3] R. Isermann: *Fault-Diagnosis Systems. An Introduction from Fault Detection to Fault Tolerance*. Heidelberg: Springer, 2006.

- [4] K. Wolters, D. Söffker: Diagnoseverfahren und Notlaufkonzepte mechatronischer Systeme. *Abschlussbericht zum Forschungsvorhaben Nr. 408, Heft 724*, Forschungsvereinigung Antriebstechnik, 2004.
- [5] H. Schulte: Modeling of Hydrostatic Transmissions considering leakage losses. *IFAC Workshop on Advanced Fuzzy-Neural Control Proceedings*, Oktober 2007, Valenciennes, France.
- [6] H. Schulte: Robuster Beobachterentwurf für Takagi-Sugeno Fuzzy Systeme zur Überwachung hydrostatischer Fahrtriebe. *Proc., 18. Workshop Computational Intelligence*, Universitätsverlag Karlsruhe, 2008.
- [7] G. F. Franklin, J. D. Powell, A. Emami-Naeini: *Feedback Control of Dynamic Systems*. 2nd edition, Addison-Wesley, 1998.
- [8] K. Tanaka, H. Wang: *Fuzzy Control Systems Design and Analysis*. New York, Chichester: John Wiley & Sons, 2001.
- [9] C. Edwards, S. K. Spurgeon: On the development of discontinuous observers. *Int. Journal of Control*, Vol. 59, 1994, S. 1211-1229.
- [10] C. Edwards, S. K. Spurgeon: *Sliding mode control: Theory and applications*. London: Taylor & Francis, 1998.
- [11] B. L. Walcott, S. H. Žak: Combined observer-controller synthesis for uncertain dynamical systems with applications. *IEEE Transactions on Systems, Man and Cybernetics*, Vol. 18, 1988, S. 88-104.
- [12] G. Balas, R. Chiang, A. Packard, M. Safonov: *Robust Control Toolbox*, The Mathworks Inc., 2006.
- [13] C. Edwards, S. K. Spurgeon: A Sliding Mode Control Matlab Toolbox
<http://www.le.ac.uk/eg/ce14/vscbook/vscbook.html>
- [14] R. Palm: Sliding Mode Observer for a Takagi Sugeno Fuzzy system. *Ninth IEEE International Conference on Fuzzy Systems*, Mai 2000, San Antonio, USA, S. 665-670.

Schnelles Rescheduling von Gridjobs mit Heuristiken und dem Evolutionären Algorithmus GLEAM

Wilfried Jakob, Alexander Quinte, Karl-Uwe Stucky, Wolfgang Süß

Forschungszentrum Karlsruhe GmbH
Institut für Angewandte Informatik (IAI)
Postfach 3640, 76021 Karlsruhe
Tel. (07247) 82-4663
Fax (07247) 82-2602

E-Mail: {wilfried.jakob, alexander.quinte, uwe.stucky, wolfgang.suess}@iai.fzk.de

Zusammenfassung

Auf Grund der dynamischen Struktur eines Grids bestehend aus heterogenen Rechner-Ressourcen ist eine permanente Umplanung innerhalb weniger Minuten notwendig. Anlass dafür können beispielsweise neue Jobs, Ressourcenausfall oder neue Ressourcen sein. Dies kommt im Grid im Gegensatz zu vielen anderen Produktionsplanungsaufgaben häufiger vor, was die Anforderungen an die multikriterielle Optimierung unter den gegebenen Zeitbeschränkungen noch verschärft. Die hier vorgestellten Ergebnisse sind eine Fortführung von Arbeiten zur Neuplanung in einem quasi leeren Grid, mit denen wir unseren Global Optimising Resource Broker and Allocator GORBA erprobt und verbessert haben und über den in diesem Rahmen bereits berichtet wurde.

Nach einer formalen Definition und Klassifikation des Problems werden neue Heuristiken zur Umplanung vorgestellt, die die Informationen des alten Plans ausnutzen und es wird untersucht, inwieweit sie zur Umplanung beitragen. Außerdem wird über Experimente zur maximalen Last (Jobs und Rechnerressourcen), die in drei Minuten bearbeitbar ist, berichtet.

1 Einleitung

Der Begriff „computational grid“ oder „Grid“ bezeichnet ein heterogenes Rechnernetz, das als virtualisiertes und verteiltes Rechenzentrum verstanden werden kann [1]. Benutzer beschreiben ihre *Anwendungsjobs* durch so genannte *Workflows*, die aus einzelnen elementaren *Gridjobs* bestehen. Ein Workflow kann durch einen gerichteten azyklischen Graphen dargestellt werden, der die Reihenfolgebeziehungen zwischen den einzelnen Gridjobs beschreibt. Außerdem legt der Benutzer fest, welche Ressourcen, wie Anwendungsprogramme, Datenspeicher oder Rechnerleistung benötigt werden, um die einzelnen Gridjobs auszuführen. Dabei können Ressourcen andere Ressourcen benötigen. Z.B. erfordert ein Softwarewerkzeug ein bestimmtes Betriebssystem und eine Hardware, um darauf ausgeführt werden zu können. Daraus folgt das Konzept der Koallokation von Ressourcen. Weitere Angaben der Benutzer betreffen das zur Verfügung stehende Kostenbudget, eine späteste Fertigstellungszeit und eine Präferenz zwischen schneller und kostengünstiger Ausführung. Dazu kommt die erwartete Ausführungszeit der einzelnen Gridjobs, was bei neuen Anwendungen auf Schätzungen beruhen kann und bei älteren auf Erfahrungswerten. Die Betreiber von Ressourcen bieten diese zu unterschiedlichen Tages- oder Wochenzeiten an, wobei auch tageszeit- oder wochentagabhängige Kostenmodelle vorgegeben werden können. Weiterhin unterscheiden sich heterogene Ressourcen meist in ihrer Performance und weisen unterschiedliche Preis-

Leistungs-Verhältnisse auf. Schließlich können manche Ressourcen auf Grund vorhandener Lizenzen im Vergleich zur angebotenen Hardware nur eingeschränkt zur Verfügung stehen.

Ressourcennutzer und –anbieter haben unterschiedliche Interessen: So sind die Anbieter an einer hohen *Auslastung* und damit an kurzer *Gesamtbearbeitungszeit* interessiert. Die Nutzer wollen hingegen, dass die vorgegebenen *Kostenbudgets* und *Zeitlimits* eingehalten oder, noch besser, unterschritten werden. Ein Teil dieser Kriterien steht offensichtlich miteinander in Konflikt, wie z.B. der Wunsch nach schneller *und* billiger Ausführung.

Da davon ausgegangen werden kann, dass Gridjobs Bearbeitungszeiten benötigen, die eher im Stunden- als im Minutenbereich liegen, steht eine bestimmte, wenn auch begrenzte Zeit für Planung und Optimierung zur Verfügung. Im Rahmen dieser Untersuchung wurde ein Zeitraum von drei Minuten als vertretbar angesehen. Wenn ein Umplanungsereignis eintritt, werden alle bereits begonnen Gridjobs und alle, die innerhalb dieses Zeitraums begonnen werden sollen, fixiert und sind damit nicht Gegenstand der Umplanung. Somit wird die Abarbeitung der Gridjobs durch den wiederholten Prozess des Reschedulings nicht gestört.

Während die hier vorgestellte Rescheduling-Aufgabe auf Grund ihrer Besonderheiten vergleichsweise neu ist, ist die Behandlung von Schedulingaufgaben dagegen eine klassische Disziplin des Operations Research. Einen guten Überblick und eine Klassifikation mit entsprechenden Lösungsansätzen liefert z. B. Bruckner [2, 3].

Der zweite Abschnitt enthält eine Zusammenfassung der in [4] ausführlich behandelten formalen Definition des Problems zusammen mit der ebenfalls in [4] detailliert beschriebenen multikriteriellen Bewertungsfunktion. Anschließend erfolgt eine Klassifizierung der Aufgabe und es wird eine Übersicht über vergleichbare Arbeiten gegeben. Der dritte Abschnitt ist der Beschreibung der verwendeten Algorithmen und darunter insbesondere der neuen Umplanungsheuristiken gewidmet. Der Frage nach dem Effekt der neuen Heuristiken und nach Obergrenzen für die in drei Minuten bearbeitbarer Gridjobs und Ressourcenmengen wird in Abschnitt 4 nachgegangen.

2 Definition des Optimierungsproblems und Einordnung

Gegeben seien entsprechend der in [2] beschriebenen Notation eine Ressourcen-Menge $M = \{M_1, \dots, M_m\}$, eine Menge $J = \{J_1, \dots, J_j\}$ von Anwendungsjobs und eine Menge O von Gridjobs. Die n Gridjobs eines Anwendungsjobs J_i werden mit O_{i1}, \dots, O_{in} bezeichnet. Folgende Funktionen seien gegeben:

- Eine Vorgängerfunktion $p: O \times O \rightarrow \{TRUE, FALSE\}$ für die Gridjobs.
- Eine Zuordnungsfunktion $\mu: O \rightarrow \mathcal{P}(\mathcal{P}(M))$ von Gridjobs zu Ressourcenmengen, wobei $\mathcal{P}(M)$ die Potenzmenge von M bezeichnet. Mit μ_{ij} wird die Menge aller möglichen Kombinationen von Ressourcen aus M bezeichnet, die gemeinsam zur Bearbeitung von O_{ij} fähig sind.
- Eine Zeitbedarfsfunktion $t: O \times \mathcal{P}(M) \rightarrow \mathbb{R}$, die für jeden Gridjob O_{ij} den Zeitbedarf zur Ausführung auf einer Ressourcenmenge $R_{ij} \in \mu_{ij}$ angibt.
- Eine Kostenfunktion $c: \mathbb{R} \times \mathcal{P}(M) \rightarrow \mathbb{R}$, die zu jedem Zeitpunkt $z \in \mathbb{R}$ die Kosten pro Zeiteinheit einer Ressourcenmenge angibt.

Die Optimierung erfolgt durch eine geeignete Wahl der Startzeitpunkte $s(O_{ij}) \in \mathfrak{R}$ und der Ressourcenzuordnungen $R_{ij} \in \mu_{ij}$. Eine gültige Lösung muss folgende zwei Kriterien erfüllen:

1. Alle Gridjobs wurden geplant und die Ressourcen wurden exklusiv belegt:

$$\forall O_{ij} : \exists s(O_{ij}) \in \mathfrak{R}, R_{ij} \in \mu_{ij} : \forall M_j \in R_{ij} : \quad (1)$$

M_j ist in $[s(O_{ij}); s(O_{ij}) + t(O_{ij}, R_{ij})]$ exklusiv belegt durch O_{ij} .

2. Vorgängerbeziehungen wurden eingehalten:

$$\forall i, j \neq k : p(O_{ij}, O_{ik}) \Rightarrow s(O_{ik}) \geq s(O_{ij}) + t(O_{ij}, R_{ij}) \quad (2)$$

Die Bewertung einer Verletzung der beiden nachfolgenden Beschränkungen erfolgt durch Straffunktionen, welche sowohl den Umfang der Kosten- oder Zeitüberschreitungen bewerten als auch die Anzahl der davon betroffenen Anwendungsjobs. Sie ergeben einen Faktor zwischen Null und Eins mit dem die „unbestrafte Fitness“ multipliziert wird.

1. Fristen d_i werden eingehalten:

$$\forall J_i : d_i \geq s(O_{in}) + t(O_{in}, R_{in}) \quad \text{wobei } O_{in} \text{ der letzte Gridjob von } J_i \text{ ist.} \quad (3)$$

2. Kostenlimits c_i werden eingehalten:

$$\forall J_i : c_i \geq \sum_{j=1}^n \int_{s(O_{ij})}^{s(O_{ij}) + t(O_{ij}, R_{ij})} c(z, R_{ij}) dz \quad (4)$$

Die in [4] ausführlicher beschriebene Fitnessfunktion basiert auf der gewichteten Summe der zuvor genannten vier Kriterien plus einem Hilfskriterium. Es unterstützt die Verkürzung der Bearbeitungszeit und damit die Einhaltung des *Zeitlimits* der Anwendungsjobs, indem es die frühere Fertigstellung nicht-terminaler Gridjobs belohnt. Ein Gridjob ist nicht-terminal, wenn er keine Nachfolger hat. Damit wird ein früherer Beginn der terminalen Gridjobs und in der Folge eine Verkürzung der Gesamtbearbeitungszeit ermöglicht. Zur Bewertung werden die Verzögerungen aller nicht-terminalen Gridjobs gegenüber ihrer frühest möglichen Startzeit ermittelt und der Durchschnitt gebildet.

Die Berechnung der Fitnesswerte der einzelnen Kriterien basiert auf Unter- und Obergrenzen für Kosten und Zeiten, die in der ersten Planungsphase durch eine Analyse des Workflows ermittelt werden. Mit Ausnahme der *Auslastung* ergibt sich der Fitnesswert f_i eines Kriteriums i aus dem aktuellen Wert $Kriterium_{i,akt}$ durch

$$f_i = \frac{Kriterium_{i,akt} - Kriterium_{i,min}}{Kriterium_{i,max} - Kriterium_{i,min}} \quad (5)$$

Dadurch werden die Werte f_i unabhängig von der aktuellen Aufgabe. Die sich ergebenden prozent-ähnlichen Werte werden gewichtet und zur Rohfitness aufsummiert, welche bei Budgetverletzungen mit den entsprechenden Straffunktionen multipliziert wird oder andernfalls unverändert bleibt und die Endfitness darstellt. Um den bei der gewichteten Summe möglichen Kompensationseffekten entgegenzuwirken werden die Kriterien einzeln oder in Gruppen gemäß einer Priorität sortiert und erhalten einen so genannten Erfüllungswert. Die Kriterien der höchsten Priorität tragen immer zur Summenbildung bei, während die anderen erst zugeschaltet werden, wenn die Kriterien der nächst höheren Ebene jeweils mindestens den Erfüllungswert erreicht haben. Die Gewichtung beruht auf Erfahrung und ist derzeit so eingestellt, dass ein Ausgleich zwischen den Interessen der Gridakteure angestrebt wird.

Die geschilderte Aufgabenstellung enthält das NP-vollständige Job-Shop-Scheduling Problem als Spezialfall. Es gibt folgende Erweiterung gegenüber dieser klassischen Aufgabe des Operations Research:

- Ressourcenalternativen statt einer Ressource pro Gridjob
- Heterogene Ressourcenalternativen
- Zeitlich beschränkte statt unbeschränkter Verfügbarkeit von Ressourcen
- Gridjobs können mehrere statt nur einen Nachfolger oder Vorgänger haben (Parallele Bearbeitungsstränge in den Workflows)
- Koallokation von Ressourcen möglich statt nicht zugelassen
- Mehrere statt nur ein Bewertungskriterium, insbesondere Kosten

Allein schon auf Grund der NP-Vollständigkeit kann generell nur mit Näherungslösungen gerechnet werden.

Ein vergleichbares Problem konnte in der Literatur nicht gefunden werden, siehe auch [2, 3]. In [5] wird z.B. festgestellt, dass es nur wenig Arbeiten zu multikriteriellem Scheduling gibt, die sich meistens auch nur mit Ein-Maschinen-Problemen befassen. Im Grid-Kontext sind in jüngster Zeit weitere Arbeiten zu multikriterieller Bewertung erschienen. In [6] wird berichtet, dass sich die meisten mit nur zwei Kriterien beschäftigen, wovon das eine optimiert wird und das andere lediglich als Beschränkung dient, siehe z.B. [7, 8]. Der in [8] gewählte Ansatz matrix-förmiger Chromosomen dürfte mit dafür verantwortlich sein, dass nur eine geringe Anzahl von Gridjobs (ca. 30) innerhalb einer Stunde geplant werden konnte. Kurowski et al. [9] verwenden eine modifizierte Form der gewichteten Summe zur multikriteriellen Bewertung, planen aber keine Workflows. Insgesamt konnte keine Arbeit gefunden werden, die in wenigen Minuten eine Menge an Ressourcen und durch Workflows organisierte Gridjobs einer global optimierenden multikriteriellen Planung unterzieht, welche mit dem hier behandelten Umfang vergleichbar ist. Natürlich gibt es eine Vielzahl von Veröffentlichungen und Verfahren zu Teilaspekten des Problems. Zum Beispiel den bekannten Giffler-Thompson-Algorithmus [10, 11], der für die vorliegende Aufgabenstellung erweitert wurde, aber schlechtere Ergebnisse lieferte als die nachfolgend beschriebenen Heuristiken [12].

3 Algorithmen von GORBA und Ergebnisse früherer Arbeiten

GORBA [4, 12, 13] plant in zwei Schritten: Im ersten Schritt werden die Planungsdaten auf Plausibilität geprüft und einer statischen Analyse unterzogen, die erste Ergebnisse für die Unter- und Obergrenzen der einzelnen Kriterien liefert. Danach werden erste Schedules mit Hilfe von Heuristiken erzeugt, die die Unter- und Obergrenzen verfeinern und im Falle ausbleibender Belegungskonflikte bereits das Endergebnis darstellen. Andernfalls dienen sie zur Initialisierung der Startpopulation des im zweiten Schritt folgenden Laufs des Evolutionären Algorithmus (EA) GLEAM (General Learning Evolutionary Algorithm and Method) [14, 15, 16].

Zunächst werden die alten Heuristiken zur Erzeugung von Schedules in einem leeren Grid [4, 12, 13] beschrieben, da die neuen auf diesen aufbauen. Als erster Schritt wird eine Gridjobsequenz gemäß einer der folgenden drei heuristischen Regeln unter Beachtung der Vorgängerbeziehungen gebildet:

1. Gridjobs des Anwendungsjobs mit dem frühesten Fertigstellungstermin zuerst
2. Gridjobs des Anwendungsjobs mit der kürzesten Bearbeitungszeit zuerst
3. Gridjobs mit der kürzesten Bearbeitungszeit zuerst.

Danach werden die drei folgenden Ressourcenallokationsstrategien (RAS) angewandt, so dass insgesamt neun Schedules entstehen:

- RAS-1: billigste Ressource, die zum frühest möglichen Zeitpunkt frei ist, wählen
- RAS-2: schnellste Ressource, die zum frühest möglichen Zeitpunkt frei ist, wählen
- RAS-3: Verwendung von RAS-1 oder RAS-2 für alle Gridjobs eines Anwendungs-jobs gemäß seiner Präferenz für kostengünstige oder schnelle Ausführung

Die ersten beiden RAS gelten jeweils für alle zu planenden Anwendungsjobs, so dass deren Gridjobs die gleiche Ressourcenpräferenz anwenden, während die dritte diese Zuordnung differenzierter auf der Ebene der Anwendungsjobs vornimmt.

Die neuen Heuristiken für das Rescheduling verwenden die Gridjobsequenz des alten Plans für alle noch zu planenden Gridjobs. Das sind alle, die zum Zeitpunkt des Umplanungsereignisses noch nicht gestartet wurden und auch nicht innerhalb der Umplanungszeit von 3 Minuten gestartet werden. Gridjobs neuer Anwendungsjobs werden an die Sequenz der alten entsprechend den drei heuristischen Regeln zur Sequenzbildung angehängt. Auf die so entstandenen drei Sequenzen werden jeweils die drei RAS angewandt, woraus wieder neue Schedules resultieren, diesmal aber basierend auf den Vorgaben des alten Plans.

Zusammen mit den alten Heuristiken entstehen so insgesamt 18 Schedules, die bei der Initialisierung der Startpopulation des nachfolgenden GLEAM-Laufs Verwendung finden. Die Standardversion von GLEAM [15, 16] enthält bereits eine Reihe genetischer Operatoren, die für kombinatorische Aufgabenstellungen gut geeignet sind. Aus Platzgründen werden sie hier nur kurz beschrieben und der interessierte Leser wird auf [4, 13, 16] verwiesen. Neben der üblichen Mutation zur Veränderung der Genreihenfolge gibt es Mutationen, die ganze Gensequenzen (so genannte Segmente) verschieben oder ihre interne Reihenfolge umkehren. Die Segmente stellen eine der evolutionären Veränderung unterworfenen Metastruktur der Chromosomen dar. 1- und n-Punkt Crossover setzen an den Segmentgrenzen an. Beiden Operatoren ist eine genotypische Reparatur nachgeschaltet, die dafür sorgt, dass bei den resultierenden Nachkommen keine Gene fehlen. Dazu kommt das *order-based crossover* [17], das die relative Genreihenfolge der Eltern bewahrt und hier an die Segmentierung angepasst wurde [12, 4].

In einer ersten Entwicklungsphase von GORBA wurden die Algorithmen und Heuristiken basierend auf Neuplanungen getestet und bei zwei unterschiedlichen Codierungen und Reparaturarten evaluiert [13, 4]. Dazu wurden vier unterschiedliche Benchmark-Szenarien verwendet, die die Kombinationen von kleinen und großen Ressourcenalternativen und geringen und großen Gridjobabhängigkeiten wiedergeben. Sie werden mit *sR* und *lR* für *small/large Resource alternatives* und *sD* und *lD* für *small/large grid job Dependencies* bezeichnet. Diese wurden mit vier unterschiedlichen Lastmengen getestet, so dass alle Varianten an Hand von 16 Benchmarks verglichen wurden. Die erste Codierung bildet die Ressourcenzuordnung auf die Parameter der Gene ab und die Gridjobsequenz auf die Genreihenfolge im Chromosom. Damit sind zwar alle Kombinationen erreichbar, aber der zugehörige Suchraum ist entsprechend groß. Bei der zweiten Codierung entfallen die Ressourcenparameter und die Ressourcenauswahl erfolgt durch die RAS, die durch ein spezielles Gen ausgewählt wird. Die zweite Codierung hat sich innerhalb der kurzen Bearbeitungszeit ab einer Last von 100 Gridjobs als günstiger erwiesen [13, 4]. Die phänotypische Reparatur von Reihenfolgeverletzungen hat bessere Resultate geliefert als eine genotypische. Die Details und die Erklärung dafür sind in [13, 4] zu finden. Als Ergebnis der Voruntersuchungen verwenden wir für die hier be-

handelten Fragestellungen die zweite Codierung, phänotypische Reparatur, das bereits erwähnte Hilfskriterium und das *order-based crossover* [13, 12, 4].

Da die zweite Codierung Grundlage der vorliegenden Untersuchung ist, soll sie hier näher vorgestellt werden. Jedem Gridjob entspricht ein Gen. Die Belegungsmatrix, die die zeitliche Belegung der Ressourcen durch Gridjobs abbildet und die man sich als Gantt-Chart vorstellen kann [13], wird in der Reihenfolge der Gridjob-Gene im Chromosom aufgebaut. Ein besonderes Gen legt dabei die zu verwendende RAS fest und bringt dadurch ein Element der Koevolution mit ein. Der Scheduling-Vorgang pro Gen sieht folgendermaßen aus:

1. Die früheste Startzeit des Gridjobs ist entweder die früheste Startzeit seines Anwendungsjobs oder die späteste Fertigstellungszeit seiner Vorgänger, sofern vorhanden.
2. Gemäß der durch das RAS-Gen festgelegten RAS wird eine Liste alternativer Ressourcen für jede primäre Ressource des Gridjobs angelegt.
3. Beginnend mit den ersten Ressourcen dieser Listen wird die Bearbeitungszeit berechnet und ein freies Zeitfenster bei allen primären und abhängigen Ressourcen gesucht, wobei mit der frühesten Startzeit von Schritt 1 begonnen wird.
4. Die gefundenen Ressourcen werden für den Gridjob und die berechnete Zeit reserviert, d.h., er wird entsprechend in die Belegungsmatrix eingetragen.

4 Experimentelle Ergebnisse für schnelles Rescheduling

Es gibt eine ganze Reihe von Gründen für ein Rescheduling, wobei das Eintreffen eines oder mehrerer neuer Anwendungsjobs das häufigste sein dürfte. Weitere derartige Ereignisse sind Jobabbruch, neu hinzukommende Ressourcen, Ressourcenausfall, deutliche Abweichung von Jobausführungszeiten von der Planzeit, Preisänderungen bei Ressourcen, usw. Die Experimente basieren auf dem am häufigsten erwarteten Umplanningereignis neuer Jobs und sollen die folgenden drei Fragen beantworten:

1. Profitiert das Rescheduling vom alten Schedule? Wenn ja, bis zu welchem Anteil beendeter und neuer Gridjobs?
2. Wie effektiv sind die alten und neuen Heuristiken und der nachfolgende EA-Lauf?
3. Bis zu welcher Menge an Gridjobs und Ressourcen verbessert GLEAM den besten heuristisch erzeugten Schedule?

Da sich die beiden Benchmark-Szenarien basierend auf einem großen Abhängigkeitsgrad der Gridjobs als die schwierigeren erwiesen haben [4, 12], werden sie für die Experimente zu Grunde gelegt. Sie werden mit *sRID* und *lRID* (*small* oder *large Resource alternatives* / *large Dependencies*) bezeichnet. Die Zeit- und Kostenbudgets wurden bewusst so gewählt, dass die Heuristiken keine Schedules erzeugen konnten, bei denen die Budgets vollständig eingehalten wurden. Ein Kriterium für den Erfolg des Einsatzes von GLEAM war daher das Auffinden von Schedules ohne Budgetverletzungen, was auch in fast allen Benchmarkszenarien erreicht wurde [13, 12, 4].

Die Rescheduling-Experimente zu den ersten beiden Fragestellungen basieren auf diesen Benchmarks, so dass zwei Bewertungskriterien verglichen werden können: Budgeteinhaltung und Fitness-Wert der Schedules. Der einzige EA-Parameter, der bei den Versuchen variiert wurde, ist die Populationsgröße, die für die ersten beiden Fragestellungen zwischen 90 und 900 lag. Bei der Ermittlung der maximal verarbeitbaren Last musste, wie nachfolgend dargestellt, auch mit kleineren Größen gearbeitet werden. Für

jede Benchmark-Situation und Populationsgröße wurden 50 Läufe durchgeführt und die Ergebnisse gemittelt. Zur Absicherung der statistischen Relevanz unterschiedlicher Mittelwerte wurden Konfidenzintervalle und t-Tests bei 99% Sicherheit verwendet.

Zur Beantwortung der ersten beiden Fragen wurde die Abarbeitung der Planungen für die beiden Benchmarkszenarien bei 100 und 200 Gridjobs sowie 10 Ressourcen nach Abarbeitung von 10 oder 20% der Gridjobs gestoppt. Dann kamen jeweils 10, 20, 30, oder 50% neue Gridjobs hinzu, wobei sich der Prozentsatz auf die ursprüngliche Größe bezieht. Daraus ergeben sich 32 Benchmarks für die Umplanung. Die Konzentration auf relativ wenig abgearbeitete Gridjobs wurde gewählt, da diese Situation in der Praxis wahrscheinlicher ist. Außerdem würden mehr abgearbeitete Gridjobs der bereits bearbeiteten Situation „Neuplanung“ näher kommen.

Bild 1 vergleicht die Ergebnisse aller 32 Benchmarks. Zur Interpretation der dargestellten normalisierten Fitness muss darauf hingewiesen werden, dass bestimmte geringe Schwankungen wegen Abweichungen bei den Benchmarks unerheblich sind. Werte zwischen 50 und 70% können als gute Ergebnisse interpretiert werden, da die oberen und unteren Grenzen theoretischer Natur sind und in der Regel nicht erreicht werden können. Ergebnisse nahe bei 100% würden entweder auf einen trivialen Fall oder einen Softwarefehler hindeuten. Das wichtigste Ergebnis ist, dass für 10% neue Gridjobs alle acht Szenarien gute Resultate liefern. Der Beitrag der beiden Heuristikgruppen ist deutlich situationsabhängig und in Fällen schwacher Ergebnisse kann GLEAM diese kompensieren. Mit anderen Worten, wenn die Heuristiken gut funktionieren, bleibt wenig Raum für weitere Verbesserungen innerhalb der geringen Planungszeit.

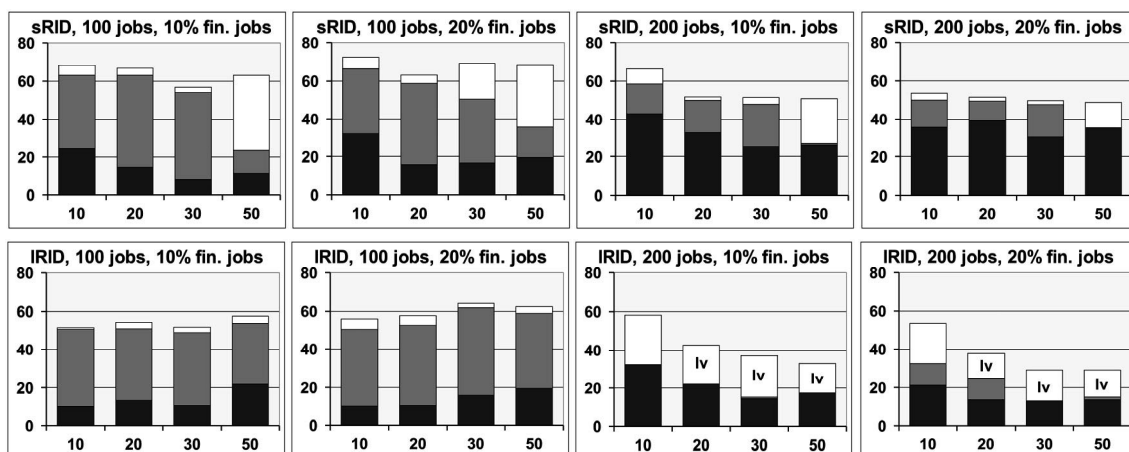


Bild 1: Vergleich der Fitnessanteile, die von den alten Heuristiken (dunkelgrau), den neuen Umplanungsheuristiken (hellgrau) und von GLEAM (weiß) stammen. X-Achse: Anteil neuer Gridjobs in Prozent relativ zur ursprünglichen Anzahl, Y-Achse: normalisierte Endfitness.

Alle GLEAM-Läufe verbessern die Fitness signifikant. Selbst im Falle der geringsten Verbesserung von Benchmark IRID bei 100 Gridjobs und 10% fertigen und neuen Gridjobs ist das beste heuristische Resultat eindeutig außerhalb des Konfidenzintervalls des GLEAM-Ergebnisses (Der Abstand ist dreimal größer als notwendig).

Abkürzungen soweit nicht im Text erklärt: lv: Budgetverletzungen (limit violations), wobei meistens nur noch ein bis drei Anwendungsjobs Endtermine in geringem Maße überschreiten.

Ein weiteres erfreuliches Ergebnis ist, dass diese Kompensationen auch in gewissem Umfang bei mehr neuen Gridjobs als 10% stattfinden, auch wenn die Budgetverletzungen nicht immer vollständig beseitigt werden können (siehe die sechs mit lv gekennzeichneten).

zeichneten Säulen). Es kann erwartet werden, dass mehr neue Gridjobs den Beitrag der Umplanungsheuristiken schmälern und Bild 1 bestätigt dies auch tatsächlich für 50% neue Gridjobs. Der Fall *IRID* bei 200 Gridjobs und 10% abgearbeiteten ist insofern eine Ausnahme, als die Umplanungsheuristiken keinen oder nur einen geringen Beitrag leisten können.

Tabelle 1 fasst den Anteil aller Heuristiken am Gesamtergebnis der heuristischen Planung zusammen. Für alle 32 Benchmarks wird die Fitness jeder Heuristik als Prozentwert der besten Heuristik dieser Parametrierung berechnet. Aus den vier Werten für die beiden Gridjobmengen und *sRID* sowie *IRID* wird der Durchschnitt gebildet und in die Tabelle eingetragen. Die rechte Spalte enthält dann den Durchschnitt der Werte für die verschiedenen Anteile beendeter und neuer Gridjobs. Die alten Heuristiken basierend auf kurzen Arbeitszeiten in der unteren Hälfte der Tabelle zeigen das gleiche schlechte Verhalten wie bei der Neuplanung eines leeren Grids [12]. Setzt man sie dagegen zum Rescheduling ein, liefern sie durchaus einige gute Beiträge. Gemäß der Tabelle könnte vermutet werden, dass RAS-3 immer am besten abschneidet. Das hier aus Platzgründen nicht gezeigte Rohmaterial enthält aber dreizehn Fälle, in denen die beiden anderen RAS die besseren Ergebnisse liefern. Daher ist es sinnvoll, sie alle beizubehalten, zumal die Berechnung eines Schedules sehr schnell in Millisekunden erfolgt. Es könnte sich aber als sinnvoll erweisen, die sechs alten Heuristiken basierend auf kurzen Arbeitszeiten wegzulassen.

Tabelle 1: Vergleich der Beiträge aller Heuristiken für unterschiedliche Anteile fertiger und neuer Gridjobs. Die besten Werte jeder Spalte sind dunkelgrau hinterlegt und Werte, die mindestens 90% des besten Wertes erreichen, hellgrau.

Abkürzungen: RH: Rescheduling-Heuristik, KBZ: kürzeste Bearbeitungszeit, RAS: Ressourcenallokationsstrategie, siehe Abschnitt 3.

Abgearbeitete Gridjobs:	10%				20%				Durchschnitt
Neue Gridjobs:	10%	20%	30%	50%	10%	20%	30%	50%	
RH früheste Fertigstellung & RAS-3	0.90	0.96	0.88	0.92	0.86	0.83	0.89	0.92	0.90
RH früheste Fertigstellung & RAS-2	0.70	0.44	0.73	0.80	0.64	0.59	0.75	0.61	0.66
RH früheste Fertigstellung & RAS-1	0.48	0.44	0.54	0.45	0.59	0.22	0.53	0.45	0.46
RH KBZ des Gridjobs & RAS-3	0.97	0.86	0.81	0.69	0.94	0.78	0.81	0.62	0.81
RH KBZ des Gridjobs & RAS-2	0.74	0.42	0.63	0.53	0.66	0.50	0.68	0.39	0.57
RH KBZ des Gridjobs & RAS-1	0.47	0.41	0.46	0.28	0.57	0.24	0.54	0.26	0.40
RH KBZ des Anwendungsjobs & RAS-3	0.90	0.88	0.82	0.70	0.86	0.83	0.77	0.70	0.81
RH KBZ des Anwendungsjobs & RAS-2	0.70	0.41	0.70	0.56	0.64	0.51	0.57	0.46	0.57
RH KBZ des Anwendungsjobs & RAS-1	0.48	0.44	0.57	0.31	0.59	0.24	0.49	0.43	0.44
früheste Fertigstellung & RAS-3	0.45	0.42	0.35	0.56	0.45	0.41	0.47	0.51	0.45
früheste Fertigstellung & RAS-2	0.58	0.52	0.38	0.72	0.51	0.43	0.56	0.69	0.55
früheste Fertigstellung & RAS-1	0.42	0.42	0.39	0.54	0.39	0.37	0.37	0.45	0.42
KBZ des Gridjobs & RAS-3	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01
KBZ des Gridjobs & RAS-2	0.01	0.01	0.01	0.01	0.02	0.01	0.01	0.01	0.01
KBZ des Gridjobs & RAS-1	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01
KBZ des Anwendungsjobs & RAS-3	0.02	0.02	0.02	0.02	0.03	0.02	0.02	0.02	0.02
KBZ des Anwendungsjobs & RAS-2	0.02	0.02	0.02	0.02	0.02	0.02	0.02	0.02	0.02
KBZ des Anwendungsjobs & RAS-1	0.02	0.02	0.02	0.02	0.02	0.02	0.02	0.02	0.02

Zur Untersuchung der dritten Fragestellung wird das Umplanungsereignis mit 10% abgearbeiteten und neuen Gridjobs basierend auf den beiden Benchmarkszenarien *sRID* und *IRID* bei einer proportional steigenden Zahl von Gridjobs und Ressourcen zu Grunde gelegt. Die Bewertung basiert erstens auf dem durch GLEAM gewonnenen Fitnesszuwachs im Vergleich zum besten heuristischen Ergebnis und zweitens auf der Erfolgsrate. Letztere ist der Anteil der Läufe ohne Budgetverletzung an allen 50 Läufen pro Benchmarkparametrierung. Bild 2 zeigt die Ergebnisse. Wie erwartet, fallen Erfolgsrate und EA-Verbesserung mit zunehmender Last. Bis etwa 400 - 500 Gridjobs und 40 - 50 Ressourcen können bei beiden Benchmarkszenarien Budgetverletzungen beseitigt werden. Die relativ großen Schwankungen der EA-Verbesserungen können mit den unterschiedlichen Fähigkeiten der Heuristiken zur Vermeidung von Budgetverletzungen erklärt werden. Da Verletzungen vergleichsweise hart bestraft werden, können relativ kleine Unterschiede große Fitnessdifferenzen erzeugen, was dann seinerseits die Verbesserungsrate entsprechend schwanken lässt. In zwei Fällen erzeugen bereits die Heuristiken straffreie Schedules, sodass dem EA wenig Raum für Verbesserungen bleibt. Sie sind in Bild 2 durch ein weißes statt graues Dreieck gekennzeichnet.

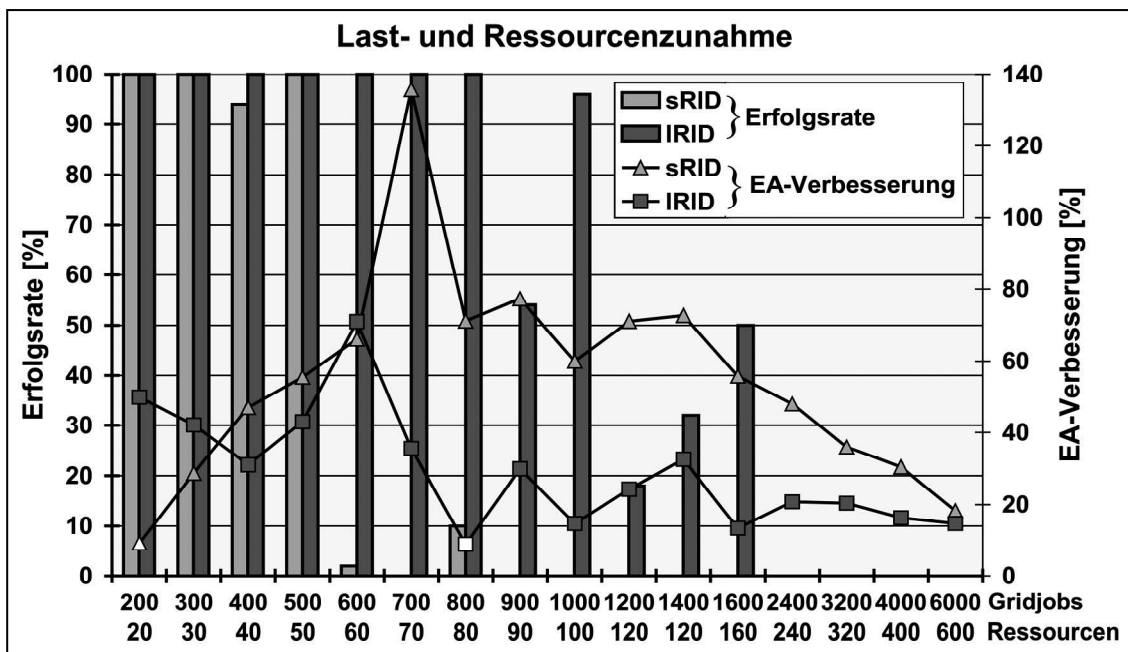


Bild 2: Auswirkungen von steigender Last auf die Erfolgsrate und die EA-Verbesserung gegenüber der besten heuristischen Planung bei beiden untersuchten Benchmarkszenarien und 10% abgearbeiteten und neuen Gridjobs.

Bild 2 zeigt auch, dass es mit wachsender Last beim *sRID*-Benchmark im Vergleich zum *IRID*-Fall schwieriger ist, die Budgetvorgaben einzuhalten. Es ist offenbar leichter dieses Ziel zu erreichen, wenn mehr Ressourcen zur Auswahl stehen. Selbst für mehr als 1600 Gridjobs und 160 Ressourcen können noch Verbesserungen durch den EA erreicht werden. Ab dieser Last sinkt die Verbesserungsrate kontinuierlich. Daraus kann geschlossen werden, dass selbst bei bis zu 6000 Gridjobs und 600 Ressourcen immer noch Verbesserungen unterhalb des Niveaus der vollständigen Einhaltung der Budgets durch GLEAM erreicht werden können. Mit anderen Worten, die Anzahl verletzungsfreier Anwendungsjobs wird durch den EA-Lauf gegenüber der heuristischen Planung immer noch vergrößert.

Je mehr Gridjobs geplant werden müssen und je mehr Ressourcen zur Verfügung stehen, desto größer wird die Belegungsmatrix und desto weniger Schedules können in der gegebenen Zeit von drei Minuten durchgerechnet werden. Bild 3 zeigt, wie dieser Betrag bei steigender Last kontinuierlich sinkt. Je größer die Anzahl alternativer Ressourcen, desto mehr Ressourcen müssen durch die RAS geprüft werden. Das dauert länger und erklärt die geringeren Werte für *IRID* im Vergleich zu *sRID*. Am Ende sinkt die Anzahl möglicher Evaluationen so stark ab, dass die Populationsgröße auf 20 bis 30 reduziert werden muss, um wenigstens noch etwa zwei Dutzend Generationen durchrechnen zu können. Dies sind geringe Werte für einen EA ohne Unterstützung durch ein lokales Suchverfahren, siehe [18]. Daraus ergibt sich auch, dass die vorliegende Implementierung kaum größere Lasten verkraften können. Abhilfe kann hier zum Beispiel eine verbesserte Implementierung der Evaluation oder ein schnellerer Rechner schaffen.

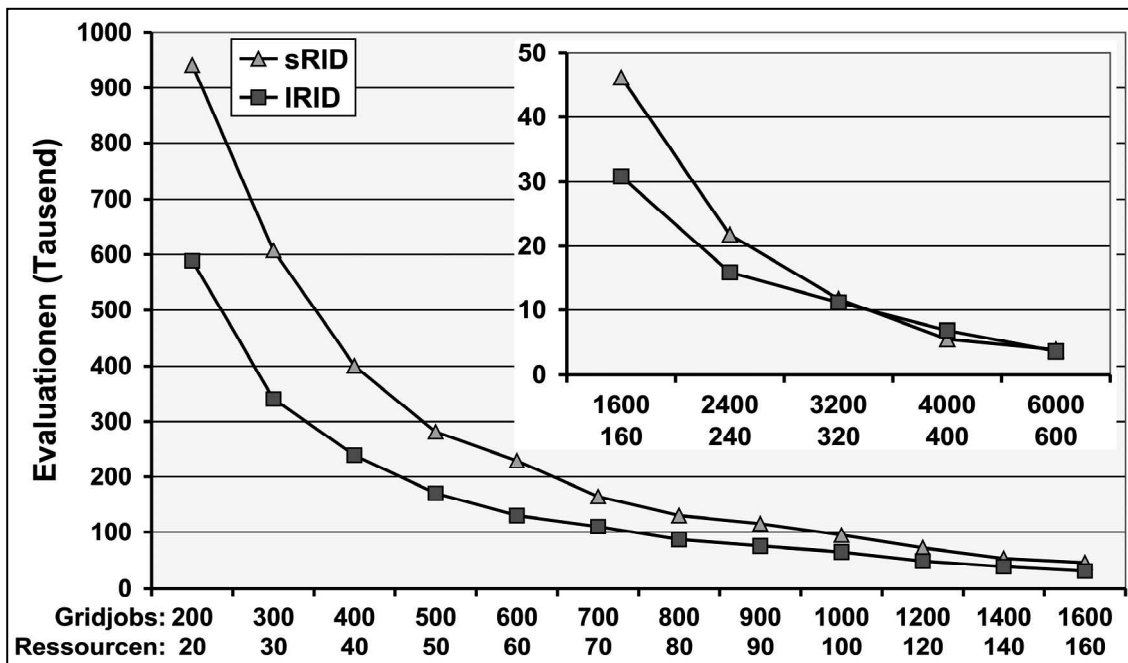


Bild 3: Innerhalb von drei Minuten bewertbare Schedules bei wachsender Last. Zur besseren Darstellung wird die Kurve ab 2400 Gridjobs getrennt mit angepasster Skalierung der Y-Achse dargestellt.

5 Zusammenfassung und Ausblick

Es konnte gezeigt werden, dass das Scheduling-Problem von Gridjobs unter realistischen Voraussetzungen komplexer ist als das klassische NP-vollständige Job-Shop-Scheduling Problem. Beim Gridjob-Scheduling kommen zur klassischen Aufgabenstellung noch alternative und heterogene Ressourcen, Ressourcen-Koallokation, Workflows mit paralleler Abarbeitung und eine multikriterielle Bewertung hinzu. Letzteres hat es bisher verhindert, ein lokales Suchverfahren zu finden, das erfolgreich als Meme in die evolutionäre Suche integriert werden kann [19, 20]. Das liegt daran, dass der Effekt kleiner Änderungen, wie z.B. das Vorziehen eines Gridjob-Gens oder die Wahl einer anderen alternativen Ressource nicht abgeschätzt werden kann, ohne die Belegungsmatrix zumindest bis zu der vorgenommenen Änderung aufzubauen. Bei anderen kombina-

torischen Aufgabenstellungen, wie z.B. dem „Problem des Handlungsreisenden“ ist es dagegen leicht zu entscheiden, ob eine Teilstrecke durch eine Veränderung kürzer wird oder nicht.

Das untersuchte Problem des Reschedulings anlässlich neuer Gridjobs nach einem relativ geringen Grad der Abarbeitung des aktuellen Schedules ist einer der Standardfälle beim Scheduling und Ressourcenmanagement im Grid. Neben dem Ressourcenausfall dürfte er auch einer der schwierigeren sein. Denn beim Hinzukommen neuer Ressourcen oder beim Jobabbruch verringert sich die Last gegenüber dem ursprünglichen Schedule. Für den Fall neuer Gridjobs wurden neue Heuristiken vorgestellt, die die im alten Plan enthaltene Information ausnutzen. Es konnte gezeigt werden, dass sie für den häufigeren Fall geringerer Änderungen von bis zu 20% abgearbeiteten und neuen Gridjobs das Ergebnis der heuristischen Planungsphase in der Regel deutlich verbessern.

Die verarbeitbare Last wurde für den Fall von 10% abgearbeiteten und neuen Gridjobs bei steigender Gridjob- und Ressourcenanzahl untersucht. Bis zu einer Last von 6000 Gridjobs und 600 Ressourcen konnten mit Hilfe des EA-Laufs Verbesserungen gegenüber der heuristischen Planung erzielt werden, auch wenn diese gegen Ende hin immer geringer ausfielen. Da bei der maximalen Last nur noch etwa zwei Dutzend Generationen bei einer kleinen Population von 20 oder 30 Individuen innerhalb der gewählten drei Minuten möglich war, kann die erfolgreiche Bearbeitung größerer Lasten mit der gegebenen Soft- und Hardwarekonfiguration (ein Prozessor eines AMD Athlon 64 4400+, 2.0 GHz) nicht erwartet werden. Ein Tuning der ersten prototypischen Implementierung oder schnellere Hardware könnte hier Abhilfe schaffen oder bei gegebener Last und Zeit bessere Ergebnisse liefern. Letzteres ist deshalb zu erwarten, da der EA ab etwa 200 Gridjobs bei weitem noch nicht in die Konvergenzphase gelangt.

Aktuelle Arbeiten haben die Untersuchung weiterer Heuristiken und Schedulingverfahren zum Gegenstand. Dies soll auch die Grundlage für ein adaptives GORBA liefern, dass entsprechend dem aktuellen Gridzustand geeignete Heuristiken und Optimierungsverfahren auswählt und durch Erfolg und Misserfolg lernt.

6 Literatur

- [1] Foster, I., Kesselman, C.: *The Anatomy of the Grid: Enabling Scalable Virtual Organisations*. Int. J. of Supercomputer Applications, vol. 15(3), 2001, S.200-222.
- [2] Brucker, P.: *Scheduling Algorithms*. Springer, Berlin Heidelberg, 2004.
- [3] Brucker, P.: *Complex Scheduling*. Springer, Berlin Heidelberg, 2006.
- [4] Jakob, W., Hahnenkamp, B., Quinte, A., Stucky, K.-U., Süß, W.: *Schnelles Scheduling mit Hilfe eines hybriden Evolutionären Algorithmus*. Automatisierungstechnik (57)3, Oldenbourg-Verlag, 2009. DOI:10.1524/auto.2009.0759
- [5] Setamaa-Karkkainen, A., Miettinen, K., Vuori, J.: *Best Compromise Solution for a New Multiobjective Scheduling Problem*. Computers & Operations Research, vol. 33(8), 2006, S.2353-2368.
- [6] Wiecek, M., Hoheisel, A., Prodan, R.: *Taxonomy of the Multi-criteria Grid Workflow Scheduling Problem*. In: Talia, D., Yahyapour, R., Ziegler, W. (eds.): *Grid Middleware and Services - Challenges and Solutions*. Springer, New York, 2008, S.237-264. DOI:10.1007/978-0-387-78446-5_16
- [7] Tsiakkouri, E., Sakellariou, S., Dikaiakos, M.D.: *Scheduling Workflows with Budget Constraints*. In: Gorlatch, S., Danelutto, M. (eds.): *Conf. Proc. CoreGRID Workshop "Integrated Research in Grid Computing"*, 2005, S.347-357.

- [8] Y. J., Buyya, R.: *A Budget Constrained Scheduling of Workflow Applications on Utility Grids using Genetic Algorithms*. In: Conf. Proc. 15th IEEE Int. Symp. on High Performance Distributed Computing (HPDC 2006), IEEE CS Press, Los Alamitos, 2006.
- [9] Kurowski, K., Nabrzyski, J., Oleksiak, A., Weglarz, J.: *Scheduling Jobs on the Grid - Multicriteria Approach*. Computational Methods in Science and Technology 12(2), Scientific Publishers OWN, Polen, 2006, S.123-138
- [10] Giffler, B., Thompson, G.L.: *Algorithms for Solving Production Scheduling Problems*. Operations Research 8, 1960, S.487-503.
- [11] Neumann, K., Morlock, M.: *Operations Research*. Carl Hanser, München, 2002.
- [12] Jakob, W., Quinte, A., Stucky, K.-U., Süß, W.: *Fast Multi-objective Scheduling of Jobs to Constrained Resources Using a Hybrid Evolutionary Algorithm*. In: Rudolph, G., et al. (eds.): PPSN X LNCS 5199, Springer, Berlin, 2008, S.1031-1040.
- [13] Jakob, W., Quinte, A., Stucky, K.-U., Süß, W., Blume, C.: *Schnelles Resource Constrained Project Scheduling mit dem Evolutionären Algorithmus GLEAM*. In: Mikut, R., Reischl, M. (Hrsg.): Conf. Proc. 17. Workshop Computational Intelligence, Universitätsverlag Karlsruhe, S.152-164, 2007.
- [14] Blume, C.: *GLEAM - A System for Intuitive Learning*. In: Schwefel, H.P., Männer, R. (eds.): Proc. of PPSN I, LNCS 496, Springer, Berlin, 1991, S.48-54
- [15] Blume, C., Jakob, W.: *GLEAM – An Evolutionary Algorithm for Planning and Control Based on Evolution Strategy*. In: Cantó-Paz, E. (ed.): GECCO 2002, vol. LBP, 2002, S.31-38.
- [16] Blume, C.: *GLEAM - Ein EA für Prozessabläufe am Beispiel von Steuerungen für Industrieroboter*. In: Mikut, R., Reischl, M. (Hrsg.): Conf. Proc. 16. Workshop Computational Intelligence, Universitätsverlag Karlsruhe, S.11-24, 2006.
- [17] Davis, L. (eds): *Handbook of Genetic Algorithms*. V. Nostrand Reinhold, New York, 1991.
- [18] Jakob, W.: *Towards an Adaptive Multimeme Algorithm for Parameter Optimisation Suiting the Engineers' Need*. In: Runarsson, T. P., et al. (eds.): PPSN IX LNCS 4193, Springer, Berlin, 2006, S.132-141.
- [19] Hahnenkamp, B.: *Integration anwendungsneutraler lokaler Suchverfahren in den adaptiven memetischen Algorithmus HyGLEAM für komplexe Reihenfolgeoptimierung*. Diplomarbeit, Uni Karlsruhe, Fak. f. Wirtschaftswissenschaften, AIFB, 2007
- [20] Sonnleithner, D.: *Integration eines Giffler-Thompson-Schedulers in GORBA*. Studienarbeit, Uni Karlsruhe, Fak. f. Maschinenbau, AIA, 2008

Advantages of Hierarchical versus Flat Model Structures for High-Dimensional Mappings

Benjamin Hartmann, Oliver Nelles

Institut für Mechanik und Regelungstechnik - Mechatronik
Universität Siegen
Paul-Bonatz-Str. 9-11
Tel.: (0271) 740-2271
Fax: (0271) 740-2365
E-Mail: benjamin.hartmann@uni-siegen.de

1 Introduction

In all model-based applications, a strong trend for ever increasing complexity can be observed in order to meet the more sophisticated market demand. Additionally, higher model accuracies are required to fulfill higher performance goals and meet stricter regulations. Both trends call for models with more and more input variables, i.e., higher-dimensional models.

Many modeling schemes severely suffer from the curse of dimensionality [1], that is a strong increase in modeling effort with a growing number of inputs. Recently very popular modeling approaches based on local model networks (LMN) generated with axes-orthogonal incremental construction algorithms also perform suboptimal for high-dimensional input spaces [2].¹ In [3] an axes-oblique partitioning strategy was proposed to overcome these weaknesses of the axes-orthogonal strategies.

The local model network is hierarchically organized and well suited for high-dimensional input spaces. But one difficulty in application of this algorithm is the a-priori fixation of the local model overlaps, because the resulting hierarchical model structure is not known beforehand. To overcome this drawback in this paper an automatic adjustment of the transitions between local models is introduced.

The article is organized as follows. At first in Sect. 2 the basics of local model networks are introduced. Section 3 analyzes the influence of the covariance matrix used in the validity functions and side effects of normalization in local model networks with Gaussian basis functions. In Sect. 4 a strategy of hierarchical modeling is presented. The comparison of a flat and a hierarchical model structure is given in Sect. 5. Besides the discussion of the key features of global estimation, local estimation and interpolation smoothness in local model networks, Sect. 6 introduces a new algorithm for the adjustment of the transitions between local models in hierarchical model networks. The performance of the hierarchical model strategy in combination with an automatic smoothness adjustment is evaluated with two examples in Sect. 7. This paper ends by summarizing the important conclusions.

¹For example, nonlinearities along multi-dimensional directions can only be modeled adequately with axes-orthogonal partitioning by generating a huge amount of local models.

2 Local model networks

The output \hat{y} of a local model network with p inputs $\underline{u} = [u_1 \ u_2 \ \dots \ u_p]^T$ can be calculated as the interpolation of M local model outputs \hat{y}_i , $i = 1, \dots, M$,

$$\hat{y} = \sum_{i=1}^M \hat{y}_i(\underline{u}) \Phi_i(\underline{u}) \quad (1)$$

where the $\Phi_i(\cdot)$ are called interpolation or validity or weighting functions [4]. These validity functions describe the regions where the local models are valid; they describe the contribution of each local model to the output. From the fuzzy logic point of view (1) realizes a set of M fuzzy rules where the $\Phi_i(\cdot)$ represent the rule premises and the \hat{y}_i are the associated rule consequents. Because a smooth transition (no switching) between the local models is desired here, the validity functions are smooth functions between 0 and 1. For a reasonable interpretation of local model networks it is furthermore necessary that the validity functions form a *partition of unity*:

$$\sum_{i=1}^M \Phi_i(\underline{u}) = 1. \quad (2)$$

Thus, everywhere in the input space the contributions of all local models sum up to 100%.

In principle, the local models can be chosen of arbitrary type. If their parameters shall be estimated from data, however, it is extremely beneficial to choose a linearly parameterized model class. The most common choice are polynomials. Polynomials of degree 0 (constants) yield a neuro-fuzzy system with singletons or a normalized radial basis function network. Polynomials of degree 1 (linear) yield local linear model structures, which is by far the most popular choice. As the degree of the polynomials increases, the number of local models required for a certain accuracy decreases. Thus, by increasing the local models' complexity, at some point a polynomial of high degree with just one local model ($M = 1$) is obtained, which is in fact equivalent with a global polynomial model ($\Phi_1(\cdot) = 1$).

Besides the possibilities of transferring parts of mature linear theory to the nonlinear world, local *linear* models seem to represent a good trade-off between the required number of local models and the complexity of the local models themselves. Due to the overwhelming importance and for simplicity of notation the rest of this paper will deal only with local models of linear type:

$$\hat{y}_i(\underline{u}) = w_{i,0} + w_{i,1}u_1 + w_{i,2}u_2 + \dots + w_{i,p}u_p. \quad (3)$$

However, an extension to polynomials of higher degree or other linearly parameterized model classes is straightforward.

3 Flat model structure

This section gives a theoretical investigation about axes-orthogonal and axes-oblique input space partitioning strategies in local model networks with Gaussian basis functions. One key feature of this network type is its flat structure, i.e., all validity functions can be computed in parallel. Side effects of normalization in local model networks are analyzed.

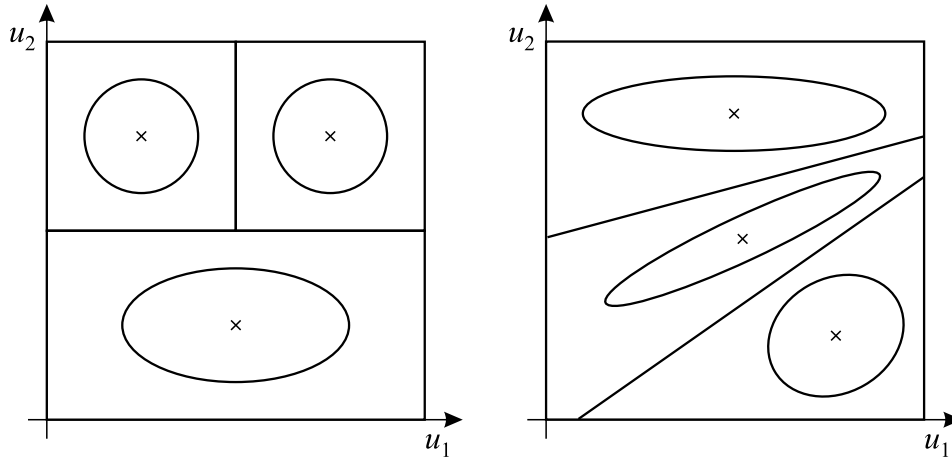


Figure 1: The circles and ellipses represent the contour lines of the multidimensional membership functions μ_i (i.e., before normalization) in case of axes-orthogonal (left) and axes-oblique partitioning (right).

3.1 Influence of the covariance matrix

To construct a Gaussian basis function the distance x_i from a data point to each center \underline{c}_i is calculated with the help of the covariance matrix $\underline{\Sigma}_i$, which scales and rotates the axes:

$$x_i = \|\underline{u} - \underline{c}_i\|_{\underline{\Sigma}_i} = \sqrt{(\underline{u} - \underline{c}_i)^T \underline{\Sigma}_i^{-1} (\underline{u} - \underline{c}_i)}. \quad (4)$$

The membership functions $\mu_i(\cdot)$ of a Gaussian basis function network are given by:

$$\mu_i(\underline{u}) = \exp\left(-\frac{1}{2}\|\underline{u} - \underline{c}_i\|_{\underline{\Sigma}_i}^2\right). \quad (5)$$

To achieve a *partition of unity* the membership functions have to be normalized to obtain the validity functions:

$$\Phi_i(\underline{u}) = \frac{\mu_i(\underline{u})}{\sum_{j=1}^M \mu_j(\underline{u})}. \quad (6)$$

Depending on the covariance matrix two cases for partitioning the input space can be distinguished, see Fig. 1:

1. *Axes-orthogonal partitioning*: If the input space is divided into rectangular regions by axes-orthogonal splits the covariance matrix becomes diagonal:

$$\underline{\Sigma}_i = \begin{bmatrix} \sigma_{i,1}^2 & 0 & \cdots & 0 \\ 0 & \sigma_{i,2}^2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \sigma_{i,p}^2 \end{bmatrix}, \quad (7)$$

where $\sigma_{i,1}^2, \dots, \sigma_{i,p}^2$ are the variances of each input direction. In case of univariate membership functions the tensor product

$$\mu(u_1) \cdot \mu(u_2) \cdot \dots \cdot \mu(u_p) = \mu(u_1, u_2, \dots, u_p) \quad (8)$$

is utilized as construction mechanism. The multidimensional membership function realized by each neuron can be described with p center values $c_{i,j}$ and p variances $\sigma_{i,j}^2$. One big advantage of axes-orthogonal partitioning is its easy interpretability in terms of fuzzy logic. This way of partitioning allows a projection of the validity regions to the one-dimensional input variables. Undesirable normalization side effects and extrapolation behavior can be improved compared to clustering or data-based strategies [5, 4]. Common approaches are e.g. CART [6] and LOLIMOT [7], see also [8, 9].

2. *Axes-oblique partitioning*: In case of an axes-oblique partitioning strategy that is used for e.g. Gustafson-Kessel [10] or Gath-Geva clustering [11] the covariance matrix becomes symmetric. A fuzzy interpretation of the model is not really possible, because the projection to the input axes is not possible without loss of information.

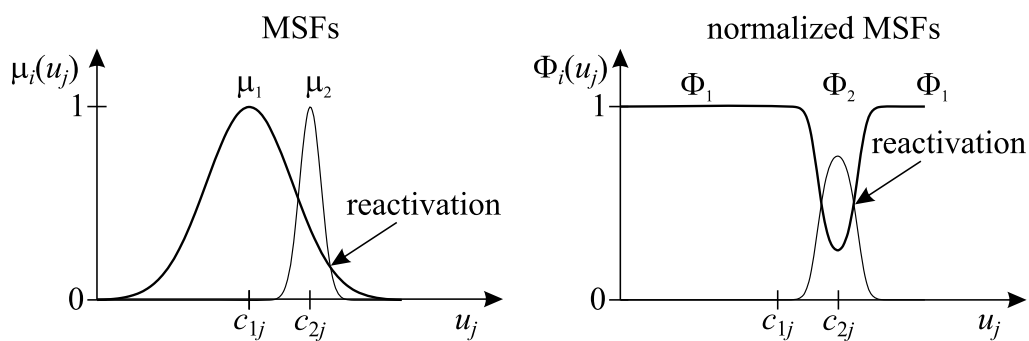


Figure 2: Reactivation of basis functions occurs if the activation of the Gaussian BF with the largest standard deviation becomes higher than the activation of all other Gaussian BFs.

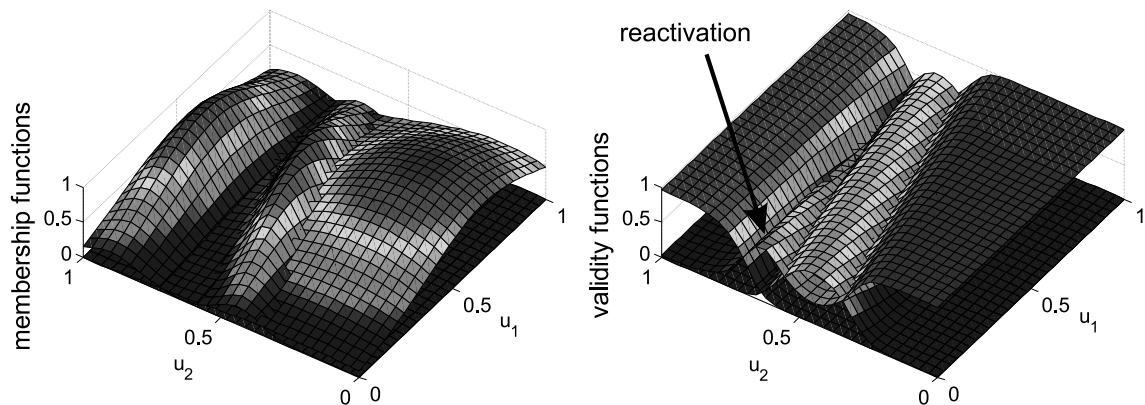


Figure 3: Normalization of axes-oblique membership functions leads to unexpected reactivation regions in the input space. Projection to the one-dimensional input variables delivers no adequate information about these side effects.

3.2 Side effects of normalization

The normalization (6) can lead to some very unexpected and usually undesirable effects, which are illustrated in [5]. In case of a diagonal covariance matrix these effects do *not*

occur if all basis functions (BFs) possess identical standard deviations for each dimension, i.e.,

$$\sigma_{1j} = \sigma_{2j} = \dots = \sigma_{Mj}. \quad (9)$$

The side effects are caused by the fact that for all very large input values the activation of the Gaussian BF with the largest standard deviation becomes higher than the activation of all other Gaussian BFs. As shown in Fig. 2, the normalization then results in a reactivation of the basis function with the largest width. This reactivation makes the validity functions non-local and multi-modal – both are properties usually not intuitively assumed and not desired in such networks.

These problems translate to the multidimensional case. If axes-orthogonal partitioning is used, these side effects can always be analyzed by examining the one-dimensional projections to the input variables. However, in case of scaled and rotated Gaussian BFs (axis-oblique partitioning) the normalization can lead to highly nontransparent reactivation regions in the input space, see Fig. 3. These side effects are typically not very significant for the performance of a Gaussian BF network, but they are of fundamental importance with regard to its interpretation.

The next section introduces a hierarchical model structure that can circumvent the above mentioned drawbacks of the flat structure.

4 Hierarchical model structure

Almost all of the common partitioning strategies yield flat models. Even if the *algorithm* is hierarchically organized like e.g. LOLIMOT, the constructed *model* itself is flat in the sense that all validity functions $\Phi_i(\cdot)$ can be calculated in parallel. This is an important feature if the network really should be realized in hardware or by some parallel computer. For any standard software implementation a hierarchical model structure as shown in Fig. 4 is not disadvantage. Strictly speaking hierarchical model networks do not belong to the group of neural networks because the parallelization of neurons is not applicable. But in case of sequential computer architectures hierarchical model approaches can be favorably. Truly hierarchical model structures are e.g. pursued with CART [6], MARS [12], hinging hyperplane trees [2], the hierarchical local model network [13] and the axes-oblique partitioning algorithm proposed in [3].

As an example, a model with four rules shall be constructed in a hierarchical manner. The model is represented by a binary tree as shown in Fig. 4. At each knot i of the tree the input space is softly partitioned into two areas by two splitting functions $\Psi_i(\cdot)$ and $\tilde{\Psi}_i(\cdot)$ which sum up to one:

$$\Psi_i(\underline{u}) + \tilde{\Psi}_i(\underline{u}) = 1. \quad (10)$$

These regions are further subdivided by succeeding knots (if they exist). Each leaf of the tree realizes a local model and its contribution to the overall model output is given by the multiplication of all splitting functions from the root of the tree to the corresponding leaf. For the tree with seven knots and four leaves in Fig. 4 this means:

$$\begin{aligned} \Phi_3(\underline{u}) &= \tilde{\Psi}_1(\underline{u}), & \Phi_4(\underline{u}) &= \Psi_1(\underline{u})\Psi_2(\underline{u}), \\ \Phi_6(\underline{u}) &= \Psi_1(\underline{u})\tilde{\Psi}_2(\underline{u})\Psi_5(\underline{u}), & \Phi_7(\underline{u}) &= \Psi_1(\underline{u})\tilde{\Psi}_2(\underline{u})\tilde{\Psi}_5(\underline{u}). \end{aligned}$$

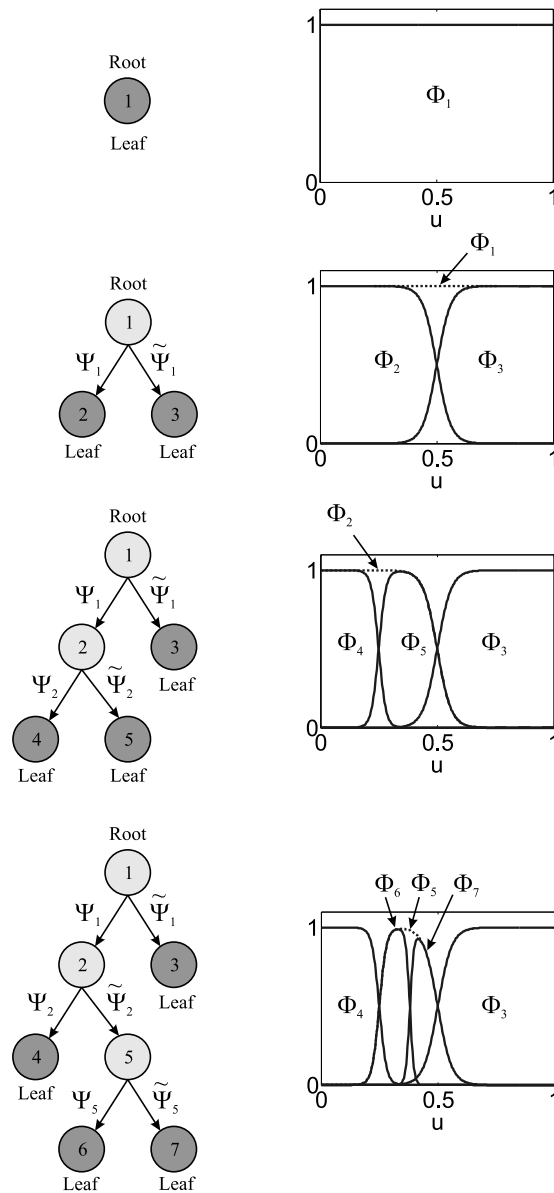


Figure 4: Hierarchical tree construction algorithm. Every leaf of the tree represents one validity function Φ_i and is generated by multiplying the corresponding splitting functions Ψ_i .

Once these validity functions are determined, the overall model output is given similarly to (1) as the sum the local models weighted with their associated validity functions:

$$\hat{y} = \sum_{i \in \mathcal{L}} \hat{y}_i(\underline{u}) \Phi_i(\underline{u}) \quad (11)$$

where \mathcal{L} is the set of indices of the leaf knots, which is in the above example $\mathcal{L} = \{3, 4, 6, 7\}$.

5 Comparison of flat and hierarchical model structure

This section compares the main features of the flat and the hierarchical model structures. The advantages and drawbacks of each approach are described and the benefits of the hierarchical structure are explained.

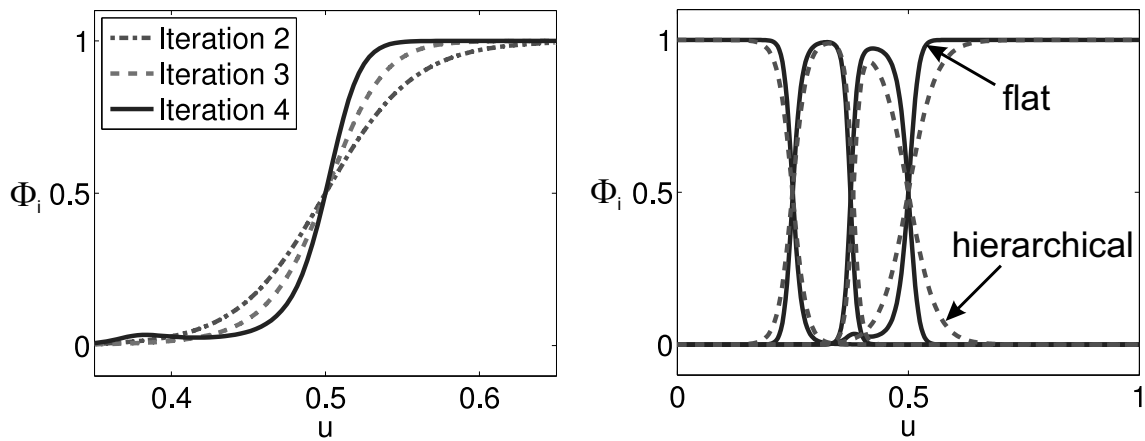


Figure 5: Left: In the case of a flat structure the most right validity function changes its shape in each iteration although the splits are taken in different regions. Right: Comparison of the partitioning with a flat and a hierarchical model structure.

A key feature of the flat model structure is that the validity functions can be calculated in parallel. This enables fast computation times, if parallel computers are used. Furthermore the interpretability of the model is high in terms of fuzzy logic, if axes-orthogonal partitioning is applied. The Gaussian membership functions can be projected to the input axes and a rule-based interpretation is possible. The partitioning generated with a flat model structure is independent of the splitting order.

However, there are also some disadvantages of the flat model structure. Firstly, each split of the input space has an effect on all existing validity functions. That means that strongly spoken all local models must be re-estimated after each split of the model. Figure 5 (left) illustrates this effect. After each iteration of the partitioning algorithm the validity function changes its shape, although the split has been taken in a different input space region. Secondly, due to the normalization of the membership functions the flat structure suffers from the side effects of normalization that are described in Section 3.2, especially in the case of axes-oblique partitioning. Another drawback is that for axes-oblique splits full symmetric covariance matrices are required. Therefore, many parameters (variances and covariances) have to be estimated, if the partitioning is not axes-orthogonal.

The advantages of the hierarchical model structure are the following: The hierarchical approach allows a partitioning without normalization. The partition of unity in (2) holds automatically if only (10) is fulfilled. That means that all addressed undesirable effects of normalization are avoided (comp. Sect. 3.2), because the normalization in (6) is not needed to create a partition of unity [5, 4]. Furthermore the computation times are decreased and numerical problems are circumvented, because the normalization has not to be calculated. Sigmoidal functions fulfill all necessary properties for splitting functions and are therefore well suited for the axes-oblique approach. The number of parameters is decreased in comparison to normalized Gaussian validity functions. The hierarchical validity functions are guaranteed to possess the properties of locality and approximately compact support. That means that whole model subtrees can be changed or adapted without causing any effect on other tree regions.

The remedy of the above mentioned disadvantages of flat model structures goes along with new challenges in terms of the hierarchical model structure. Firstly, a parallel computation of the model is not possible due to the hierarchical tree approach. Secondly, the

partitioning is affected by the order of splitting. A different partitioning is obtained, if the order of splitting changes. Lastly, there is a issue that concerns the adjustment of the overlaps between local models. This is discussed more detailed in section 6.

The differences between the flat and the hierarchical partition are illustrated in Fig. 5 (right). The smoothness was fixed in a way that the smoothnesses of the first splits are equal in both, the hierarchical and the flat case. The comparison of the model structures and the positive experiences of the authors with the hierarchical approach show that, especially in the framework of high-dimensional modeling and axes-oblique partitioning, the application of the hierarchical structure is almost indispensable.

6 Smoothness adjustment

The goal of approximation with local model networks is i) the interpretability of local models and ii) a smooth global model behaviour. The necessity of adjustment of overlaps between local models depends on the used parameter estimation approach. Two different approaches for optimization of the local model parameters can be distinguished: *global* and *local* estimation. While global estimation represents the straightforward application of the least squares algorithm, local estimation neglects the overlap between the validity functions in order to exploit the local features of the model [14], [13].

6.1 Global and local estimation:

In case of global estimation the following characteristics can be listed:

- High training effort.
- High flexibility in case of large overlaps of validity functions, but then interpretability of local models is lost.
- High accuracy if data has low noise level.
- In case of hard switching between local models global and local estimation tend to the same results.

Local estimation seems to be advantageous to global estimation in most applications [4, 15]. The following benefits can be expected:

- Fast training.
- Regularization effect.
- Good interpretability of local models.
- Advantages when applied in a recursive algorithm for online learning.

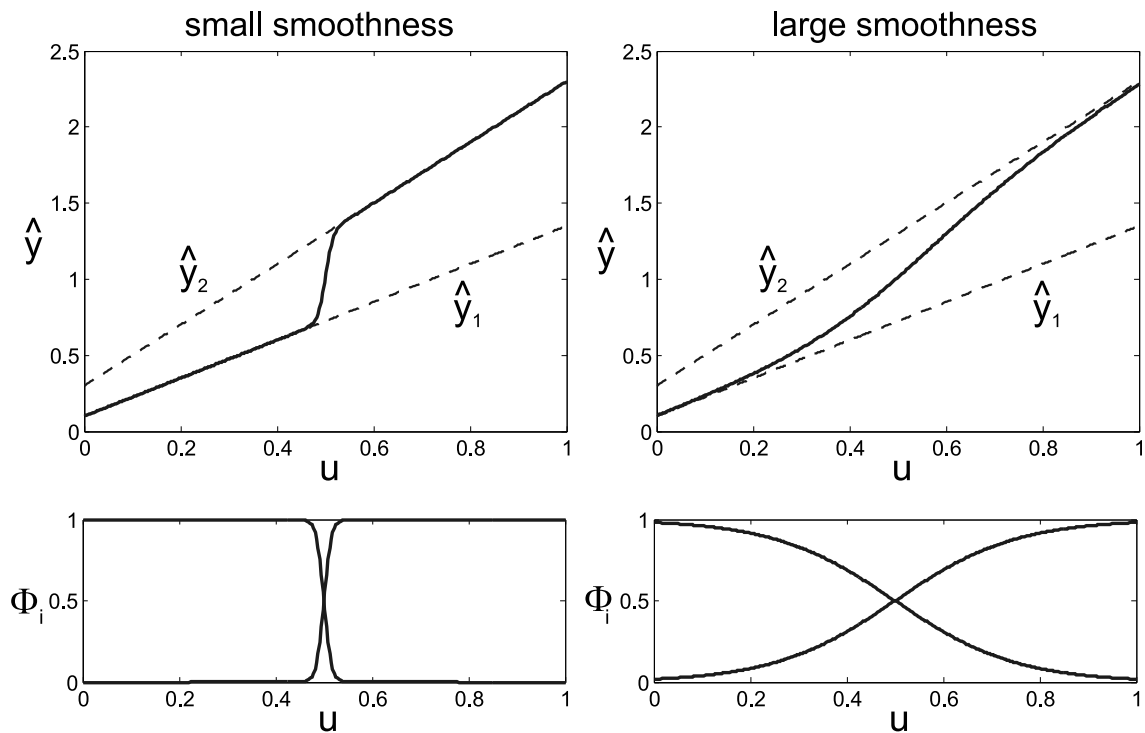


Figure 6: Influence of interpolation smoothness on the model and relation between model output \hat{y} and validity function Φ_i .

6.2 How smooth should the interpolation be?

For the choice of the interpolation smoothness it is interesting to consider two special cases:

1. *smoothness too small*: hard switches between local models.
2. *smoothness too large*: big overlaps of validity functions and loss of locality.

Hence the smoothness should be a compromise between these cases. In Fig. 6 it is depicted how the interpolation smoothness influences the validity functions Φ_i and the transition between two local linear models \hat{y}_1 and \hat{y}_2 . In most cases it seems to be sufficient to roughly choose some reasonable *a-priori* smoothness value for all local models. The activation of all validity functions should reach almost 100% in order to ensure a proper interpretation of the associated local models.

6.3 Smoothness optimization or a-priori fixation?

With local estimation an optimization of the smoothness does not work satisfactorily in most cases. The reason is that a hard switching between local models also minimizes the error caused by neglecting the overlap between the validity functions. However, even if optimal performance is achieved such small overlaps should not be realized in practice where the model is usually required to be smooth. In contrast with a global estimation approach a smoothness optimization tends to yield very large smoothness values ($\rightarrow \infty$), in order to overcome the restrictions in model flexibility due to the locality of the validity

functions. Considering the low sensitivity of the model quality on the smoothness and all the difficulties arising during nonlinear optimization of the smoothness it can be generally recommended to fix it a-priori. But in case of networks with hierarchical structure an additional unpleasant effect occurs: The overlap of validity functions depends on the hierarchy level of the local models. Since the splitting functions Ψ_i have to be multiplied with each other to generate the validity functions Φ_i , the activity of subsequent neurons is reduced with each hierarchy level. For example, the validity functions associated with knot 6 and 7 in Fig. 4 are always less active than the validity function associated to their mother knot 5. A reliable fixation of the smoothness a-priori is only possible if also the final depth (or hierarchy level) of the hierarchical model structure would be known. To circumvent this drawback a new algorithm has been developed where the smoothness is adapted automatically after adding a local model to the hierarchical tree structure.

6.4 Smoothness of sigmoid splitting functions:

The in [3] proposed axes-oblique partitioning algorithm uses sigmoid splitting functions:

$$\Psi_i(\underline{u}) = \frac{1}{1 + e^{-\kappa_i(v_{i,0} + v_{i,1}u_1 + \dots + v_{i,p}u_p)}}. \quad (12)$$

Because of the symmetric properties of $\Psi_i(\cdot)$, its counterpart can be written as:

$$\tilde{\Psi}_i(\underline{u}) = 1 - \Psi_i(\underline{u}) = \Psi_i(-\underline{u}). \quad (13)$$

The vector $\underline{v}_i = [v_{i,1} \ v_{i,2} \ \dots \ v_{i,p}]^T$ determines the direction of the soft split, the offset term $v_{i,0}$ determines the position of the split (its distance from the origin), and the redundant parameter κ_i determines the smoothness of the split. The smoothness information is already contained in the size of $|\underline{v}_i|$ and strictly speaking κ_i is not needed. However, it makes the split optimization easier if the κ_i parameter is explicitly introduced and the direction vector \underline{v} is constrained to be of unit length in order to remove the redundancy. This can be done by performing the following operation on (12): $v_{i,j} \leftarrow v_{i,j}/|\underline{v}|$ for $j = 0, 1, \dots, p$.

The smoothness has to be chosen heuristically. The key idea is to select a smoothness which is roughly proportional to the size of the validity regions. Thus between large regions the interpolation is performed more smoothly than between small regions. This gives the algorithm an adaptive resolution feature that enables it to cover arbitrarily tiny details in the process behavior. The centers of the validity regions are approximated by the mean value of the data samples weighted with their validity function value

$$\underline{c}_i = \frac{1}{N} \sum_{k=1}^N \Phi_i(\underline{u}(k)) \cdot \underline{u}(k), \quad (14)$$

where N is the number of data samples. Then the κ_i is chosen as

$$\kappa_j = \frac{1}{\sigma |\underline{c}_j - \underline{c}_i|}, \quad \sigma > 0, \quad (15)$$

where j is the neighbor of knot i in the tree and σ is the smoothness factor for the overall model.

6.5 Automatic transition adjustment algorithm:

For each local model split the smoothness of the overall model is adjusted. The following optimization problem can be defined:

$$J = |\Phi_{\text{threshold}} - \min(\Phi_i(\mathcal{C}_i), \Phi_j(\mathcal{C}_j))| \rightarrow \min_{\sigma} \quad (16)$$

where $\Phi_{\text{threshold}}$ is an user defined value that gives a threshold for the minimal activation of all validity functions. The validity functions Φ_i and Φ_j are generated by division of an existing local model one level up in hierarchy. As long as J is larger than a minimum accuracy ε the transitions between all local models has to be sharpened by iteratively decreasing the smoothness factor σ .

Depending on the underlying process $\Phi_{\text{threshold}}$ has to be chosen by the user. For example, if the process possesses high gradients and the training data is not significantly disturbed this threshold should be chosen high, i.e., close to 1. On the other hand, the threshold has to be chosen low if the process is highly disturbed and little knowledge about the process behaviour is available. Experiments showed that the choice $\Phi_{\text{threshold}} = 0.9$ delivers reasonable results in most applications.

After manipulation of the model smoothness actually the parameters of all local models have to be re-estimated consecutively. But since the smoothness influences the model parameters only insignificantly ([4]) the additional re-estimation procedure can be skipped in most applications. Experimental results showed a relative parameter difference of less than 0.1%. The advantages of this smoothness adjustment approach are shown with demonstration examples in the following section.

7 Demonstration examples

The following static SISO process will be modeled:

$$y = \frac{1}{(u - 2)^2 + 0.1}, \quad (17)$$

with $0 \leq u \leq 5$. To model this function accurately a locally high resolution in the region $u \approx 2$ is required and thus it is a kind of worst case example for the hierarchical structure identification algorithm using a-priori fixed smoothness. In Fig. 7 both the model with a-priori fixed smoothness and with automatic smoothness adjustment are illustrated. The convergence plot shows that the models of both methods yield to nearly the same normalized root mean squared error in case of 5 local models. But if the number of local models is increased the model with a-priori fixed smoothness becomes too smooth. The refinement of the model around $u \approx 2$ achieves no improvement. In other words, the local model network loses its universal approximation capability. This unsatisfactory model behavior is a consequence of the too small validity function values. They directly result from the multiplications of the splitting functions due to the hierarchical model structure that is not known beforehand. The here proposed smoothness adjustment algorithm automatically adapts the smoothness of the overall model. As in the convergence plot illustrated, the model with automatic smoothness adjustment can be improved significantly with each iteration in contrast to the a-priori fixed smoothness approach. An application example is

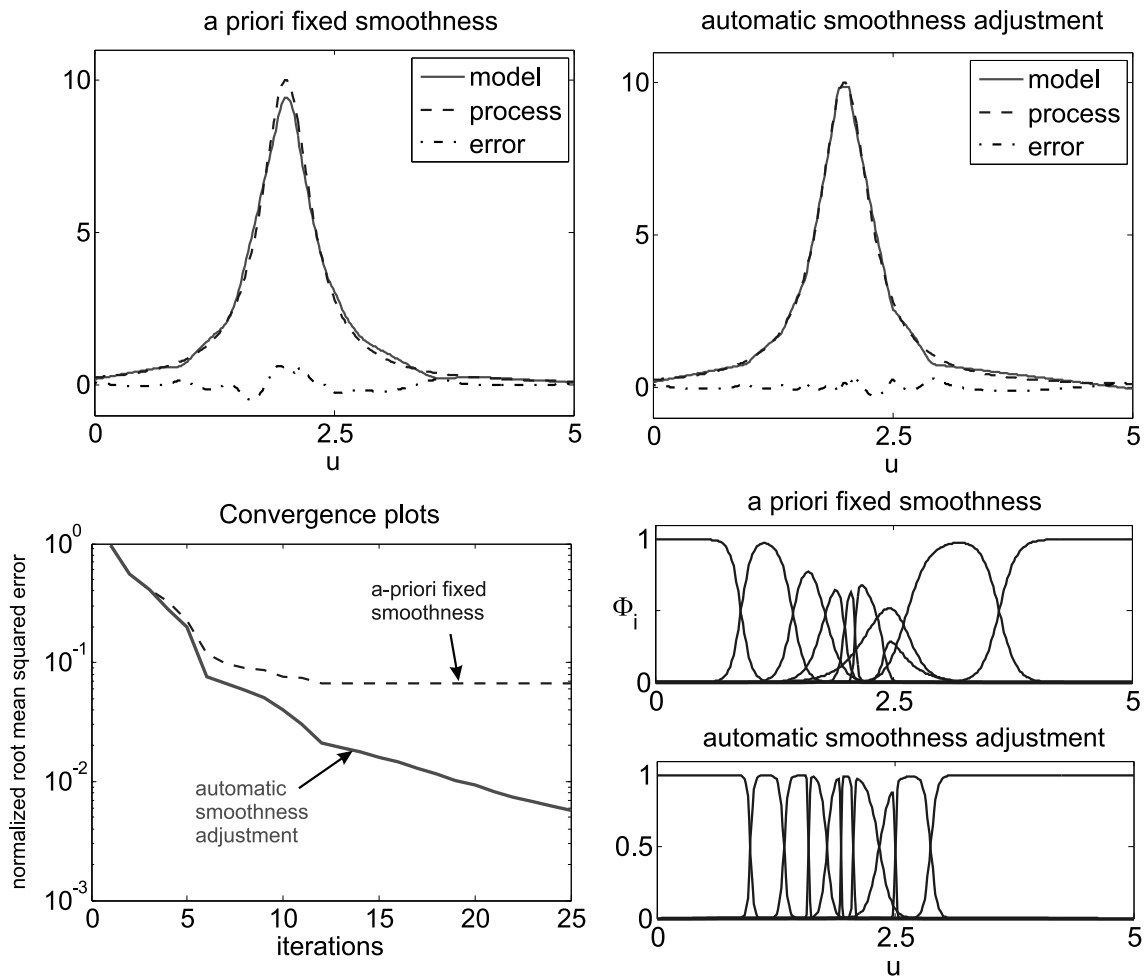


Figure 7: Modeling of a test-process with a-priori fixed smoothness (top left) and automatic transition adjustment algorithm (top right). The convergence plots (bottom left) and the corresponding validity functions (bottom right) show the advantages of automatic smoothness adjustment.

the modeling of an engine characteristic map with and without the proposed algorithm, see Fig. 8. The inputs are the engine speed in *rpm* and the injection mass in *mg*. The map output is the engine torque in *Nm*. The training data consists of 256 input-output pairs. The partitioning algorithm was utilized with and without automatic transition adjustment.² Table 1 shows the number of models, the training error (NRMSE) and the computation time for the engine map model with a-priori fixed smoothness on the one hand and with automatic smoothness adjustment on the other hand. The hierarchical local model network achieves with the new algorithm improved results while the computation times are lower both under specification of a maximum training error of 5% and under specification of the maximum number of local models (here: 15LMs).

8 Conclusions

In this article local model networks with flat model structures are compared with hierarchical model structures. One advantage of hierarchical approaches is that they are not affected by undesirable side effects due to normalization. Eminently in the framework of

²Computed with Intel Core 2 processor, 1.83 GHz, 2 GB RAM

Table 1: Results of engine map modeling

Smoothness Adjustment	#LM	NRMSE	CPU time [s]
off	17	0.0500	7.03
on	14	0.0500	5.01
off	15	0.0517	6.13
on	15	0.0470	5.36

axes-oblique partitioning the usage of a hierarchical organized model structure becomes advantageous. However, the attribute of an universal approximator gets lost if the overlaps of the local models are fixed a-priori. The proposed new algorithm for automatic smoothness adjustment between local models can overcome this drawback.

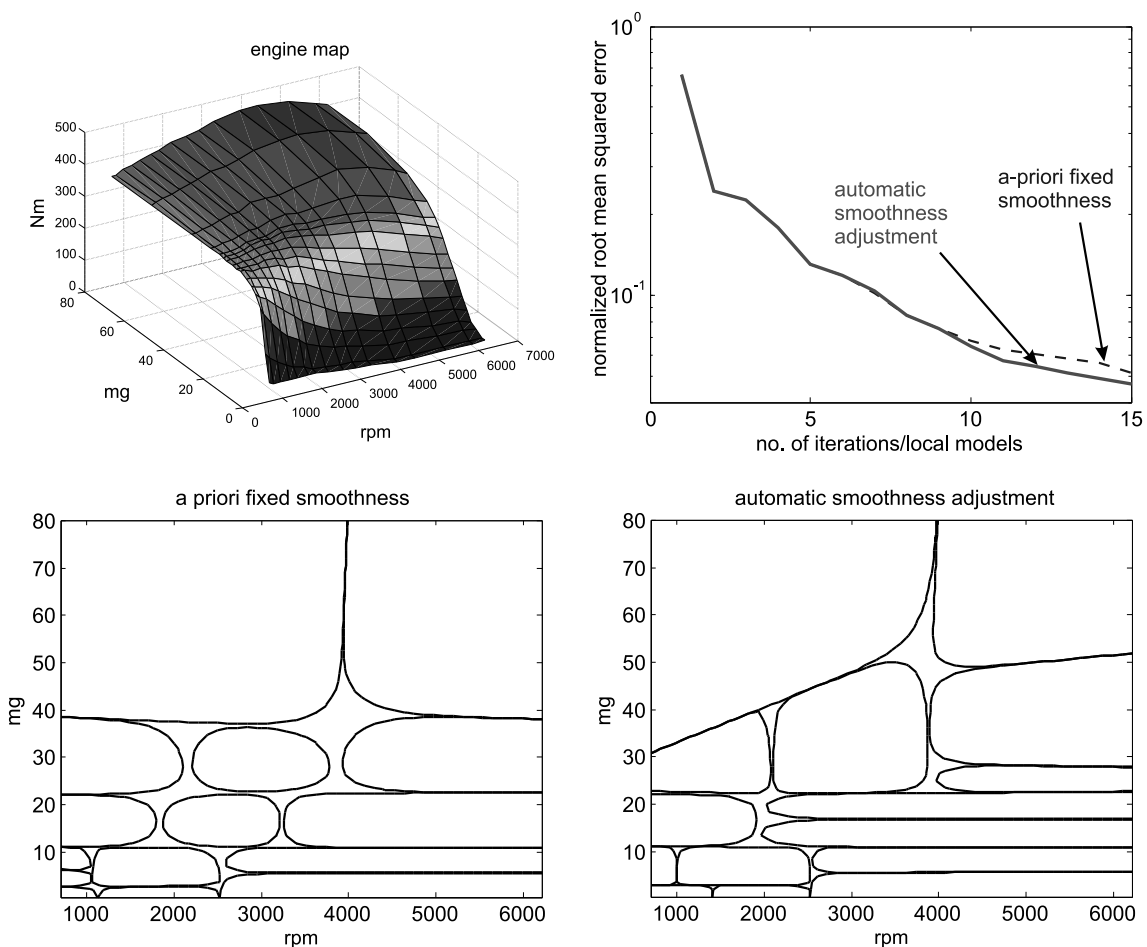


Figure 8: Application of a hierarchical local model network on an engine characteristic map (top left). The convergence plots (top right) and the contour plots of the partitioning (bottom left/right) illustrate the improvement of the model by the new transition adjustment algorithm.

Literatur

- [1] Bellman, R.: *Adaptive Control Processes*. New Jersey: Princeton University Press. 1961.
- [2] Ernst, S.: Hinging Hyperplane Trees for Approximation and Identification. In: *IEEE Conference on Decision and Control (CDC)*, S. 1261–1277. Tampa, USA. 1998.
- [3] Nelles, O.: Axes-Oblique Partitioning Strategies for Local Model Networks. In: *International Symposium on Intelligent Control (ISIC)*. Munich, Germany. 2006.
- [4] Nelles, O.: *Nonlinear System Identification*. Berlin, Germany: Springer. 2001.
- [5] Shorten, R.; Murray-Smith, R.: Side-Effects of Normalising Basis Functions in Local Model Networks. In: *Multiple Model Approaches to Modelling and Control* (Murray-Smith, R.; (Eds.), T. J., Hg.), Kap. 8, S. 211–229. Taylor & Francis, London. 1997.
- [6] Breiman, L.; J.H. Friedman R. Olshen R., C. S.: *Classification and Regression Trees*. New York: Chapman & Hall. 1984.
- [7] Nelles, O.; Sinsel, S.; Isermann, R.: Local Basis Function Networks for Identification of a Turbocharger. In: *IEE UKACC International Conference on Control*, S. 7–12. Exeter, UK. 1996.
- [8] Sugeno, M.; Kang, G.: Structure Identification of Fuzzy Model. *Fuzzy Sets & Systems* 28 (1988) 1, S. 15–33.
- [9] Johansen, T.: Identification of Non-linear System Structure and Parameters Using Regime Decomposition. *Automatica* 31 (1995) 2, S. 321–326.
- [10] Gustafson, D.; Kessel, W.: Fuzzy Clustering with a Fuzzy Covariance Matrix. In: *IEEE Conference and Decision and Control*, S. 761–766. San Diego, USA. 1979.
- [11] Gath, I.; Geva, A.: Unsupervised Optimal Fuzzy Clustering. *IEEE Trans. Pattern Analysis and Machine Intelligence* 11 (1989) 7, S. 773–781.
- [12] Friedman, J.: Multivariate Adaptive Regression Splines (with discussion). *The Annals of Statistics* 19 (1991) 1, S. 1–141.
- [13] Murray-Smith, R.: *A Local Model Network Approach to Nonlinear Modeling*. Dissertation, University of Strathclyde, Strathclyde, UK. 1994.
- [14] Cleveland, W.; Devlin, S.; Grosse, E.: Regression by Local Fitting: Methods, Properties, and Computational Algorithms. *Journal of Econometrics* 37 (1996), S. 87–114.
- [15] Murray-Smith, R.; Johansen, T.: Local learning in local model networks. In: *IEE International Conference on Artificial Neural Networks*, S. 40–46. 1995.

Genetic Programming Applied to Predictive Control in Environmental Engineering

**Oliver Flasch, Thomas Bartz-Beielstein,
Patrick Koch, and Wolfgang Konen**

Fakultät für Informatik und Ingenieurwissenschaften, Fachhochschule Köln
E-Mail: {oliver.flasch | thomas.bartz-beielstein |
patrick.koch | wolfgang.konen}@fh-koeln.de

Abstract

We introduce a new hybrid *Genetic Programming* (GP) based method for time-series prediction in predictive control applications. Our method combines existing state-of-the-art analytical models from predictive control with a modern typed graph GP system. The main idea is to pre-structure the GP search space with existing analytical models to improve prediction accuracy. We apply our method to a difficult predictive control problem from the water resource management industry, yielding an improved prediction accuracy, compared with both the best analytical model and with a modern GP method for time series prediction. Even if we focus this first study on predictive control, the automatic optimization of existing models through GP shows a great potential for broader application.

1 Introduction

Predictive control is a key technology for improving the efficiency of systems and processes in environmental engineering. It can efficiently reduce environmental pollution. Today, low priced sensors are available to measure data which describe the state of the system. These data can easily be stored on modern computer systems. One possible barrier which prevents a wide applicability of predictive control is the difficulty to rapidly develop and deploy the required software. Time series regression constitutes one core functionality of predictive control software. Although many different methods, ranging from classical statistical regression to modern computational statistics, can be used for time series regression [1], it is often not at all clear which method offers acceptable precision for a concrete regression problem.

Genetic Programming (GP) is unique among the methods of Evolutionary Computation (EC) in the sense that it can easily incorporate and recombine other techniques, especially time series regression methods. GP is able to automatically generate solution programs for arbitrary application problems in a very flexible manner [2]. The downside of this generality is a very large and unstructured search space. For many difficult real-world problems, the size and complexity of this search space quickly becomes unmanageable; our first experiments on applying GP on predictive control illustrate these difficulties (see subsection 3.3).

Bartz-Beielstein et al. [3] and Konen et al. [4] introduced an *analytic approach* to predictive control in environmental engineering. They developed an analytical regression model which was customized for this problem, the so-called INT2 model. Integrating this existing analytical approach into GP, we are able to *pre-structure* the search space

to allow for a much more effective evolutionary search. We are able to significantly improve on the results obtained by applying standard GP, as well as on the results obtained by applying existing analytical time series regression methods alone. The pre-structuring does not necessarily mean that we have to abandon the complete generality of the GP approach, because every GP algorithm configuration creates a certain bias in the evolutionary search [5]. By pre-structuring the search space with existing solutions, this bias is simply adjusted from randomly chosen to more promising directions.

We conduct a first case study to illustrate how our approach works in a real-world setting. This case study is devoted to the prediction of water levels in storm-water overflow tanks based on current rainfall data, in order to implement predictive control of water drain rate. Such predictions are of immense practical utility in preventing costly and damaging over- or under-loading of the sewage system connected to these storm-water overflow tanks. The task of predicting the current fill levels from the past rain data alone—not using past fill levels—is rather challenging since the hidden state of the surrounding soil influences the impact of rain in a nonlinear fashion. We pre-structure the GP search space with the INT2 model [3, 4]. The case study compares the prediction RMSE of our method with the prediction RMSE of standard GP and INT2.

Training and verification time series for this case study consist of more than 100,000 data records, ranging from April to July 2007. New data from 2008 and 2009 will be used in the near future of our ongoing research project and will validate the generalizability of our approach. These time series are real historical data, collected from storm-water overflow tanks in Germany.

This paper is structured as follows: Section 2 describes the GP approach and shows how we apply this approach to time series prediction. Section 3 presents and discusses first experimental results. Section 4 concludes this paper with a short summary of the results and presents directions for further research.

2 Optimization of existing analytical models through GP

GP is an EC method that automatically generates solutions from high-level problem definitions. GP proceeds by generating a population of computer programs and then refining them in an evolutionary process of variation and selection. To guide this evolutionary process, a solution quality measure, called a fitness function, has to be provided. This fitness function maps candidate solutions (called individuals, taking the form of computer programs) to numerical fitness values. In contrast to other Evolutionary Computation methods, like Evolutionary Systems (ES) or Genetic Algorithms (GA), the structure of the solution does not have to be known in advance. In contrast to solutions generated by sub-symbolic EC methods, like Artificial Neural Networks or Support Vector Machines (SVM), solutions generated by GP are symbolic expressions. This has the benefit of GP solutions being readable and understandable by humans. Furthermore, GP's symbolic solutions are accessible to formal methods of program construction and verification. First, this means that GP solutions are modular and scalable; they can easily be combined to form larger units. Second, the correctness of GP solutions can (at least in principle) be formally proved.

This flexibility of GP comes at a certain cost: The algorithm has an exceptionally large parameter space, containing not only typical EC parameters, like population size and

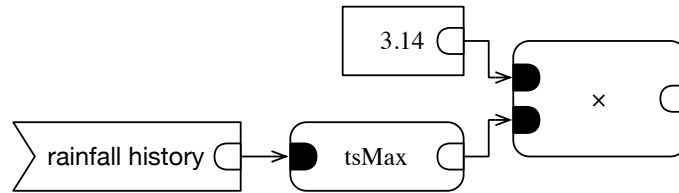


Figure 1: A simple example for an individual graph in our GP system.

selection method, but also much more complex parameters concerning program representation (e.g. linear, tree, or graph), program variation (mutation, crossover), and initial program generation (random, seeding with pre-specified individuals). A highly important parameter of GP is the set of primitive building blocks programs are created from, called the GP function and terminal set. Automatic parameter optimization techniques, such as *Sequential Parameter Optimization* (SPO) [6], can be employed to alleviate the difficulty of tuning this large parameter space.

An even more difficult problem is posed by the GP search space, the space of all expressions in a certain formalism (programming language). The computationally tractability of this search space depends on a complex combination of several factors, including the fitness function, the function and terminal sets, and the variation operators. For an effective evolutionary search, a certain causality between a program and its fitness value has to be maintained, in the sense that a small mutation of the program must not lead to an unbounded mutation of its fitness value. Maintaining this causality when parameterizing GP can be surprisingly difficult, for with many function and terminal sets, small changes to a program can cause radical changes in behaviour. Even if an effective evolutionary search is maintained by careful parameterization, the GP search space may be much too large to reliably find good solutions for more difficult real-world problems. In practice, it is often unavoidable to pre-structure this search space, as we show in section 2.3.

2.1 Typed graph GP for general time series prediction

Our GP implementation is a slightly generalized version of *vTrader*, a commercial typed graph GP system provided by DIP Dortmund Intelligence Project GmbH¹. *vTrader* represents GP individuals as term graphs of the expressions of the strict and strongly-typed functional programming language $gp\mathcal{L}$. These graphs are stored in a layered array data structure to allow efficient mutation and interpretation.

As $gp\mathcal{L}$ is strongly-typed, we define the set of types T first: A type $\tau \in T$ is exactly either one of the base types $B := \{\text{boolean}, \text{integer}, \text{double}, \text{string}, \text{date_time}, \text{time_series}\}$, or a n -ary function type $(\delta_1, \dots, \delta_n) \rightarrow \nu$, for base types $\delta_1, \dots, \delta_n$, and ν .

The set of terms of $gp\mathcal{L}$ is then defined as follows: A term of in $gp\mathcal{L}$ is exactly either a constant literal c of base type $\beta \in B$, or an input variable x of base type $\beta \in B$, or a n -ary function application $f(t_1, \dots, t_n)$ of type $\tau \in T$, where f must be of type $(\delta_1, \dots, \delta_n) \rightarrow \tau$ and the terms t_i of base types $\delta_i \in B$ for $i \in [1, n]$.

¹*vTrader* is primarily used in financial time series prediction and portfolio optimization. See <http://www.dortmundintelligence.com/> for more information.

Consider as an example the valid $gp\mathcal{L}$ term $3.14 \times tsMax(\text{rainfall history})$. This term is a function application and has the base type `double`. The function \times is of type $(\text{double}, \text{double}) \rightarrow \text{double}$, the function `tsMax` is of type $(\text{time_series}) \rightarrow \text{double}$, the constant literal `3.14` is of type `double`, and the input variable `rainfall history` is of type `time_series`. Figure 1 shows how this term would be represented as term graph in our GP system.

Building on this basic structure, `vTrader` provides a comprehensive default function and terminal set consisting of literals of all supported scalar types, arithmetic functions, statistical functions, boolean operators, as well as if-then-else branches and comparison functions. The resulting language is minimal in the sense that it does not contain higher-order or generic functions, but it is expressive enough to compactly represent interesting algorithms for time series prediction.

Populations can be initialized by randomly generating graphs of the required type, or by seeding with prefabricated graphs. The required graph type depends on the application problem and on the available input data. `vTrader` implements grow-, shrink-, strong-, and weak-mutation operators. These operators respectively add, remove, change, and perturbate random graph nodes. The strong-mutation operator replaces subgraphs with randomly generated graphs of matching type, while the weak-mutation operator perturbates numerical literals. An implementation of a crossover operator is planned, but not finished at the time being.

To employ our GP system for time series regression, we evolve individuals of the type $(\text{time_series}, \dots, \text{time_series}) \rightarrow \text{double}$. To use such an individual as a predictor, we just apply it to a window of our current set of input time series and interpret the resulting `double` value as the predicted value. The prediction horizon, as well as which time series to predict, is encoded in the fitness function as follows: We use a training set containing historical data of S_{target} , the time series to predict, as well as of $\mathcal{S} := \{S_{input_1}, \dots, S_{input_n}\}$, the set of input time series (S_{target} may be contained in \mathcal{S}). To measure the prediction accuracy of an individual I at time t with horizon length h , we first window all input time series to the interval $[t - (h+w), t - h]$ for some fixed window length w . We then apply I to the set of windowed input time series to calculate I 's prediction of $S_{target}(t)$ and record the squared error SE between the real value at $S_{target}(t)$ and this prediction. To approximate the prediction RMSE, we take the root mean of a number of randomly sampled values of SE on the training dataset, and take this approximation as the fitness value of I .

When later deploying an individual I as a predictor in a predictive control application, I simply has to be applied as a function to a window (of length w) of the current real-time input dataset.

2.2 The INT2 model for predictive control of stormwater tanks

We investigated in our previous work [3, 4] the stormwater tank problem as described in Sec. 1 with different modeling approaches, among them FIR, NARX, ESN, a dynamical system based on ordinary differential equations (ODE) and a dynamical system based on integral equations (INT2). All models were systematically optimized using SPO [6]. Among these models the INT2 approach turned out to be the best one, although the full modeling potential was not yet assessed in our previous work [3]. In the following we

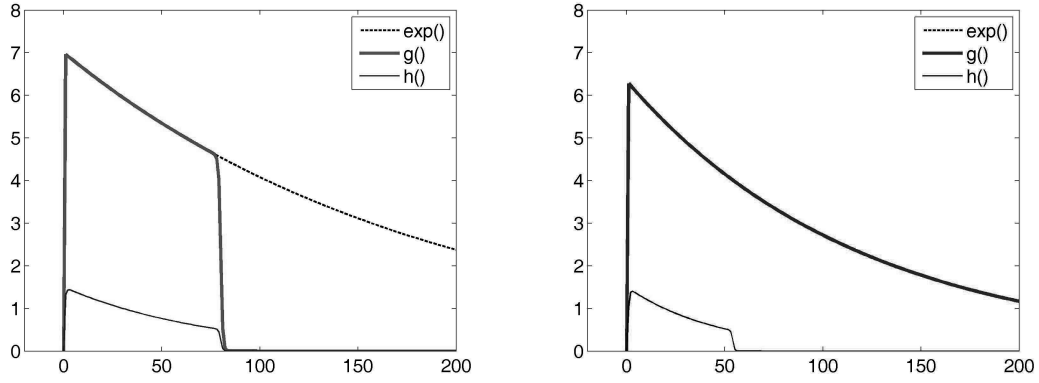


Figure 2: Analytic convolution kernels of the INT2 model. Left: manual setting, right: after SPO-optimization (from [3]).

describe briefly the INT2 model (for a more detailed presentation see [4]) and show how GP can be utilized to improve this model further.

The INT2 model

The causal relationships between incoming rain $r(t)$ and the resulting stormwater tank fill level $y(t)$ are modeled by the following integral equations:

$$L(t) = \int_{-\infty}^t \beta_L r(\tau) e^{-\alpha_L(t-\tau)} d\tau \quad (1)$$

$$K(t) = \max(0, L(t) - \Delta) \quad (2)$$

$$y(t) - B = \int_{-\infty}^t r(\tau - \tau_{\text{rain}}) g(t - \tau) d\tau + \int_{-\infty}^t K(\tau - \tau_{\text{rain}}) h(t - \tau) d\tau \quad (3)$$

As an intermediate quantity we introduce in Eq. (2) the 'leaky rain' $L(t)$ which is a leaky integration of the past rainfall and thus may help to characterize the hidden state of the soil. $K(t)$ is simply a clipped version of $L(t)$. Eq. (3) models the stormwater tank fill level as the convolution of rainfall and leaky rain with certain filter kernels $g(t)$, $h(t)$. A specific filter kernel like

$$g(t) = \beta e^{-\alpha t}$$

would correspond to the leaky integration of Eq. (1). But other forms are equally well possible. As an example one might consider truncated exponential functions with smooth on- and offset

$$g(t) = \beta \sigma(t - \tau_{in}) e^{-\alpha t} \sigma(\tau_{out} - t) \quad (4)$$

$$h(t) = h_0 \sigma(t - \tau_{in3}) e^{-\alpha_H t} \sigma(\tau_{out3} - t) \quad (5)$$

$$\text{with } \sigma(t) = \frac{1}{2}(1 + \tanh(\kappa t)), \quad (6)$$

which are depicted in Figure 2.

Kernel optimization The flexibility of the INT2 model—in principle any causal kernel function can be used—constitutes an opportunity and a burden at the same time. The opportunity is that a rich set of pre-structured model functions is at hand. The burden is that it is manually impossible to investigate in a systematic way a large set of possible functions. The parameterization example Eqs. (4), (5)—although containing 9 tunable parameters—characterizes only a small set of possible kernel functions. The application of GP was motivated by the need to investigate a much richer set of possible kernel functions. the following.

2.3 Optimizing INT2 through GP

Even though the INT2 model has been established as the best alternative in our previous work (see [3]), it still has some drawbacks, as already described in Sec. 2.2. Especially the requirement to choose a specific kernel function is an important problem, which can have a significant impact on the results. Of course it might be possible to build other kernel functions by the practitioner, but one can not be sure to find the optimal kernel. Due to the infinite function space, it seems to be as looking for the needle in a haystack.

With the integration of the kernel function into the GP system, this additional burden for the practitioner can be removed. Also, this integration might be a promising way to evolve more suitable kernel functions. When automatically evolving kernel functions for the INT2 model, it is of course of large interest whether it is possible to achieve an improvement over the hand- and SPO-tuned model.

Pre-structured GP

One can think of integrating other models as pre-structuring elements into the GP system and to compare the results of these new GP-evolved models with the results of the constituent models. The optimization of models based on ordinary differential equations (ODE) might be such an alternative. However, as we have achieved the best results with the INT2 model in our previous research, we first concentrate on joining the INT2 model with our GP system. We identified three possible approaches for doing so:

- *Pre-structuring individuals* for the initial population of the GP algorithm: Individuals are initialized with the INT2 model itself. These individuals are optimized by the general evolutionary process, e.g., mutation and recombination. The underlying concept of this approach can be described as follows: The INT2-initialized population provides an established model in the solution set from the beginning of the search which might accelerate the evolution. This approach is very general and powerful, but requires the reformulation and reimplementing of the established model in the programming language used as the individual representation of the underlying GP system ($gp\mathcal{L}$ in our case).
- *Integration of the INT2 model* into the function set of the GP system: In this approach, variation operators can choose the model as a building block for creating new individuals. With this approach to model integration, the function set gains a much more powerful element in comparison to the standard functions generally used in GP.

- *Embedding the INT2 model* into the fitness function of the GP system: Each time the fitness function is evaluated, the INT2 model is calculated based on GP-generated convolution kernels. The prediction RMSE of the INT2 model is then used as a fitness value for the GP system.

We choose the third approach in this work, because it offers a simple and practicable way to use the INT2 model with the GP system. The main advantages of this solution are that there is no need to extend the standard function set of GP, or to restrict the search space to certain areas, as done in the first option. The integration into the fitness function does not require any changes to the GP system itself and is therefore easy to implement in any given GP system. Also, the individuals themselves can remain rather simple and do not need to grow too much during the search, which avoids a drawback of genetic programming (see e.g. *bloat* in genetic programming). It is of course possible to combine all three approaches to model integration, for example by implementing only some components of the model as GP functions to allow for free recombination of these components, and keeping other parts of the model fixed in the fitness function. The exploration of these possibilities is a theme for further research.

3 Experimental results

3.1 Experimental setup

In this section, we present first experimental results of applying our method to the real-world problem of water level prediction in stormwater tanks based on historic rainfall data. To generate a baseline for comparisons, we applied the INT2 model, as the current best known prediction model for this problem. We then applied standard GP, i.e. GP with a standard function set, to this problem. Finally, we applied our GP-optimized version of INT2. All experiments were performed on the same set of input data.

Our input data consists of measurements of the current water level and the current rainfall at a real stormwater tank in Germany. These measurements were taken every 5 minutes, ranging from 01 April 2007 (10:15) to 18 July 2007 (10:10). We divided this dataset into a training dataset, ranging from 21 April 2007 (00:00) to 28 April 2007 (00:00), and a test dataset, ranging from 28 April 2007 (00:05) to 18 July 2007 (10:10). The first 20 days of April 2007 were nearly completely dry, so they were omitted from the training dataset. All the results presented in the following are based on the test dataset.

Since both GP and SPO are randomized algorithms, each experimental setting was repeated 5 times on consecutive random seeds. Our experiments show that the quality of the results did not vary much. We only present the best results obtained in the 5 runs of each experimental setting. The experimental settings involving GP were parameterized to allow them a comparable quota of computing time. Each of the total of 10 GP runs took 4 hours of computing time a single core of an Intel Xeon E5530 (2.4 GHz) CPU. Each run was allowed a maximum memory usage of 4 GiB.

3.2 INT2 with analytic kernels

As a baseline for comparisons, we applied the INT2 model to our test data, using the parameters as described in [3, 4]. In these studies, INT2 was found to give the best results

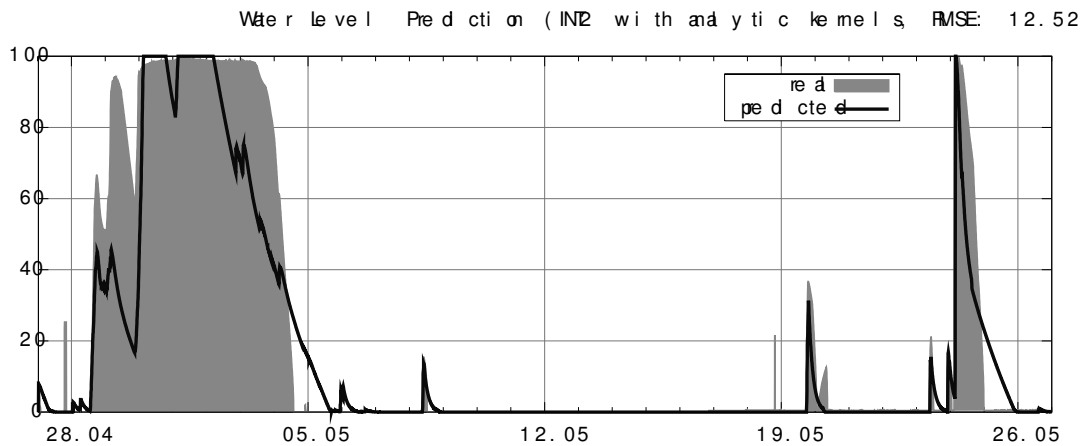


Figure 3: Water level prediction with INT2, using SPO-optimized analytic kernels.

of all investigated modeling approaches, including FIR, NARX, ESN, and a dynamical system based on ordinary differential equations (ODE).

We used the SPO-optimized analytical INT2 convolution kernels $g(t)$ and $h(t)$ based on the parametrization of Eqs. (4), (5), and shown on the right of Figure 2. The parametrized functions of these kernels were defined manually, taking background knowledge about the water level prediction into account. Then the 9 parameters of these functions together with other model parameters were optimized using SPO, giving a near optimal parameterization of the INT2 model. In contrast, with our GP method, we will try in Sec. 3.4 to evolve suitable kernels without any problem-specific background knowledge or human intervention. Figure 3 shows the results of applying the INT2 model with SPO-optimized convolution kernels to our test dataset. The filled curve represents the real water levels, while the dark solid line represents the predicted water levels.

3.3 Standard GP

To our knowledge, this is the first application of GP to the stormwater tank water level prediction problem. We proceeded as described in Sec. 2.1, using the GP parameters shown in Table 1. We deliberately choose a simple "standard" function set, containing only basic arithmetic, logic, comparison, control flow, and time series processing functions. These functions should be available by default on every modern typed GP system. Another reason for choosing this simple function set is that practitioners often do not have neither the time nor the resources to extensively tune a GP function set.

We parameterized the GP system so that only rainfall data could be used as input to the evolved predictor functions. The predictions of the best individual found are shown in figure 4. This rather disappointing result was to be expected, given the vast search space of predictor functions and given the other reasons already mentioned in Sec. 2.

A customized problem-specific function set would probably greatly improve GP performance. But for our problem, as for many other real-world problems, it is very difficult to define such an problem-specific function set without detailed background knowledge. That is why we turn to existing analytical models for guidance. Our approach of integrating these existing models into GP can be seen as a first step towards formalizing

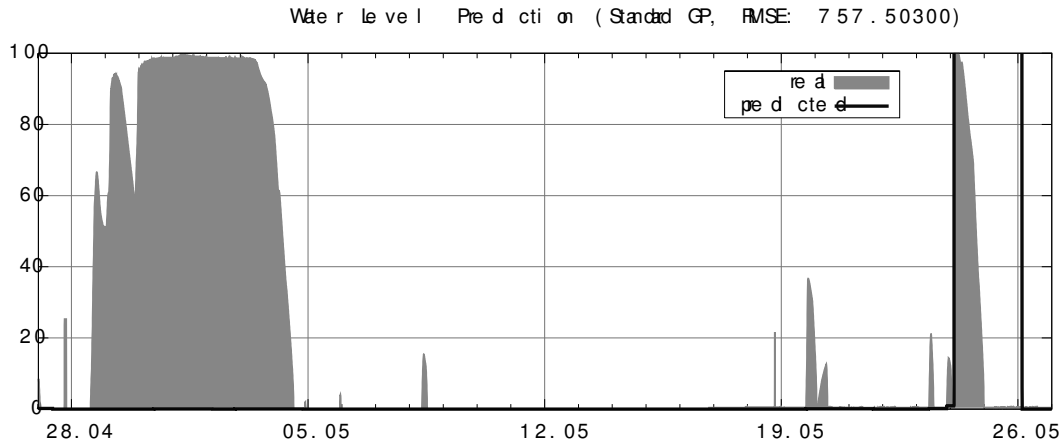


Figure 4: Water level prediction through GP with a standard function set.

the process of inferring improved evolutionary algorithms for real-world problems from existing analytical models.

Table 1: Parameters used for standard GP runs.

Objective:	Find a predictor function for the water level in stormwater tank, based on historical rainfall data. The prediction horizon is 0 minutes.
Terminal set:	$\{\text{rainfall history}\} \cup [-1.0, 100.0]$
Function set:	$\{+, \times, -, \%(safe\ divide), abs, max, min, ln, ifThenElse, <, <=, >, >=, not, and, or, tsMax, tsMin, tsSlope, tsArithmeticMean, tsStdDev, tsSum\}$ Functions prefixed with <i>ts</i> are defined on time series.
Fitness:	$RMSE(\text{water level}_{predicted}, \text{water level}_{real})$
Selection:	Tournament selection with tournament size 2.
Initialization:	Random graphs with maximum depth of 20 and maximum size of 100 nodes. The population size is 20 graphs, with a maximum depth of 40 and a maximum size of 200 nodes.
Variation:	3 steps per individual variation, with a probability for grow-mutation of 0.6, shrink-mutation of 0.0, strong-mutation of 0.8, and weak-mutation of 0.2 at each step.
Termination:	Terminate after 2,500 generations (25,000 fitness evaluations).

3.4 INT2 with GP-optimized kernels

In the following experiments, we evolved the INT2 convolution kernels $g(t)$ and $h(t)$ through our GP system, using the parameterization shown in Table 2. Both kernels are functions of type $(\text{double}) \rightarrow \text{double}$ that are simultaneously co-evolved in two separate populations in our GP system. Settings that only evolved $g(t)$ or $h(t)$, using the respective SPO-optimized kernels $h(t)$ or $g(t)$ to complete the model, were also tried. While these settings delivered good results in less time due to lesser problem complexity, the results after 2,500 generations were comparable to the results of the co-evolution setting described here.

To ensure an unbiased GP search space of manageable size, the GP function set in this configuration contains the polyline function only. This function behaves as a piecewise linear function of its first parameter defined by the control points given as its remaining

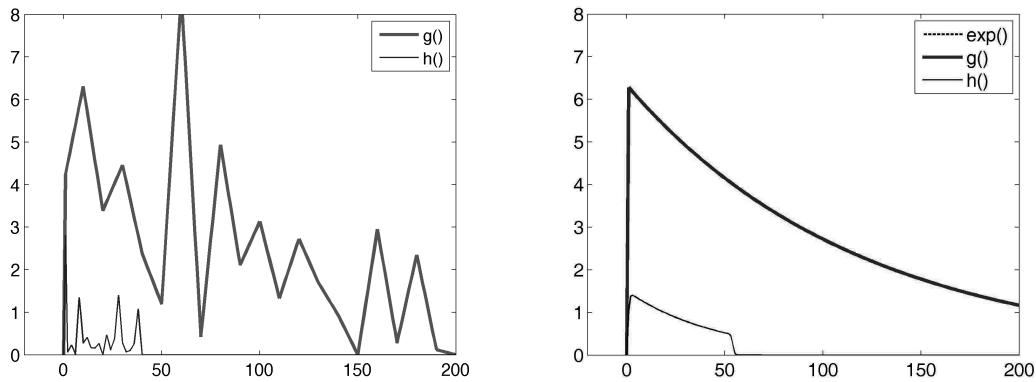


Figure 5: Left: GP-evolved convolution kernels for the INT2 model. Right: SPO-optimized kernels for comparison.

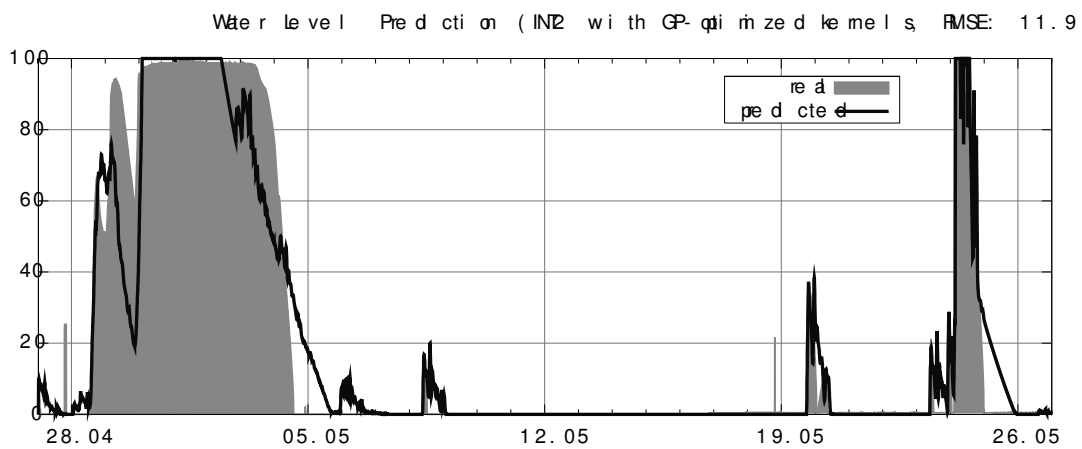


Figure 6: Water level prediction with INT2, using the GP-evolved kernels of figure 5.

parameters. In this setting, we used 20 control points, given by 40 values of type `double`, resulting in a polyline function of arity 41. A standard GP function set, as in subsection 3.3 could have also been used, which would be topic for further research. We chose the a piecewise linear function in this setting to enable an unbiased search for kernel functions of all forms, while keeping the search space reasonable small.

Figure 5 shows the best kernels found in 5 runs. The predictions of the INT2 model using these kernels are shown in Figure 6. As can be seen in this figure, these results are slightly better than the results based on SPO-optimized analytic kernels presented in subsection 3.2. It might be possible to further improve on these results by employing SPO to optimize the remaining INT2 model parameters (such as τ_{rain}) for the GP-evolved kernels. Here, we just used the same settings as in subsection 3.2 for these parameters.

3.5 Discussion

Table 3 summarizes our experimental results. The prediction RMSE column shows the best result out of 5 runs for each method. It was calculated on the test dataset defined in Sec. 3.1. With a prediction RMSE more than 60 times larger than the other two methods,

Table 2: Parameters used for GP-optimized INT2 runs.

Objective:	Find optimal convolution kernels $g(t)$ and $h(t)$ for the INT2 model.
Terminal set:	$\{t \text{ (i.e. input variable the kernel function)}\} \cup [0.0, 200.0]$
Function set:	{polyline}
Fitness:	RMSE(water level _{predicted} , water level _{real})
Selection:	Tournament selection with tournament size 2.
Initialization:	Random graphs with maximum depth of 2 and maximum size of 100 nodes. The population size is 10 graphs, with a maximum depth of 2 and a maximum size of 200 nodes.
Variation:	5 steps per individual variation, with a probability for grow-mutation of 0.0, shrink-mutation of 0.0, strong-mutation of 0.0, and weak-mutation of 1.0 at each step.
Termination:	Terminate after 2, 500 generations (12, 500 fitness evaluations).

GP with a standard function set does not lead to sensible results. Pre-structuring the GP search space with the INT2 model on the other hand gives a result even slightly better than the so far best known approach for this problem, INT2 with analytic kernels.

Table 3: Overview of the experimental results. The prediction RMSE for each method was calculated on the test dataset.

<i>Method</i>	<i>prediction RMSE</i>
INT2 with analytic kernels	12.52496
Standard GP	757.50300
INT2 with GP-optimized kernels	11.91573

At first sight it might seem a bit awkward that the GP-evolved kernels in Figure 5 show such a rugged shape. Wouldn't a smooth shape be more natural? Although this might seem reasonable, one has to keep in mind that there was no evolution pressure towards smooth kernel functions and that the rain data itself show on the relevant timescale a very non-smooth, burst-like behaviour (sharp 'needles'). Therefore it is quite natural for the GP-system to 'invent' also a non-smooth kernel function $g(t)$ which tries to optimize the effects of certain needles on the subsequent fill levels in our training data. Keep in mind that the time span of the training data in this study is rather small. We expect the GP-evolved kernels to become smoother when we are using more training data, which has however to be confirmed by future research.

4 Conclusions and outlook

GP is difficult to apply directly to real-world time series prediction problems, at least when using a standard function set. Hand-tuning the function set promises improved results, but is often very difficult to do in practice. Furthermore, for many real-world problems, it is not at all clear if it is easier to find a suitable GP function set than to find a suitable modeling function directly. GP shows its strength in domains where there are many modeling functions that work only some of the time, without clear rules when each of these modeling functions should be applied. The most striking example for this kind of problem domain is financial time series prediction, the problem domain our GP system vTrader has been designed for. The vTrader GP function set contains building blocks

for a wide array of financial time series regression models, which are then selected and recombined during GP search based on the accuracy they give on the most current financial time series data.

We showed that by pre-structuring the GP search space with an established analytical model of the prediction problem to solve, it is possible to improve on the results of even an SPO-optimized version of this model. Given an easy to apply pre-structuring method for the GP search space, GP could be very useful in optimizing arbitrary components of existing models. Such a method could be perceived not so much as an extension of GP, but as a design method for application-specific evolutionary algorithms.

In this first study, by integrating our analytical model in the GP fitness function, we only implemented the simplest approach to model integration. By implementing the other approaches to model integration described in subsection 2.3, many other interesting opportunities for applying GP in the optimization of analytical models would open. By integrating the constituent parts of the INT2 model as functions in the GP function set, the GP system could recombine these parts in novel ways to find a better configuration of the model. By controlling the granularity of these parts, the size of the GP search space can be controlled.

As already pointed out in Sec. 3.5, we currently are in the process of applying our methods to larger datasets (bigger time span, more different stormwater tanks) in order to test the generalizability of our model and to see whether GP-evolved kernels trained on larger datasets would become smoother.

Additional directions of further research would be the evaluation of function sets other than the one used in subsection 3.4 for the GP-evolution of INT2 convolution kernels. Also, applying our approach to other problems in predictive control that have existing analytical models is of great interest.

5 Acknowledgements

This work has been supported by the Bundesministerium für Forschung und Bildung (BMBF) under the grants FIWA and SOMA (AIF FKZ 17N1009, "Ingenieurnachwuchs") and by the Cologne University of Applied Sciences under the research focus grant COSA. We are grateful to Prof. Dr. Michael Bongards and his research group (GECO-C) for discussions and for the stormwater tank data and to Dr. Wolfgang Kantschik (DIP GmbH) for making the vTrader GP system available to us as well as for many helpful discussions.

References

- [1] Brockwell, P. J.; Davis, R. A.: *Introduction to Time Series and Forecasting*. New York NY: Springer. 2002.
- [2] Smits, G.; Vladislavleva, E.: Ordinal Pareto Genetic Programming. In: *Proceedings of the 2006 IEEE Congress on Evolutionary Computation* (Yen, G. G.; Lucas, S. M.; Fogel, G.; Kendall, G.; Salomon, R.; Zhang, B.-T.; Coello, C. A. C.; Runarsson, T. P., Hg.), S. 3114–3120. Vancouver, BC, Canada: IEEE Press. URL <http://ieeexplore.ieee.org/servlet/opac?punumber=11108>. 2006.

- [3] Bartz-Beielstein, T.; Zimmer, T.; Konen, W.: Parameterselktion für komplexe Modellierungsaufgaben der Wasserwirtschaft – Moderne CI-Verfahren zur Zeitreihenanalyse. In: *Proc. 18th Workshop Computational Intelligence* (Mikut, R.; Reischl, M., Hg.), S. 136–150. Universitätsverlag, Karlsruhe. 2008.
- [4] Konen, W.; Zimmer, T.; Bartz-Beielstein, T.: Optimierte Modellierung von Füllständen in Regenüberlaufbecken mittels CI-basierter Parameterselktion. *at – Automatisierungstechnik* 57 (2009) 3, S. 155–166.
- [5] Poli, R.; Langdon, W. B.; McPhee, N. F.: *A field guide to genetic programming*. Published via <http://lulu.com> and freely available at <http://www.gp-field-guide.org.uk>. URL <http://www.gp-field-guide.org.uk>. (With contributions by J. R. Koza). 2008.
- [6] Bartz-Beielstein, T.: *Experimental Research in Evolutionary Computation—The New Experimentalism*. Natural Computing Series. Berlin, Heidelberg, New York: Springer. 2006.

Beschleunigung der Selbstoptimierung durch Selbstsimulation

Nils Rosemann, Andreas Buschermöhle, Werner Brockmann

Universität Osnabrück
Albrechtstraße 28
49069 Osnabrück
Tel.: (0541) 969 2488
E-Mail: nils.rosemann@uos.de

Zusammenfassung

Moderne Regelungssysteme werden in zunehmend komplexeren Szenarien und Umgebungen eingesetzt. Klassische Entwurfstechniken stoßen dabei an ihre Grenzen, da sie dynamische Veränderungen, wie sie durch Verschleiß und unsichere Interaktionen mit der Umgebung entstehen, nur mit hohem Aufwand berücksichtigen können. Aus diesem Grund wurden verschiedene Arten von selbstoptimierenden Verfahren entwickelt. Sie lassen sich in zwei Kategorien einteilen, nämlich direkte und indirekte Adaption. Beide Adaptionarten haben ihre spezifischen Vor- und Nachteile. Entsprechend wünschenswert ist ein Lösungsverfahren, das deren Vorteile vereint, die Nachteile aber möglichst ausschließt. Zu diesem Zweck wird in dieser Arbeit die IRIS-Architektur entwickelt, die einen allgemeinen Ansatz zur Integration verschiedener Regelungen unter Berücksichtigung dynamisch veränderlicher Unsicherheiten darstellt. Dadurch können selbstoptimierende Systeme trotz fehlender, formaler Stabilitätsnachweise eine hohe Vertrauenswürdigkeit erreichen. Ein simuliertes Kran-Szenario dient in dieser Arbeit als erstes Demonstrationsbeispiel für den IRIS-Ansatz.

1 Motivation

Moderne Automatisierungssysteme werden in zunehmend komplexeren Szenarien und Umgebungen eingesetzt. Dadurch wird nicht nur die formale Modellierung erschwert. Neben den damit verbundenen Herausforderungen an das Engineering kommen je nach Anwendung weitere Herausforderungen durch ein zeitvariantes Systemverhalten und/oder dynamisch veränderliche Unsicherheiten bei der Erfassung oder Beeinflussung des Systemzustandes. Neben Sensorrauschen oder Messfehlern können dies Lose, Alterung oder ähnliches sein. Klassische Entwurfstechniken stoßen hier an ihre Grenzen, da sie dynamische Veränderungen, wie sie durch Verschleiß und unsichere Interaktionen mit der Umgebung entstehen, oder dynamisch veränderliche Unsicherheiten nur mit hohem Aufwand berücksichtigen können.

Selbstoptimierende¹, lernfähige Regelungen haben sich als eine mögliche Technik erwiesen, um in der Automatisierungstechnik solche Aufgaben leichter zu bewältigen [1]. Der Hauptgrund liegt darin, dass sich die vollständige formale Modellierung des zu

¹Die Begriffe „Selbstoptimierung“ und „Adaption“ werden hier synonym verwendet. Das eine ist aus der Sicht des Systems, das andere aus der des Reglers.

regelnden Prozesses oftmals umgehen lässt. Stattdessen werden aus Beobachtungen des Verhaltens im laufenden Betrieb Informationen abgeleitet, um die Regelung kontinuierlich zu optimieren. Dazu wurden verschiedene Arten von selbstoptimierenden Verfahren entwickelt, die im Betrieb ihr Regelverhalten anpassen. Sie lassen sich grob in zwei Kategorien einteilen, nämlich direkte und indirekte Adaption. Bei ersterer werden aus dem beobachteten Systemverhalten online Lernvorgaben für einen lernfähigen Funktionsapproximator (z. B. ein künstliches neuronales Netz oder lernfähiges Fuzzy-System) generiert und direkt in eine Änderung des lernfähigen Systems umgesetzt. Bei der indirekten Adaption wird dagegen aus dem beobachteten Systemverhalten zunächst online oder offline ein Modell gelernt. Aus dem Modell wird dann mittels vorgegebener Entwurfsverfahren direkt das gewünschte Systemverhalten, also der eigentliche Regler, abgeleitet.

Eine solche Vorgehensweise ist aber im Allgemeinen nur möglich zu dem Preis einer deutlich reduzierten Vorhersagbarkeit des Gesamtsystemverhaltens. Denn das, was zu einem gegebenen Zeitpunkt gelernt wird, verändert das zukünftige Verhalten des Systems, und das wiederum, was zukünftig gelernt werden wird, und so weiter. Dadurch erhält die Selbstoptimierung den Charakter eines chaotischen Systems, denn kleine Änderungen der Anfangsbedingungen können zu ganz unterschiedlichen Lernverläufen führen. Außerdem kann sich schnell ein (sicherheits)kritisches Verhalten einstellen. Dies zu verhindern stellt wegen der nur sehr indirekten Einflussmöglichkeiten des Entwicklers eine besondere Herausforderung dar.

Durch den geschlossenen Wirkungskreis ist die Selbstoptimierung von Reglern besonders kritisch. Neben der erforderlichen Stabilität gilt es daher die Robustheit gegenüber (kurzfristigen) Störungen zu optimieren bzw. sicherheitskritische Betriebszustände zuverlässig zu vermeiden. Zudem stellt sich der Lernerfolg nicht unmittelbar ein. Je besser das lernfähige System vorstrukturiert und je mehr (Vor-)Wissen mitgegeben ist, desto eher ist die Adaption abgeschlossen.

Ein weiteres grundsätzliches Problem ist, dass das selbstoptimierende, d. h. online lernende System natürlich nicht a priori für alle Betriebsfälle, also den gesamten Eingangsraum, vollständige und gleich gute Lernvorgaben bekommen kann. Denn die technischen Systeme werden i. d. R. gar nicht in kritischen Bereichen und nur selten in deren Nähe betrieben. Im normalen Betrieb durchlaufen sie üblicherweise nur noch weiter eingeschränkte Betriebsbereiche. Folglich ist (zumindest im worst case) davon auszugehen, dass weite Bereiche des Eingangsraums aufgrund fehlender Lerndaten unspezifiziert sind. Kritisch wird dies, sobald das System z. B. aufgrund von unerwarteten Anregungen wie Störungen in solche Bereiche gerät.

Um gleichzeitig den Engineering-Aufwand gering zu halten, aber auch einen guten und sicheren Betrieb eines selbstoptimierenden Systems zu gewährleisten, lassen sich die Anforderungen an selbstoptimierende Systeme wie folgt zusammenfassen:

- wenig prozessspezifisches Vorwissen, wenig formaler Modellierungsaufwand erforderlich,
- Engineering möglichst auf einer abstrakteren, verhaltensorientierten Ebene,
- schnelles Erreichen eines gut und sicher funktionierenden Systemverhaltens

(bei der Selbsteinstellung² wie auch bei der Anpassung an ein zeitvariantes Systemverhalten),

- trotz unvollständiger, inkrementeller Lernvorgaben schnelle und gute Konvergenz der Lernverfahren,
- sicheres Einhalten von Randbedingungen, z. B. hinsichtlich dynamischer oder anderer sicherheitsrelevanter Eigenschaften,
- Interpretierbarkeit der Lernergebnisse zum besseren Verständnis des Systems und zur Ergründung von Fehlerquellen.

Beide oben genannten Adaptionsarten erfordern ausdrucks mächtige Funktionsapproximatoren als lernende Regler und haben vor diesen Hintergründen ihre spezifischen Vor- und Nachteile, die nachfolgend diskutiert werden.

2 Stand der Technik

Selbstoptimierende Systeme nutzen allgemeine Funktionsapproximatoren als Wissensrepräsentation. Dadurch zielen sie auf Problemstellungen ab, die sich der formalen Modellierung, z. B. aufgrund des zu hohen Aufwandes, entziehen. Grundlegerend ist dabei die Technik der *direkten Adaption*. Man spricht von direkter Adaption, wenn Eigenschaften des Reglers im Betrieb direkt aus dem beobachteten Prozessverhalten abgeleitet werden. Dazu ist eine passende, inkrementelle Lernarchitektur notwendig [2, 3]. Sie lässt sich im Prinzip mit einem beliebigen Funktionsapproximator nutzen. Auf Grund der lokalen und sequentiellen Natur der Lernvorgaben bieten sich aber inkrementelle, lokale Funktionsapproximatoren an. Dazu eignen sich vor allem spezielle Fuzzy-Systeme, die einen guten Kompromiss aus Ausdrucksmächtigkeit, Lerndauer und Interpretierbarkeit darstellen [4, 5]. Die direkte Selbstoptimierung hat so den Vorteil, dass sie ohne formalen Entwurf mit nur wenig grundlegendem Expertenwissen (z. B. Prozessmonotonien, Latenzen) auskommt. Solche Regelungen können sich durch eine schnelle Selbsteinstellung, hohe Flexibilität und geringe rechnerische Komplexität auszeichnen. Im Gegenzug können sie nur in tatsächlich aufgetretenen Situationen sinnvolles Wissen ansammeln. Zudem ist ein formaler Nachweis der Konvergenz und Stabilität nicht möglich. Für einen solchen Nachweis ist ein formales Prozessmodell nötig, zumindest in Form von einer Prozessklasse. Nimmt man eine feste Prozessklasse an, so kann man mächtige direkte Adaptionsverfahren entwickeln [6]. Dabei müssen aber oftmals schwer interpretierbare Entwurfsentscheidungen getroffen werden, deren Auswirkung auf das spätere Regelverhalten, z. B. bei Störungen, kaum absehbar ist. Ein allgemeiner Nachteil von direkten Verfahren ist natürlich die fehlende Prädiktionsmöglichkeit und das potentiell kritische Regelverhalten in bis dato ungelerten Bereichen, da dafür ein Modell nötig wäre.

²Unter Selbsteinstellung wird im Folgenden verstanden, dass ein Regler auch gänzlich ohne prozessspezifische Parametrierung autonom „from scratch“, also bei der Inbetriebnahme, durch Selbstoptimierung eingestellt wird.

Diese Nachteile sind durch die *indirekte Adaption* teilweise vermeidbar, bei der zunächst ein formales Prozessmodell identifiziert wird, aus dem dann Eigenschaften des Reglers algorithmisch bestimmt werden. Auch hier ist eine *a priori* Einschränkung der Prozessklasse zunächst vorteilhaft, weil sich so eine geeignete Modellstruktur angeben lässt und sich damit die Modellidentifikation auf eine Schätzung von (wenigen) Parametern zurückführen lässt. Das identifizierte Modell ist (potentiell) global korrekt. Online-Schätzungen sind aber problembehaftet, so dass Zusatzmechanismen nötig sind, um z. B. Drift³ und Bursting⁴ zu vermeiden [7, 8].

Es bietet sich aber nicht für jede Prozessklasse an, ein parametrisiertes Modell zu nutzen. Stattdessen kann auch die Modellstruktur im Betrieb gelernt werden. Dies erfordert jedoch eine starke Einschränkung der erlaubten Modellstrukturen, da ansonsten der Suchraum zu groß wird. Ein Beispiel ist der *Starfish*-Roboter, der seine kinematische Struktur, also kein Prozessmodell sondern ein Modell auf einer höheren Abstraktionsebene, zielgerichtet mithilfe evolutionärer Verfahren ermittelt [9]. Nachteilig ist hierbei natürlich die extrem lange und rechnerisch aufwändige Explorationsphase. Außerdem erlauben die meisten praxisrelevanten Prozesse keine oder nur eine sehr eingeschränkte Modellidentifikation durch aktive Exploration. Einen Kompromiss aus flexibler Modellstruktur und akzeptabler Lerndauer stellen vorstrukturierte Funktionsapproximatoren dar [10]. Als wichtiger Nachteil bleibt, dass das gelernte Modell nur an Stellen im Eingangsraum korrekt ist, an denen Lerndaten vorliegen.

Ein wichtiger Vorteil eines gelernten oder angenommenen Modells ist die Möglichkeit, *prädiktiv* zu regeln, das heißt potentielle Stellaktionen im Vorhinein zu beurteilen. Prädiktivregelung (*Model Predictive Control, MPC* [11]) erlaubt es nämlich, komplexe Randbedingungen und Kriterien einzuhalten bzw. zu optimieren. Beispiele sind Beschränkungen von Stellgrößen, Energieeinsparungen und Sicherheitsgrenzen. Die Güte einer Prädiktivregelung hängt natürlich empfindlich von der Genauigkeit des Modells ab. In den meisten Fällen wird MPC daher mit einem festen, d. h. nicht adaptiven, Modell genutzt.

In Szenarien, in denen ein festes Modell nicht verfügbar ist, muss allerdings mit adaptiven Modellen gearbeitet werden. Eine gestufte Herangehensweise besteht dann darin, zentrale Parameter eines Modells im Betrieb durch eine *open-loop*-Strategie zunächst zu identifizieren und erst dann für *closed-loop*-Prädiktivregelung zu verwenden, wie das [12] an einem medizintechnischen und damit sicherheitskritischen Beispiel der Verabreichung von Anästhetika zeigt.

Die Vor- und Nachteile der einzelnen Ansätze sind in Tabelle 1 zusammengestellt. Es zeigt sich, dass kein Ansatz alle Kriterien gleichzeitig oder gleich gut erfüllt. Die direkte Selbstoptimierung bietet eine schnelle Selbsteinstellung, kann jedoch keine Sicherheit bieten, da das zukünftige Verhalten nicht bekannt ist. Dies wird von der indirekten Selbstoptimierung erreicht, benötigt jedoch einen formalen Entwurf. Daher bietet eine Architektur, die die beiden Ansätze verbindet und so versucht, die Vorteile beider zu nutzen und die Nachteile zu schmälern, eine gute Chance, die gestellten Anforderungen zu erfüllen.

³Drift bedeutet, dass die Parameterschätzung des Modells nicht konvergiert, wobei der geschlossene Regelkreis dennoch stabil sein kann.

⁴Bursting bezeichnet einen plötzlichen Ausbruch der Regleraktivität, sobald das identifizierte Modell instabil wird. Durch die Anregung wird die Parameterschätzung verbessert und das identifizierte Modell kurzfristig wieder stabil. Der Ausbruch kann quasizyklisch immer wieder auftreten.

Anforderungen	direkte Selbstopt.	Starfish	direkte Adaption	indirekte Adaption	indirekte Selbstopt.
schnelle Selbsteinstellung	++	-	+	+	+
formaler Entwurf nötig	nein	nein	ja	ja	ja
globale Konvergenz	nein	ja	ja	ja	nein
Flexibilität	+	-	-	-	-
Stabilitätsnachweis	-	-	+	+	+
Sicherheit / Prädiktion	-	+	-	+	+

Tabelle 1: Vergleich der unterschiedlichen Ansätze bezüglich der Anforderungen.

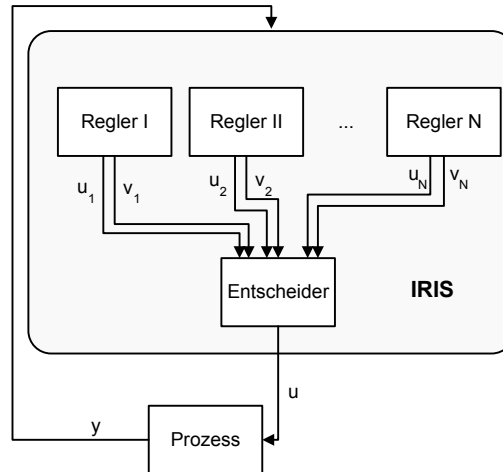


Abbildung 1: Architektur des vorgestellten IRIS-Ansatzes.

3 Ansatz

3.1 Gesamtarchitektur

Die IRIS-Architektur (*Integrative Robust Intelligent System*) basiert auf der Verwendung von N Reglern, die nach verschiedenen Verfahren arbeiten bzw. sich selbst optimieren. In jeder Situation werden entsprechend N Stellgrößen (u_1, \dots, u_N) für den zu regelnden Prozess generiert. Dies ist in Abbildung 1 dargestellt. Neben der Stellgröße liefern die Regler zusätzlich ein Vertrauensmaß (v_1, \dots, v_N), welches angibt, wie sicher sich der jeweilige Regler ist, dass die gegebene Stellgröße zur aktuellen Situation passt. Ein nachgelagerter Entscheider integriert diese Informationen anhängig von den Vertrauensmaßen zu einer endgültigen Stellgröße u , die an den Prozess gegeben wird. Zwei Varianten, diese Integration durchzuführen, werden in Kapitel 3.3 näher erläutert.

Dieses Konzept ermöglicht es, verschiedene Regler zu kombinieren. So kann zwischen den gegebenen Reglern zum einen räumlich im Eingangsraum gewechselt werden und zum anderen dynamisch in der Zeit durch lernfähige Regler, deren Vertrauensmaß vom Lerninput abhängt. Auf diese Weise kann zusätzlich zu einem lernfähigen Regler ein möglicherweise ungenauer, aber sicherer Regler als „Fallback-Lösung“ gegeben werden, der im Zweifelsfall verwendet werden kann oder den Prozess in kritischen

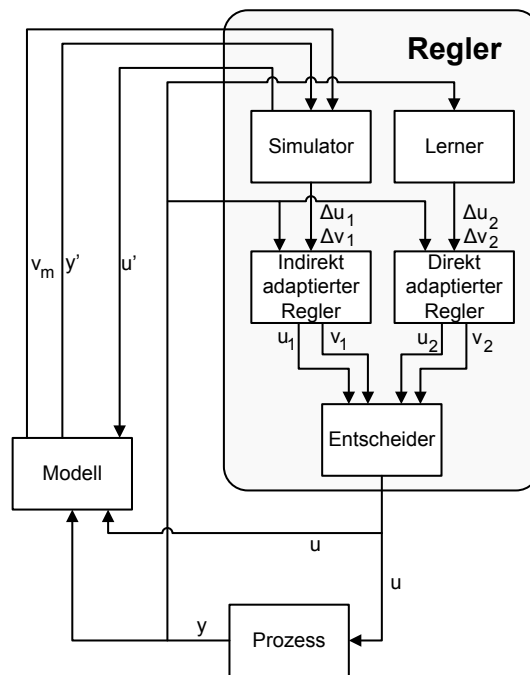


Abbildung 2: Spezielles Szenario der IRIS-Architektur.

Situationen wieder in den sicheren Bereich bringt. Ein lernfähiger Regler kann darauf aufbauend während der Regelung Informationen über den Prozess sammeln, um durch indirekte Adaption seine Regelstrategie anzupassen. Sobald das gelernte Wissen ausreicht, um den Prozess selbst zu regeln, übernimmt der lernfähige Regler das System, um eine exaktere Regelung zu erreichen. Durch dieses Konzept können auch bereits existierende Regler, die zum Beispiel in Form eines PID-Reglers oder als Fuzzy-Expertensystem vorliegen, in die Architektur eingebunden werden.

3.2 Architekturvariante für lernfähige Regler

Während die Integration eines statischen und eines lernenden Reglers recht anschaulich ist, ist die Integration von direkt und indirekt adaptierenden Reglern diffiziler. Deshalb wird hierauf genauer eingegangen. Zur Verdeutlichung ist eine spezielle Anwendung des Konzeptes in Abbildung 2 dargestellt. Hier werden zwei lernfähige Regler in Kombination eingesetzt. Ein Regler basiert auf direktem Lernen nach dem Prinzip des *Directed Self-Learning* (DSL, [3]) und kann so mit wenig Vorwissen erlernen, den Prozess zu regeln (rechter Zweig in Abbildung 2). Der zweite Regler basiert auf indirektem Lernen nach dem MPC-Prinzip (linker Zweig in Abbildung 2). Das Modell für den prädiktiven Regler wird hier allerdings parallel am Prozess erlernt und generiert ebenfalls ein Vertrauensmaß v_m , das angibt, wie sicher das Modell das lokale Verhalten des Prozesses beschreibt. Mit diesem online erstellten Modell kann der Regler quasi offline trainiert werden, indem am Modell weitere Lernstimuli erzeugt werden. Außerdem kann eine Folge y' der Regelgröße für eine Stellgrößenfolge u' mit dem Modell vorhergesagt werden, sodass die aktuell beste Stellgröße ausgewählt wird. Eine Beschreibung der Optimierung wird in Kapitel 3.4 gegeben.

Durch das Lernen wird bei den Reglern nicht nur die Stellgrößenvorgabe durch

$\Delta u_{1,2}$ angepasst. Es wird auch das Vertrauensmaß, das ein Regler ausgibt, durch $\Delta v_{1,2}$ angepasst. Das Vertrauensmaß v_1 der modellbasierten Regelung hängt von der Genauigkeit, mit der die Zielvorgabe in der Simulation erreicht werden konnte, und dem Vertrauensmaß des Modells ab. Beim DSL hängt das Vertrauensmaß indirekt von dem Lernerfolg ab und wird aufgrund des aktuellen Lernschrittes angepasst.

3.3 Integrationsmethoden

Der Entscheider muss die Stellaktionen, die von den verschiedenen Reglern ausgegeben werden, sinnvoll zu einer an den Prozess zu gebenden Stellaktion integrieren. Hierzu stützt er sich auf die zusätzlich generierten Vertrauensmaße v_1 und v_2 . Zwei einfache Varianten, diese Integration durchzuführen, sind die harte Auswahl der Vorgabe mit dem höchsten Vertrauensmaß und die Interpolation mit dem Vertrauensmaß als Gewichtung.

Bei der harten Auswahl wird die Stellgröße ausgewählt, die das höchste Vertrauensmaß liefert. Somit wird hart zwischen den verschiedenen Lösungen umgeschaltet. Dies hat jedoch zur Folge, dass unterschiedliche Lösungsstrategien konkurrieren. Sofern sie zu unterschiedliche Ausgaben liefern, kann jedes Umschalten auf einen anderen Regler starke Stellgrößenänderungen zur Folge haben und im ungünstigsten Fall bewirken, dass die Lösungen sich temporär widersprechen und der Prozess nicht sauber geregelt wird. Hat man jedoch zwei Lösungen, die eine vergleichbare Strategie verfolgen und bei denen eine der beiden Lösungen in unterschiedlichen Bereichen besser ist, dann führt die harte Auswahl dazu, dass man immer die vertrauenswürdigeren Information verwendet und so insgesamt die bestmögliche Strategie verfolgt.

Da die zuvor genannte Methode immer die Lösung nur eines Reglers übernimmt, können Sprünge in der Stellgröße auftreten. Einen glatteren Stellgrößenverlauf erreicht die gewichtete Interpolation zwischen den Regelstrategien. Hierbei geht jede Stellgröße u_i eines Reglers mit ihrem Vertrauensmaß v_i anteilig in die Gesamtlösung nach (1) ein.

$$u = \frac{\sum_{i=1}^N v_i \cdot u_i}{\sum_{i=1}^N v_i} \quad (1)$$

Da die Vertrauensmaße gewichtet addiert werden, führt dies bei Prozessen, die sich stetig im Eingangsraum bewegen, auch in der Regel zu einer stetigen Änderung der Stellgröße. Vorsicht ist geboten, wenn für eine Situation überhaupt kein Regler ein $v_i > 0$ liefert. Das kann verhindert werden, indem man einen statischen Regler mit konstantem Vertrauensmaß parallel zu den lernfähigen Reglern verwendet, indem man die lernfähigen Regler vorinitialisiert und beim Lernen ausschließt, dass das Vertrauensmaß auf 0 fällt, oder indem man (wie im Demonstrationsbeispiel) einem Regler den Vorzug gibt.

Die Lösungen werden bei der Interpolationsmethode ineinander überblendet. Dies hat zur Folge, dass widersprechende Lösungen, die vergleichbar vertrauenswürdig sind, in der Interpolation zu einer Stellgröße führen können, die keiner der Regelstrategien entspricht. Das Regelverhalten kann dadurch kontraproduktiv beeinflusst werden. Ein Beispiel für einen solchen Fall ist ein inverses Pendel, das aus der

hängenden Position aufgeschwungen und dann balanciert werden soll. Zu Beginn sind die Lösungen, zur einen oder anderen Seite zu fahren, gleichermaßen legitim und würden in der Interpolation zum Stillstand führen, wodurch das Pendel nicht zum Schwingen angeregt wird.

Wie diese Integrationsmöglichkeiten zeigen, treten vor allem Probleme auf, wenn mehrere gleichwertige, aber unterschiedliche Lösungsstrategien vorliegen. Daher sollte versucht werden, die Problemstruktur so zu verändern, dass sie nur eine optimale Lösung zulässt. So werden die verschiedenen Regler zur gleichen Lösungsstrategie geleitet und eine Integration kann gefahrlos vorgenommen werden. Beim Pendelwagen-Beispiel ist dies möglich, indem eine Vorzugsrichtung angegeben wird, in die sich der Wagen bewegen soll. So wird das Aufschwingen zu einer eindeutig lösbarer Aufgabe und die Integration der Regler funktioniert.

3.4 Optimierung in der Online-Simulation

Die indirekte Adaption verwendet das online generierte Modell des Prozesses, um eine geeignete Regelung zu finden bzw. zu optimieren. Da das Modell online parallel zur Regelung erlernt wird, hängt es vom bisherigen Betrieb ab, in welchen Bereichen bereits Wissen über den Prozess gesammelt wurde. Der Prozess wird dabei bereits durch andere Regler, z. B. mittels direkter Adaption, geführt und lernt somit nur in dem Bereich des Eingangsraumes, in dem der Prozess sich bewegt, sodass keine zufällige Exploration stattfindet und eine größere Sicherheit gegeben ist. Durch die Extrapolation im Modell können nun gezielt weitere Lernvorgaben generiert werden und auch an den Grenzen zu nicht explorierten Bereichen bereits eine grobe Vorgabe des Regelverhaltens vorgelernt werden. Da das Modell bereits sicher Wissen über den Prozess im letzten Zeitschritt gesammelt hat und bei stetigen Prozessen die aktuelle Position im Eingangsraum in der Nähe der letzten liegt, kann man die lokal extrapolierenden Eigenschaften des Modells ausnutzen und die aktuellen Position als Szenario für eine Simulation verwenden. Durch das Modell kann zudem lokal vorhergesagt werden, wie sich der Prozess in der Zukunft verhalten wird, und man kann so den Regler über mehrere Zeitschritte vorausoptimieren. Diese Optimierung über zukünftige Stellgrößen wird im Folgenden detaillierter beschrieben.

Für ein gegebenes Szenario muss also eine Stellgrößenfolge $U = \{u_1^1, \dots, u_1^{N_h}\}$ bis zum Prädiktionshorizont N_h optimiert werden. Hierzu wird die Auswirkung der Stellgrößen mit dem Modell voraussimuliert und das resultierende Verhalten für eine Bewertung der Stellgrößenfolge verwendet. Dabei gilt zu beachten, dass das Modell nicht nur eine reine Ein-Ausgabe-Relation liefert, die die Bewertung ermöglicht, sondern zudem eine ausreichend valide Beschreibung des Verhaltens des Prozesses vorhersagen muss, damit eine sinnvolle Mehrschritt-Prädiktion durchgeführt werden kann.

Die Bewertungsfunktion kann dabei unterschiedliche Anforderungen an die Regelung beschreiben, um diese bei der Optimierung zu berücksichtigen. Die Folge von Stellgrößen wird nun so optimiert, dass sie die Bewertungsfunktion minimiert und sich somit der optimalen Stellgrößenfolge \hat{U} annähert. Diese Minimumssuche ist mit den üblichen Problemen der Optimierung behaftet, sodass bei einer ungünstigen Topologie des Prozesses ein Optimierungsalgorithmus in lokalen Minima hängen bleiben

kann und das globale Minimum nicht findet. Daher ist die Architektur modular gehalten, damit ein beliebiger Optimierungsalgorithmus angewendet werden kann, der mit den Eingaben an das Modell und der resultierenden Bewertung, also ohne Kenntnis über deren Ableitung oder weitere Größen, arbeiten kann. So kann für jedes Problem ein Optimierungsalgorithmus verwendet werden, der den jeweiligen Anforderungen an Genauigkeit und Geschwindigkeit genügt.

Um zu erreichen, dass das Optimierungsproblem nicht zu komplex wird, kann die Länge der Stellgrößenfolge zudem auf $N_u \leq N_h$ Zeitschritte beschränkt werden. Die Zahl der Schritte, über die optimiert wird, wird als Stellhorizont bezeichnet. Ab dem Stellhorizont wird für den Rest des Prädiktionshorizontes konstant die letzte Stellgröße beibehalten. So wird das Optimierungsproblem vereinfacht und das Regelverhalten kann dennoch weiter in die Zukunft vorhergesagt und bewertet werden.

Eine einfache Bewertungsfunktion $b(u)$ verwendet eine Linearkombination der Norm der Regelabweichung e und der zeitlichen Ableitung des Istwertes, die durch die Differenz zweier aufeinanderfolgender Messungen approximiert wird $\frac{dy}{dt} \approx \Delta y = y_t - y_{t-1}$, um die Stellgrößenfolge U zu bewerten. So wird einerseits der Fehler mit einem proportionalen Strafterm belegt, um den Prozess einzuregulieren und andererseits eine zu starke Dynamik der zu regelnden Größe vermieden. Zusätzlich werden Verletzungen von Systembeschränkungen B_i durch einen weiteren konstanten additiven Term C_i in die Bewertungsfunktion eingebracht. So können die Beschränkungen priorisiert werden, indem wichtigere Beschränkungen einen größeren additiven Strafterm erhalten. Für N_B Beschränkungen und einen Prädiktionshorizont N_h ergibt sich die Funktion somit zu (2), wobei w_e das Gewicht für den Regelfehler und w_d das Gewicht für die Dynamik der zu regelnden Größe angibt und $\delta(B_i, U)$ ist Eins, falls die i te Beschränkung durch die Anwendung von U nicht eingehalten wurde, sonst Null. Der hochgestellte Index U weist darauf hin, dass die Größe aus dem Modell durch die Mehrschritt-Prädiktion mit der Stellgrößenfolge U entstanden ist.

$$b(U) = w_e \cdot \sqrt{\sum_{i=1}^{N_h} (e_i^U)^2} + w_d \cdot \sqrt{\sum_{i=1}^{N_h} (\Delta y_i^U)^2} + \sum_{i=1}^{N_B} \delta(B_i, U) \cdot C_i \quad (2)$$

Die Optimierung der Stellgrößenfolge reduziert sich durch die Bewertungsfunktion auf eine Minimierung. Es kann modular ein beliebiger Minimierungsalgorithmus eingesetzt werden. Beispielsweise wird die Minimierung durch einen *Random Walk* erreicht, der die Gefahr in lokalen Minima hängen zu bleiben verringert. Der Random Walk führt über N_w Schritte eine lokal zufällige Suche nach dem Minimum durch, wobei ein größerer Wert für N_w die Laufzeit erhöht, aber auch den Suchraum für das Optimum genauer durchsucht. Dabei wird in jedem Schritt die aktuelle Approximation der Position des Minimums im Suchraum zufällig durch eine gleichverteilte Zufallsvariable innerhalb einer Schrittweite $\pm d_{rw}$ variiert, sodass der Suchraum stochastisch in dieser lokalen Umgebung abgesucht wird. Sofern der neue Punkt im Suchraum einen kleineren Wert für die Bewertungsfunktion aufweist als der bisherige, wird dieser als Schätzung für das Minimum übernommen. Andernfalls wird er verworfen und ein neuer zufälliger Schritt ausgewählt. Durch eine große Schrittweite kann dieses Verfahren zu einer global zufälligen Suche führen, die bei einer ausreichend großen Zahl von Schritten mit größerer Wahrscheinlichkeit das globale Minimum findet.

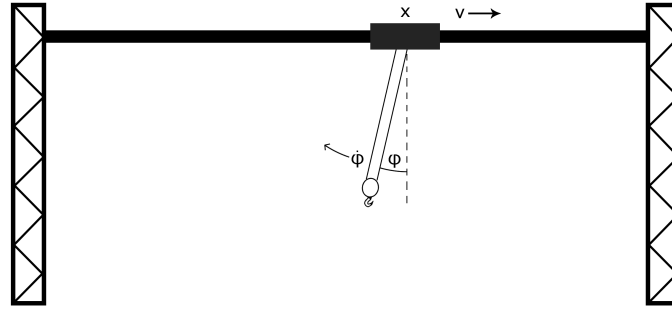


Abbildung 3: Schematische Darstellung des untersuchten Kranszenarios.

4 Untersuchungen

4.1 Testszenario

Um die Eigenschaften der IRIS-Architektur zu demonstrieren, wird nachfolgend ein erster Funktionstest vorgestellt. In dieser Untersuchung wird ein Kranwagen simuliert (siehe Abbildung 3). Der Wagen befindet sich an einer Position x und bewege sich mit einer Geschwindigkeit v . Unter dem Wagen hängt eine Last, die mit einem Winkel φ ausgelenkt ist und sich mit einer Winkelgeschwindigkeit von $\dot{\varphi}$ bewegt. Die Aufgabe der Regelung ist es, das Ziel möglichst schnell anzufahren. Das Sicherheitskriterium ist, dass der Kran dabei nicht überschwingen darf.

Als eine einfache Näherung zur Beschreibung dieses Systems werden folgende zeitdiskrete Differenzgleichungen verwendet:

$$\begin{aligned}
 x_{t+\Delta t} &= x_t + v_t \cdot \Delta t \\
 v_{t+\Delta t} &= v_t + 100 \cdot u \cdot \Delta t - \beta_x \cdot v_t - \text{sgn}(v_t) \cdot \gamma_x \cdot v_t^2 \\
 \varphi_{t+\Delta t} &= \varphi_t + \dot{\varphi}_t \cdot \Delta t \\
 \dot{\varphi}_{t+\Delta t} &= \dot{\varphi}_t - 9.81 \cdot \Delta t \cdot \sin(\varphi_t) + 0.1 \cdot u \cdot \cos(\varphi_t) - \beta \cdot \dot{\varphi}_t - \text{sgn}(\dot{\varphi}_t) \cdot \gamma \cdot \dot{\varphi}_t^2
 \end{aligned}$$

Hierbei gibt β den linearen und γ den quadratischen Reibungskoeffizienten der Aufhängung der Last und β_x sowie γ_x die Äquivalente für die Bewegung des Wagens an. Die Stellgröße u_t ist die Beschleunigung, durch die die Geschwindigkeit des Wagens beeinflusst werden kann, um die Last zu einer Zielposition zu führen. Diese Stellgröße ist auf $u_t \in [-1, 1]$ beschränkt. Der Zeitschritt wird für die folgenden Experimente zu $\Delta t = 0.01$ und die Reibungskoeffizienten zu $\beta = 10^{-3}$, $\gamma = 10^{-4}$, $\beta_x = 10^{-3}$ und $\gamma_x = 10^{-4}$ gewählt.

Hier wird der Wagen exemplarisch aus einer Startposition $x_0 = 0.0$ zu einer Zielposition $x_z = 5.0$ gefahren und startet mit selbsteinstellenden Teilreglern. Das bedeutet, dass jeder Teilregler global mit einer Stellgröße von 0 vorinitialisiert ist. Der Ablauf wird in vier aufeinanderfolgenden Zyklen von je 500 Zeitschritten wiederholt, um die Verbesserung der lernfähigen Regler zu betrachten, die vom ersten Zyklus an mitlernen. Dabei wird die Architektur aus Abbildung 2 verwendet.

4.2 Teilregler

Für diese Untersuchungen wird aus Gründen der Anschaulichkeit auf den eigentlich sinnvollen „Fallback“-Regler, also einen manuell eingestellten, statischen Regler, verzichtet. Dadurch lassen sich die Effekte der lernfähigen Regler besser verstehen. Die lernfähigen Regler werden sowohl für das DSL als auch für die prädiktive Regelung durch einen lokal linearen Funktionsapproximator auf Basis eines Takagi-Sugeno-Fuzzy-Systems 0. Ordnung beschrieben. Initialisiert sind diese mit einer Stellgröße $u = 0$. Die Eingangsgrößen sind die Abweichung vom Zielwert $x_t - x_z$, die aktuelle Geschwindigkeit \dot{x}_t , der Winkel ϕ_t und die Winkelgeschwindigkeit $\dot{\phi}_t$ gegeben. Daraus ermittelt der Funktionsapproximator jeweils die anzuwendende Stellgröße $u_{1,t}$ bzw. $u_{2,t}$ und die zugehörigen Vertrauensmaße $v_{1,t}$ bzw. $v_{2,t}$. Zu Beginn haben beide Teilregler ein Vertrauensmaß von 0. Es ist dann fest eingestellt, dass die Stellgröße des direkt lernenden Teilreglers verwendet wird.

Für die Optimierung durch Prädiktion wird ein Prädiktionshorizont von $N_h = 30$ und ein Stellhorizont $N_u = 5$ verwendet, um die Auswirkungen weit genug vorhersehen zu können, aber die Optimierung handhabbar zu halten. Der Random Walk sucht mit $N_w = 200$ Schritten der Schrittweite $\pm d_{rw} = 0.1$ nach dem Optimum, wobei die Bewertungsfunktion (2) mit $w_e = 1$ und $w_d = 0$ nur die Abweichung vom Zielwert berücksichtigt. Ein Überschwingen über die Zielposition wird mit $C_1 = 100$ bestraft. Für die so erlernte Stellaktion wird das Vertrauensmaß nach

$$v_1 = \max \left(0, \frac{D_{max} - |x_z - x_e|}{D_{max}} \right) \cdot \prod_{i=1}^{N_h} v_{m,i} \quad (3)$$

ermittelt. Die Funktion ist vergleichbar mit (4), wobei hier der Fehler in der finalen Position x_e , in der die Prädiktion endet, verwendet wird, um zu das Ergebnis der Regelung zu bewerten, und kein zusätzlicher additiver Term verwendet werden muss, da direkt die verwendete Stellgröße bewertet werden kann. Zudem wird das Vertrauensmaß durch die Sicherheit des Modells beeinflusst, die durch das Produkt der Vertrauensmaße der einzelnen Prädiktionsschritte $v_{m,i}$ gegeben ist. Somit ist für diesen Regler das Vertrauen dadurch gegeben, wie nah die Regelung an die Zielposition gelangt und wie sicher die Vorhersage des Modells ist.

Die direkte Adaption verwendet als *Lerner* (siehe Abbildung 2) zur Anpassung des Reglers eine Aktualisierung nach der Vorschrift

$$\Delta u_2 = (x_z - x_t - \chi_v \cdot \dot{x}_t) \cdot \chi_u$$

mit den Proportionalitätsfaktoren $\chi_v = 0.2$ für die Restgeschwindigkeit und $\chi_u = 0.01$ für die Adaptionstärke. Die Anpassung ist damit von der Regelabweichung und einer dazu proportionalen Restgeschwindigkeit abhängig und kann durch die Adaptionstärke χ_u eingestellt werden. Einer Abweichung vom Ziel wird also nur dann entgegengewirkt, wenn die Geschwindigkeit nicht ausreicht, oder aber es wird schon im Voraus abgebremst, falls die Geschwindigkeit zu hoch ist. Die durch Δu_2 geänderte Stellgröße wird mit einem Lernfaktor von $\lambda = 0.9$ in den Funktionsapproximator, also zu 90%, übernommen und zu 10% die bisherige Stellgröße beibehalten, um im Lernvorgang nicht auf einzelne Lernvorgaben zu stark zu reagieren.

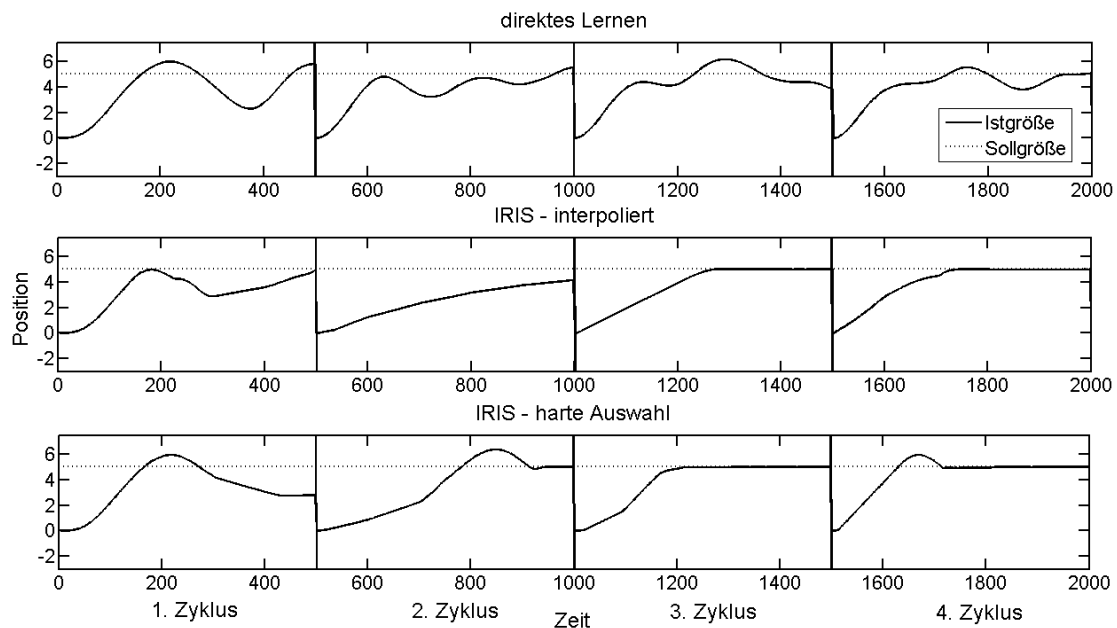


Abbildung 4: Regelverhalten der verschiedenen Varianten.

Directed Self-Learning	1. Zyklus	2. Zyklus	3. Zyklus	4. Zyklus
mittlerer Regelfehler	1.8053	1.2497	1.2004	1.0768
Integral Überschwingen	98.5099	13.1251	98.9705	27.9147
IRIS (Interpolation)				
mittlerer Regelfehler	1.7985	2.5135	1.2999	1.0125
Integral Überschwingen	0	0	0.4265	0
IRIS (harte Auswahl)				
mittlerer Regelfehler	1.9193	2.0829	1.1444	0.81807
Integral Überschwingen	65.379	114.8766	0	51.4982

Tabelle 2: Kennzahlen des Regelverhaltens bei den verschiedenen Lerntypen.

Das Vertrauensmaß, das für die so veränderte Stellaktion angegeben wird, ermittelt sich nach (4), wobei $D_{max} = 4$ die maximale zulässige Abweichung vom Zielwert angibt, ab der das Vertrauensmaß auf Null gesetzt wird. Auf diese Weise ergibt sich das Vertrauensmaß des Reglers danach, wie nah die Position an den Zielwert gebracht werden kann. Eine zusätzliche Erhöhung des Vertrauens um $\delta v = 0.1$ spiegelt die erwartete Verbesserung der Regelung durch den aktuellen Lernschritt wider.

$$v_1 = \min \left(1, \max \left(0, \frac{D_{max} - |x_z - x_i|}{D_{max}} \right) + \delta v \right) \quad (4)$$

4.3 Ergebnis

Den Verlauf von Soll- und Istgröße zeigt Abbildung 4 für die drei Szenarien. Die Kennzahlen des Regelverhaltens sind in Tabelle 2 zu sehen. Dabei ist zu erkennen, dass die beiden IRIS Varianten den mittleren Regelfehler nicht wesentlich verringern,

dafür aber versuchen, das Überschwingen zu vermeiden. Beim IRIS-Ansatz mit Interpolation kann ein Überschwingen nahezu vollständig verhindert werden, da die Vorausschau es ermöglicht, den Wagen früh genug abzubremsen, sofern das Modell das zukünftige Verhalten genau genug vorhersagt. Das Wissen der Prädiktion wird bei einer harten Auswahl des verwendeten Reglers nicht optimal genutzt, da das Vertrauensmaß geringer ist, und so kann das Überschwingen nicht immer verhindert werden. Bei der Interpolation ist der Einfluss des prädiktiven Reglers groß genug, um das Überschwingen zu vermeiden. Über die einzelnen Zyklen wird das Regelverhalten besser, nur bei der harten Auswahl zeigt der Verlauf im vierten Zyklus ein verstärktes Überschwingen.

5 Diskussion

Der Fokus der vorgestellten Untersuchung ist die Kombination von zwei Paradigmen zur Selbstoptimierung. Die direkte Selbstoptimierung bietet dabei die Flexibilität, auch langfristig auf Veränderungen des unterlagerten Prozesses zu reagieren. Das liegt an der Modellfreiheit der direkten Selbstoptimierung. Sie ermöglicht die Adaption, solange grundlegende Monotoniezusammenhänge erhalten bleiben. Diese Monotoniezusammenhänge sind aber gewöhnlich auch Kernpunkte beim Systementwurf und daher zum einen bekannt und zum anderen nicht zeitvariant.

Ein weiterer Vorteil der direkten Selbstoptimierung ist die schnelle Selbsteinstellung, also das Lernen von Grund auf (*learning from scratch*). Solches Lernen ist für modellbasierte, adaptive Regler schwierig, da eine gute Adaption eine präzise Modellidentifikation erfordert. Diese Problematik umgeht die direkte Selbstoptimierung, indem sie nur einen lokal adaptierten Regler erlernt. Damit ist die Vorgehensweise aber auch relativ vorsichtig (es wird nicht extrapoliert), und in jedem bisher unbesuchten Bereich des Eingangsraum muss das Lernen bei Null beginnen. Zudem geschieht das Lernen und das Regeln rein reaktiv, vorausschauendes Vermeiden von sicherheitskritischen Grenzen ist nicht möglich. Genau das zeigt sich auch in den Untersuchungen. Die Selbsteinstellung durch DSL erfolgt zügig und zielgerichtet, schwingt aber über und verletzt damit das Sicherheitskriterium. Erst die Kombination mit indirekter Selbstoptimierung in der IRIS-Architektur erlaubt es, Sicherheitsgrenzen prädiktiv einzubeziehen.

Die indirekte Selbstoptimierung verwendet allgemeine Funktionsapproximatoren (z. B. spezielle Takagi-Sugeno Fuzzy-Systeme) zur Modellbildung. Solche Approximatoren sind flexibler als parametrisierte, physikalische Modelle (*white box*), sind allerdings nur dort gültig, wo schon Lerndaten vorliegen, also nur lokal. Ein lokal gültiges Modell erlaubt aber zumindest eine vertrauenswürdige Prädiktion für wenige Prädiktionsschritte, so dass Sicherheitsgrenzen leichter vermieden werden können. Genauso wichtig wie die Prädiktionsmöglichkeit ist, dass durch ein Modell mehr oder weniger unabhängig vom normalen Regelungsbetrieb Selbstsimulationen durchgeführt werden können. Damit kann die direkte Selbstoptimierung auch auf Bereiche des Eingangsraums ausgedehnt werden, in denen das System sich gerade nicht aufhält. Diese Fähigkeit der IRIS-Architektur wurde in dem Demonstrationsbeispiel jedoch noch nicht eingesetzt und ist damit Teil der zukünftigen Arbeiten.

Das mitgelernte Selbstmodell ist natürlich tendentiell ungenau oder sogar völlig

falsch in Regionen des Eingangsraumes, für die noch wenige oder gar keine Lern-
daten vorliegen. Dementsprechend unverlässlich ist das Wissen, das durch Selbst-
simulationen für diese ungenauen Regionen generiert wurde. Der richtige Umgang
mit solchen dynamischen, also im Betrieb veränderlichen Unsicherheiten ist daher
essentiell für die IRIS-Architektur.

Die Untersuchungen haben dabei gezeigt, dass dem Entscheidungsmodul bzw. dem
darin genutzten Integrationsprinzip eine zentrale Rolle zukommt. Eine harte Aus-
wahl des aktuell vertrauenswürdigsten Reglers kann zu Justierungsproblematiken
führen, so dass wie im Untersuchungsbeispiel die angenommene Vertrauenswürdigkeit
des (besseren) Reglers (der Selbstsimulation) minimal zu niedrig ist und damit
das System überschwingt. Die Interpolationsmethode kann diese Problematik re-
duzieren, so dass im Untersuchungsbeispiel kein Überschwingen auftritt. Allerdings
muss dazu sichergestellt sein, dass die interpolierten Regler keine fundamental un-
terschiedlichen Lösungsstrategien verfolgen.

Die volle Flexibilität der IRIS-Architektur konnte in den Untersuchungen jedoch
noch nicht abschließend gezeigt werden. So ist es durchaus zielführend, mit vertret-
barem Aufwand zur Entwurfszeit generierbares Wissen in die Architektur einzu-
binden. Dies kann in Form von nicht-optimalen statischen Reglern geschehen. Sie
werden mit einer manuell vorgegebenen Vertrauenswürdigkeit parallel zu den ad-
aptiven Modulen betrieben, um für die ersten Betriebsmomente oder bestimmte
Ausnahmesituationen ein vorhersehbares, sicheres Systemverhalten garantieren zu
können. Hinzu kommt die oben angesprochene Möglichkeit, weitere Lernvorgaben
zu generieren. Ebenso kann das Selbstmodell vorstrukturiert oder an einem Proto-
typen vortrainiert werden, um im Betrieb möglichst schnell ein global verlässliches
Modell zu erhalten. Die IRIS-Architektur ermöglicht es dem Entwickler also, die
Balance zwischen Entwurfsaufwand, Konvergenzdauer und Flexibilität entsprechend
der spezifischen Anforderungen und Randbedingungen des Entwicklungsprojektes zu
wählen. Zukünftige Arbeiten werden genau diese Gesichtspunkte vertiefen und er-
kunden, nach welchen Entwurfsrichtlinien die IRIS-Architektur in realen Systemen
eingesetzt werden kann. Ein Schwerpunkt wird dabei auf zeitvarianten Systemen
liegen.

6 Zusammenfassung

Selbstoptimierende Systeme ermöglichen es bei vielen Anwendungen, mit geringem
Entwurfsaufwand nichtlineare und zeitvariante Prozesse zu regeln. Direkte Selbst-
optimierung stellt dabei einen guten Kompromiss zwischen Flexibilität und der Eignung
zur Selbsteinstellung dar. Indirekte Selbstoptimierung bietet im Gegenzug die
Möglichkeit zur Prädiktion und damit zum leichteren Einhalten von Sicherheits-
beschränkungen. In dieser Arbeit wurde die IRIS-Architektur vorgestellt. Sie stellt
einen Ansatz dar, um zum einen die beiden Paradigmen der direkten und indirekten
Selbstoptimierung zu kombinieren. Zum anderen erlaubt die IRIS-Architektur,
gezielt mit den beim Betrieb von lernfähigen Reglern notwendig auftretenden Unsi-
cherheiten umzugehen. Dies ist essentiell für einen praktischen Einsatz, denn selbst-
optimierende Systeme zielen auf Einsatzszenarien ab, die sich der formalen Model-
lierung entziehen, und damit auch dem formalen Stabilitätsnachweis. Ein explizites

Unsicherheitsmanagement trägt also dazu bei, die nötige Vertrauenswürdigkeit von selbstoptimierenden Systemen sicherzustellen.

Literatur

- [1] Tolle, H.: Lernende Regelungen für die Automatisierungstechnik. In: *Wissensverarbeitung in der Automatisierungstechnik* (Krebs, V., Hg.), Bd. 897, S. 27–54. VDI Verlag. 1991.
- [2] Shao, S.: Fuzzy self-organizing controller and its application for dynamic processes. *Fuzzy Sets and Systems* 26 (1988) 2, S. 151–164.
- [3] Brockmann, W.: Online Machine Learning For Adaptive Control. In: *Workshop on Emerging Technologies and Factory Automation ETFA, CRL Publishing Ltd., London*, S. 190–195. 1992.
- [4] Pomares, H.; Rojas, I.; Gonzalez, J.; Damas, M.; Pino, B.; Prieto, A.: Online global learning in direct fuzzy controllers. *IEEE Transactions on Fuzzy Systems* 12 (2004) 2, S. 218–229.
- [5] Rosemann, N.; Brockmann, W.: Concept for Controlled Self-optimization in Online Learning Neuro-fuzzy Systems. In: *KI 2007: Advances in Artificial Intelligence, LNAI* (Hertzberg, J.; Beetz, M.; Englert, R., Hg.), Bd. 4667, S. 498–501. Springer, Berlin. 2007.
- [6] Leonessa, A.; Haddad, W.; Hayakawa, T.; Morel, Y.: Adaptive control for nonlinear uncertain systems with actuator amplitude and rate saturation constraints. *International Journal of Adaptive Control and Signal Processing* 23 (2009) 1, S. 73 – 96.
- [7] Dozal-Mejorada, E.; Thakker, P.; Ydstie, B.: Supervised adaptive predictive control using dual models. In: *Proc. 8th Int. Symp. Dynamics and Control of Process Systems*, S. 109 – 114. 2007.
- [8] Hill, J.; Ydstie, B.: Adaptive control with selective memory. *International Journal of Adaptive Control and Signal Processing* 18 (2004) 7, S. 571 – 587.
- [9] Bongard, J.; Zykov, V.; Lipson, H.: Resilient Machines Through Continuous Self-Modeling. *Science* 314 (2006) 5802, S. 1118–1121.
- [10] Farrell, J.; Polycarpou, M.: *Approximation Based Control: Unifying Neural, Fuzzy and Traditional Adaptive Approximation Approaches*. Wiley Interscience, Hoboken, New Jersey. 2006.
- [11] Camacho, E. F.; Bordons, C.: *Model Predictive Control*. Springer. ISBN 1852336943, 9781852336943. 2004.
- [12] Nino, J.; De Keyser, R.; Syafie, S.; Ionescu, C.; Struys, M.: EPSAC-controlled anesthesia with online gain adaptation. *International Journal of Adaptive Control and Signal Processing* 23 (2009) 5, S. 455 – 471.

Konzeptionierung eines Green-Supercomputer-Clusters zur Berechnung von Künstlichen Neuronalen Netzen mit MATLAB

S. Schäfer, U. Lehmann, J. Krone, M. Schneider, J. Brenig

Institut für Computer Science, Vision and Computational Intelligence,
Fachhochschule Südwestfalen, Frauenstuhweg 31, 58644 Iserlohn
Tel. (02371) 566-124, Fax -209

E-Mail: {Schaefer.S, Lehmann, Krone, MSchneider, Brenig}@fh-swf.de

Abstract

Es soll eine Software entwickelt werden, die die Erstellung einer beliebigen Anzahl an Künstlichen Neuronalen Netzen zur n-fach Kreuzvalidierung mit MATLAB erlaubt. Die Trainingsaufträge werden anschließend auf dem angeschlossenen Computer-Cluster automatisch verteilt, um eine maximale Auslastung des Clusters zu gewährleisten. Die Software soll es dem Anwender ermöglichen das Training und die Auswertung einer großen Anzahl von KNN effektiver von der eigenen Workstation aus vorzunehmen. Weiterhin soll sich die Software um ein effizientes Energiemanagement des Clusters kümmern. Der Aufwand zur datenbasierten Modellbildung soll durch diese Vorgehensweise unter zeitlichen und energetischen Gesichtspunkten optimiert werden.

1 Einleitung

Da das Training von Künstlichen Neuronalen Netzen (KNN) aufgrund einer Vielzahl von beeinflussenden Faktoren, wie z.B. Trainingsverfahren, Art und Anzahl der Trainingsdaten und der Architektur, viel Zeit in Anspruch nehmen kann, besonders auf normalen Bürocomputern, gibt es in den letzten Jahren vermehrt Bestrebungen in Wirtschaft und Forschung das Training von Künstlichen Neuronalen Netzen mit Hilfe von unterschiedlichen Parallelisierungstechniken zu beschleunigen.

Ein Ansatz ist dabei die Parallelisierung des Trainings eines einzelnen Künstlichen Neuronalen Netzes durch Aufsplitten der einzelnen Rechenschritte auf mehrere Prozessoren/Prozessorkerne. Des Weiteren werden neuerdings auch Berechnungen auf die Graphics Processing Unit (GPU) einer modernen Hochleistungs-Grafikkarte [1][2] ausgelagert. Jedoch gibt es bei dieser Parallelisierungsmethode den Nachteil, dass sich aufgrund des mathematischen Modells eines Künstlichen Neuronalen Netzes das Training nur sehr schwierig parallelisieren lässt, da zur Berechnung des folgenden Layers immer erst die Ergebnisse des aktuellen vorliegen müssen. Daher kann immer nur die Berechnung der Neuronen des aktuellen Layers zur parallelen Berechnung

aufgeteilt werden, was je nach Architektur des KNN nur zu einer teilweisen Auslastung der verfügbaren Prozessoren/Prozessorkerne führt.

Weite Verbreitung findet besonders die parallele Durchführung mehrerer unabhängiger Trainingsdurchläufe für ein und dasselbe Künstliche Neuronale Netz mit Hilfe eines Computer Clusters. Dabei werden jedem Knoten des Clusters je nach Anzahl der Prozessoren/Prozessorkerne ein oder mehrere KNN-Trainings zugewiesen, um diese zu berechnen. Nach der Beendigung wird das Ergebnis ausgegeben oder an einen zentralen Rechner zur Auswertung weiterzuleiten. Problematisch ist oft die Verteilung der Berechnungen an die einzelnen Worker des Clusters und die anschließende Auswertung der Ergebnisse. Diese Aufgaben werden bei kleinen Clustern oftmals von Hand vorgenommen. Wünschenswert wäre es an dieser Stelle natürlich eine einheitliche Schnittstelle nutzen zu können, welche die vorgenannten Aufgaben automatisiert durchführt und somit das Potential hat die Effizienz des Trainingsprozesses deutlich zu steigern.

Ein weiteres Problem beim Einsatz eines Computer-Clusters ist der hohe Energiebedarf. Beim Betrieb einer Vielzahl an Computern ist dieser nicht zu unterschätzen, besonders wenn die Worker des Clusters, trotz Leerlauf, nicht ausgeschaltet oder in den Ruhezustand versetzt werden, kann dies zu hohen Energiekosten führen. Daher wäre es im Hinblick auf die aktuelle Wirtschafts- und Klimalage wünschenswert bei der Erstellung einer einheitlichen Trainingsschnittstelle auch ein Energiemanagement für die Worker des Computer-Clusters zu integrieren. Mit dessen Hilfe könnten nicht benötigte Worker in einen Energiesparmodus geschickt werden und bei Bedarf wieder aufgeweckt werden, um Energie zu sparen.

Zusammenfassend können folgende Ziele für die Software, welche innerhalb dieses Projekts entwickelt wird, definiert werden:

- Intuitiv zu benutzende Befehlssyntax
- Automatisierte Verteilung der vorhanden Aufgaben auf dem Cluster
- Automatisierte Auswertung der berechneten Ergebnisse
- Individuelle und einfache Modifizierbarkeit von Netzarchitektur, Trainings-, Parallelisierungs- und Auswertungsparametern
- Funktionen zur Steigerung der Energieeffizienz

Um diese Ziele zu erreichen werden in diesem Projekt die Neural Network Toolbox, die Parallel-Computing-Toolbox, sowie der Distributed Computing Server der Mathematik-Software MATLAB zum Einsatz kommen. MATLAB bietet bei der Umsetzung der oben genannten Projektziele entscheidende Unterstützung:

1. Durch die weite Verbreitung von MATLAB in Industrie und Forschung ist oftmals nur noch der Zukauf der Parallel-Computing-Toolbox, sowie von Worker-Lizenzen für den Distributed Computing Server erforderlich
2. Plattformunabhängige Nutzung unter Linux, Mac OS X, Solaris 64, Unix, Windows [3]
3. Einfache und effiziente Einrichtung eines Computer Clusters

2 Zielarchitektur

Die Software zielt auf Computer-Cluster ab, welche mit dem MATLAB Distributed Computing Cluster erstellt und verwaltet werden. Dabei ist für die Funktionalität der Software unerheblich welche hardwaretechnische Basis den Workern des Clusters zugrunde liegt. Es muss lediglich sichergestellt sein, dass MATLAB auf der vorhandenen Hardware lauffähig ist und dass alle Computer über ein entsprechend performantes Netzwerk mit dem JobManager des Clusters verbunden sind. Der JobManager muss wiederum mit dem Client-Computer verbunden sein, auf welchem die Trainings-Aufträge erstellt und anschließend an den JobManager weitergeleitet werden, welcher diese wiederum auf den Cluster verteilt. Natürlich kann gegebenenfalls ein und derselbe Computer Client und JobManager sein, was bei kleinen Computer-Clustern sinnvoll sein kann. Bei großen Clustern könnte dies jedoch zu Performance-Problemen auf dem Client-/JobManager-Computer führen, da dieser eventuell mit Verwaltungs- beziehungsweise Auswertungsarbeiten vollständig ausgelastet ist.

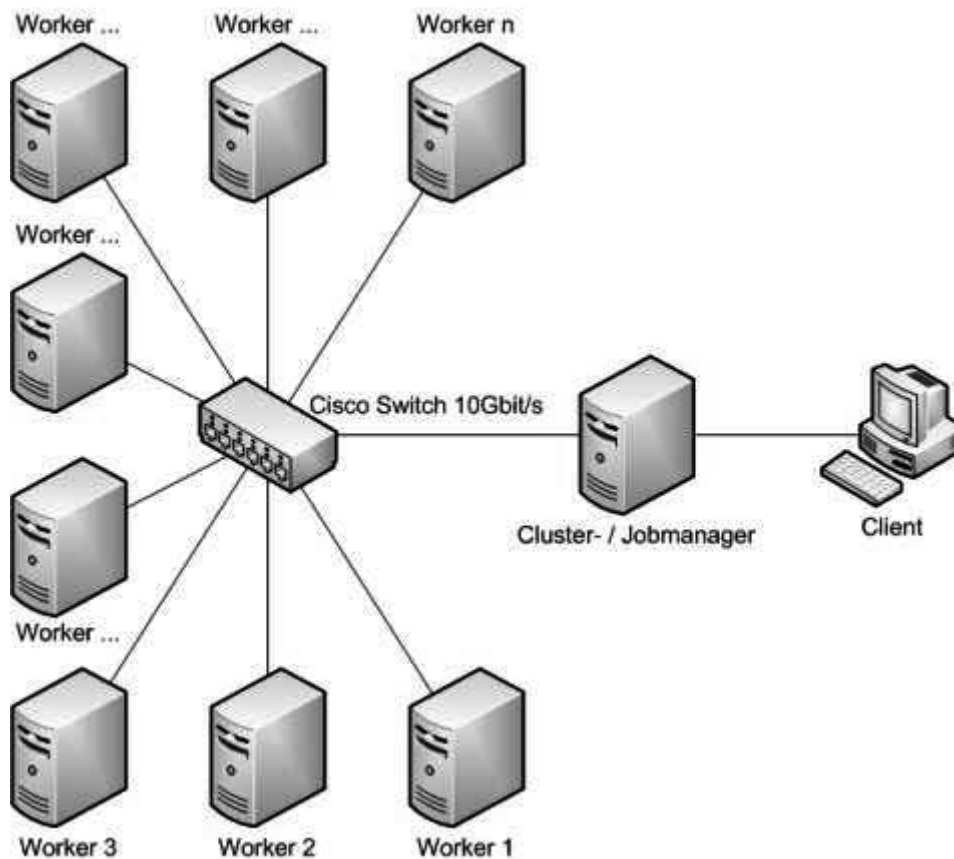


Abbildung 1: Netzplan eines mit der Software kompatiblen Computer-Clusters

3 Verteilungskonzept

Der Anwender legt zunächst die Architektur des zu trainierenden Künstlichen Neuronales Netzes fest. Er soll dabei auf eine große Anzahl Trainingsalgorithmen (z.B. Backpropagation, Levenberg-Marquardt, etc. [4]), Aktivierungsfunktionen (z.B. Identität, Tangens Hyperbolicus, Sigmoid, etc. [4]) und Trainingsparameter (z.B. Lernrate, Momentum, etc.) zurückgreifen können, um den Trainingsablauf so flexibel und individuell wie möglich gestalten zu können. Die Konfigurationsmöglichkeiten sollen dabei ähnlich mächtig wie bei der MATLAB Neural Network Toolbox ausfallen. Gleichzeitig werden dem Benutzer umfassende Möglichkeiten zur Verfügung stehen, um das Verhalten der Software bei der Verteilung der Trainingsaufträge auf den Computer-Cluster, der Funktionalität des Energiemanagements und der Auswertung aller Ergebnisse nach Abschluss aller Trainingsaufträge zu beeinflussen.

In den folgenden Abschnitten werden jeweils der geplante Workflow des Trainings und der Auswertung erläutert.

3.1 Workflow – Training

Nach Erhalt eines Trainingsauftrages wird der JobManager zuerst eine Aufwands- und Umfangsabschätzung durchführen, um die tatsächliche Menge der Rechenaufgaben und den voraussichtlichen Ressourcenverbrauch des Auftragspakets einzuschätzen. Daraufhin wird der Status des Computer-Clusters abgefragt, um zu ermitteln welche Ressourcen aktuell zur Verfügung stehen. Dabei wird die Anzahl der Worker und deren Hardware-Ressourcen, wie verfügbare CPU-Kerne und nutzbarer Hauptspeicher, evaluiert. Der für die Verteilung der Aufträge auf den Cluster zuständige Algorithmus wird sich danach, entsprechend der vorgegebenen Parameter, um die Verteilung der Aufträge auf die Worker des Clusters kümmern.

Dabei wird der Algorithmus zum Beispiel bei einem Bedarf an maximaler Trainingsgeschwindigkeit, jedem Worker Trainingsaufträge zuweisen, wobei bei manchen Workern nicht alle verfügbaren CPU-Kerne genutzt werden, da eine vollständige Auslastung aller CPU-Kerne nur bedingt höhere Leistung bedeutet. Bei Priorisierung der Energieeffizienz wird die Software versuchen, bei möglichst wenigen Workern alle CPU-Kerne maximal auszulasten und gleichzeitig die restlichen Worker in Schlafmodi zu schicken bzw. dort zu belassen.

3.2 Workflow – Auswertung

Nachdem ein Trainingsauftrag angestoßen und vom JobManager entsprechend auf dem Cluster verteilt wurde, wartet der JobManager auf die Ergebnisse aller Worker. Diese arbeiten die ihnen zugewiesenen Aufgaben nacheinander ab und liefern alle erreichten Ergebnisse an den JobManager zurück. Wenn alle Worker die Bearbeitung der ihnen gestellten Aufgaben abgeschlossen und damit alle Ergebnisse beim JobManager abgeliefert haben, werden diese anhand der vom Anwender eingestellten Parameter ausgewertet.

Es werden dabei aus allen trainierten KNN, diejenigen mit dem besten Trainingsergebnis ausgewählt und deren Initialisierung, Architektur, Lernkurven (Trainingsfehler, Validierungsfehler), sowie die verwendete Datenaufteilung gespeichert und anschließend an den Client zurückgeliefert und angezeigt. Anschließend werden, falls kein weiterer Trainingsauftrag ansteht, alle Worker in einen Schlafzustand versetzt, um Energie zu sparen.

4 Benutzerfreundlichkeit

Nach Fertigstellung der Software wird es möglich sein ein Künstliches Neuronales Netz mit einer n -fach Kreuzvalidierung anzulegen und daran anschließend einen Trainingszyklus mit einer beliebigen Anzahl an verschiedenen initialisierten Varianten dieses angelegten Netzes anzustoßen. Trotz der automatisierten Verteilung aller Trainingsaufträge soll der Benutzer ein Maximum an Kontrolle über alle Parameter des Trainingsablaufs behalten. Weiterhin soll die Software eine vergleichbare Konfigurationsvielfalt für KNN bieten wie die Neural Network Toolbox von MATLAB; z.B. die Auswahl des Trainingsalgorithmus, die Einstellung der Lernrate, die Wahl der Aktivierungsfunktionen, etc.. Die Kontrolle über die Aufteilung der Datensätze in Trainings- und Validierungsdaten, wie die anschließende Aufteilung auf dem Cluster, sowie die Auswertung der erreichten Ergebnisse, soll ebenfalls beim Benutzer der Software liegen. Diese Prozesse müssen selbstverständlich so transparent wie möglich ablaufen, weswegen es wichtig ist, dem Benutzer eine ausführliche Dokumentation der Software, sowie bei Bedarf ausführliche Statusmeldungen und Informationen während der Benutzung der Software zur Verfügung zu stellen. Daher wird während der Entwicklung der Software ein Schwerpunkt auf die oben genannten Punkte gelegt, um sie so benutzerfreundlich wie möglich zu gestalten und es dem Benutzer so zu ermöglichen, schnell die gewünschten Ergebnisse zu erzielen.

5 Zusammenfassung

Die in diesem Beitrag beschriebene Software soll das Training von Künstlichen Neuronalen Netzen auf einem Computer-Cluster bestehend aus n Workern, durch Verwendung der Software MATLAB und deren Erweiterungen Neural Network Toolbox, Parallel Computing Toolbox, sowie Distributed Computing Server, deutlich komfortabler und effektiver gestalten als es heute bei zumeist kleinen Computer-Clustern der Fall ist. Durch Verwendung der Software wird es dem Benutzer ermöglicht, einen Trainingszyklus auf einem Computer-Cluster von seiner eigenen Workstation aus zu starten.

Einer der Hauptpunkte, der bei der Entwicklung der Software besondere Beachtung finden wird, ist, dem Anwender ein Maximum an Kontrolle über alle Softwareabläufe während des Trainingsvorgangs zu bieten. Dem Anwender soll damit volle Kontrolle über den Verlauf des Trainings ermöglicht werden. Es soll ihm ermöglicht werden, alle Parameter des KNN, die Strategie des Algorithmus zur Verteilung der Trainingsaufgaben auf den Cluster, sowie den Auswertungsalgorithmus detailliert den eigenen Bedürfnissen anzupassen. Der Algorithmus, welcher die Verteilung der

Trainingsaufgaben übernimmt, wird um das automatisierte Aktivieren einer Stromsparfunktion für nicht benötigte Worker erweitert, wodurch ein effizientes Energiemanagement im Cluster möglich wird.

6 Ausblick

Nach Abschluss der Konzeptionsphase wird mit der Entwicklung der in diesem Beitrag beschriebenen Software und deren Algorithmen am Institut für Computer Science, - Vision and Computational Intelligence begonnen. Anschließend soll die in diesem Projekt erstellte Software am Institut CV&CI zum Einsatz kommen, stetig weiterentwickelt und um neue Funktionalitäten erweitert werden.

Eine der bereits angedachten Erweiterungen ist die zusätzliche Verwendung von Hochleistungsgrafikkarten auf den Workern zur Beschleunigung des Trainings von KNN. Der Vorteil bei dem Einsatz von Hochleistungsgrafikkarten ist unter anderem die hohe Energieeffizienz durch eine, verglichen mit herkömmlichen Mehrkern-CPU's, enorm hohe Rechenleistung pro Watt [5].

7 Danksagung

Diese Arbeit entstand am Institut CV&CI der Fachhochschule Südwestfalen im Rahmen des vom Bundesministerium für Bildung und Forschung geförderten Projekts „Neuroadaptiver Bauplatz im Flugzeugbau“.

8 Literatur

- [1] Lehmann, U.: *Unmenschliches Wissen*. fh16 – Lebendige Forschung an Fachhochschulen in NRW, Köln; Journal 2009.
- [2] Lahabar, Sheetal; et. Al.: High Performance Pattern Recognition on GPU. Internet Publikation: <http://cvit.iiit.ac.in/papers/Sheetal08High.pdf>. Stand: 18.06.2008.
- [3] The MathWorks, Inc. (Hrsg.): *MATLAB – Requirements*. Internet-Publikation: <http://www.mathworks.com/products/matlab/requirements.html>. Stand: 23.09.2009.
- [4] Zell, Andreas: *Simulation neuronaler Netze*. 4., unveränderte Auflage. München: Oldenbourg Wissenschaftsverlag GmbH 2003.
- [5] Dipl.-Inform. Sven Turck, Dipl.-Inform. Sebastian Schäfer, cand. Inform. Christian Löwen: *NVIDIA CUDA - NVIDIA's neue Parallel Computing Architektur*. SIHK Bildungszentrum, Hagen, Eugen-Richter-Str. 110 im Juni 2009.

KnowledgeBase-System aus CI-basierten Prognosemodellen zur Machbarkeitsprüfung von Blechumformprozessen

S. Turck^{1,2}, M. Schneider¹, U. Lehmann¹, J. Krone¹, Ch. Klimmek²

¹Institut für Computer Science, Vision und Computational Intelligence
Fachhochschule Südwestfalen, Frauenstuhlgweg 31, 58644 Iserlohn
Tel. (02371) 566-303 Fax (02371) 566-209
E-Mail: {mschneider, lehmann, krone}@fh-swf.de

²SimuForm GmbH

Zentrum für Produktionstechnologie, Carlo-Schmid-Allee 3, 44263 Dortmund
Tel. (0231) 398189-0 Fax (0231) 398189-99
E-Mail: {turck, klimmek}@simuform.de

1 Kurzfassung

Die SimuForm GmbH ist ein High-Tech Unternehmen mit Sitz in Dortmund. SimuForm entwickelt eine weltweit einzigartige, intelligente Software-Lösung für den Automotive-Sektor mit spezieller Ausrichtung auf die Prozesskette Blech. Das Ziel dieser neuartigen, prozessbegleitenden CAE-Applikationen (Computer Aided Engineering) ist die Unterstützung des Ingenieurs bei der virtuellen Produktentwicklung. Kern dieser Applikationen ist ein KnowledgeBase-System, welches erworbenes Prozesswissen im Unternehmen, wie zum Beispiel aus bereits durchgeführten Finite-Elemente-Simulationen, mit Hilfe von Data Mining und künstlichen neuronalen Netzen (KNN) aufbereitet. Die generierten Prognosemodelle werden zusammen mit einem digitalen Fingerabdruck der Geometrie in der KnowledgeBase abgelegt. Das gespeicherte Wissen kann anschließend auf neue, ähnliche Geometrien und Fertigungsprozesse angewendet werden. Mit Hilfe dieser innovativen Technologie kann der Produktentwickler eine Machbarkeitsprüfung komplexer Blechumformprozesse innerhalb von Sekunden durchführen. Dies stellt einen enormen Vorteil gegenüber einer vollständigen Simulation des Prozesses mittels der numerischen Finite-Elemente-Methode dar und steigert somit maßgeblich die Effizienz und die Wirtschaftlichkeit im Unternehmen.

Im Rahmen einer Kooperation zwischen der SimuForm GmbH und dem Institut für Computer Science, Vision und Computational Intelligence der Fachhochschule Südwestfalen wurden die CI-Methoden zur Generierung der Prognosemodelle weiterentwickelt, um eine bessere Performance zu erzielen. Primär wurden die statistische Datenaufbereitung des Prozesswissens sowie das Verfahren zum maschinellen Lernen erweitert. Verglichen mit den zuvor erzielten Ergebnissen konnten die Generalisierungsleistung der künstlichen neuronalen Netze verbessert und die Trainingszeiten deutlich reduziert werden.

2 Umformtechnik

In der DIN 8580 wird der Begriff Umformen folgendermaßen definiert: Fertigen durch bildsames (plastisches) Ändern der Form eines festen Körpers; dabei werden sowohl die Masse als auch der Zusammenhalt beibehalten [1]. Das Umformen selbst kann in Abhängigkeit der Beanspruchungsart des umgeformten Bauteils gemäß DIN 8582 in mehrere Gruppen unterteilt werden. Diese weisen wiederum jeweils mehrere Untergruppen auf. Die Abbildung 2-1 gibt einen Überblick über alle Gruppen und Untergruppen des Fertigungsverfahrens Umformen. Für die CAE-Applikationen der SimuForm GmbH ist insbesondere das Tiefziehen von Bedeutung, welches nachfolgend kurz betrachtet wird.

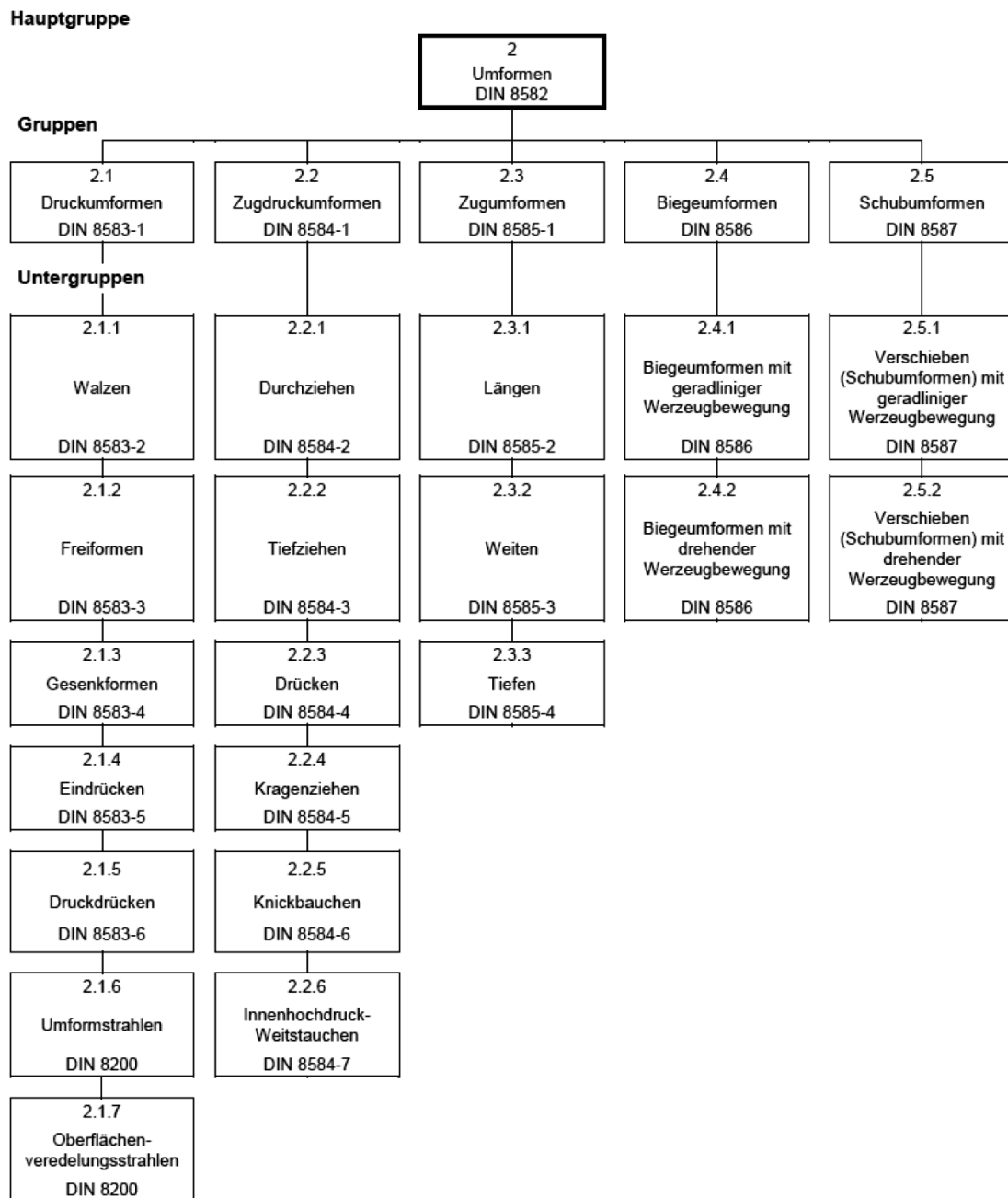


Abbildung 2-1: Übersicht über alle Gruppen und Untergruppen des Fertigungsverfahrens Umformen nach DIN 8580 [1]

In der deutschen Norm wird Tiefziehen definiert als das Umformen eines Blechzuschnittes in einen einseitig offenen Hohlkörper (Tiefziehen im Erstzug) oder eines vorgezogenen Hohlkörpers zu einem Hohlkörper mit geringerem Umfang ohne beabsichtigte Veränderung der Blechdicke (Tiefziehen im Weiterzug). Der Tiefziehprozess kann dabei sowohl mit einem Werkzeug, einem Wirkmedium oder einer Wirkenergie erfolgen. In Abbildung 2-2 ist zur Verdeutlichung eines solchen Prozesses das Tiefziehen im Erstzug dargestellt.

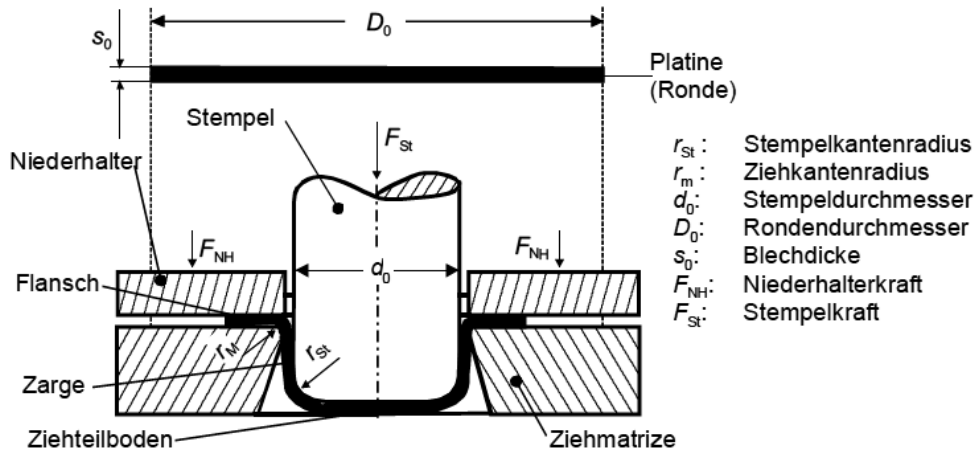


Abbildung 2-2: Tiefziehen im Erstzug [2]

Generell lässt sich der Tiefziehprozess in zwei Phasen einteilen. In der ersten Phase, welche auch als Streckziehen bezeichnet wird, formt der Ziehstempel den Ziehteilboden aus. Dabei kommt es insbesondere an dem Stempelkantenradius und dem Ziehkantenradius zu einer Reduktion der Blechdicke, in der Umformtechnik auch Abstreckung genannt [2]. Die zweite Phase stellt den eigentlichen Tiefziehvorgang dar und wird eingeleitet, sobald die Stempelkraft die Flanscheinzugskraft überschreitet. Der Werkstoff fließt dabei aus dem Flansch in die Zarge des Ziehteils [2]. Während dieser beiden Phasen wird das Ziehteil mit unterschiedlichen Kräften beansprucht, wie in Abbildung 2-3 zu sehen ist.

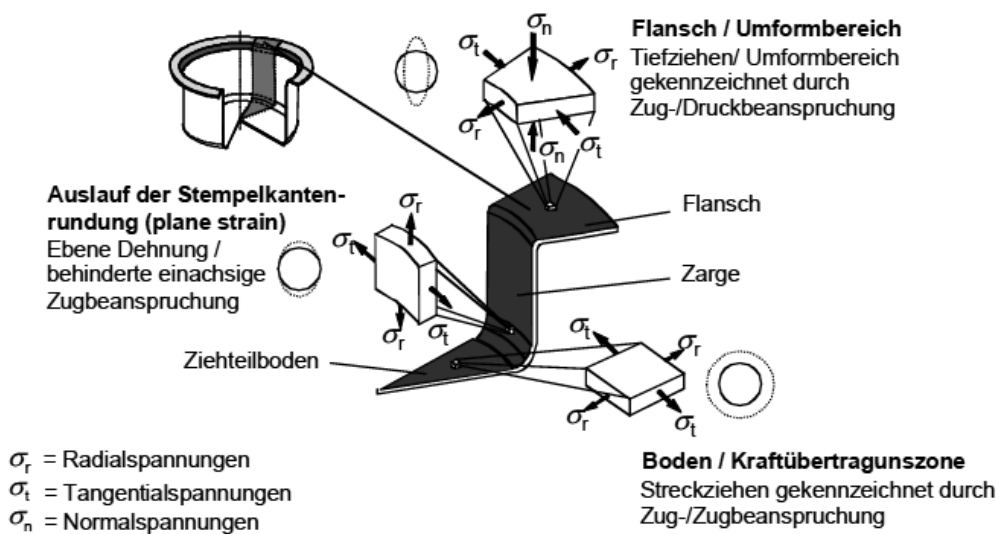


Abbildung 2-3: Beanspruchungen beim Tiefziehen am Beispiel eines rotationssymmetrischen Bauteils [2]

3 KnowledgeBase-System

Mit Hilfe der neuartigen Technologie lässt sich das bestehende Know-How in einem Unternehmen frühzeitig für die Produktentwicklung nutzen. Dadurch lassen sich Konzeptionsfehler bereits in der Anfangsphase identifizieren und Entwicklungen kosteneffizient in die optimale Richtung lenken. Kern dieser Softwareprodukte ist ein intelligentes KnowledgeBase-System, welches das Wissen aus vergangenen Umformprozessen vollkommen automatisiert aufbereitet und in Prognosemodelle für ähnliche Prozesse überführt. Zu diesem Zweck werden die Ergebnisse aus bereits durchgeführten Finite-Elemente-Simulationen [3] extrahiert und mit den Informationen der Bauteilgeometrie und des Werkstoffs verknüpft [4]. In Anbetracht der unterschiedlichen Dimensionierung der einzelnen Größen werden diese mittels einer Min-Max-Normalisierung skaliert. Zusätzlich werden während der Vorverarbeitung der Daten automatisch umformkritische Zonen der Geometrie identifiziert. Mit Hilfe dieser Datensätze wird anschließend das Prognosemodell antrainiert. Dabei handelt es sich um ein vollständig verbundenes, mehrschichtiges Feed-Forward-Netz. Als Trainingsverfahren wird eine Eigenimplementierung auf Basis des Backpropagation-Algorithmus [5] verwendet. Nach dem erfolgreichen Training wird das künstliche neuronale Netz zusammen mit einem digitalen Fingerabdruck der Bauteilgeometrie in der zentralen KnowledgeBase gespeichert.

Für die Machbarkeitsprüfung eines neuen, ähnlichen Fertigungsprozesses wird lediglich die Geometrie des Bauteils und die Auswahl des Materials benötigt. Es ist keine Diskretisierung und Vernetzung oder Prozessmodellierung im Sinne einer FEM erforderlich. SimuForm garantiert daher Laufzeiten von maximal 60 Sekunden für die Anwendung der Prognosemodelle [6]. Stattdessen greift die Software für die Machbarkeitsprüfung auf das gespeicherte Prozesswissen der KnowledgeBase zurück. Die Auswahl des Prognosemodells erfolgt dabei anhand eines Abgleichs der Fingerabdrücke der Geometrie. Die Prognose kann die vollständige Simulation eines Prozesses mit der Finite-Elemente-Methode nicht ersetzen, gibt jedoch innerhalb von Sekunden Aufschluss über die zu erwartende Verteilung der Blechdicke und die auftretenden Beanspruchungen innerhalb des Bauteils sowie mögliches Auftreten von Bauteilversagen. Dadurch lassen sich frühzeitig die vielversprechendsten Konzepte während der Produktentwicklung identifizieren.

4 Optimierung der Prognosemodelle

Im Rahmen eines kooperativen Projektes zwischen der SimuForm GmbH und dem Institut für Computer Science, Vision und Computational Intelligence der Fachhochschule Südwestfalen wurde ein Kreuzvalidierungsverfahren [7] in den Trainingsprozess der künstlichen neuronalen Netze integriert. Dieses ermöglicht eine bessere Abschätzung der Generalisierungsleistung und steigert die Robustheit des Prognosemodells gegenüber Ausreißern in den Datensätzen.

Ein großer Teil der Optimierungsarbeiten während des Kooperationsprojektes wurde in die Implementierung des Trainingsalgorithmus investiert. Eine Analyse des Quellcodes der bisherigen Lösung zeigte, dass die Umsetzung des Backpropagation-Algorithmus in drei verschachtelten for-Schleifen über die einzelnen Layer, Neuronen und Gewichte

iteriert und kaum hinsichtlich der Laufzeit optimiert ist. Dabei eignen sich gerade vollständig verbundene Feed-Forward-Netze zur Verwendung von Vektor- und Matrixoperationen, welche einen Großteil der Berechnungen in klassischen HPC-Anwendungen (High Performance Computing) ausmachen und dementsprechend extrem optimiert sind. Im Rahmen des Projektes wurde eine Neuentwicklung des Online-Backpropagation-Algorithmus auf Basis hochperformanter Vektor- und Matrixoperationen [8] vorgenommen. Wie deutlich anhand von Abbildung 4-1 und Abbildung 4-2 zu erkennen ist, konnte die Trainingszeit pro Epoche durch den Einsatz der optimierten mathematischen Funktionen um ein Vielfaches reduziert werden.

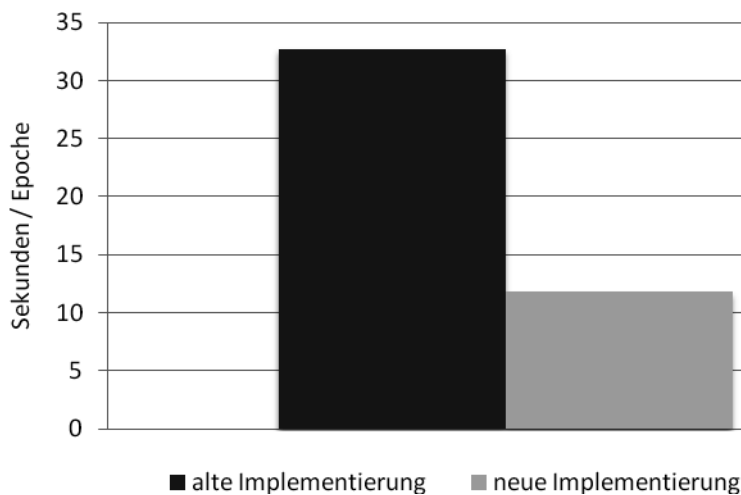


Abbildung 4-1: Gegenüberstellung der Laufzeit in Sekunden pro Epoche für das Training einfacher Wannen-Geometrien mit ca. 1,5 Millionen Datensätzen

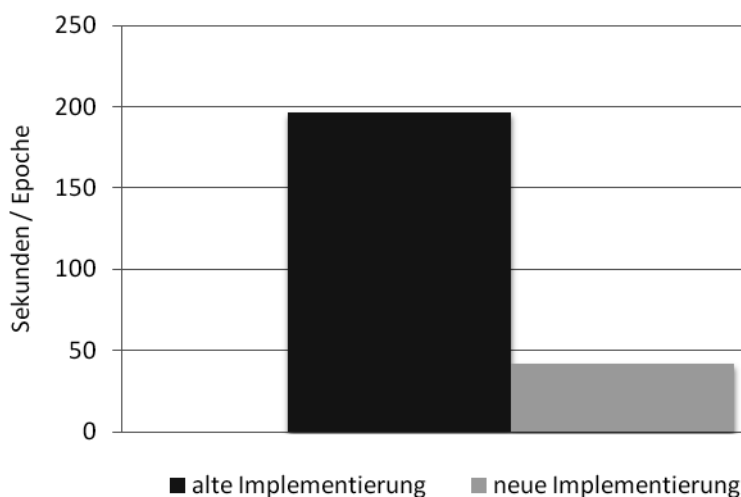


Abbildung 4-2: Gegenüberstellung der Laufzeit in Sekunden pro Epoche für das Training realer Karosseriebauteile mit ca. 5,6 Millionen Datensätzen

5 Ausblick

Aufgrund der deutlich gesteigerten Performance in Bezug auf die Trainingsdauer soll kurzfristig eine neue Versuchsreihe durchgeführt werden. Dabei sollen unterschiedliche Netztopologien und ihre Auswirkungen auf die Generalisierungsfähigkeit der künstlichen neuronalen Netze untersucht werden. Ein langfristiges Ziel bei der Fortführung dieses Projektes ist die Entwicklung eines adaptiven Verfahrens zur automatischen Bestimmung der Trainingsparameter und der Netztopologie. Basierend auf einer statistischen Analyse der Datenbeschaffenheit soll durch die adaptive Anpassung ein möglichst optimales Trainingsergebnis erzielt werden.

Danksagung

Diese Arbeit entstand in Kooperation zwischen der Fa. SimuForm GmbH und dem Institut für Computer Science, Vision und Computational Intelligence an der Fachhochschule Südwestfalen. Das Institut wird vom Ministerium für Innovation, Wissenschaft, Forschung und Technologie des Landes Nordrhein-Westfalen gefördert.

Literatur

- [1] DIN Deutsches Institut für Normung e.V. [Hrsg.]: *DIN 8580:2003-09. Fertigungsverfahren. Begriffe, Einteilungen*. Berlin: Beuth Verlag GmbH, 2003.
- [2] Doege, Eckart und Behrens, Bernd-Arno: *Handbuch Umformtechnik: Grundlagen, Technologien, Maschinen*. 1. Auflage. Berlin: Springer Verlag, 2006.
- [3] Wittel, Falk K.: *Eine kurze Einführung in die Finite Elemente Methode*. [Online] <http://www.ifb.ethz.ch/education/FEMeinfuehung/Skript>. Abgerufen am: 20.09.2009.
- [4] SimuForm GmbH [Hrsg.]: *SimuForm Perceptron*. [Online] <http://www.simuform.de/index.php?id=193&L=0>. Abgerufen am: 05.09.2009.
- [5] Zell, Andreas: *Simulation neuronaler Netze*. 3., unveränderter Nachdruck. München: Oldenbourg Wissenschaftsverlag GmbH, 2000.
- [6] SimuForm GmbH [Hrsg.]: *SimuForm Forecast*. [Online] <http://www.simuform.de/index.php?id=169&L=0>. Abgerufen am: 05.09.2009.
- [7] Reed, Russel D. and Marks II, Robert J.: *Neural Smithing. Supervised Learning in Feedforward Artificial Neural Networks*. Cambridge, MA: MIT Press, 1999.
- [8] Dongarra, Jack J. et al.: An extended set of FORTRAN basic linear algebra subprograms. In: *Transactions on Mathematical Software (TOMS), Volume 14, Issue 1*. New York, NY: ACM, 1988.

Automatisierter Entwurf von Hydraulikventilreglern: Eine Evolutionsstrategie für Optimierungsprobleme mit variabler Parameteranzahl

Jan Braun, Johannes Krettek, Frank Hoffmann, Torsten Bertram

Lehrstuhl für Regelungssystemtechnik, Technische Universität Dortmund

Otto-Hahn-Str. 4, 44221 Dortmund

Tel.: (0231) 755-2745

Fax: (0231) 755-2752

E-Mail: Jan.Braun@tu-dortmund.de

1 Einführung

Viele Optimierungsprobleme im Umfeld der Regelungstechnik, vor allem die Auslegung von Reglern sowie die Modellierung und Identifikation von Systemen lassen sich nur bedingt durch eine konstante Anzahl von Optimierungsparametern beschreiben. Bei der Auslegung eines optimalen Reglers bedeutet das, dass nicht nur die Reglerparameter optimal einzustellen sind, sondern auch die Reglerstruktur optimal zu wählen ist. Klassische Evolutionsstrategien sind darauf ausgelegt, die Lösung für eine a priori festgelegte Anzahl von Parametern in vorher definierten Grenzen zu optimieren. Dieser Beitrag stellt einen neuartigen Ansatz zur multikriteriellen evolutionären Optimierung von Regelungssystemen vor, deren Struktur nicht oder nur unvollständig bekannt ist und daher auf ein Optimierungsproblem mit variabler Anzahl von Lösungsparametern führt. Der Ansatz macht, nur allgemeine Vorgaben über die grundsätzlich zulässigen Strukturen, um so ausgehend von zunächst einfachen Grundformen im Laufe der Evolution komplexere Strukturen zu entwickeln. Die Komplexität des Systems, messbar an der Anzahl der benötigten Parameter, kann als ein zusätzliches Kriterium der im Allgemeinen multikriteriellen Optimierung verwendet werden, um die Generierung unnötig komplexer Systeme zu verhindern, und die resultierende pareto-optimale Lösungsmenge bietet dem Experten einen Kompromiss zwischen Komplexität und Leistungsfähigkeit zur Auswahl. So werden beim Systementwurf die Kosten höherer Komplexität gegen die Verbesserung der Funktionalität objektiv gegeneinander abgewogen.

Optimierungsprobleme mit variabler Struktur wurden bereits für zahlreiche Anwendungsgebiete entwickelt. Die Problemstellungen sind prinzipiell sehr vielfältig, die Beispiele in diesem Beitrag sind jedoch auf den Bereich der Regelungstechnik beschränkt. Bei der genetischen Programmierung sind die als Baum oder als Sequenz elementarer Operatoren kodierten Lösungen praktisch immer von variabler Länge. Interpretiert werden Lösungen als Baumstrukturen, bei denen während der Rekombination einzelne Äste vertauscht werden. Olhofer et al. sowie Jin und Sendhoff verwenden variable Repräsentationen für die Strukturoptimierung von Turbinenschaufeln in [1] und [2]. Andere problemspezifische Optimierungsmethoden mit variabler Parameteranzahl finden sich zum Beispiel bei der Optimierung von Fachwerkstrukturen, bei denen die Gesamtstruktur aus einer unterschiedlichen Anzahl verschiedenartig miteinander kombinierter Einzelstäbe besteht. Beispielhaft erwähnt seien hier Kawamura et al. die in [3] einen modifizierten genetischen Algorithmus sowie eine Lösungsrepräsentation und eine dem Problem angepasste Rekombinationsmethode beschreiben. In [4] werden von Spreve und Rolf Ansätze

zur Entwicklung optimaler Schichtstrukturen und Ansätze zur Optimierung von ARMA-Modellen vorgestellt.

Dieser Beitrag widmet sich Mechanismen zur Verwendung multikriterieller Algorithmen zur Systemmodellierung und zum Reglerentwurf. Die zentralen Aspekte sind zum einen die Abbildung von Nichtlinearitäten durch abschnittsweise lineare Kennlinien, die sowohl bei der Modellierung als auch als Kompensation bei der Regelung zum Einsatz kommen. Die zweite Anwendung hat die Optimierung von linearen zeitinvarianten Übertragungsfunktionen durch das evolutionäre platzieren von Pol- und Nullstellen zum Gegenstand. Die Mechanismen erfordern an verschiedenen Stellen die Modifikation und Erweiterung der herkömmlichen evolutionären Operatoren sowie problemspezifische Anpassungen zur Reduktion und Erweiterung der Struktur von Lösungen.

Kapitel 2 erläutert den zu Grunde liegenden multikriteriellen Algorithmus und führt die notwendigen Erweiterungen zur Behandlung von Problemen mit variabler Parameteranzahl ein. Kapitel 3 stellt den entwickelten Mechanismus zur Optimierung nichtlinearer Kennlinien vor und beschreibt die Funktionsweise der notwendigen Mechanismen. Das 4. Kapitel geht im Detail auf die Optimierung von Übertragungsfunktionen im Pol-Nullstellenraum ein. Kapitel 5 widmet sich der evolutionären Modellbildung unter Verwendung der entwickelten Mechanismen für nichtlineare Kennlinien, Übertragungsfunktionen mit variabler Parameteranzahl sowie weiterer einfacher Übertragungs- und Totzeitglieder. Als beispielhafte Anwendungen werden die Identifikation eines simulierten Systems sowie die eines realen Hydraulikventils vorgestellt. Kapitel 6 stellt den evolutionären Reglerentwurf dar. Kapitel 7 liefert einen Ausblick auf die weitere Entwicklung und mögliche Anwendungen des vorgestellten Konzepts.

2 Evolutionäre Optimierung

Als Basis der hier vorgestellten Methoden dient der NSGA-II [5] als bewährter evolutionärer Algorithmus für multikriterielle Probleme. Die Parameter des Optimierungsproblems x sind reellwertig kodiert, weshalb für diesen Algorithmus pseudobinäre Mutations- und Rekombinationsvarianten verwendet werden, wie sie in [6] und im Anhang von [7] erläutert sind.

Eine Evolutionsstrategie für Probleme mit variabler Parameteranzahl wie in Bild 1 dargestellt, unterscheidet sich bei der Populationsinitialisierung und Rekombination vom Standardverfahren, und beinhaltet einen zusätzlichen Operator zur Strukturveränderung. Die Veränderungen, denen die Lösungsstruktur unterliegt, werden für die evolutionären Operatoren problemspezifisch behandelt und in den nächsten Kapiteln im Detail vorgestellt. Problemunabhängig erzeugt die Initialisierung zunächst ausschließlich Lösungen von einfacher Struktur, da die Wahrscheinlichkeit zufällig eine ansatzweise geeignete Lösung höherer Komplexität zu erzeugen mit steigender Parameteranzahl abnimmt [8]. In den ersten Generationen setzen sich daher Individuen erhöhter Komplexität kaum durch. Erst wenn das Entwicklungspotential einfacher Lösungsstrukturen ausgereizt ist, treten sukzessive komplexere Lösungsvarianten auf, die durch ihre zusätzlichen Parameter weitergehende Optimierungspotentiale erschließen. Ein Kernaspekt der problemspezifischen Strategieelemente ist daher die neutrale Strukturveränderung. Der Evolutionäre Algorithmus ist im Wesentlichen gegenüber einer reinen Zufallssuche dadurch abgegrenzt, dass die Lösungen durch Mutation gezielt inkrementell anstatt rein zufällig verändert werden. Ziel

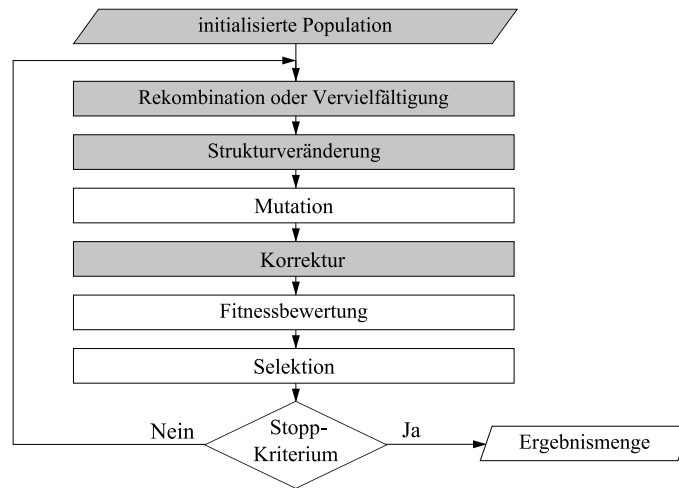


Bild 1: Struktur der Evolutionsstrategie für Optimierungsprobleme mit variabler Parameteranzahl; Abweichungen vom klassischen Algorithmus sind grau hinterlegt.

der im Folgenden beschriebenen Methoden ist es daher, auch bei der Erweiterung oder Reduzierung der Struktur der Lösung den ursprünglichen bereits adaptierten Zusammenhang zwischen Lösungs- und Kriterienraum zu erhalten.

3 Kennlinien zur Approximation von Nichtlinearitäten

Nichtlineare Verstärkungskennlinien treten in der Regelungstechnik häufig auf, beispielsweise um im System auftretende statische Nichtlinearitäten zu kompensieren. Eine solche abschnittsweise lineare Kennlinie wird durch mehrere Stützstellen definiert, zwischen denen jeweils linear interpoliert wird. Ist für einen Regler beispielsweise eine unbekannte Nichtlinearität der Strecke zu kompensieren, ist die Anzahl der Stützstellen a priori schwer festzulegen. Eine zu geringe Anzahl führt zu einer schlechten Kompensation, eine unnötig hohe Anzahl zu einem komplexen und schwer optimierbaren System. Deshalb wird ein evolutionärer Algorithmus verwendet, der parallel zur Optimierung der Lage der Stützstellen auch deren Anzahl optimiert.

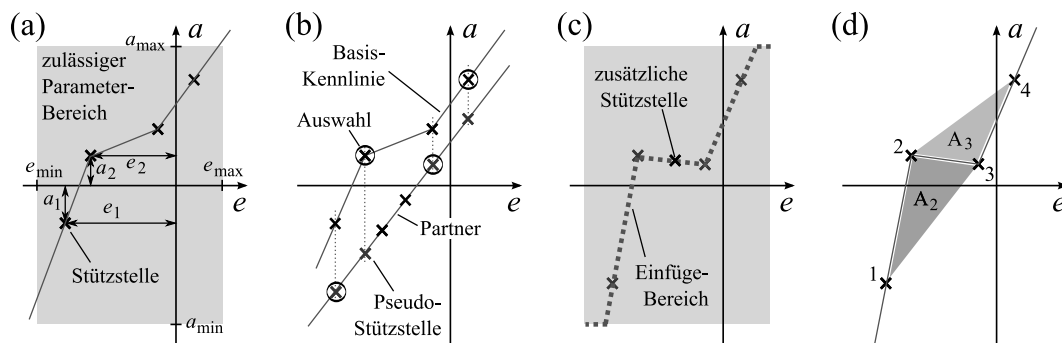


Bild 2: Darstellung einer Kennlinie mit vier Stützstellen. (a) Zulässiger Parameterbereich, Kodierung sowie Inter- und Extrapolation. (b) Rekombination von Basis-Kennlinie und Rekombinationspartner. (c) Struktur-erweiterung durch Hinzufügen eines Stützpunktes auf der gepunkteten Linie. (d) Struktur-reduktion anhand der aufgespannten Flächen.

Die Kennlinien sind in der Lösung als Liste von Stützpunkten ($\mathbf{e} = [e_1, e_2, \dots, e_g]^T$) und

$\mathbf{a} = [a_1, a_2, \dots, a_g]^T$) mit variabler Länge g kodiert, wobei sowohl Eingangswerte \mathbf{e} als auch Ausgangswerte \mathbf{a} , Optimierungsparameter sind, für die ein zulässiger Wertebereich $e_i \in [e_{min}, e_{max}]$ und $a_i \in [a_{min}, a_{max}]$ mit $i = 1, 2, \dots, g$ definiert ist. Vor Berechnung der Kennlinie werden die e_i - a_i -Wertepaare anhand ihres e -Wertes aufsteigend sortiert. Zwischenwerte werden anhand der beiden benachbarten Stützpunkte linear interpoliert. Liegt ein Eingabewert außerhalb des Stützstellenbereiches wird anwendungsabhängig entweder der a -Wert des nächsten Nachbarn übernommen, oder auf Basis der beiden nächsten Nachbarn linear extrapoliert, wie in Bild 2 (a) dargestellt.

3.1 Initialisierung von Kennlinien

Die Initialisierung erfolgt durch zufälliges gleichverteiltes Setzen von je zwei Stützstellen pro Lösung. Die beiden Stützstellen definieren eine Gerade, diese stellt die minimale Struktur für eine Kennlinie dar und entspricht als Übertragungsglied einer proportionalen Verstärkung.

3.2 Rekombination von Kennlinien

Die vorgeschlagene Rekombination wählt zufällig zwei Lösungen aus den Eltern und führt eine diskrete Rekombination durch. Mit einer Wahrscheinlichkeit von 50 % werden beide Lösungen unverändert in die Nachkommenmenge übernommen. Andernfalls werden die Lösungen rekombiniert. Um die Platzierung der Werte \mathbf{e} der Stützstellen zu erhalten, und mit Lösungen unterschiedlicher Stützstellenanzahl zu arbeiten, stellt eine Lösung die Basis dar, indem die Werte \mathbf{e} seiner Stützstellen für den Nachkommen übernommen werden. Zur Rekombination werden an diesen Stellen pseudo \mathbf{a} -Werte anhand der Kennlinie des Rekombinationspartners interpoliert (siehe Bild 2 (b)). Aus diesen zwei Sätzen von \mathbf{a} -Werten werden die Parameter des Nachkommen zufällig gewählt. Anschließend wird ein zweiter Nachkomme gebildet, wobei diesmal die andere Lösung als Basis Verwendung findet. Ist die Anzahl der Stützstellen g gleichzeitig Optimierungskriterium des Algorithmus, dann ist die Population in Bezug auf die Anzahl der Stützstellen divers. In diesem Fall ist es vorteilhaft die Lösungen der Elternmenge anhand der Anzahl von Stützstellen zu sortieren und nur benachbarte Lösungen zur Rekombination zu verwenden. Dies bietet Vorteile, da die Rekombination von Lösungen mit stark unterschiedlicher Stützstellenanzahl, aufgrund der Strukturunterschiede der beiden Eltern im Allgemeinen zu schlecht angepassten Nachkommen führt.

3.3 Strukturveränderung von Kennlinien

Die Strukturveränderung wird zufällig bei 30 % der Lösungen durchgeführt, eine Strukturveränderung erfolgt bei 10 %. Das Hinzufügen von Stützstellen erfolgt durch das Ziehen eines neuen e_{g+1} -Wertes aus einer Gleichverteilung im definierten Wertebereich $[e_{min}, e_{max}]$. Anschließend wird auf Basis der bereits vorhandenen Stützstellen der a_{g+1} -Wert ermittelt. Liegt der ermittelte Wert a_{g+1} außerhalb des Wertebereichs wird er auf den überschrittenen Grenzwert korrigiert. Da der eingefügte Stützpunkt, außer bei Überschreitung des Wertebereichs, auf der bisherigen Kennlinie liegt, ist die Erweiterung ergebnisneutral.

Die Strukturreduktion erfolgt durch Entfernen eines einzelnen Stützpunktes (e_x, a_x) , wobei die beiden äußeren Stützpunkte der Lösung ausgenommen sind $(x \in \{2, 3, \dots, g-1\})$. Für alle Stützpunkte mit $i = 2, 3, \dots, g-1$ wird der Flächenbeitrag A_i berechnet, den dieser Stützpunkt in Verbindung mit seinen beiden Nachbarn aufspannt. Bild 2 (d) veranschaulicht dies am Beispiel einer Kennlinie mit vier Stützstellen. Aus den mit dem Flächeninhalt bewerteten Punkten wird einer zufällig zum Löschen ausgewählt. Die Wahrscheinlichkeit der Löschung ergibt sich aus den Flächenbeiträgen $\frac{1/A_i}{1/\sum_{j=2}^{g-1} A_j}$. Dieses Vorgehen bevorzugt das Entfernen von Stützstellen, die nur einen kleinen Beitrag zum Gesamtverlauf der Kennlinie liefern.

4 Optimierung von Übertragungsfunktionen

Die Darstellung des Übertragungsverhaltens eines Systems erfolgt im Zeitbereich als lineare Differentialgleichung. Für die Optimierung wird das System Laplacetransformiert, und als Übertragungsfunktion dargestellt. Diese Darstellung entspricht einer gebrochen rationalen Funktion, wie sie in Gleichung 1 angegeben ist. Charakterisiert wird dieses Übertragungsverhalten durch die Verstärkung K sowie die Lage der Pol- und Nullstellen. Von besonderer Bedeutung ist, dass Pol- und Nullstellen sowohl reellwertig als auch als konjugiert-komplexes Paar auftreten können.

$$G(s) = K \frac{\prod_{k=1}^m (s - n_{R,k}) \cdot \prod_{h=1}^q (s - n_{C,h})(s - \bar{n}_{C,h})}{\prod_{j=1}^v (s - p_{R,j}) \cdot \prod_{i=1}^u (s - p_{C,i})(s - \bar{p}_{C,i})} \quad (1)$$

mit: $n_{R,k}, p_{R,k} \in \mathbb{R}, \quad n_{C,k}, p_{C,k} \in \mathbb{C}$

Sind einzelne Typen nicht vorhanden ($m = 0, q = 0, v = 0$ oder $u = 0$) gilt:

$$\prod_{k=1}^m (s - n_{R,k}) = 1 \quad \text{für } m = 0, \quad \prod_{h=1}^q (s - n_{C,h})(s - \bar{n}_{C,h}) = 1 \quad \text{für } q = 0,$$

$$\prod_{j=1}^v (s - p_{R,j}) = 1 \quad \text{für } v = 0, \quad \prod_{i=1}^u (s - p_{C,i})(s - \bar{p}_{C,i}) = 1 \quad \text{für } u = 0.$$

4.1 Parametrisierung von Übertragungsfunktionen

Es zeigt sich, dass es für die Optimierung vorteilhaft ist, die Pol- und Nullstellen nicht in kartesischen- sondern in Polarkoordinaten darzustellen, da Frequenz und Dämpfung des Systems in dieser Repräsentation entkoppelt sind. Eine komplexe Zahl z wird durch $z = r \cdot e^{i\varphi}$ beschrieben. Da für reellwertige Pol- und Nullstellen immer gilt: $\varphi = \pi$ wird diese Information nicht als Parameter der Lösung mitgeführt. Ebenso wird bei konjugiert komplexen Pol- und Nullstellen nur diejenige gespeichert, für die gilt: $\varphi \in [\pi/2, \pi]$. Rechts in Bild 3 ist die Parametrisierung eines Systems mit reellwertiger Nullstelle und

konjugiert komplexem Polpaar dargestellt. Der Parametervektor \mathbf{x} ergibt sich folglich zu:

$$\begin{aligned}
 \text{reelle Nullstellen: } \mathbf{r}_{nR} &= [r_{nR,1} \ r_{nR,2} \ \dots \ r_{nR,m}]^T \\
 \text{komplexe Nullstellen: } \mathbf{r}_{nC} &= [r_{nC,1} \ r_{nC,2} \ \dots \ r_{nC,q}]^T \\
 \varphi_{nC} &= [\varphi_{nC,1} \ \varphi_{nC,2} \ \dots \ \varphi_{nC,q}]^T \\
 \text{reelle Polstellen: } \mathbf{r}_{pR} &= [r_{pR,1} \ r_{pR,2} \ \dots \ r_{pR,v}]^T \\
 \text{komplexe Polstellen: } \mathbf{r}_{pC} &= [r_{pC,1} \ r_{pC,2} \ \dots \ r_{pC,u}]^T \\
 \varphi_{pC} &= [\varphi_{pC,1} \ \varphi_{pC,2} \ \dots \ \varphi_{pC,u}]^T \\
 \mathbf{x} &= [K^T \ \mathbf{r}_{nR}^T \ \mathbf{r}_{nC}^T \ \varphi_{nC}^T \ \mathbf{r}_{pR}^T \ \mathbf{r}_{pC}^T \ \varphi_{pC}^T]^T
 \end{aligned} \tag{2}$$

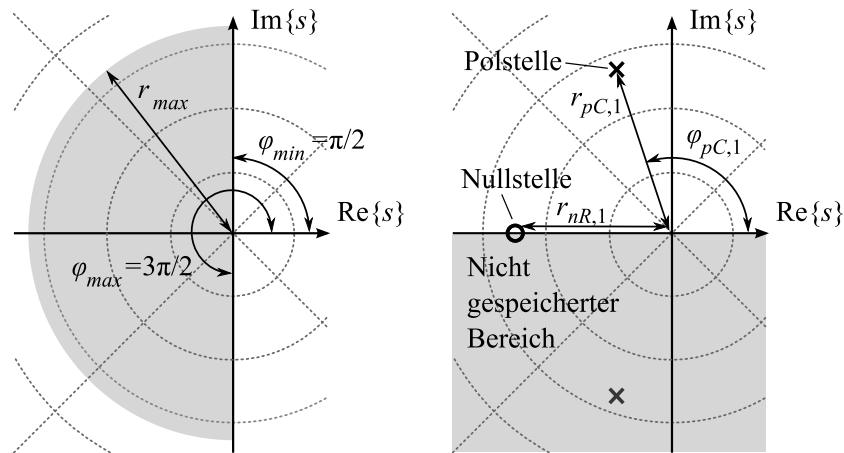


Bild 3: Links: Zulässiger Parameterbereich in der komplexen Ebene. Rechts: Parametrisierung eines Beispielsystems mit reeller Nullstelle und konjugiert komplexem Polpaar.

Die zulässigen Parameterbereiche sind in Bild 3 links dargestellt. $\varphi_{min} = \pi/2$, $\varphi_{max} = 3\pi/2$ und $r_{min} = 0$ beschränkt die Lage von Pol- und Nullstellen auf die linke Halbebene, während r_{max} so gewählt wird, dass die verwendete Hardware oder Simulation die entsprechenden Frequenzen noch korrekt verarbeitet. Bei der Modellidentifikation kann für Nullstellen auch $\varphi_{nC} \in [0, 2\pi]$ als Beschränkung verwendet werden.

4.2 Initialisierung von Übertragungsfunktionen

Die Initialisierung von $G(s)$ erfolgt als zufällig aus dem Wertebereich gewählte statische Verstärkung K , ohne Pol- oder Nullstellen. Dies entspricht dem strukturell einfachsten Übertragungsglied, dem P-Glied.

4.3 Rekombination von Übertragungsfunktionen

Durch den variablen Aufbau der Kodierung der Übertragungsfunktion ist die Rekombination von Lösungen schwierig. Bei einer zufälligen diskreten Rekombination entstehen potentiell Lösungen die alle oder keinen der dominanten Pole der Eltern erben. In beiden Fällen unterscheidet sich das Übertragungsverhalten der Lösung fundamental von dem seiner Eltern. Ebenso eignen sich intermediäre Varianten der Rekombination kaum, da hier mit hoher Wahrscheinlichkeit dominante Pole mit untergeordneten Polen verrechnet

werden, was wiederum zu stark abweichendem Übertragungsverhalten führt. Des Weiteren ergibt sich die Fragestellung nach der Rekombination von Eltern mit stark unterschiedlicher Pol- und Nullstellenanzahl. Versuche mit zufälliger diskreter Rekombination zeigen ein schlechtes Konvergenzverhalten des Algorithmus. Ein möglicher Ansatz ist die Analyse der Struktur der einzelnen Lösungen und eine darauf aufbauende Rekombination ähnlicher Strukturen. Im hier verfolgten Ansatz wird jedoch auf eine Rekombination innerhalb des Übertragungsverhaltens gänzlich verzichtet.

4.4 Strukturveränderung von Übertragungsfunktionen

Veränderungen der Struktur erfolgen mit einer Wahrscheinlichkeit von 50 %. Diese Modifikationen erfolgen immer auf eine Art, die einzelne Lösungen nicht zu stark verändert. Bei 20 % der Lösungen wird die Ordnung reduziert, während bei 30 % eine Erhöhung der Ordnung erfolgt. Die Strukturveränderung teilt sich dabei in sechs gleichberechtigte Varianten, die mit einer Wahrscheinlichkeit von jeweils 5 % zum Einsatz kommen.

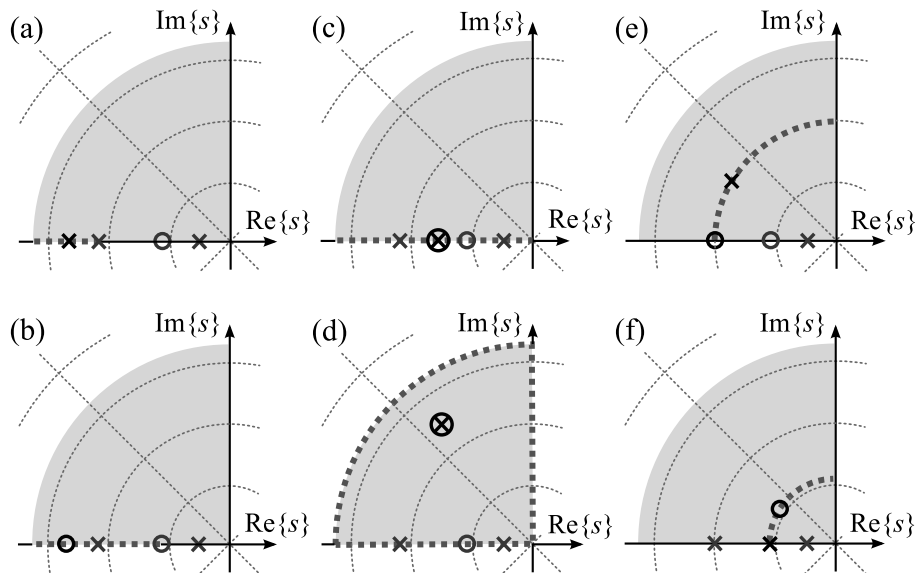


Bild 4: Mögliche Strukturveränderungen einer Beispiel-Übertragungsfunktion mit zwei Polstellen und einer Nullstelle (grau). Die zulässigen Wertebereiche für das Hinzufügen sind in dicker Strichpunktlinie gekennzeichnet, hinzugefügte Elemente sind schwarz dargestellt. (a) Hinzufügen einer Polstelle. (b) Hinzufügen einer Nullstelle. (c) Hinzufügen einer Kombination aus Pol- und Nullstelle. (d) Hinzufügen einer konjugiert komplexen Pol-Nullstellenkombination. (e) Erweitern einer existierenden Polstelle beziehungsweise einer Nullstelle (f) durch eine Pol-Nullstellenkombination und Hinzufügen eines komplexen Anteils.

- **Einfügen von „schnellen“ reellen Pol- und Nullstellen:** Das Übertragungsverhalten wird primär von den Pol- und Nullstellen mit dem größten Realteil bestimmt. Die Strukturveränderung durch Hinzufügen eines Punktes erfolgt somit „links“ von allen bisherigen Punkten. Bei Kodierung in Polarkoordinaten gilt für die Erweiterung durch ein Polstelle $r_{nR,m+1} \in [\max(\mathbf{r}_{nR}, \mathbf{r}_{nC}), r_{max}]$. Das Hinzufügen einer Nullstelle erfolgt analog, dabei gilt zusätzlich die Bedingung, dass beim Erweitern ein kausales System entsteht; der Zählergrad den Nennergrad nicht übersteigt ($m + 2q < v + 2u$). Bild 4 veranschaulicht diese Erweiterung durch eine Polstelle (a) und eine Nullstelle (b).

- **Einfügen von Pol-Nullstellen-Kombinationen:** Das Hinzufügen von reellwertigen Pol- und Nullstellen an der gleichen Stelle ist neutral, da sie sich gegenseitig kompensieren, beziehungsweise sich in der mathematischen Darstellung kürzen lassen. Das Hinzufügen erfolgt an beliebiger Stelle im zulässigen Bereich der reellen Achse $r_{nR,m+1} = r_{pR,v+1} \in [0, r_{max}]$. Einfluss erhält diese Erweiterung durch die anschließende Mutation, welche die Lage zufällig zueinander versetzt, und so Elemente mit Lead- oder Lag-Charakteristik entstehen lässt.
- **Einfügen von komplexen Pol-Nullstellen-Kombinationen:** Analog zum vorher erläuterten Fall werden hier konjugiert komplexe Pol-Nullstellen-Paare mit $r_{nC,q+1} = r_{pC,u+1} \in [0, r_{max}]$ und $\varphi_{nC,q+1} = \varphi_{pC,u+1} \in [\pi/2, \pi]$ an zufälliger Stelle im zulässigen Wertebereich initialisiert. Dies erlaubt die Einführung einer schwingenden Charakteristik in das bestehende System.
- **Umwandlung einer Pol- oder Nullstelle in ein konjugiert-komplexes Paar:** Eine bestehende reellwertige Pol- oder Nullstelle wird mit einer Pol-Nullstellen-Kombination überlagert. Die nun doppelt vorhandene Pol- oder Nullstelle erhält nachfolgend einen konjugiert-komplexen Anteil, indem φ einen Zufallswert aus dem Intervall $[\pi/2, \pi]$ erhält. Dargestellt wird diese Erweiterung in Bild 4, für eine Polstelle (e) und eine Nullstelle (f). Wird die erläuterte Kodierung benutzt, bedeutet dies im Falle der Erweiterung einer Polstelle $r_{pR,1}$ das Hinzufügen seiner Nullstelle $r_{pR,x} = n_{pR,m+1}$ sowie die Umwandlung der originalen Polstelle in ein konjugiert komplexes Polpaar $r_{pC,q+1} = r_{pR,x}$ und $\varphi_{pC,q+1} \in [\pi/2, \pi]$. Diese Erweiterung ist speziell für die Systemidentifikation sinnvoll, da Systeme mit zu niedriger Ordnung den Anstiegsbereich gut approximieren können, jedoch nicht in der Lage sind schwingendes Verhalten zu reproduzieren. Diese Strukturserweiterung ermöglicht die Umwandlung in ein schwingfähiges System mit ähnlicher Anstiegscharakteristik.
- **Reduktion der Ordnung:** Die Strukturreduktion einer bestehenden Übertragungsfunktion ist schwierig, da der Einfluss einzelner Pol- und Nullstellen nur im Kontext aller vorhandenen Pol- und Nullstellen zu beurteilen ist. Denkbar ist die Struktur analog zu den Erweiterungen nach „schnellen“ beziehungsweise sich kompensierenden Pol- und Nullstellen zu durchsuchen und diese zu entfernen. Dies birgt jedoch die Gefahr, den positiven Einfluss von Erweiterungen in vorherigen Generationen durch das Entfernen ebendieser Pol- und Nullstellen wieder zunichte zu machen. Um die Charakteristik des Systems weitestgehend zu erhalten, wird hier auf das analytische Verfahren der Pade-Approximation [9] zurückgegriffen. Die Ordnung des bestehenden Systems wird ermittelt und bei möglichst gleich bleibendem Systemverhalten um eins reduziert.

4.5 Korrektur von Übertragungsfunktionen

Der an die Standard-Mutation angeschlossene Korrekturschritt ermöglicht es Pol- und Nullstellen von der reellen Achse in die imaginäre Ebene beziehungsweise von dieser auf die reelle Achse zu wechseln. Die beschriebenen Operatoren gelten für Pol- und Nullstellen in gleicher Weise, weshalb nachfolgend von *Singularitäten* gesprochen wird.

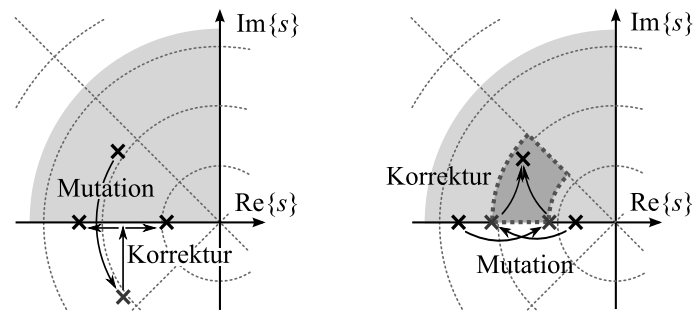


Bild 5: Korrektur von Übertragungsfunktionen. Links: Wechsel von komplex zu reell. Rechts: Wechsel von reell zu komplex.

- **Wechsel von komplex zu reell:** Da Singularitäten mit Imaginäranteil nur konjugiert komplex auftreten können, wird nur die Singularität mit positivem Imaginäranteil gespeichert, während der zulässige Parameterbereich $[\pi/2, 3\pi/2]$ ist. Tritt nach der Mutation ein negativer Imaginäranteil auf, ist dieser auf einen Sprung über die reelle Achse durch die Mutation zurückzuführen. In diesem Fall wird diese konjugiert-komplexe Singularität in zwei reellwertige zerlegt. Dazu erfolgt eine Umwandlung in kartesische Koordinaten $a + i \cdot b$ und die Erstellung zweier reeller Singularitäten $r_{R,x} = -a + b/2$ und $r_{R,y} = -a - b/2$. Bild 5 (links) stellt dieses Vorgehen graphisch dar.
- **Wechsel von reell zu komplex:** Vor der Mutation wird die Reihenfolge aller reellen Singularitäten protokolliert. Wechseln zwei Singularitäten die Abfolge, findet ein Wechsel in die komplexe Ebene statt (vergleiche Bild 5 (rechts)). Die beiden bisherigen reellen Singularitäten $r_{R,x}$ und $r_{R,y}$ werden entfernt, während eine konjugiert komplexe Singularität mit $r_C \in [r_{R,x}, r_{R,y}]$ und $\varphi_C \in [3\pi/4, \pi/2]$ hinzugefügt wird.

5 Systemidentifikation

Grundlage einer Systemidentifikation mit dem vorgestellten Ansatz ist die Messreihe des Ausgangssignals eines mit einem Stimulus beaufschlagten Systems. Die Übertragungsglieder, mit denen das Verhalten des realen Systems angenähert wird, sind problemabhängig. Zur Verfügung stehen das Übertragungsverhalten zur Bestimmung der Systemdynamik, eine Kennlinie zur Bestimmung der statischen Charakteristik, Totzeit und Hysterese.

Die Anforderungen an den Stimulus ergeben sich aus den verwendeten Übertragungsgliedern, die evolutionär identifiziert werden. Zur Bestimmung des Übertragungsverhaltens eignen sich Sprungfolgen, deren Abstände das zwischenzeitliche Einschwingen des Systems erlauben. Zur Bestimmung der nichtlinearen Verstärkungskennlinie ist ein Durchfahren des gesamten Arbeitsbereichs erforderlich, idealerweise in einer Geschwindigkeit, bei der die Dynamik des Systems eine untergeordnete Rolle spielt. Zur Bestimmung der Hysterese werden Positionen in der Mitte des Arbeitsbereichs von beiden Seiten angefahren.

Das Hinzufügen von Pol- und Nullstellen verändert die Gesamtverstärkung der Übertragungsfunktion, was dementsprechend auch deren stationären Endwert verändert, auf den

das System einschwingt. Bei der Identifikation ist eine derartige Systemveränderung unerwünscht. Zur Kompensation erhält $G(s)$ einen zusätzlichen Verstärkungsfaktor K_N den Einfluss der Pol- und Nullstellen von $G(s)$ auf eins normiert:

$$K_N = \frac{\prod_{k=1}^m (-n_{R,k}) \cdot \prod_{h=1}^q (n_{C,h} \cdot \bar{n}_{C,h})}{\prod_{j=1}^v (-p_{R,j}) \cdot \prod_{i=1}^u (p_{C,i} \cdot \bar{p}_{C,i})} = \frac{\prod_{k=1}^m (-r_{nR,k}) \cdot \prod_{h=1}^q (r_{nC,h}^2 + \varphi_{nC,h}^2)}{\prod_{j=1}^v (-r_{pR,j}) \cdot \prod_{i=1}^u (r_{pC,i}^2 + \varphi_{pC,i}^2)}. \quad (3)$$

5.1 Systemidentifikation in der Simulation

Zur Verifikation des Ansatzes wird ein System $P(s) = 0,02 \frac{s+0,5}{(s+0,075)(s+0,03+0,3i)(s+0,03-0,3i)}$ evolutionär durch eine Übertragungsfunktion approximiert. Die bei der Optimierung verwendeten Bewertungskriterien sind die Fläche zwischen dem gemessenen und approximierten Systemausgang (IAE) sowie die Ordnung der jeweiligen Übertragungsfunktion $G(s)$. Der evolutionäre Optimierungsalgorithmus arbeitet mit 25 Eltern und 50 Nachkommen über 300 Generationen. Die Übertragungsfunktion $G(s)$ unterliegt der Beschränkung der Ordnung $m + q \leq 7$.

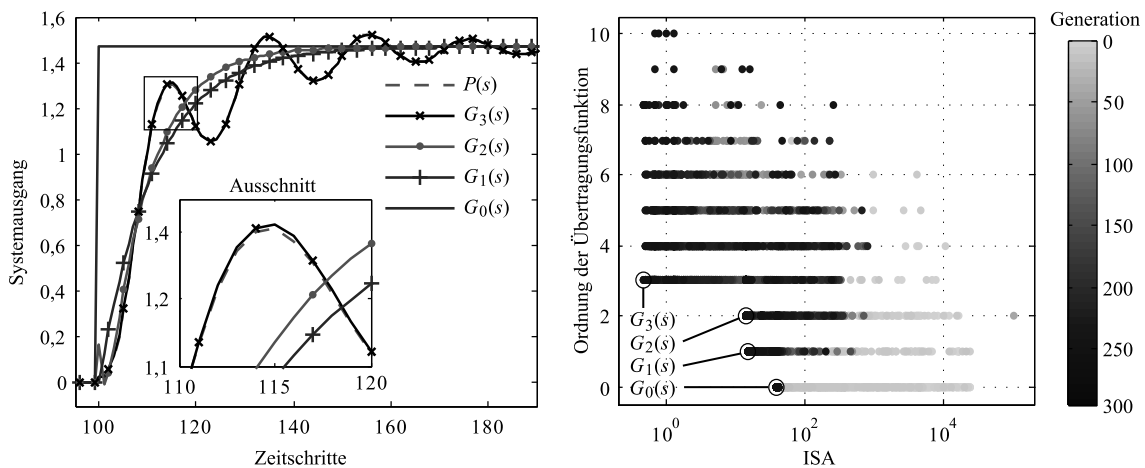


Bild 6: Ergebnisse der Systemidentifikation.

Bild 6 zeigt rechts die erzielten Kriterien über die Anzahl der Generationen. Neben den Verbesserungen des ISA, die durch die normale Mutation erzielt werden, erschließt sich die Evolution durch Strukturveränderungen weitergehende Verbesserungspotentiale. Bei der dargestellten, approximierten Übertragungsfunktion dritter Ordnung bedeutend dies, dass Fortschritte oft von Übertragungsfunktionen von Ordnung fünf und sechs erzielt werden, was die hohe Anzahl an Lösungen in diesem Bereich erklärt. Diese Fortschritte werden in folgenden Generationen durch die Strukturreduktion auf Lösungen niedrigerer Ordnung propagiert, wodurch die Lösungen höherer Ordnung aus der Elternmenge verschwinden. Ist in den späten Generationen erst einmal die tatsächliche Struktur des Systems gefunden, spielen weitere Strukturveränderungen eine untergeordnete Rolle.

Die Ergebnisse der finalen Elite mit den Übertragungsfunktionen:

$$G_3(s) = 0,027059 \frac{(s + 0,3665)(s - 0,2498 + 0,9683i)(s - 0,2498 - 0,9683i)}{(s + 0,0740)(s + 0,0309 + 0,3006i)(s + 0,0309 - 0,3006i)},$$

$$G_2(s) = 0,17085 \frac{(s - 0,5630 + 0,8265i)(s - 0,5630 - 0,8265i)}{(s + 0,1163)(s + 1)},$$

$$G_1(s) = 0,13085 \frac{1}{(s + 0,08898)} \quad \text{und} \quad G_0(s) = 1,4716$$

sind in Bild 6 links zum Vergleich aufgetragen. Dargestellt ist der erste Sprung des verwendeten Stimulus. Die Referenzantwort des Systems $P(s)$ wird von $G_3(s)$ so gut approximiert, dass Abweichungen nur im Ausschnitt erkennbar sind. Die Übertragungsfunktionen niedrigerer Ordnung sind prinzipiell nicht in der Lage den Verlauf korrekt zu approximieren. Die Verwendung der Ordnung als Optimierungskriterium verhindert das Entstehen von Lösungen unnötig hoher Komplexität und erlaubt des Weiteren die subjektive Abwägung zwischen Komplexität und Performance.

5.2 Systemidentifikation eines Hydraulikventils

Identifiziert wird das Systemverhalten eines hydraulischen Wegeventils, das verwendet wird, um Richtung und Stärke eines Volumenstroms zu regeln. Dazu wird ein Kolben positioniert, dass er die entsprechenden Kanäle im Ventil öffnet oder schließt. Die Positionierung des Kolbens erfolgt durch einen Elektromagneten, der gegen eine Rückstellfeder arbeitet. In der hier dargestellten Anwendung wird die unterlagert geregelte Stromstärke im Magneten als Eingang verwendet, während die Kolbenposition, gemessen über einen eingebauten induktiven Wegaufnehmer, den Systemausgang darstellt.

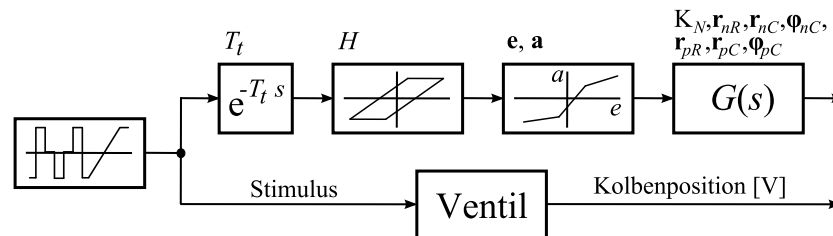


Bild 7: Aufbau des Identifikationsmodells.

Grundlage für die Identifikation ist ein allgemeines Streckenmodell, wie in Bild 7 dargestellt, das aus Hysterese, Totzeit und einem Hammerstein-Modell besteht. Letzteres setzt sich aus einer nichtlinearen Kennlinie und einer linearen Übertragungsfunktion zusammen. Da es sich bei der Hysterese und Kennlinie um nichtlineare zeitinvariante Systeme handelt, hat die Reihenfolge der Übertragungsglieder Einfluss auf das Ergebnis, was die Allgemeingültigkeit der Struktur des allgemeinen Streckenmodells einschränkt. Durch das Platzieren dieser Komponenten vor der Übertragungsfunktion vereinfacht sich die Bestimmung der Parametergrenzen für diese Komponenten aufgrund des bekannten Wertebereichs der Eingangssequenz. Das Hysteresemodell filtert je nach innerem Status Signalanteile mit Amplituden kleiner als die Hysterese H aus dem Signal, weshalb es sinnvoll ist, sie vor der Kennlinie und Übertragungsfunktion einzubinden, da ansonsten

Signalanteile dieser Blöcke keinen Einfluss auf das Approximationsergebnis mehr haben. Das bei der Optimierung verwendete Bewertungskriterium ist die Fläche zwischen dem gemessenen und approximierten Systemausgang (IAE). Der Optimierungsalgorithmus verwendet 10 Eltern und 50 Nachkommen über 160 Generationen. Die Anzahl der Kennlinien-Stützstellen g ist auf 10 beschränkt. Die Übertragungsfunktion $G(s)$ unterliegt der Beschränkung $m + q \leq 7$. Die Kombination von Kennlinie und Übertragungsfunktion birgt die Besonderheit, dass die statische Verstärkung des gesamten Systems von beiden Blöcken beeinflusst wird. Um diese Redundanz zu vermeiden wird nur die Verstärkung K_n verwendet, der Parameter K entfällt.

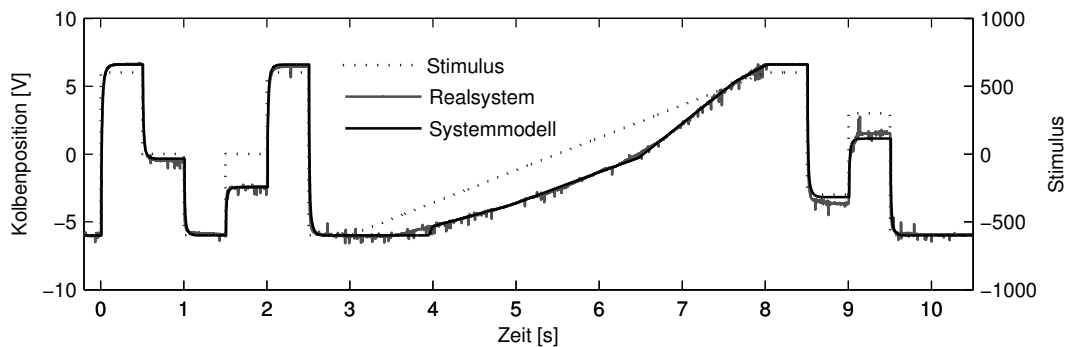


Bild 8: Ergebnis der Systemidentifikation des Ventils.

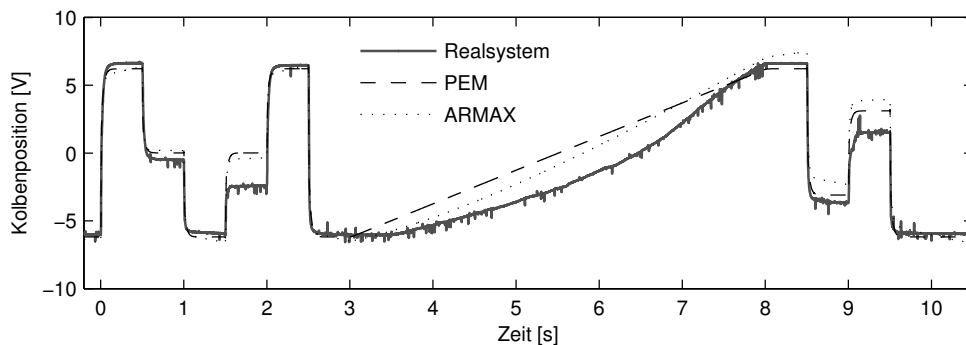


Bild 9: Identifizierung des Ventils durch die Matlab-Toolbox.

Bild 8 zeigt das Ergebnis des evolutionär identifizierten Modells im Vergleich mit der Systemantwort des Ventils. Abgebildet ist der als Anregung verwendete Stimulus sowie die reale Systemantwort und die Antwort des identifizierten Modells. Der durch das IAE-Kriterium bewertete Bereich liegt bei 0 bis 8,5 s, während die beiden folgenden Sprünge lediglich der Verifikation dienen.

Um das Ergebnis der Identifikation einzuordnen, wird es in Abbildung 9 mit der Qualität herkömmlicher Verfahren verglichen. Exemplarisch gezeigt werden die Stimulus-Antworten von Modellen die mittels Verfahren aus der Matlab System Identification Toolbox identifiziert worden sind, zum einen als kontinuierliche Übertragungsfunktion (dritter Ordnung), zum anderen als ARMAX Polynom-Modell mit jeweils 6 Koeffizienten [10]. Diese werden mittels eines iterativen nichtlinearen Verfahrens zum Minimieren des quadratischen Fehlers ermittelt. Für beide Modelltypen wurde die jeweils beste Ordnung anhand des Modellfehlers auf Optimierungs- und Validierungsdaten gewählt. Beide Modelle zeigen einen deutlich größeren Ausgangsfehler als das evolutionär entwickelte Modell. Dies ist zum größten Teil auf die fehlende Möglichkeit zur Modellierung von Hysterese

zurückzuführen. Ebenso wird erwartungsgemäß der vor allem im Rampenabschnitt der Stimulusfunktion auftretende Effekt einer nichtlinearen Verstärkung nicht modelliert.

Für das hinsichtlich des IAE-Kriteriums ermittelte optimale Systemmodell gilt, dass die Totzeit des Systems nahezu vernachlässigbar ist, die statische Nichtlinearität durch eine Kennlinie mit 10 Stützstellen abgebildet und die Dynamik durch eine Übertragungsfunktion vierter Ordnung angenähert wird. Das Systemverhalten zeigt darüber hinaus eine signifikante Hysterese.

Generell ist das evolutionär identifizierte Modell sehr gut in der Lage das Systemverhalten zu approximieren. Die deutlichste Abweichung zeigt sich bei Beginn der Rampe (3 bis 4 s). Der Grund liegt im einfachen Hysteresemodell, das nicht in der Lage ist das Systemverhalten korrekt zu approximieren. Die entstandene Lösung stellt einen Kompromiss zwischen der Hysterese H dar, die benötigt wird, um die beiden Sprünge bei 0,5 s und 1,5 s korrekt darzustellen, und dem Beginn der Rampe, der einen kleineren Wert für H erfordert, um korrekt zu approximieren.

6 Regelung eines Hydraulikventils

Bei komplexen Regelungsproblemen erfordert die Wahl der Reglerstruktur Vorwissen über das Übertragungsverhalten der Regelstrecke. Ziel des vorgestellten Verfahrens ist es, die Strukturauswahl ebenfalls auf die evolutionäre Optimierung zu verlagern, indem nur allgemeine Vorgaben über die grundsätzlich zulässigen Strukturen gemacht werden, um so basierend auf zunächst einfachen Grundformen im Laufe der Evolution komplexere Strukturen zu entwickeln.

Das mit der Identifikation entwickelte nichtlineare Streckenmodell des Ventils ist Grundlage der Reglerentwicklung. Optimiert wird ein Regler bestehend aus einer Übertragungsfunktion. Theoretisch ist es möglich, anhand der durch die Identifikation erhaltenen Daten zusätzliches Expertenwissen in den Regler einfließen zu lassen, beispielsweise durch Kompensation der Kennlinie und der Hysterese, was in der hier dargestellten Reglerentwicklung jedoch nicht erfolgt.

Der Evolutionäre Algorithmus arbeitet mit 25 Eltern, 50 Nachkommen und 300 Generationen. Kriterien sind ISE zwischen dem Sollwert und Istwert des geregelten Systems, sowie der Grad der Übertragungsfunktion. Bild 10 stellt Details des ersten Sprungs der zur Optimierung verwendeten Testsequenz dar. Der entwickelte P-Regler $R_0(s)$ ist in der Lage das System zu stabilisieren, er zeigt jedoch eine deutliche bleibende Regelabweichung. $R_1(s)$, der Regler ersten Grades, erreicht den Endwert, jedoch mit ähnlicher Schwingneigung wie $R_0(s)$. Der Regler zweiten Grades $R_2(s) = 493,4 \frac{(s+94,23)(s+5134)}{(s+0,0006329)(s+8280)}$ ist annähernd ein PIDT₁-Regler, mit der Einschränkung, dass die dominante Polstelle bei -0,0006329 nicht exakt auf Null konvergiert, was eine leichte Abweichung zur PIDT₁-Struktur bedeutet. Dieser Regler ist in der Lage, das System schnell und ohne Schwingneigung auszuregeln. $R_3(s)$ zeigt nur minimal besseres Verhalten und eine annähernd gleiche Systemantwort wie $R_2(s)$. Deutlichere Verbesserungen erzielen $R_4(s)$ und $R_5(s)$ wie rechts oben in Bild 10 zu sehen ist. $R_4(s)$ und $R_5(s)$ sind sich dabei so ähnlich, dass auf eine Darstellung von $R_5(s)$ verzichtet wurde.

Die Erkenntnis, dass ein PID-Regler zu den optimalen Lösungen gehört, deckt sich mit vorherigen Experimenten an diesem System. Als Referenz dient ein PID-Regler mit optimierten Parametern, dessen Sprungantwort praktisch deckungsgleich mit den Ergebnissen

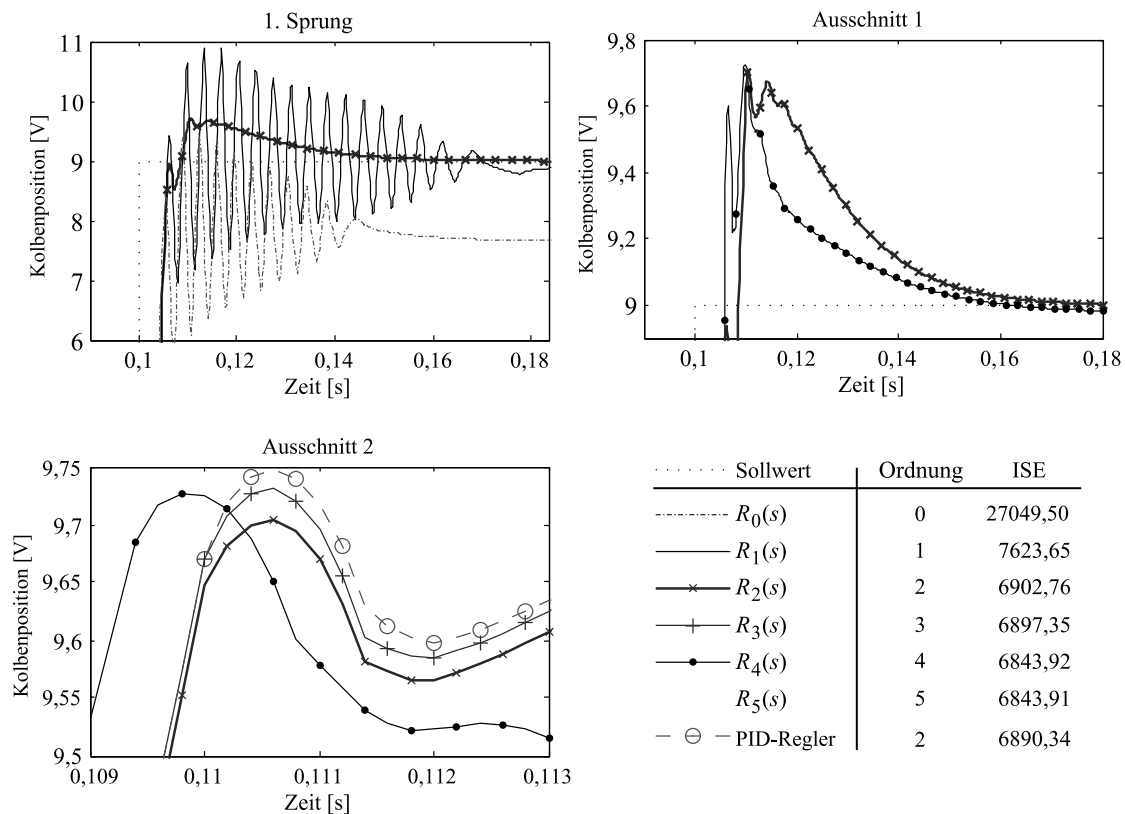


Bild 10: Ausschnitte der Sprungantwort des simulierten geregelten Systems. Dargestellt sind die Elite-Individuen der Optimierung sowie ein PID-Regler zum Vergleich.

$R_2(s)$ und $R_3(s)$ ist. Unterschiede sind lediglich in der stark skalierten Darstellung links unten in Bild 10 erkennbar. Das Ergebnis zeigt, dass der hier dargestellte evolutionäre Algorithmus in der Lage ist, zweckmäßige aus der klassischen Regelungstechnik bekannte Strukturen zu entwickeln und eine gleichzeitige Parameteroptimierung durchzuführen.

7 Ausblick

Dieser Beitrag präsentiert einen Ansatz zur Lösung von Optimierungsproblemen mit variabler Parameteranzahl durch einen multikriteriellen evolutionären Algorithmus. Als Beispielprobleme angeführt werden die Modellidentifikation und der Entwurf von Reglern, jeweils mit Hilfe von nichtlinearen Kennlinien und Übertragungsfunktionen variabler Ordnung. Die problemspezifischen Operatoren und Methoden, die dabei zur Anwendung kommen, lassen sich darüber hinaus jedoch auch auf andere Probleme anpassen. Im Bereich der Regelungstechnik ist dies eine Ausweitung auf zeitdiskrete Systeme und Mehrgrößensysteme. Bei der Identifikation des Wegeventils ist das zu einfache Hysterese-Modell das größte Hemmnis. Eine Verknüpfung der evolutionären Optimierung mit fortgeschrittenen oder detaillierteren Methoden zur Hysteresemodellierung wie zum Beispiel den Preisach-Modellen ist geeignet hier Verbesserungen zu schaffen.

Literatur

- [1] Olhofer, M.; Jin, Y.; Sendhoff, B.: Adaptive encoding for aerodynamic shape optimization using Evolution Strategies. In: *Congress on Evolutionary Computation (CEC)*, Bd. 2, S. 576–583. Seoul, Korea: IEEE Press. 2001.
- [2] Jin, Y.; Sendhoff, B.: A systems approach to evolutionary multi-objective structural optimization and beyond. *IEEE Computational Intelligence Magazine* (2009). (Accepted).
- [3] Kawamura, H.; Ohmori, H.; Kito, N.: Truss topology optimization by a modified genetic algorithm. *Structural and Multidisciplinary Optimization* 23 (2002) 6, S. 467–473.
- [4] Sprave, J.; Rolf, S.: Variable-Dimensional Optimization with Evolutionary Algorithms Using Fixed-Length Representations. In: *Evolutionary Programming*, S. 261–269. 1998.
- [5] Deb, K.; Pratap, A.; Agarwal, S.; Meyarivan, T.: A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation* 6 (2002), S. 182–197.
- [6] Agrawal, R. B.; Deb, K.: Simulated Binary Crossover for Continuous Search Space. 1995.
- [7] Deb, K.; Agrawal, S.: A niched-penalty approach for constraint handling in genetic algorithms. In: *Artificial neural nets and genetic algorithms: proceedings of the international conference in Portoro , Slovenia, 1999*, S. 235. Springer Verlag Wien. 1999.
- [8] Jin, Y.; Olhofer, M.; Sendhoff, B.: On evolutionary optimization of large problems with small populations. In: *Int. Conf. on Natural Computation*, LNCS 3611, S. 1145–1154. Edinburgh: Springer. 2005.
- [9] Varga, A.: Balancing free square-root algorithm for computing singular perturbation approximations. In: *Decision and Control, 1991., Proceedings of the 30th IEEE Conference on*, S. 1062–1065 vol.2. 1991.
- [10] Ljung, L.: *System Identification: Theory for the User (2nd Edition)*. Prentice Hall PTR. ISBN 0136566952. 1998.

Adaptive Fuzzy-Regelung einer Gummispritzpresse

Ferdinand Svaricek¹, Jan Gogolin¹ und Volker Härtel²

¹Institut für Steuer- und Regelungstechnik
Universität der Bundeswehr München

Werner-Heisenberg-Weg 39, 85577 Neubiberg, Tel. 089-6004-2077, Fax 089-6004-4565, E-Mail: ferdinand.svaricek@unibw.de, jan.gogolin@arcor.de

²EST-Consulting

Fichtenstr. 50, 82110 Germering,
Tel. 089-8414564, E-Mail: volker.haertel@web.de

1 Einleitung

Der Gummispritzguß ist eines der wichtigsten formgebenden Verfahren zur Herstellung von Formartikel in der Kautschukindustrie. Es ist ein zyklisches, diskontinuierliches Verfahren, bei dem sich die Verfahrensschritte Plastifizieren, Einspritzen und Vulkanisieren des Kautschuks sowie Entformen der Werkstücke zyklisch wiederholen. Die Formartikel werden unter Druck und Hitze ausvulkanisiert. Die Dauer und die Güte des Vulkanisationsprozesses ist hierbei von der Wanddicke des Werkstücks, der Gummitemperatur nach dem Einspritzen und der Formtemperatur abhängig, so daß der Kautschuk nach dem Einspritzen in die Form durch Wärmeleitung im Werkzeug von außen weiter erhitzt wird, um den Vulkanisationsvorgang zu beschleunigen.

In einem neuen Verfahren wird die Umsetzung von Druckarbeit in thermische Energie so geregelt, daß nach Einspritzen des Kautschuks in die Form nur noch kurz nachgeheizt werden muß. Damit ergibt sich bei dickwandigen Formteilen eine deutliche Verkürzung der Gesamtzykluszeit. Weiterhin liefert dieses neue Einspritzverfahren eine für eine homogene Vulkanisation günstigere Temperaturverteilung. Als zusätzliches Stellglied dient in diesem Verfahren eine verstellbare Drossel, über die durch eine Veränderung des Strömungswiderstandes die am Kautschuk geleistete Druckarbeit beeinflusst werden kann. Dabei wird durch Regelung der Spritzkolbengeschwindigkeit ein konstanter Volumenstrom eingepreßt. Die temperatur- und mischungsabhängige Viskosität der Kautschukmischung kann man als Störgröße auffassen.

Ein Prototyp einer Gummispritzpresse mit verstellbarer Drossel wurde im Rahmen des PRO INNO II - Projektes **“Entwicklung einer Gummispritzpresse mit einem temperaturgeregelten ein- und zweistufigen Einspritzprozeß“**, ein Kooperationsprojekt zwischen der LWB Steinl GmbH in Landshut und dem Institut für Steuer- und Regelungstechnik der Universität der Bundeswehr München, aufgebaut.

Standardmäßig stehen als Meßgrößen nur der Druck und der Volumenstrom des Fluids sowie die Stellung der Drossel zur Verfügung. Eine direkte Messung der Mischungstemperatur im Einspritzkanal bzw. in der Form ist nicht möglich. Darüber hinaus tritt die Mischung erst nach einer Totzeit von mehreren Sekunden in die Form ein. Die Temperaturregelstrecke hat also eine vom Volumenstrom abhängige Totzeit und ist stark nichtlinear.

In Abhängigkeit von der gewünschten Temperaturerhöhung wird jetzt der Druck vor der Drossel vorgegeben und mit Hilfe eines Fuzzy-Reglers eingestellt. Hierbei macht man sich zu Nutze, daß für die meisten Kautschukmischungen unter adiabaten Bedingungen eine Temperaturerhöhung von 4 bis 5 K pro 100 bar Druckabfall auftritt. In Abhängigkeit vom stationären Druck- und Temperaturfehler kann man den Fuzzy-Druckregler jetzt von Zyklus zu Zyklus mit einfachen Regeln adaptieren. Die Fuzzy-Regelung wurde mit einem dSPACE-System realisiert und an dem Prototyp erprobt. Die Güte und Robustheit der Adaption werden anhand experimenteller Ergebnisse diskutiert.

2 Grundidee

Eine Analyse des gesamten Spritz- und Vulkanisationsprozesses zeigt, daß für eine signifikante Absenkung der Vulkanisationszeit dickwandiger Artikel ein zusätzlicher Temperatursprung von ca. 20 bis 50°C in einem frühen Stadium des Spritzzyklus notwendig ist. Unter Berücksichtigung der typischen thermischen und rheologischen Gummieigenschaften erfordert dies einen Wärmeeintrag von bis zu 20 kW. Da sich jedoch selbst hoch rußgefüllte Kautschukmischungen als schlechte thermische und mäßige elektrische Leiter erweisen, läßt sich dies nicht durch Wärmeleitung bzw. ohmsche oder dielektrische Erwärmung erreichen. Der effektivere Weg, diese Leistung in die Mischung einzubringen, ist die Dissipation mechanischer Energie im Inneren des Materials. Aus dem Energieerhaltungsgesetz läßt sich unabhängig von den rheologischen Mischungseigenschaften die maximal erreichbare Massentemperaturerhöhung unter adiabaten Bedingungen berechnen:

$$\Delta\bar{T} = \frac{\Delta p}{c_p \cdot \rho} \quad (1)$$

Dabei sind $\Delta\bar{T}$ eine über die Zeit und Kanalquerschnitt gemittelte Temperaturdifferenz und Δp der zeitlich gemittelte Druckabfall in einer Kanalströmung. Die einzigen Materialkennzahlen, die den Temperaturanstieg bestimmen, sind die spezifische Wärmekapazität c_p und die Dichte ρ der Mischung. Für die meisten Kautschukmischungen resultiert daraus unter adiabaten Bedingungen eine Temperaturerhöhung von 4 bis 5 K pro 100 bar [1].

Der adiabate Fall läßt sich recht einfach in Ausspritzversuchen realisieren; sie eignen sich daher gut zur praxisnahen Charakterisierung der thermischen Eigenschaften einer Mischung.

Der Einspritzvorgang in ein Formnest ist wesentlich komplexer und nähert sich erst bei sehr hohen Volumenströmen und kurzer Runnersystemen (d.h.: kurzer Verweilzeit) dem adiabaten Grenzfall an. Daher muß das für die Regelung des Spritzprozesses verwendete Modell neben der dissipativen Schererwärmung auch alle sonstigen Wärmequellen und -senken berücksichtigen, die Einfluß auf die Massetemperatur nehmen. Dadurch ergeben sich Korrekturterme der Form

$$\Delta T = \bar{\alpha} \cdot A \cdot \bar{t}_r \cdot f_c (\bar{T}_i - T_w) \quad (2)$$

für den Wärmeaustausch im Aggregat und im Werkzeug [2],

mit $\bar{\alpha}$ = mittlerer Wärmeaustauschkoeffizient,
 A = Wärmeaustauschfläche,
 \bar{t}_r = mittlere Verweilzeit,
 f_c = Korrekturfaktor (> 1 aufgrund höherer Randschichttemperaturen),
 T_i, T_w = Mischungs- und Wandtemperatur.

3 Aufbau und Arbeitsweise des Prototyps

Der Mischungsstreifen wird von der Plastifizierschnecke des Spritzaggregates eingezo-
gen und in die Spritzkammer der EF-Einheit (Bild 1) gefördert.

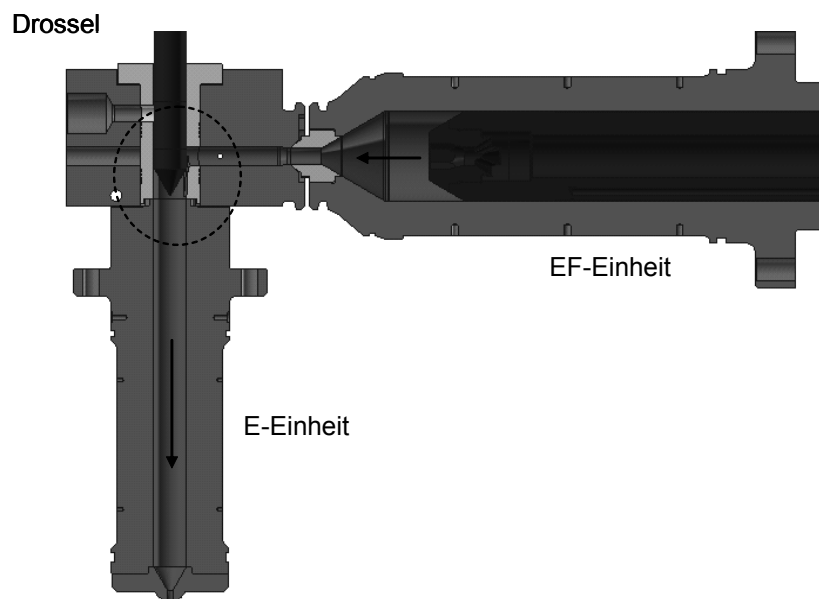


Bild 1: Ausschnitt aus der CAD-Darstellung des EF-E Spritzaggregats von LWB mit einseitiger Drossel

Wie bei konventionellen Spritzaggregaten wird die Materialtemperatur durch die Rotation der Schnecke und durch die Temperatur des Plastifizierzylinders erhöht. Nach dem Aufdosieren des eingestellten Volumens wird das Material von der EF-Einheit durch eine einstellbare Drossel in die nachgelagerte Kolbenspritzeinheit (E-Einheit) und in die Form eingespritzt. Das eingesetzte Drosselelement (Bild 1) variiert einen Spalt zwischen dem Kolben und der Wand der Kolbeneinheit. Dabei leitet eine wandhaftende Strömung einen großen Teil der in der Drossel erwärmten Mischung entlang der Kolbenspitze in die Mitte des folgenden Fließkanals (Bild 2).

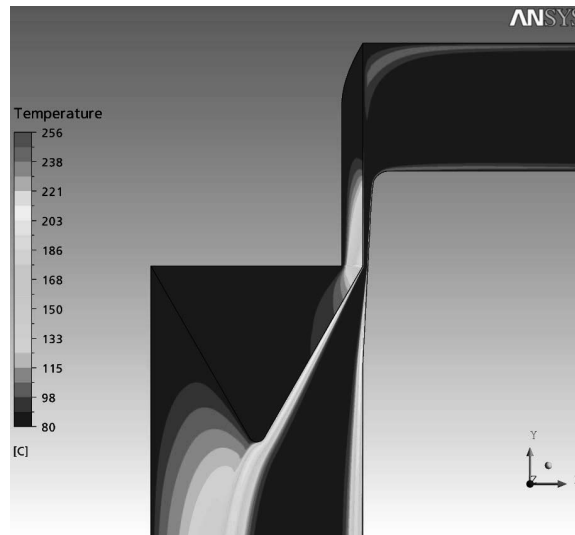


Bild 2: Simulation der Temperaturverteilung in der Drossel.

Mit der für diesen Einsatzfall speziell ausgelegten Geometrie wird im Gegensatz zu einer Runddüse das Maximum der Mischungstemperatur nicht an der Wand, sondern in der Mitte des Strömungskanals erreicht (Bild 2). Mit dem hier gezeigten System lassen sich sehr hohe Mischungstemperaturen realisieren, da der zweite Strömungskanal (E-Zylinder) nach jedem Einspritzzyklus restlos entleert wird und somit keine Gefahr besteht, den nächsten Spritzzyklus mit anverkrustetem Material zu verunreinigen.

4 Experimentelle Untersuchungen des Prototyps ohne Regelung

Den thermischen Effekt der Drossel zeigt Bild 3. Hier ist der Druck in der Mischung vor der Drossel für die Drosselstellungen 0 und 60 % für den Fall dargestellt, daß die Kolbengeschwindigkeit in beiden Versuchen gleich ist und die Mischung in Form eines dicken Gummistranges ins Freie ausgespritzt wird.

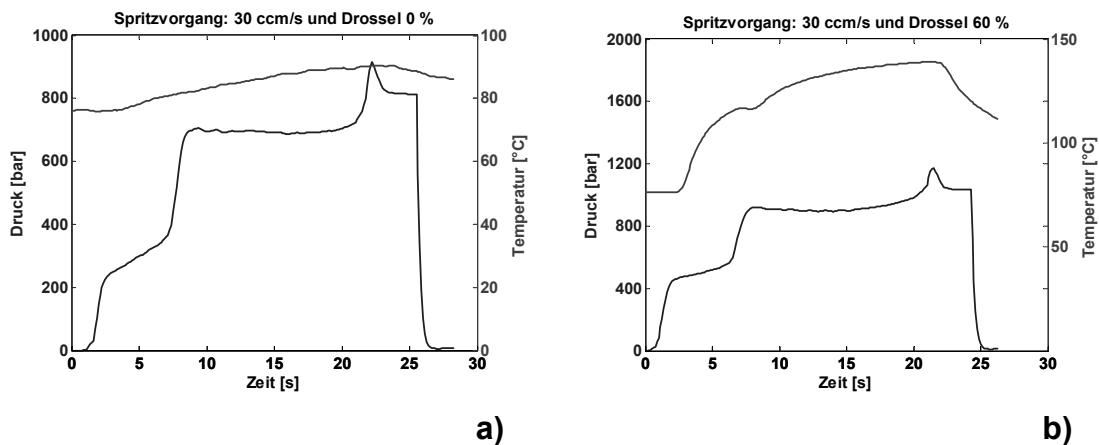


Bild 3: Typischer Verlauf von Druck und Temperatur an der Drossel für die Drosselstellungen 0 und 60 %.

Bei der Drosselstellung 60 % (Bild 3b) steigt der Druck in weniger als einer Sekunde auf etwa 400 bar. In den nächsten 3,5 Sekunden ist nur noch ein kleiner Druckanstieg zu

erkennen. In dieser Zeit wird der E-Zylinder gefüllt. Der dann folgende Druckanstieg auf etwa 900 bar resultiert daraus, daß die Mischung durch eine kleine Düse am Ende des E-Zylinders ins Freie gedrückt wird. Dies führt insgesamt zu einer Erwärmung der Mischung von 80 auf etwa 140 Grad. Bei der geöffneten Drossel (Bild 3a) ist nur ein Temperaturanstieg von etwa 10 Grad zu erkennen.

Die Abhängigkeit des Drucks von der Drosselstellung bei einem konstanten Volumenstrom Q von $40 \text{ cm}^3/\text{s}$ ist in Bild 4 dargestellt.

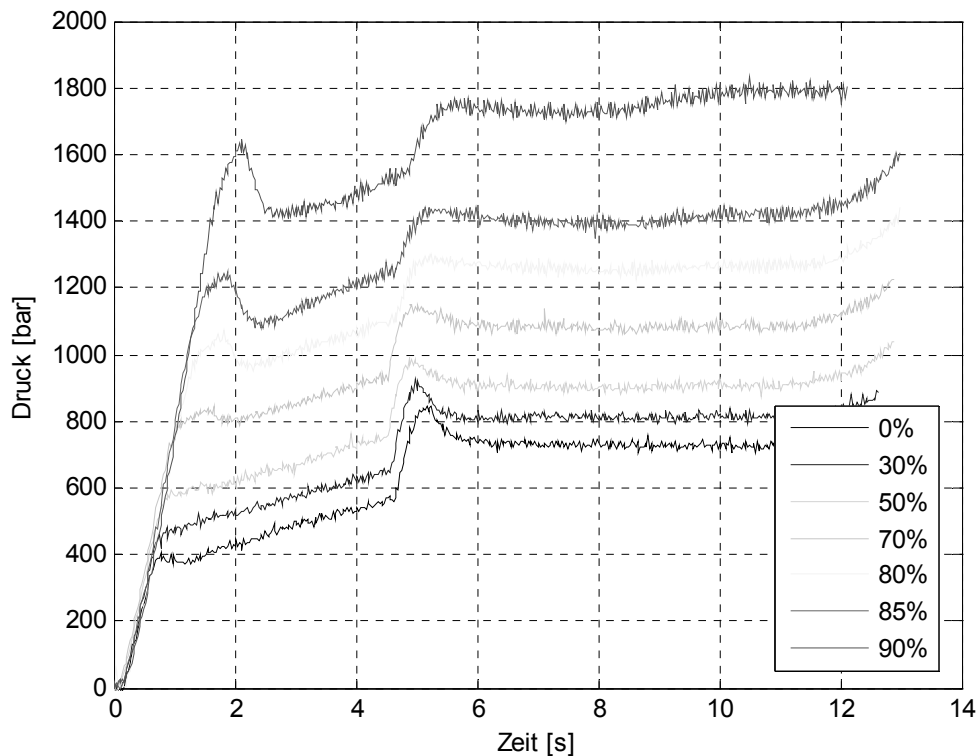


Bild 4: Druckverlauf vor der Drossel für Drosselstellungen von 0 bis 90 % bei einem Volumenstrom von $40 \text{ cm}^3/\text{s}$.

Bei der Drosselstellung 0% (Drossel offen) sind stationär etwa 700 bar notwendig um die Mischung aus der Düse der E-Einheit ins Freie zu spritzen. Bei einer Drosselstellung von 90% wird nach etwa 10 Sekunden der maximal zur Verfügung stehende Druck von 1800 bar erreicht. Der Druckanstieg am Anfang ist durch das Hydrauliksystem bestimmt und offensichtlich unabhängig von der Drosselstellung. Das erste Überschwingen im Druck - der Elastizitätsspeak - nimmt erwartungsgemäß mit kleiner werdenden Drosselöffnungen deutlich zu. Entgegengesetzt verhält sich der zweite Elastizitätsspeak bei etwa 4,5 Sekunden, der durch die Austrittsdüse verursacht wird. Hier ist das Überschwingen für 0% am größten und nimmt dann für kleinere Drosselöffnungen - also steigende Massetemperaturen - ab.

5 Regelungskonzept

Das Regelungsziel besteht eigentlich darin, die Kautschukmischung mit einer vorgegebenen Temperatur in eine Form einzuspritzen. In Bild 4 ist zu erkennen, daß die Mi-

sung bei einem Volumenstrom von $40 \text{ cm}^3/\text{s}$ erst nach etwa 4,5 Sekunden die Düse des E-Zylinders verläßt und in die Form eintritt. Erst zu diesem Zeitpunkt könnte die Temperatur der Mischung in der Form gemessen werden und der Regler reagieren. Die Temperatur der zu diesem Zeitpunkt im E-Zylinder befindlichen Mischung könnte dann allerdings nicht mehr beeinflußt werden. Die Temperaturregelstrecke hat also ein ausgeprägtes von der Spritzgeschwindigkeit abhängiges Totzeitverhalten.

Aus diesem Grund wird die Mischungstemperatur in der Form jetzt indirekt über den Druck vor der Drossel geregelt. Hierzu wurde zunächst mit Hilfe der in Abschnitt 2 besprochenen Zusammenhänge und den durchgeführten experimentellen Untersuchungen ein Prozeßmodell erstellt und validiert [2]. Mit diesem Modell kann jetzt für eine vorgegebene Temperaturerhöhung und Spritzgeschwindigkeit der notwendige Druckverlauf $p(t)$ vor der Drossel berechnet werden. Zur Berechnung der für die Temperaturerhöhung jeweils erforderlichen Einspritzleistungen berücksichtigt das Prozeßmodell neben der dissipierten Einspritzarbeit auch die mit CFD-Rechnungen für verschiedene Spritzgeschwindigkeiten und Systemtemperaturen ermittelten und experimentell verifizierten Kühlleistungen. Dieses Prozeßmodell benötigt als Eingangsgrößen das in den E-Zylinder eingespritzte Volumen V , die Spritzgeschwindigkeit, die Wandtemperaturen des Aggregates vor und hinter der Drossel sowie Druckverlust und Temperatur im Werkzeug.

Die bisherige Regelung der Geschwindigkeit v des Spritzkolbens in der EF-Einheit wird beibehalten und der mit Hilfe des Prozeßmodells berechnete Solldruckverlauf vor der Drossel wird mittels eines Fuzzy-Reglers geregelt, der als zweite Eingangsgröße auch das Volumen V erwartet.

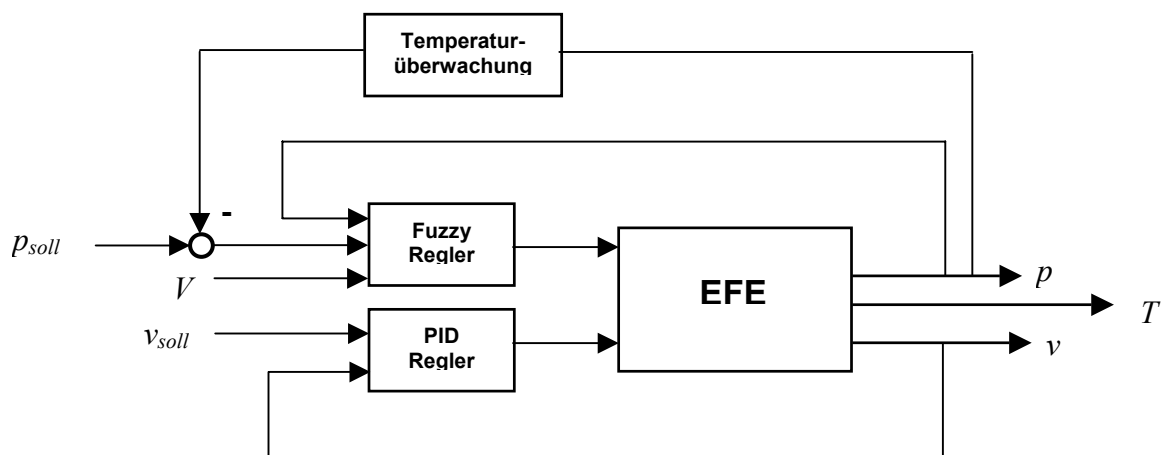


Bild 5: Blockschaltbild der MIMO-Regelung.

Die Ausgangsgröße des Fuzzy-Reglers ist die Drosselstellung; gleichzeitig wird die gemessene Temperatur T hinter der Drossel überwacht und bei Überschreitung eines Grenzwertes durch einen Eingriff der Überwachung abgesenkt (Bild 5).

Für die Regelung des Druckes im EF-Zylinder mit Hilfe der Drossel wurde aus folgenden Gründen ein Fuzzy-Regler gewählt: Der Zusammenhang zwischen Drosselstellung und Druck im EF-Zylinder ist besonders während der ersten Füllphase stark nichtlinear (Bild 4) und kann durch ein lineares Prozeßmodell nicht ausreichend gut beschrieben werden. Die minimale zulässige Öffnung der Drossel ist abhängig von der Spritzgeschwindigkeit und stellt eine weitere Nichtlinearität dar, so daß ein linearer Reglerent-

wurf nicht möglich ist. Im Fuzzy-Regler kann diese Stellgrößenbeschränkung im Entwurf direkt berücksichtigt werden.

Darüber hinaus kann das während der Simulationen und experimentellen Versuchen gewonnene Prozeßwissen mittels der von Zadeh in den 1960er Jahre entwickelte Fuzzy-Logik [4] in eine Regelbasis mit WENN-DANN-Regeln umgesetzt werden. Mit Hilfe eines entsprechenden Inferenzmechanismus [5] kann der Fuzzy-Regler diese Regeln dann auswerten und eine geeignete Drosselstellung berechnen. Für eine ausführliche Diskussion der Vor- und Nachteile von Fuzzy- und konventionellen Regelungen wird auch auf [6] verwiesen. Zur Realisierung des Fuzzy-Reglers wurde die Fuzzy Control Toolbox von Matlab und ein Rapid Control Prototyping System der Firma dSPACE [7] eingesetzt.

6 Der Fuzzy-Regler

Der Fuzzy-Regler wurde mit der Fuzzy-Toolbox von Matlab realisiert und benötigt als Eingangsgrößen das in den E-Zylinder eingespritzte Volumen sowie die Differenz zwischen Soll- und Istdruck. Mit Hilfe einer Mamdani-Inferenz wird aus diesen Eingangsgrößen die Stellgröße berechnet (Bild 6).

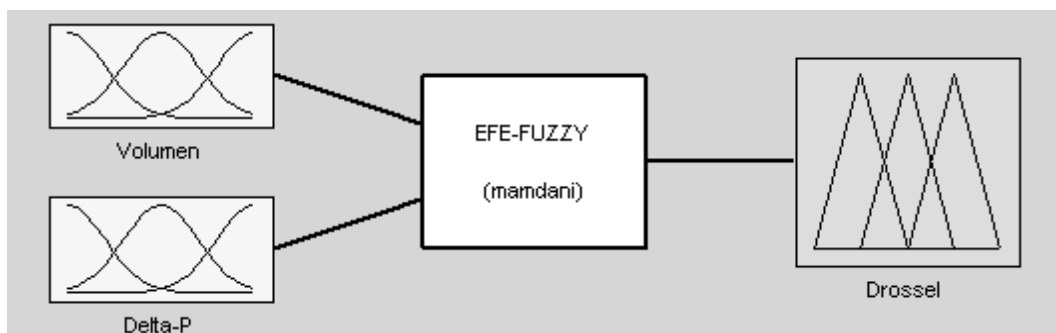


Bild 6: Struktur des Fuzzy-Reglers.

Die für die Fuzzifizierung der Eingangsgröße Volumen verwendeten Zugehörigkeitsfunktionen sind trapezförmig und in Bild 7 dargestellt.

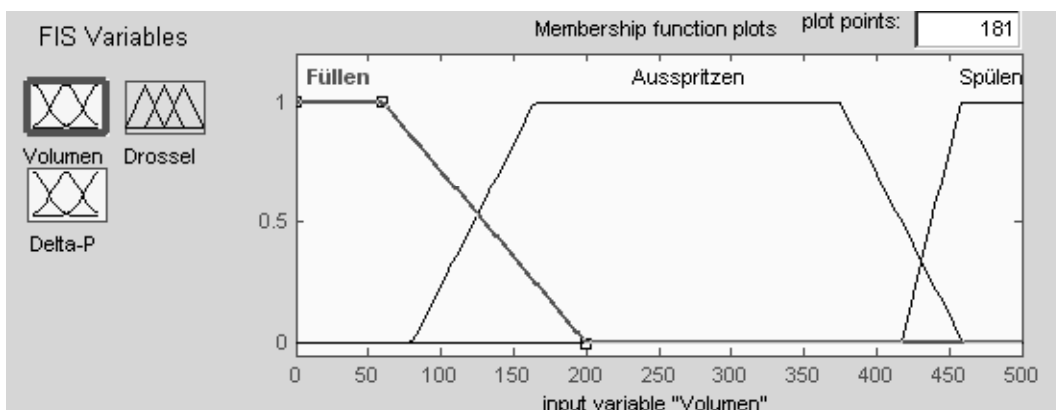


Bild 7: Zugehörigkeitsfunktionen der Eingangsgröße Volumen.

Mit diesen Zugehörigkeitsfunktionen wird der Spritzvorgang in drei Phasen unterteilt:

- Der E-Zylinder ist nach dem vorangegangenen Spritzzyklus komplett entleert und muß zunächst gefüllt werden. Die erste Zugehörigkeitsfunktion wird daher mit „Füllen“ bezeichnet.
- Nach dem Füllen des E-Zylinders tritt die Gummimischung aus der Maschinendüse des E-Zylinders aus. Dieser Phase wird die Zugehörigkeitsfunktion „Aus-spritzen“ zugeordnet.
- In der letzten Phase wird die Drossel auf etwa 20% geöffnet. Der Strömungswiderstand ist nun sehr gering (Bild 4), noch im Drosselbereich befindliche erwärmte Mischung wird durch kühle Mischung verdrängt und in den E-Zylinder gefördert. Da das eigentliche Drosselvolumen sehr gering ist ($\sim 1 \text{ cm}^3$), und der Drosselbereich sehr effektiv temperiert wird, reichen 20 - 30 cm^3 „kühler“ Mischung, um ein Anvulkanisieren oder gar Ausvulkanisieren sicher zu vermeiden. Diese Phase wird daher mit „Spülen“ bezeichnet.

Die gewählten Zugehörigkeitsfunktionen für die Fuzzifizierung der Differenz von Soll- und Istdruck zeigen Bild 8.

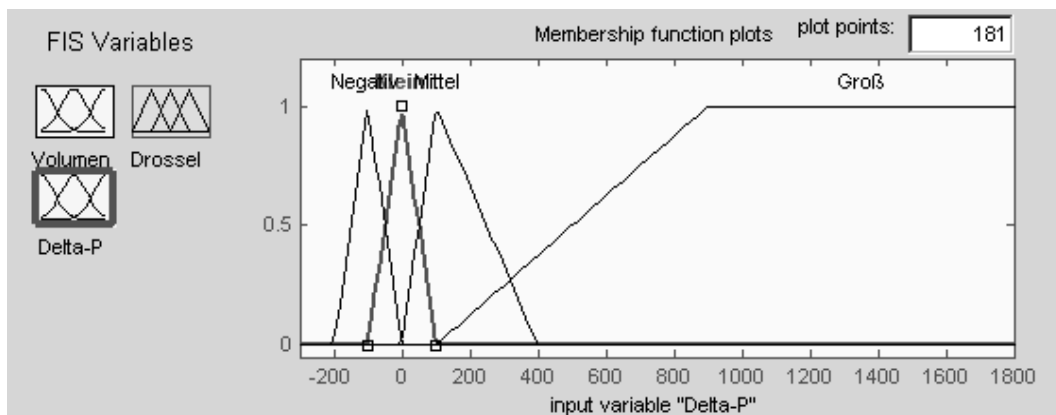


Bild 8: Zugehörigkeitsfunktionen der Eingangsgröße Delta_P.

Die Ausgangsgröße des Fuzzy-Reglers ist die Drosselstellung in Prozent, wobei 0 % voll geöffnet und 100 % voll geschlossen bedeutet. Für die Defuzzifizierung der Ausgangsgröße werden die in Bild 9 dargestellten Zugehörigkeitsfunktionen verwendet.

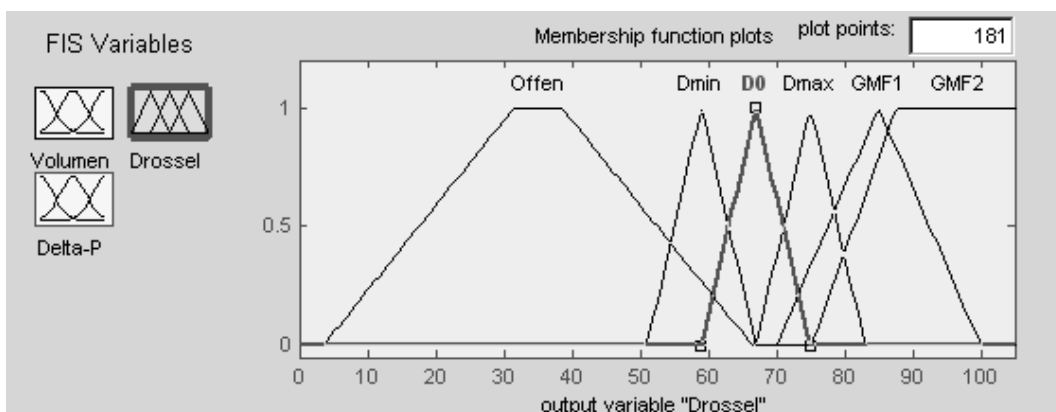


Bild 9: Zugehörigkeitsfunktionen der Ausgangsgröße Drosselstellung.

Die Zugehörigkeitsfunktion **Offen** wird für die „Spülphase“ eingesetzt und die Funktionen **GMF1** und **GMF2** für die „Füllphase“. Für die Regelung in der „Ausspritzphase“ stehen die Zugehörigkeitsfunktionen **Dmin**, **D0** und **Dmax** zur Verfügung. Die Regeln, die in der Ausspritzphase eingesetzt werden bilden das Verhalten eines Bedieners nach:

Wenn Istdruck > Solldruck DANN öffne Drossel

Wenn Istdruck < Solldruck DANN schließe Drossel

Wenn Istdruck ~ Solldruck DANN verändere Drossel nicht

Durch die Funktionen **Dmin**, **D0** und **Dmax** wird vorgegeben in welchen Bereichen die Drosselstellung variieren kann. Unzulässige Drosselstellungen können daher nicht auftreten. Je nach Arbeitsbereich (Solldruck, Kolbengeschwindigkeit) kann man diese Funktionen nach rechts oder links verschieben. Damit kann relativ einfach eine automatische Adaption realisiert werden.

7 Experimentelle Ergebnisse mit Regelung

Bei den Experimenten konnte auf ein bereits existierendes Aggregat der Firma LWB zurückgegriffen werden, bei der eine Schneckenkolbeneinheit (EF) in Kombination mit einer zweiten Kolbeneinheit (E) eine vollständige Leerung der Maschine nach jedem Spritzzyklus garantiert. Ohne Gefahr, die nächste Maschinenfüllung mit anvulkanisiertem Material der vorherigen zu verunreinigen, können jetzt mit der Regelung reproduzierbar und langzeitstabil Temperaturerhöhungen bis an die Anvulkanisationsgrenze der jeweiligen Mischung realisiert werden.

Für die im weiteren gezeigten Messungen wurde ein dickwandiger Probekörper konzipiert, bei dem an drei verschiedenen Meßstellen im Werkzeug Temperaturen gemessen werden können. Bei dem Probekörper handelt es sich um einen Zylinder mit 80 mm Durchmesser und 100 mm Höhe und einem Volumen von etwa 525 cm³. Meßfühler messen die Temperaturen im Zentrum sowie jeweils 15 mm von der Stirn- bzw. Zylinderwand entfernt. Eine identische Meßfühleranordnung wird auch benutzt um die Temperaturen im Inneren nach der Entnahme aus dem Werkzeug während des Abkühlvorganges zu verfolgen.

Bild 10 zeigt Druck- und Temperaturverlauf für einen Spritzversuch in die zylindrische Form mit 40 cm³/s und voll geöffneter Drossel. Der gezeigte Temperaturverlauf wurde mit dem Sensor gemessen, der etwa 15 mm von der Zylinderwand entfernt ist. Die Mischung hatte vor dem Einspritzen eine Temperatur von 84 Grad. In der Form wurde nach einer Minute eine Temperatur von 112,5 Grad gemessen, so daß sich bei geöffneter Drossel ein Temperaturanstieg von 28,5 Grad ergab. Bezogen auf den maximalen gemessenen Druck von etwa 850 bar ergibt sich ein Temperaturanstieg von 3,35 Grad pro 100 bar. Bezogen auf den durch Integration ermittelten mittleren Druck von 727 bar ergibt sich eine Temperaturerhöhung von 3,92 Grad /100 bar.

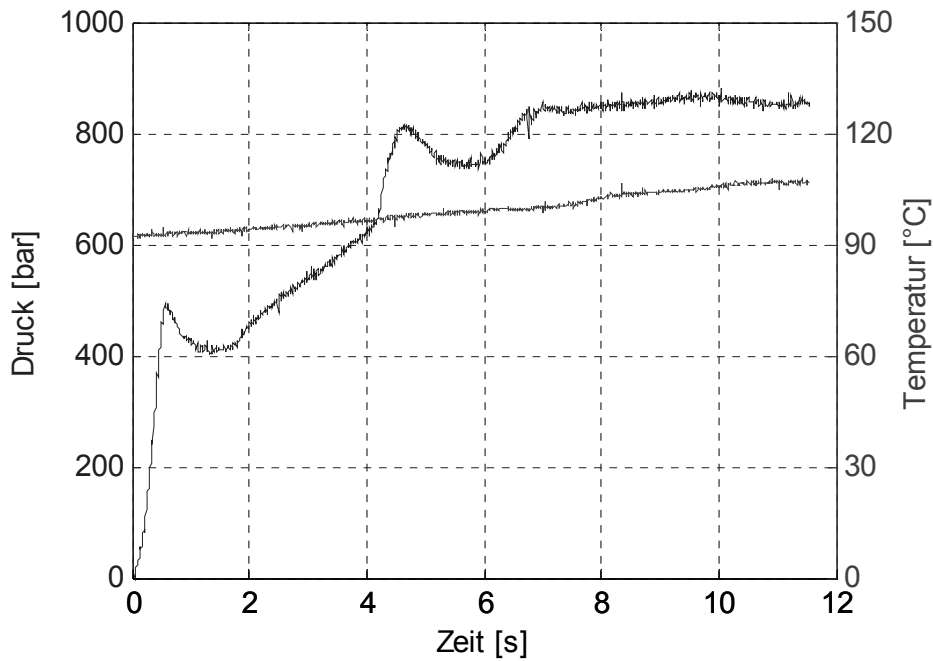


Bild 10: Druck- und Temperaturverlauf für $Q = 40 \text{ cm}^3/\text{s}$ und Drossel 0 %.

Die Ergebnisse eines Versuchs mit dem Fuzzy-Regler sind in Bild 11 dargestellt. Hier wurde der Solldruck in etwa 4 Sekunden von 600 auf 1000 bar erhöht und dann konstant gehalten. Der Istdruck folgt dem Solldruck nach etwa 1,5 Sekunden sehr gut. Die Mischung hatte vor dem Einspritzen eine Temperatur von 85 Grad. In der Form wurde nach einer Minute eine Temperatur von 122 Grad gemessen, so daß sich für 1030 bar ein Temperaturanstieg von 37 Grad ergab. Bezogen auf den mittleren Druck von 942 bar ergibt sich wiederum ein Temperaturanstieg von 3,92 Grad pro 100 bar.

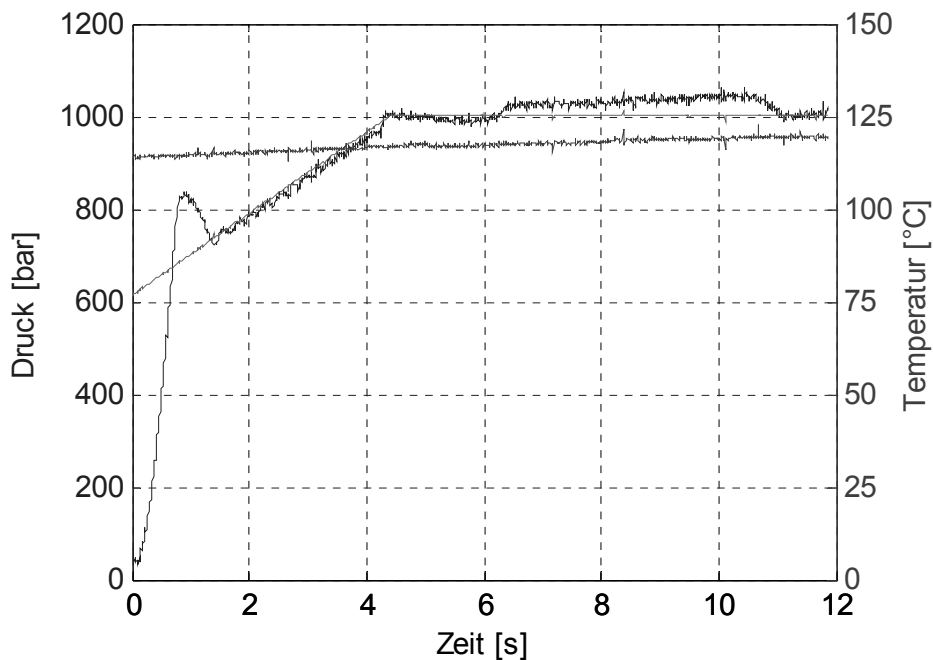


Bild 11: Druck- und Temperaturverlauf für $Q = 40 \text{ cm}^3/\text{s}$ und einem Solldruck von 1000 bar .

Einen weiteren geregelten Versuch zeigt Bild 12. Hier wurde der Solldruck in etwa 4 Sekunden von 800 auf 1300 bar erhöht und dann konstant gehalten.

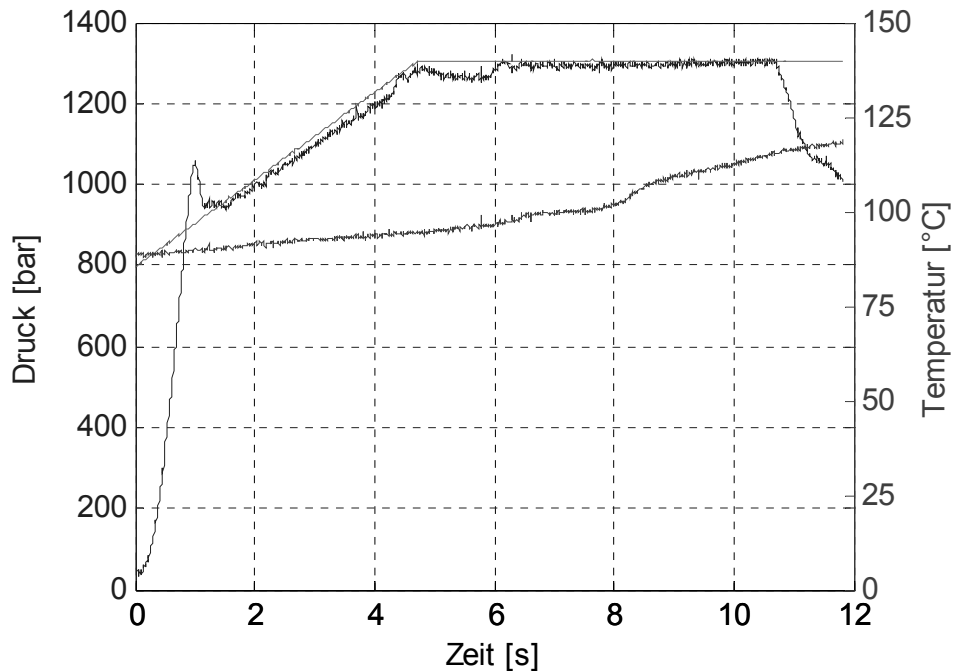


Bild 12: Druck- und Temperaturverlauf für $Q = 40 \text{ cm}^3/\text{s}$ und einem Solldruck von 1300 bar.

Der Istdruck folgt auch hier dem Solldruck nach etwa 1,5 Sekunden sehr gut. Die Mischung hatte vor dem Einspritzen wiederum eine Temperatur von 85 Grad. In der Form wurde nach einer Minute eine Temperatur von 130 Grad gemessen, so daß sich für einen stationären Druck von 1300 bar ein Temperaturanstieg von 45 Grad ergab. Bezogen auf den mittleren Druck von 1163 bar ergibt sich ein Temperaturanstieg von 3,86 Grad pro 100 bar.

Abschließend im Bild 13 ein Versuch mit einem Endsolldruck von 1600 bar. Der Istdruck folgt hier dem Solldruck nach etwa 2 Sekunden sehr gut. Die Mischung hatte vor dem Einspritzen eine Temperatur von 84 Grad. In der Form wurde nach einer Minute eine Temperatur von 138,2 Grad gemessen, so daß sich für 1600 bar ein Temperaturanstieg von 54,2 Grad ergab. Bezogen auf den gemittelten Druck ergibt sich wiederum ein Temperaturanstieg von 3,80 Grad pro 100 bar. Die etwas niedrigeren Werte bei höheren Ausgangstemperaturen sind durch die höhere Kühlleistung im E-Bereich verursacht und können im Prozeßmodell berücksichtigt werden.

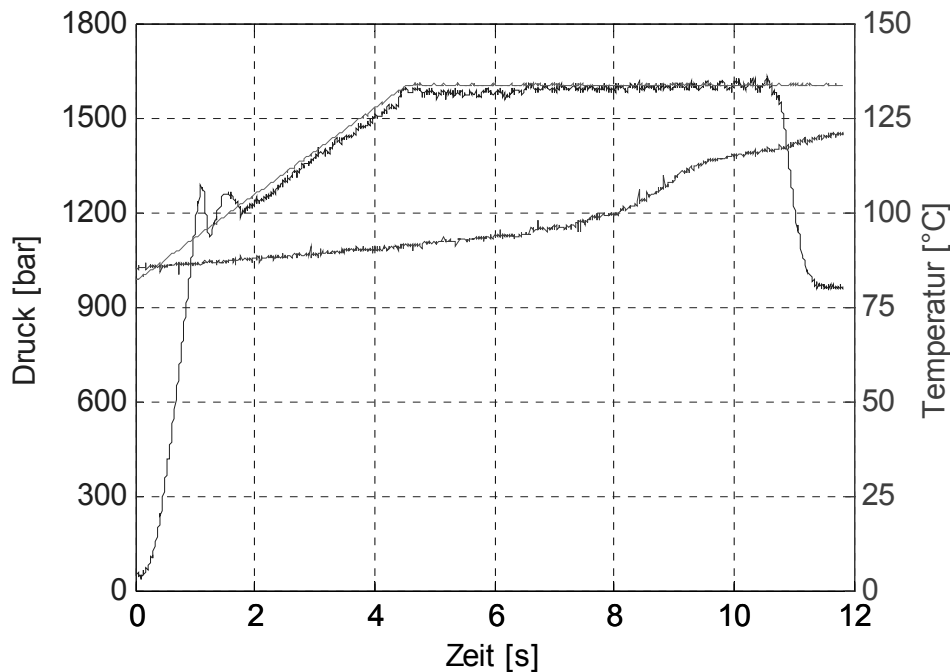


Bild 13: Druck- und Temperaturverlauf für $Q = 40 \text{ cm}^3/\text{s}$ und einem Solldruck von 1600 bar.

Begleitende Simulationsrechnungen zeigten, daß bei entsprechender Vorwahl der Starttemperatur und der Einbeziehung des Temperaturverlaufs während der Abkühlphase eine vollständige Durchvulkanisation des Zylinders in weniger als 300 s möglich ist. Diese Simulationsergebnisse konnten in unseren experimentellen Untersuchungen bestätigt werden: in Vulkanisationsversuchen mit geregelter Drossel und mit einer normal beschleunigten EPDM-Mischung wurden bei einer Werkzeugtemperatur von 165 °C reproduzierbar Fülltemperaturen von 160 °C erreicht und Heizzeiten von 200 s realisiert. Dies bedeutet, daß die Heizzeit im Vergleich zu einer konventionellen Maschine um den Faktor 6 reduziert werden konnte.

8 Automatische Adaption des Fuzzy-Reglers

Es ist bekannt [8], daß sich für Fuzzy-Regler häufig einfache Adaptionalgorithmen finden lassen. Der eingesetzte Fuzzy-Regler ist im Prinzip ein P-Regler mit einer variablen Verstärkung, die von den beiden Eingangsgrößen abhängig ist. Er enthält also keine integrierenden Anteile, die eine stationäre Genauigkeit erzwingen. Über die Zugehörigkeitsfunktion **D0** in Bild 9 wird eingestellt, welche Drosselstellung bei einer Regelabweichung von Null ausgegeben wird. Die optimale Drosselstellung für stationäre Genauigkeit ist jetzt aber vom Solldruck, der Einspritzgeschwindigkeit und der Mischung abhängig (vgl. Bild 4).

In Abhängigkeit vom stationären Druckfehler kann man die Zugehörigkeitsfunktionen **Dmin**, **D0** und **Dmax** in Bild 9 jetzt von Zyklus zu Zyklus mit einfachen Regeln adaptieren: Verschiebe die Zugehörigkeitsfunktionen nach links, wenn der Istdruck größer

als der Solldruck ist und nach rechts im umgekehrten Fall. Stabilitätsprobleme können nicht auftreten, solange man darauf achtet, daß **Dmin** nicht kleiner Null bzw. **Dmax** nicht größer als 90 % bei z.B. einem $Q = 40 \text{ cm}^3/\text{s}$ wird

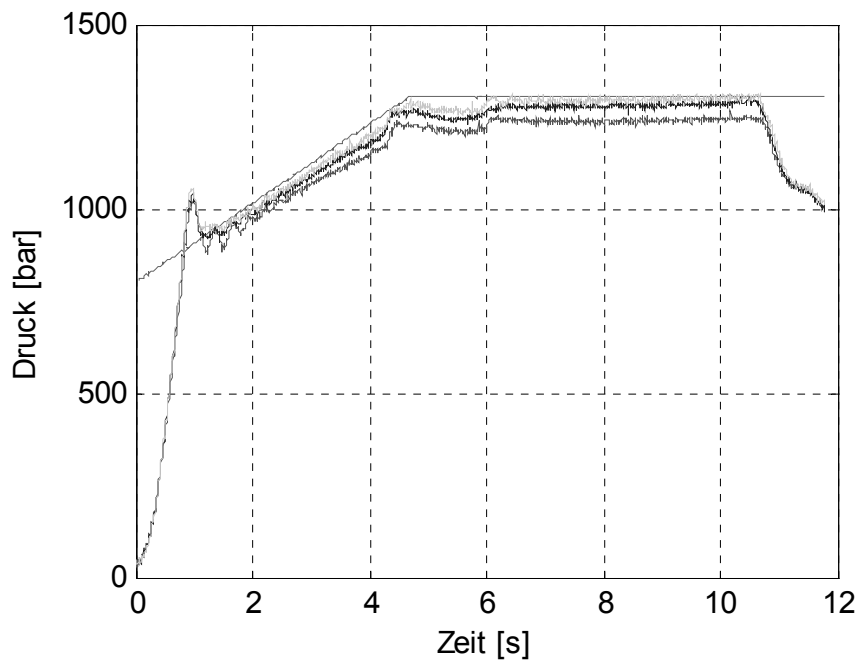


Bild 14: Adaptionsergebnisse für $Q = 40 \text{ cm}^3/\text{s}$ und einem Solldruck von 1300 bar.

In Bild 14 sind Ergebnisse für den Fall dargestellt, daß der gewählte Drosselarbeitspunkt für den eingestellten Solldruck zu klein ist, die Drosselöffnung also zu groß ist. Nach 2 Adaptionsschritten wird eine gute stationäre Genauigkeit erreicht.

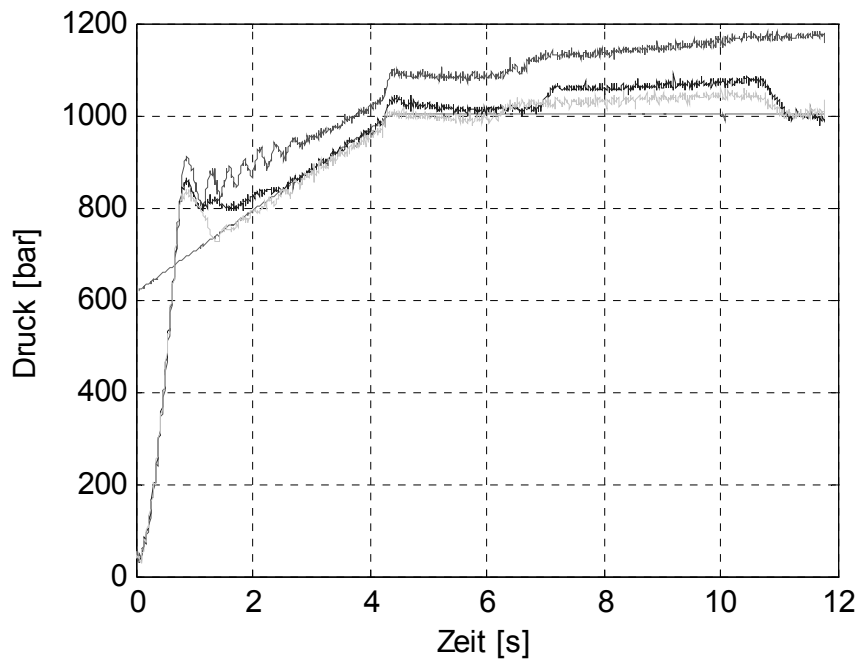


Bild 15: Adaptionsergebnisse für $Q = 40 \text{ cm}^3/\text{s}$ und einem Solldruck von 1000 bar.

Adaptionsergebnisse für den umgekehrten Fall zeigen Bild 15. Hier wurde der Drosselarbeitspunkt, der für einen Solldruck von 1500 bar gute Ergebnisse liefert, beibehalten und ein neuer Solldruck von 1000 bar vorgegeben. Auch hier ist nach zwei Adaptionsschritten schon ein guter Drosselarbeitspunkt gefunden.

9 Zusammenfassung

In diesem Beitrag wurde eine Anwendung der Fuzzy-Logik im Bereich der Regelung von Gummispritzpressen vorgestellt. Die gezeigten Ergebnisse wurden im Rahmen einer erfolgreichen Zusammenarbeit des mittelständischen Unternehmens LWB und der Universität der Bundeswehr München erzielt. Die konsequente Anwendung moderner regelungstechnischer Methoden (MIMO Control, Fuzzy Control, Rapid Control Prototyping) haben dazu beitragen, daß in kurzer Zeit ein neues Verfahren zum Spritzen und schnellem Vulkanisieren großer Gummiteile realisiert und erprobt werden konnte.

Auf der Internationalen Kautschuktagung (IRC 2009) in Nürnberg hatte die Firma LWB Steinl GmbH & Co. KG erstmalig ein Spritzaggregat mit der in diesem Beitrag vorgestellten Technologie präsentiert und für diese Entwicklung auf der Tagung den begehrten Product Award 2009 für innovative Erzeugnisse der Kautschukindustrie der Deutschen Kautschuk-Gesellschaft (DGK) erhalten.

Literatur

- [1] F. Röthemeyer und F. Sommer: *Kautschuktechnologie*. Hanser-Verlag, 2001.
- [2] V. Härtel, F. Svaricek, J. Gogolin and Th. Nieberl: A New Approach to High Performance Injection Moulding. *International Rubber Molding Conference*, Chicago, IL, April 8/9,
- [3] J. Gogolin, V. Härtel und F. Svaricek: Rheologische Auslegung eines Rapid Cure Systems: Numerische Simulation und Experimente. DKG-Fachtagung, Fulda, 17. Juni 2008.
- [4] Zadeh, L.A.: Fuzzy Sets. *Information and Control* 8 (3), 1965, S. 338-353.
- [5] T. Bertram, F. Svaricek, T. Bindel, R. Böhm, H. Kiendl, B.- M. Pfeiffer und M. Weber: Fuzzy Control: Zusammenstellung und Beschreibung wichtiger Begriffe. *Automatisierungstechnik* 42, 1994, S. 322-326.
- [6] T. Bertram und F. Svaricek: Fuzzy versus konventionelle Regelung eines inversen Pendels. *GMA-Aussprachetag "Nichtlineare Regelung"*, Langen, Mai 1993, (VDI-Berichte Nr. 1026, S. 329-338).
- [7] H. Hanselmann: Beschleunigte Mechatronik-Entwicklung durch Rapid Control Prototyping und Hardware-in-the-Loop-Simulation. *Automatisierungstechnik* 46, 1998, S. 113-119.
- [8] T. Bertram und F. Svaricek: Eine robuste adaptive Fuzzy-Kaskadenregelung für ein inverses Pendel. *GMA-Aussprachetag "Fuzzy Control"*, Langen, 1994, (VDI-Berichte Nr. 1113, S. 439-448).

Optimization and Online-Monitoring in industrial batch processes using Data-Mining methods

Christian Kühnert, Thomas Bernard

Fraunhofer-Institut für Informations- und Datenverarbeitung IITB

Fraunhoferstraße 1

Tel. (0721) 6091-9

{Christian.Kuehnert, Thomas.Bernard}@iitb.fraunhofer.de

Abstract

In general industrial batch processes are complex systems that have to be optimized due to several performance criteria. As control optimization based on the development of physical models is in many cases very time consuming and cost intensive or not even feasible an alternative approach consists in analyzing historical process data by means of computational intelligent methods. The aim is the identification of characteristic control patterns and classification of these patterns according to their contribution to process optimization. This paper presents a concept how the generated knowledge can be used for online monitoring production runs. The key idea is to derive a model from the historical process data which describes the impact of the most important features of the process variables (both manipulated and controlled variables) to a user-defined performance index (e.g. quality or losses of a production run). The set of relevant features is automatically found using an iterative algorithm based on Support Vector Machines. Further the model is used to calculate the optimal values of the relevant features for several segments. Based on the optimal values of the features an online monitoring of the process can be implemented. The proposed concept is applied to an industrial glass forming process.

1 Introduction

In general industrial batch processes are complex systems that have to be optimized due to several performance criteria. As control optimization based on the development of physical models is in many cases very time consuming and cost intensive or not even feasible an alternative approach consists in analyzing historical process data. Hereby the aim is the identification of characteristic control patterns and the classification of those patterns according to their contribution to process optimization. To reveal this information Data Mining techniques, which include machine learning theory, statistics and artificial intelligence, are increasingly used to find unknown and hidden correlations in the data [10], [14]. For optimization of complex systems these methods can be supplementary used besides physical models in order to determine unknown interconnections and important features in a process and thus can efficiently be applied for the optimization of process performance.

The key idea is to derive a model from the historical process data which describes the impact of the most important features of the process variables (both manipulated and controlled variables) to a user-defined performance index (e.g. quality or losses of a

production run). A concept for a model generation and ranking of the relevant features has been already presented in [3], [4]. This paper is based on this concept, which is summarized in section 2. The new idea for an online monitoring is presented in section 3. Finally an application of the online monitoring to an industrial glass forming process is discussed in section 4.

2 Calculation of optimal features as a basis for online monitoring

This section summarizes a concept for model generation and ranking of relevant features [3], [4]. This model can then be used as the root for the online monitoring concept proposed in this paper. The model should describe the impact of the features to a user-defined performance index (see Fig. 1).



Fig. 1 Basis for the online monitoring is a subset of relevant features and a model which describes the impact of the features to a user-defined performance index. Both the relevant features as well as the model are derived from historical process data

For the model generation the work flow as shown in Fig. 2 has to be performed. In a first step raw process data has to follow some preprocessing steps (outlier removal, resampling and normalization of the data). This helps to increase the data quality coming from the process as well as reducing the amount of stored data [5]. Then new signals are generated (e.g. derivatives or combining different time series).

In a next step features have to be generated. This includes features coming from the complete time series (e.g. mean values over the whole production run) or features extracted from segments of the measured data. Simple examples of features are minima,

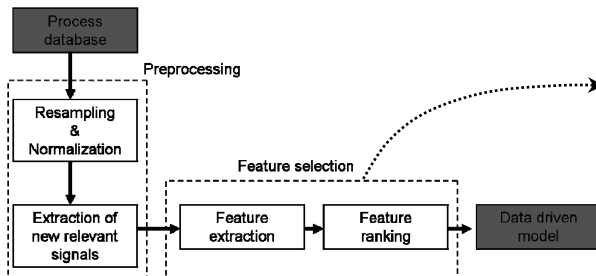


Fig. 2: Steps from measured process data to a data driven Model

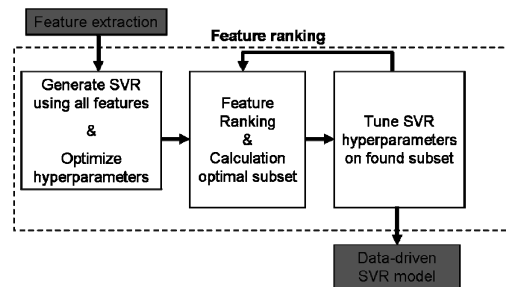


Fig. 3: Detailed work flow of feature ranking

maxima, mean values, trends of process variables or mathematically generated features like parameters from regression models. As not all of the extracted features have a meaningful relevance regarding the optimization of the process or tend to be strongly redundant to other features, a ranking procedure and the selection of an optimal feature subset has to be done. Feature ranking can be performed by discriminant or variance analysis as well nonlinear techniques [7], [8].

During all these steps it is necessary to include as much expert knowledge as possible, as the generation of suitable features is of crucial importance for the later derivation of a reliable model. The feature ranking and model building, as shown in detail in figure 3 is

performed in several iterative steps. First a model based on Support Vector Machines using all features is generated. Then a first feature ranking is performed based on the support vectors coming from the model. Out of this ranking a first subset of features is calculated and the unimportant features are removed. Using the left features in a next step a new model is calculated and the model parameters are adapted accordingly. This procedure is repeated several times until the subset of features has converged to a final number. At last the resulting SVR model is calculated. This model will be used furthermore for the calculation of restrictions for online monitoring.

SVMs have the opportunity that a feature ranking can be integrated and besides the ranking procedure a nonlinear model can be generated by including kernel functions [11], [12]. As implementation least square Support Vector Machines are used [13].

By using the prior described concept a SVR model based on the main features of the process is derived. Afterwards this model is used to calculate the feature values corresponding to the wished value of the performance index. Usually the used performance index has to be maximized (e.g. efficiency) or minimized (e.g. lost raw material) to find an optimal control pattern. To calculate extreme values an algorithm based on the Nelder-Mead simplex search [9] is used. This method is characterized to be robust concerning local minima as it doesn't use gradients.

3 Concept for online monitoring of production processes

The key idea of the online monitoring is that the calculated optimal features (which are calculated online during the process) can give a prediction of the process performance. Hence if the predicted process performance tends to be out of the specifications the operators have the opportunity to modify the control structure or process parameters. For an overview about first research topics about online monitoring for production process using Data Mining methods see [6].

The described procedure in section 2 delivers a SVR model based on the relevant features of the process. Furthermore this model is used to calculate the optimal feature values. For online monitoring purposes this model has to be further adapted. A first adaptation is carried out by selecting only interpretable features and features that can be calculated during a production. Furthermore selecting different time segments in which these features are calculated has to follow some requirements as well. Figure 4 gives an overview of the different steps that have to be carried out to make the Data Mining process capable for online monitoring.

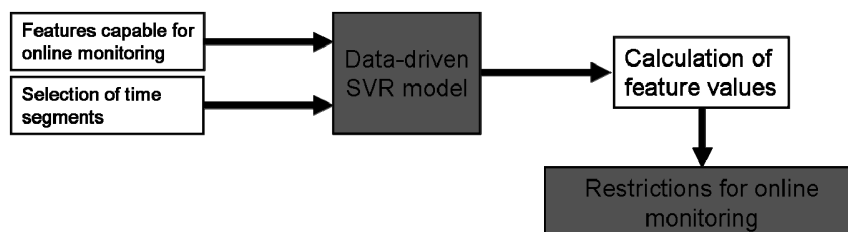


Fig. 4: Calculation of restrictions for online monitoring production processes.

3.1 Feature requirements

Features have to fulfill two major requirements. First these features have to be interpretable already during the production process. For example regression coefficients and features coming from second derivatives of the production data will be neglected. This implies already that a trade-off has to be done between the information included in a feature and its interpretability, as usually more complex features include more information. Furthermore some features can only be calculated if the production or the selected segment is already finished (i.e. mean values). That's why these features can't be used as well for giving restrictions for online monitoring the process. It has to be mentioned that reducing the available features for ranking and model building usually reduces the accuracy and increases the number of selected features. As an example for the behavior of features over time figure 5 shows the behavior of features calculated from a step response coming from a PT2 with a delay time of 1 and a damping of 0.5. By looking at the feature value behavior this shows already the difficulties for an interpretable feature selection for online monitoring. As an example the plotted values of the range and the mean of the second derivative are not intuitively clear and thus it is not sure on how to react if a production runs out of its optimal control pattern and doesn't follow its optimal values anymore. As a conclusion these features have to be removed from the analysis.

3.2 Selection of time segments

Another important requirement is the proper selection of time segments. The features shown in figure 5 only have limited information about the underlying PT2 system as they take into account the whole production process. As example the value of the range reaches its maximum as the PT2 reaches its maximum as well and doesn't change afterwards. Hence after the PT2 has passed its maximum the feature doesn't deliver new information and the feature value can't be used for describing the production further. On account of this the selection of the right segmentation is important for the later use of features for online monitoring. There are two different options for the segmentation of a graph. The first option is to select a fixed time length; the second option is to define

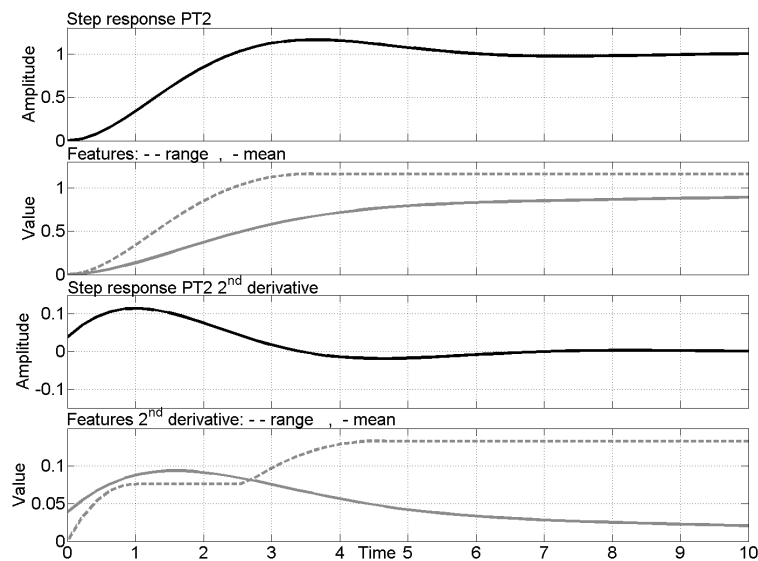


Fig. 5: Examples of features behaviour calculated from process data over time.

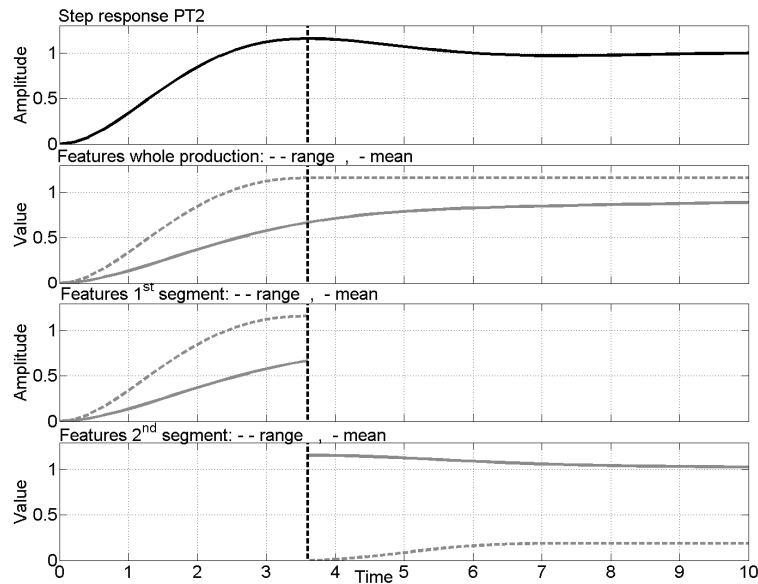


Fig. 6: Features calculated from process data after segmentation

landmarks regarding the production process. Landmarks, like local maxima or minima usually deliver better results as they can take into account more information about the underlying process. Figure 6 shows the prior defined PT2 system where the step response is segmented at the maximum for the generation of new feature values. Features coming from the second segment bring in new information compared to features describing the whole production. From these features the most significant ones have to be selected and their optimal features values can be calculated using techniques as described in section 2.

3.3 Online analysis of feature values

To perform an online monitoring of a production the selected and afterwards calculated optimal feature values can be used. As each feature symbolizes a segment during one production it can give clear restrictions for a production during a segment. Each time one segment is finished, the selected features can be used for the next segment to give restrictions for optimal control. For monitoring a process the generation of features, the selection of a small subset and the calculation of their values helps to focus on the most important process parameters for keeping the production inside its optimal control. The generation of these restrictions and how they can be used for online monitoring the process is shown in the following section on a rheological glass forming process.

4 Application to an industrial glass forming batch process

4.1 Underlying process and data set

The regarded industrial batch process is used for the production of glass tubes out of thick walled blank cylinders [1], [2]. The tubes must meet the desired specifications very precisely. For the production the cylinders are fed into the furnace and thin tubes are pulled out with the velocity v . The furnace is heated up to the temperature ϑ_{oven} . Using the heating power P_{oven} . To achieve the wished diameters D the productions are controlled by ϑ_{oven} and v . The temperature of the glass is measured with the parameter ϑ_{glass} (figure 8).

The investigated process has strong radiation effects and the material properties are changing with time which leads to a nonlinear and time variant behavior. As glass tubes not fulfilling these specifications can't be used the aim has to be to loose as less material as possible during the start-up of a production. That's why as performance index the cumulated lost glass volume is considered until the tube diameter enters a predefined tolerance band close to its nominal diameter D . The investigated dataset consists in 850 productions. Before feature extraction the measured time series P_{oven} , \mathcal{G}_{oven} , \mathcal{G}_{glass} and v , are presampled, low-pass filtered and outliers are removed. Figure 7 shows an exemplary production of one glass tube production. For the analysis two landmarks are defined which leads to 4 different segments plus the whole production process. The following section describes how to generate at first a preliminary model using as much information as available to perform a first analysis of correlations. In a further step a model only based on features that are interpretable and can be used for online monitoring are selected. Finally the optimal feature values are calculated using the prior described simplex algorithm and restrictions are generated for online monitoring the process.

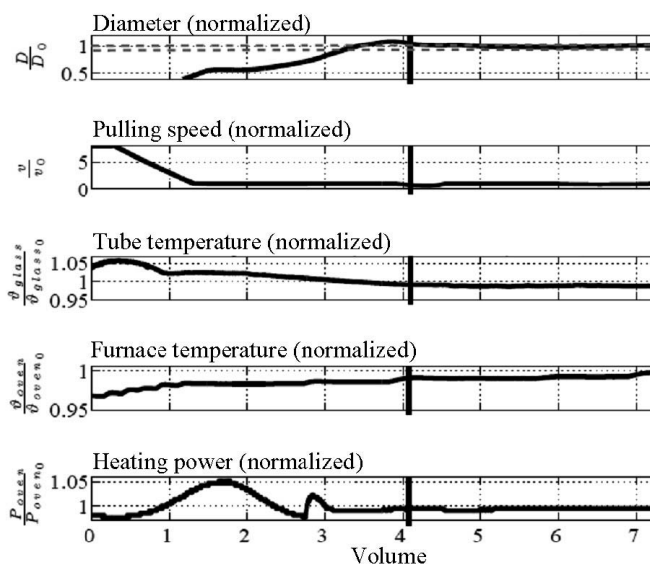


Fig. 8: Exemplary glass tube production. The vertical separation line is placed where the diameter D of the glass tube enters the defined tolerance band

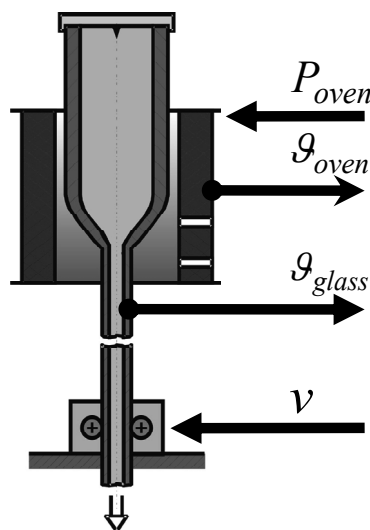


Fig. 7 Industrial batch process for the production of glass tubes.

4.2 Calculation of a preliminary SVR model

First a Data Mining analysis is performed using as much information as possible. This step is necessary to survey if there are any correlations at all between the defined performance index and the measured process parameters.

As this analysis won't be used for online monitoring, artificial features can be used as well and therefore from every time series the first and second derivative is taken. In detail from each process data measured the following six features are extracted: Minimum, maximum, mean value, range and regression coefficients (gradient and offset) coming from a linear regression model.

The results of the fund feature subset as well as the information included in each SVR model are shown in table 1. Using cross-validation techniques the final information included in the SVR model shows a fit of 79 % compared to real process data. This indicates strong correlations between the process data, in depth the tube temperature ϑ_{glass} and the pulling speed v , regarding the defined performance index. The pulling speed is used to control the diameter of the glass tube and that's why the correlations can be explained between the achieved quality and the pulling speed. Interesting is that all other features only depend on the first derivative of the tube temperature ϑ_{glass} . That means that not the actual tube temperature has an influence on the quality of the produced glass tubes but the temperature fluctuation. Regrettably almost only complex features that can't be used for online monitoring a production are selected for model building. Furthermore some features (i.e. features ranked 1 and 2) can only be calculated performing an offline analysis of the productions. Therefore in a next step a SVR model including only features capable for surveying the process online is calculated.

Table 1
Selected feature subset for preliminary SVR model

Rank	Name of feature	Declaring real process (cum)
1	1 st Derivative Gradient of ϑ_{glass} after entering tolerance band	48%
2	1 st Derivative Mean v after entering tolerance band	59%
3	1 st Derivative Mean ϑ_{glass}	70%
4	Range ϑ_{glass}	75%
5	1 st Derivative Maximum v	79%

4.3 Calculation of final SVR model and calculation of restrictions for optimal control

As described in section 3 for the generation of the final model only features and process parameters that can give intuitive results when the process is still running will be used. Regarding the preliminary SVR model only the process parameters v , ϑ_{glass} and the first derivative of ϑ_{glass} are considered for the generation of the SVR model. Furthermore, only minima, maxima and motion ranges will be extracted and the features coming from the segment after entering the tolerance band will be removed as well. Performing a second analysis leads to a bigger SVR model as now these features are less complex and therefore less information is included in each extracted feature. Comparing this model with the prior calculated one it can be seen as well that the new model doesn't reach the same amount of model fit as the prior one although more features are selected. Using cross-validation techniques a model fit results in 69 % compared to real process data. The resulting subset of features is shown in table 2. These features can now be used to calculate their optimal values and finally giving restrictions for optimal process control.

Table 2
Selected features for final SVR model

Rank	Name of feature	Restriction for optimal production (normalized)	Declaring process (cum.)
1	Range \mathcal{G}_{glass} before entering tolerance band	$< 0.085 $	31 %
2	1 st derivative Range \mathcal{G}_{glass} before entering tolerance band	$< 10.1 $	39 %
3	Range \mathcal{G}_{glass} after Diameter passes maximum	$< 0.023 $	46 %
4	1 st derivative Minimum \mathcal{G}_{glass} before entering tolerance band	> -7.3	54 %
5	Maximum v after diameter maximum	< 1.02	58 %
6	Maximum 1 st derivative \mathcal{G}_{glass} before entering tolerance band	< 5.62	62 %
7	Range v before entering tolerance band	$< 0.430 $	67 %
8	Maximum \mathcal{G}_{glass} before entering tolerance band	< 1.07	69 %

4.4 Using SVR model and features for online monitoring production processes

The calculated SVR-model can now be used for the extraction of optimal feature values. For this the prior described Nelder-Mead simplex algorithm is used. As the lost glass volume during one production should be minimized the algorithm is used to find the minimum of the 8 dimensional surface. For a brief overview figure 11 shows the first two ranked features including the defined performance index. Inside the minimum lies the optimal control pattern where the glass volume is minimized and depending on the values of the features, the production is running optimal or not.

The calculated optimal feature values can now be used for online monitoring the productions. For this restrictions are generated regarding the maximum value one feature can achieve and still stay inside the optimal control pattern. It is defined that a production is still running optimal if its value exceeds at most 20% from the optimal value found through the simplex algorithm. Finally restrictions as shown in table 2 can be drawn to describe if the process is running optimal or not. Figure 9 shows a production where the process data lies outside the restrictions. As the range from \mathcal{G}_{glass} is higher than the defined optimal value, it can be already said in the beginning of the production that more glass volume will be lost until the diameter of the tube enters the tolerance band. This leads to a slow settlement of the diameter and thus to a bad quality regarding the defined performance index. For a better view only the first three restrictions are drawn inside the plot. In contrary figure 10 shows a production where the optimal feature values don't leave the defined restrictions. Therefore this production has a fast settlement regarding the entering of the diameter into the tolerance band.

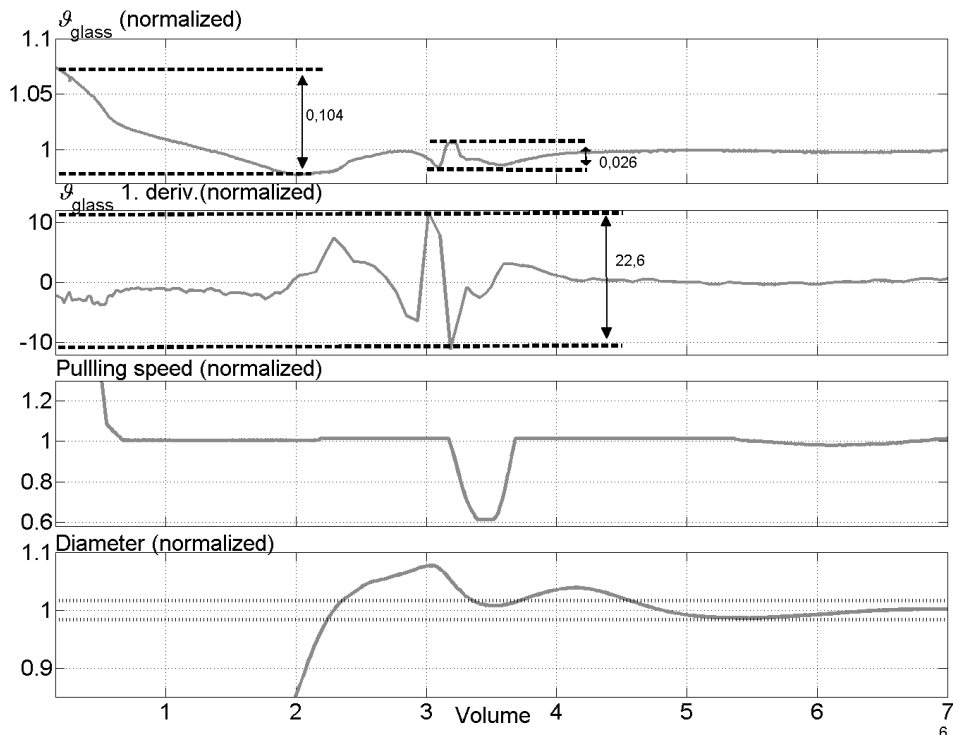


Fig. 9: Production running outside optimal control pattern.

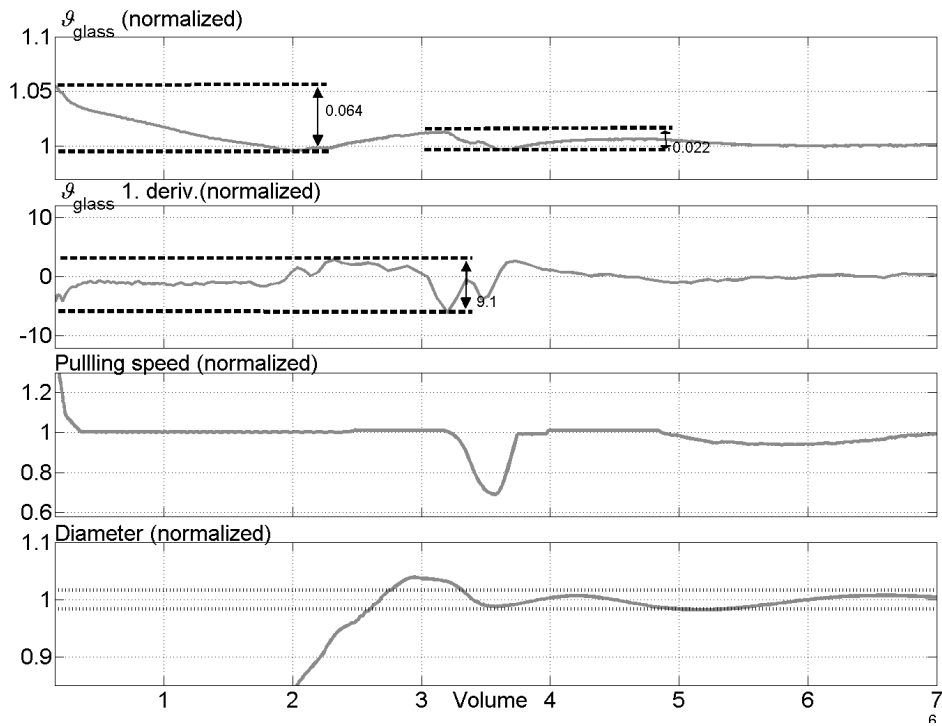


Fig. 10: Production running inside an optimal control pattern.

To confirm the found feature values over a broad range, finally 500 productions were taken from a database and tested if they fulfilled the restrictions during a production or not. Figure 12 shows productions fulfilling the restrictions defined in table 2 and plotted with productions not fulfilling these restrictions. Clearly an improvement is shown. Productions running optimal show a much faster settling towards the defined tolerance band and thus a significant improvement regarding the performance index.

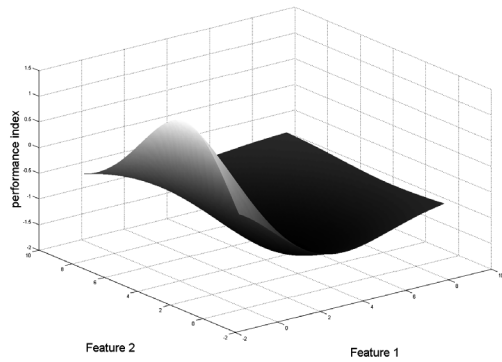


Fig. 11: Impact of the first two ranked features to the performance index

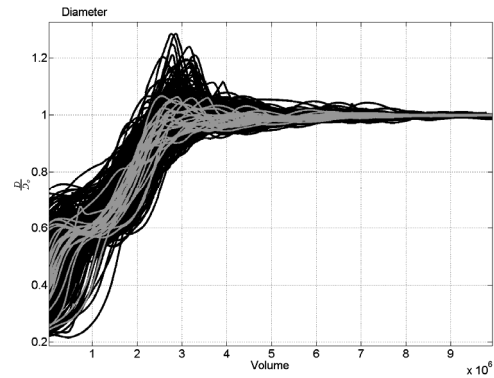


Fig. 12: Comparing productions running outside optimal control patterns (black) with productions fulfilling optimal feature values within a tolerance of 20% (grey)

5 Conclusion

In this paper a new data driven approach of analyzing and online monitoring a complex industrial production process is presented. The idea is in first place to detect hidden and not considered information in an historical database and to make the found knowledge available. Within here the generation of restrictions for online-monitoring production processes is presented. Therefore a prior SVR model is generated including a feature selection and feature ranking procedure. This calculated model including a subset of complex features can then be used to control if there are detectable correlations and interactions between different process parameters. Hereupon a model only based on easy to interpret feature values is calculated. Finally these values can be used in form of restrictions for online-monitoring the process. These restrictions are used to control if the process is running close to an optimal control pattern or not. As performance index the lost glass volume of an industrial glass forming process has been used to find correlated significant subset of features. Finally only a small number of features are needed to give clear advices on how to run the process optimal.

Future work will be concentrated on the steps how the optimal feature values can be used as soft sensors and be used directly for the design of new control concepts. Further research will also be based on a proper analysis of the found SVR model regarding generality (i.e. regarding different produced diameters) and security (i.e. does the model describe the whole feature space).

6 Literature

- [1] T. Bernard, E. Moghaddam, *Nonlinear Model Predictive Control of a Glass forming Process based on a Finite Element Model*, Proc. 2006 IEEE International Conference on Control Applications, (2006)
- [2] T. Bernard, M. Sajidmann, *A new Fuzzy based multiobjective optimization concept for process control systems*, Proc. 7th Fuzzy Days, International Conference on Computation Intelligence, 2007
- [3] C. Kuehnert, T. Bernard, *Extraction of optimal control patterns in industrial batch processes based on Support Vector Machines*, 2008 IEEE Multi-Conference on systems and control, 2009
- [4] C. Kuehnert, J. Minx, T. Bernard, H.-B. Kuntze: *Optimierung der Prozessführung komplexer verfahrenstechnischer Prozesse mit Support Vector Machines*, Automatisierungstechnische Praxis atp 51 (2009), 7, S. 47-49
- [5] M. Last, A. Kandel, H. Bunke, *Data Mining in time series databases*, World scientific publishing, 2004
- [6] R. Nikhil, J.Lakhmi, *Advanced Techniques in Data Mining and Knowledge Discovery*, Springer, 2005
- [7] R. Mikut, *Data Mining in der Medizin und Medizintechnik*, Universitätsverlag Karlsruhe, 2008
- [8] R. Mikut, *Data-Mining-Analysen mit der MATLAB Toolbox Gait-CAD*, at-Automatisierungstechnik 56(7), 2008
- [9] J. A. Nelder, R. Mead, *A simplex method for function minimization*, Computer Journal, 1965,
- [10] R. Otte, V. Otte, V. Kaiser, *Data Mining für die industrielle Praxis*, Hanser Verlag, 2004
- [11] A. Rakotomamonjy, *Analysis of SVM regression bounds for variable ranking*, Neurocomputing, Vol. 70, 2007
- [12] A. Smola, B. Schölkopf, *A Tutorial on support Vector Regression*, NeuroCOLT2 Technical Report Series, (1998)
- [13] J.A.K. Suykens, T. Van Gestel, J. De Brabanter, B. De Moor, J. Vandewalle, *Least Squares Support Vector Machines*, World Scientific, Singapore, 2002
- [14] I. Witten, E. Frank, *Data Mining Practical Machine Learning tools and techniques*, Elsevier, (2005)

Automatische Segmentierung von mikroskopischen Zellbildern für die Biokompatibilitätsprüfung mit Hilfe eines Künstlich Neuronales Netzes

S. Buhl¹, B. Neumann¹, M. Schneider¹, U. Lehmann¹, E. Eisenbarth², T. Weißbach²

¹Institut für Computer Science, Vision und Computational Intelligence,
Fachhochschule Südwestfalen, Frauenstuhlweg 31, 58644 Iserlohn

²Fachbereich Informatik und Naturwissenschaften

Tel. (02371) 566-214 Fax (02371) 566-420

E-Mail: {Buhl, Neumann.B, Eisenbarth, MSchneider, Lehmann}@fh-swf.de

1 Zusammenfassung

Im Rahmen dieses Projektes wird ein 2-stufiges Segmentierungsverfahren zur Analyse von mikroskopischen Zellbildern erarbeitet. Der erste Schritt besteht aus einer Bildvorverarbeitung zur Bildartefaktbeseitigung und Vorsegmentierung der Zellbereiche. In Stufe 2 erfolgt dann die Datengenerierung innerhalb der Zellbereiche, die durch das zuvor trainierte Künstliche Neuronale Netz (KNN) klassifiziert werden.

2 Motivation und Zielsetzung

Die Endoprothetik hat mit der Einführung neuer Materialien und physikalischer Prinzipien seit etwa der Mitte des letzten Jahrhunderts eine rasante Entwicklung erlebt und liefert heute einen großen Beitrag zur Verbesserung der Lebensqualität sowie zur Steigerung der Lebenserwartung. Weltweit werden heutzutage jährlich mehr als 1.000.000 künstliche Hüftgelenke und 800.000 künstliche Kniegelenke eingesetzt, davon in Deutschland ca. 150.000 Hüft- und 60.000 Knieendoprothesen mit steigender Tendenz der Fallzahlen. Selbstverständlich muss gewährleistet sein, dass die eingesetzten Implantatmaterialien möglichst gut vom umliegenden Körpergewebe angenommen werden. Aus diesem Grund müssen umfangreiche Biokompatibilitätsprüfungen der eingesetzten Biomaterialien vorgenommen werden [1]. Zur Unterstützung der Biotechnologen bei der derzeit aufwendigen Prüfung der Biokompatibilität wird eine Automatisierung der Analyse mikroskopischer Zellbilder angestrebt.

Die Segmentierung von mit nicht fluoreszierenden Farbstoffen behandelten Zellen in mikroskopischen Bildern stellt nach wie vor eine große Herausforderung dar. Für die Gewebeverträglichkeitsuntersuchung verbietet sich die Zellfärbung mit Fluoreszenzfarbstoffen, obwohl dadurch eine Zellsegmentierung erleichtert würde. Dies kommt daher, dass sich die Fluoreszenzfarbstoffe an spezifischen Stellen innerhalb der Zelle anlagern und somit die Bewertung der gesamten Zelle unmöglich macht. In der Literatur werden viele Verfahren für die Zellsegmentierung und -analyse mit Methoden der Bildverarbeitung (BV) beschrieben [2][3][4].

Bei der Biokompatibilitätsprüfung von Implantatwerkstoffen weisen die zu untersuchenden Zellen komplexe Zellgeometrien auf.

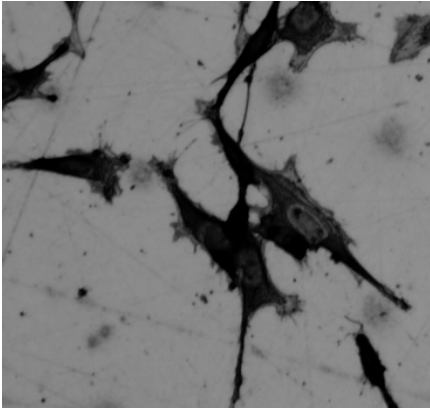


Abbildung 1: Verwendete histologisch-zytologisch gefärbte Zellen

Hier liegt der Unterschied zu vielen Publikationen. Da, wie in Abb.1 zu erkennen, die Zellmorphologie stark variiert, wird die Segmentierung von Zellen in Zellclustern sehr erschwert. Dies hat uns dazu motiviert, eine Zellsegmentierung mit BV- in Kombination mit CI-Methoden zu entwickeln.

3 Bildvorverarbeitung

Das Substrat, auf dem die Zellen aufgebracht werden, wird im Vorfeld einem Poliervorgang unterzogen. Dabei können Polierartefakte im Bildmaterial sichtbar werden, die sich ggf. negativ auf die Segmentierung der Zellen auswirken können. In Abb.2 sind einige Polierkratzer zu erkennen.

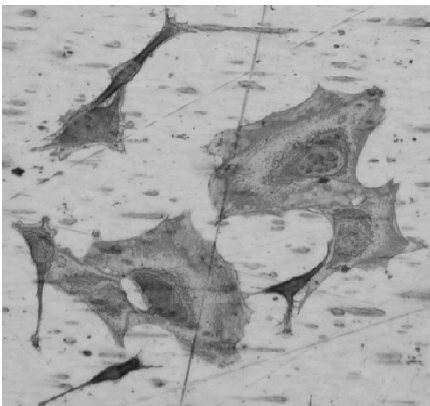


Abbildung 2: Zellen mit überlagerten Polierriefen

Für eine Bildvorverarbeitung ist es wichtig, die Kratzer zu eliminieren, die in die Zellbereiche hineinlaufen, da sie eine geschlossene Zellkontursegmentierung verhindern können. Dabei ist zu beachten, dass keine schmalen Zellausläufer fälschlicherweise als Kratzer interpretiert und eliminiert werden, weil sie ein wichtiges Kriterium für die Beurteilung der Biokompatibilität darstellen. Um Zellausläufer auszuschließen werden alle Kratzer folgenden Prüfungen unterzogen:

- Befindet sich mindestens ein Ende des Kratzers in einem Bildbereich mit hohem Blauanteil
- Ist das Verhältnis des Blaukanals zum Rotkanal > 1.1
- Ist das Verhältnis des Blaukanals zum Grünkanal > 1.1

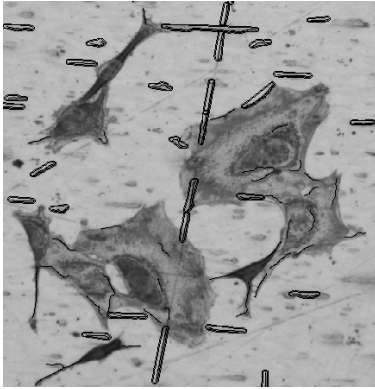


Abbildung 3: Darstellung der segmentierten Artefakte

Die Segmentierung der Kratzer wird durch eine spezielle Funktion unter Verwendung der partiellen Ableitungen einer Gaußschen Glättungsmaske durchgeführt [5]. Nachdem alle Kratzer segmentiert sind (Abb.3), erfolgt nun die Interpolation der Kratzerpixelfarbwerte in Abhängigkeit von der lokalen Umgebung. Dazu werden die Kratzerregionen jeweils zweimal einer Dilatation unterzogen und der erste dilatierte Bereich vom zweiten subtrahiert. Der auf diese Weise neu entstandene Bereich dient als Maske für die Interpolation der Kratzerpixel mit den Umgebungsfarbwerten (Abb.4).

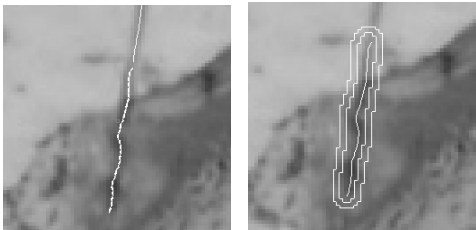


Abbildung 4: Detektierter Kratzer und Interpolationsbereich

Die Berechnung des neuen Kratzerpixelfarbwertes erfolgt dabei zeilenweise und orthogonal zur Kratzerorientierung. So werden lokale Farbänderungen in der Umgebung des Kratzers z.B. an der Zellgrenze bei der Neuberechnung des Farbwertes berücksichtigt. In Abb. 5 ist das Ergebnis einer solchen Prozedur zu sehen.

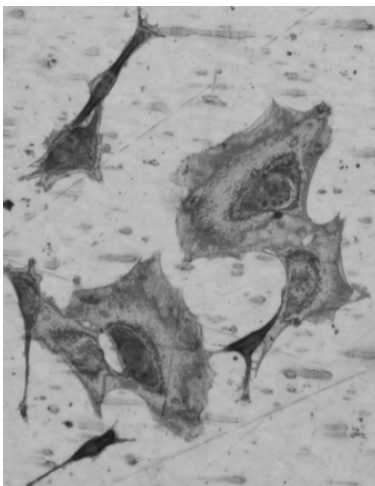
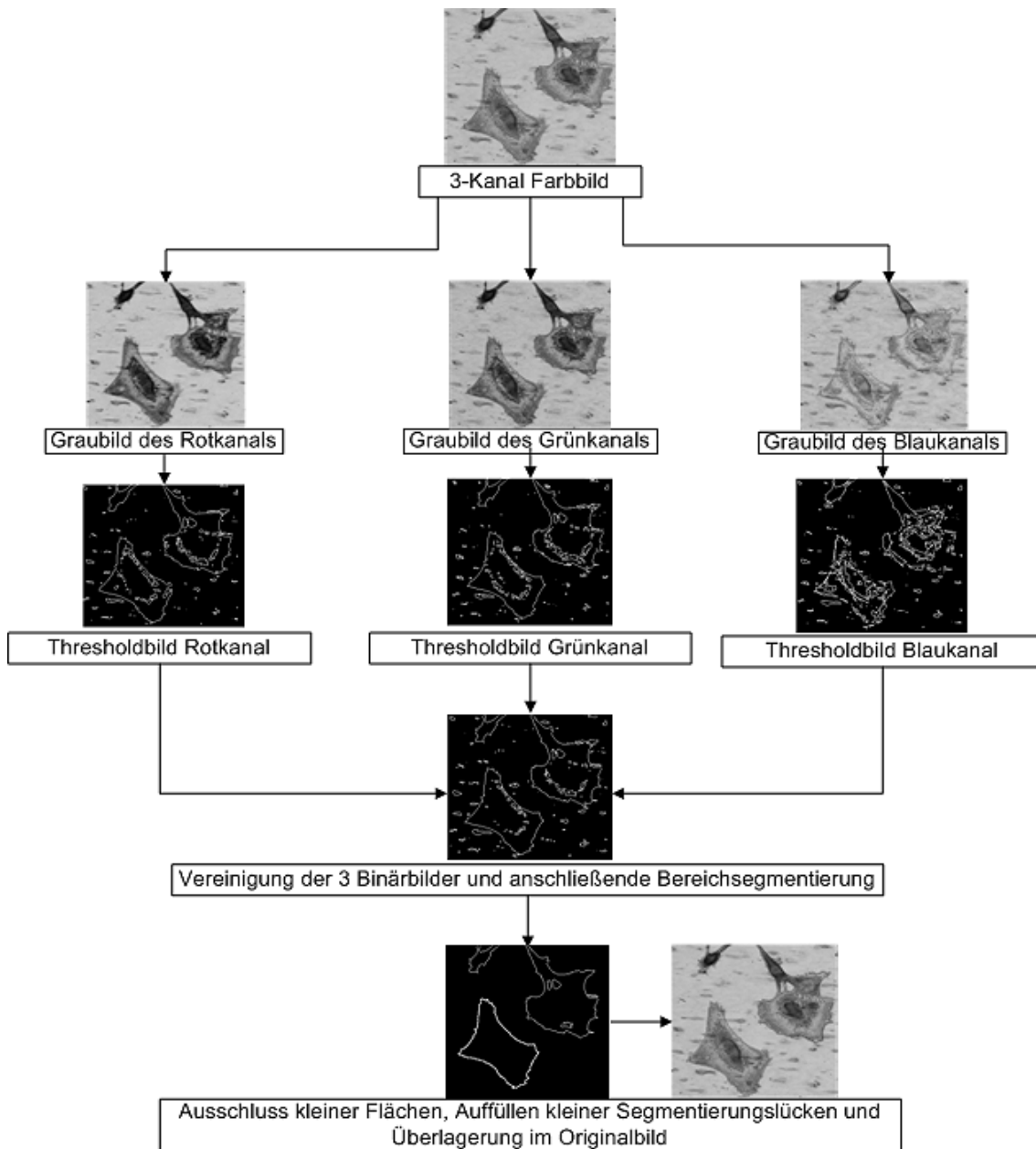


Abbildung 5: Nach Ausfilterung der Kratzer

Im Vergleich zu Abb. 2 sind die u.a. in die Zelle hineinlaufenden Kratzer völlig eliminiert worden, so dass nun eine stabilere Segmentierung der Zellkonturen gewährleistet ist.

4 Zellkontursegmentierung

Die Segmentierung der Zellkonturen erfolgt jeweils für alle 3 Farbkanaele getrennt über eine Schwellwertoperation. Die resultierenden Binärbilder werden dann vereinigt und einer Bereichsegmentierung unterzogen, damit zusammenhängende Flächen individuell auswertbar sind. Daraufhin erfolgt eine Auswahl der Regionen über das Merkmal Fläche. Schließlich werden dann kleinere Segmentierungslücken bis zu einer Fläche von 85 Pixeln innerhalb der Regionen geschlossen. Die nachfolgende Grafik veranschaulicht die geschilderte Vorgehensweise.



5 Datenerzeugung für das KNN

Da das KNN zwischen dem inneren und äußeren Ektoplasma sowie dem Endoplasma unterscheiden soll, mussten hierfür geeignete Merkmale gefunden werden. In dieser Arbeit stellen wir dem KNN vier Inputinformationen für die Klassifizierung bereit. Bei den ersten drei Inputwerten ${}^{RGB}V_R$, ${}^{RGB}V_G$, ${}^{RGB}V_B$ handelt es sich um normierte Farbkanalverhältnisse, die aus den lokalen Histogrammen berechnet werden. Hierfür werden die Schwerpunkte R_S , G_S und B_S aus den lokalen Histogrammen ermittelt und anschließend nach der Vorschrift

$${}^{RGB}V_R = \frac{R_S}{\sum}, \quad {}^{RGB}V_G = \frac{G_S}{\sum}, \quad {}^{RGB}V_B = \frac{B_S}{\sum} \text{ normiert.}$$

Dabei wird $\sum = R_S + G_S + B_S$ gesetzt. Zur Berechnung des Schwerpunktes des Rotauszuges haben wir z.B.

$$R_S = \frac{1}{\sum_{k=1}^n m_k} \sum_{k=1}^n m_k g_k$$

verwendet. Für die Schwerpunkte der anderen Farbkanäle wird analog vorgegangen. Die Normierung macht die Ergebnisse von Helligkeitsschwankungen unabhängig.

Bei dem vierten Inputwert handelt es sich um die Farbsättigung, da das Endoplasma i.d.R. eine höhere Farbsättigung besitzt als das Ektoplasma. Hierzu wird das RGB-Bild vorteilhaft in den HSV-Farbraum transformiert.

Die Inputdaten werden wie bereits erwähnt in lokalen Fenstern der Größe 10x10 Pixel generiert. Sie werden iterativ durch die Zellregion verschoben. Dabei wird im ersten Schritt das kleinste umschließende Rechteck der aktuellen Zellkontur berechnet und schrittweise die Fenstermaske durch das umschließende Rechteck bewegt (Abb.6).

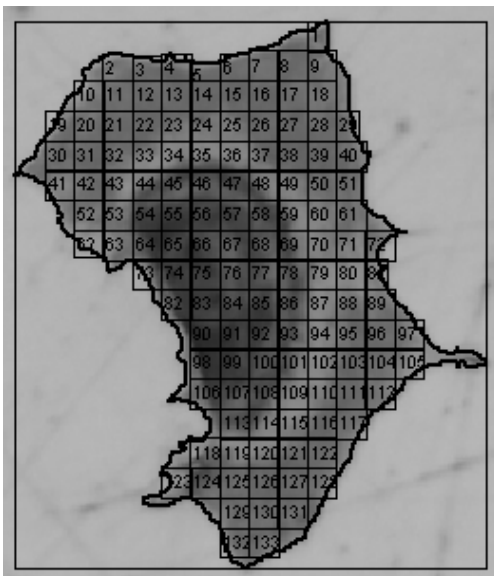


Abbildung 6: Unterteilung der Zelle in kleine Segmente

Nach jeder Iteration wird geprüft, ob mindestens 70% der Fensterfläche durch Zellpixel ausgefüllt sind. Ist dies der Fall, werden die Daten für die Inputneuronen ermittelt und gespeichert.

6 Einsatz eines Künstlichen Neuronales Netzes

Der Vorteil bei der Verwendung eines KNN liegt vor allem darin, dass ein komplexes Problem relativ einfach ohne mathematische Modellbildung und umfangreichen Algorithmen gelöst werden kann. In umfangreichen Vorversuchen mussten die optimalen Inputdaten für den erfolgreichen Einsatz eines KNN ermittelt werden.

Zur Klassifizierung der unterschiedlichen Zellbereiche wurde folgende Netzarchitektur gewählt (Abb.7):

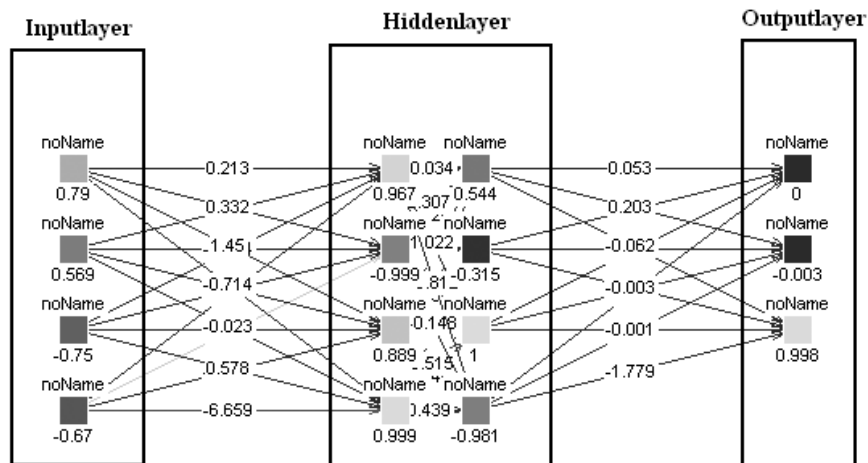


Abbildung 7: Verwendete Netzarchitektur

Für das Training wurden im Vorfeld 60 Trainings-, 5 Validierungs- und 3 Testpattern generiert. Mit dem Backpropagation-Lernalgorithmus, einer Lernrate von $\eta=0.5$ und 2000 Lernzyklen sowie einer Validierung nach jedem 10 Schritt hat sich folgendes Trainings- und Validierungsergebnis herausgestellt.

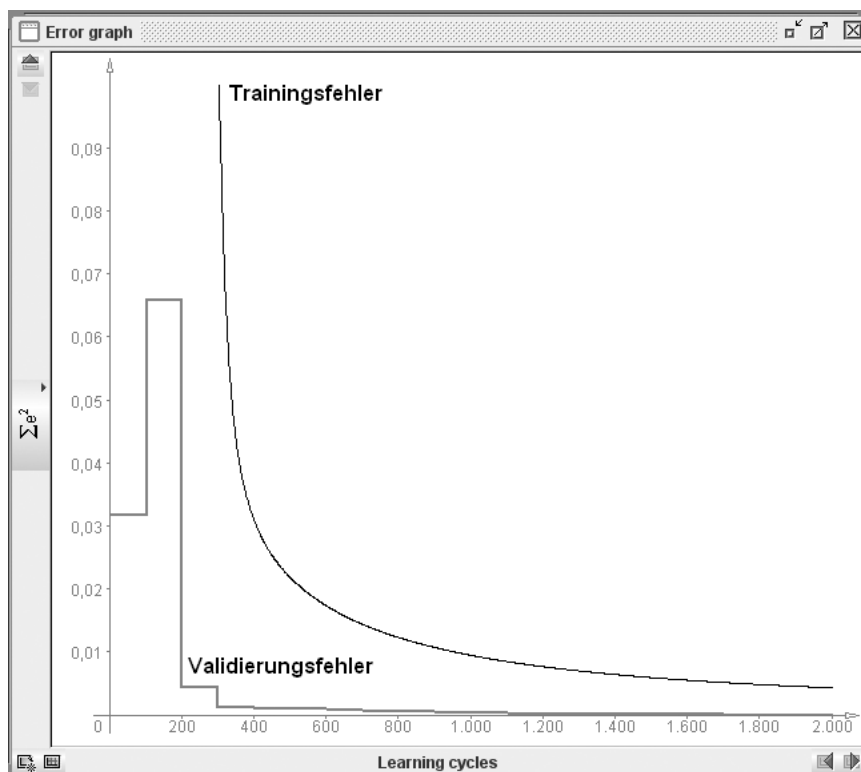


Abbildung 8: Trainings- und Validierungsfehler in Abh. der Lernzyklen

7 Ergebnisse

Grundsätzlich eignen sich KNN's zur Klassifikation der verschiedenen Zellbereiche. Zum jetzigen Zeitpunkt weisen die bisher analysierten Zellen eine relativ große Fläche und ausgeprägte Ekto- und Endoplasmabereiche auf, die von dem KNN gut klassifiziert werden.

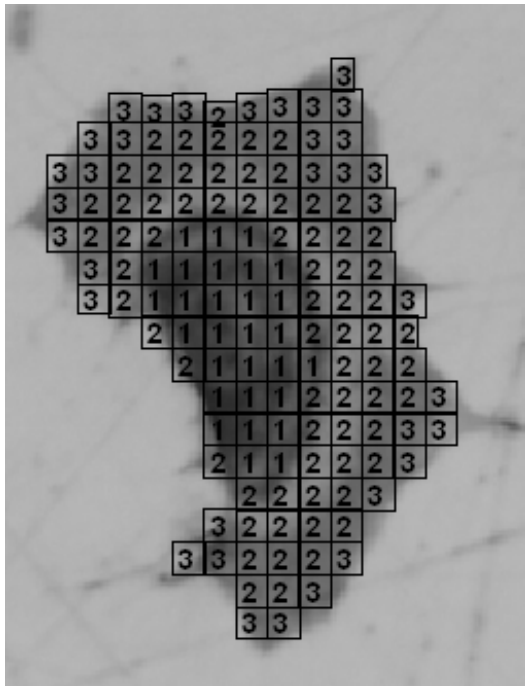


Abbildung 9: Klassifizierung der Zellbereiche

In Abb. 9 stehen die Zahlen innerhalb der kleinen Fenster für

- 1: Endoplasma
- 2: Hellblaues Ektoplasma (es umschließt das Endoplasma)
- 3: Äußeres braunes Ektoplasma.

Der Vorteil bei der lokalen Klassifizierung durch ein KNN mit Hilfe eines automatisch verschiebbaren Fensterbereiches liegt darin, dass neben der lokalen Klassifizierung auch gleichzeitig in großen Zellclustern der morphologische Kontext erhalten werden kann. Bei einer Segmentierung mit reiner Bildverarbeitung geht diese Information verloren. Vereinzelt auftretende lokale Klassifizierungsfehler können durch Überprüfung der Umgebung korrigiert werden.

8 Ausblick

Im weiteren Verlauf des Projektes soll überprüft werden, in wie weit ein KNN auf morphologisch komplexere Zellen übertragbar ist. Dabei könnten weitere Merkmale gefunden werden, die als Inputdaten für das KNN geeignet sind. Da im Endoplasma meist große Farbvariationen vorhanden sind, könnten die aus den Grauwertistogrammen berechneten Varianzen als weitere Inputdaten herangezogen werden. Vielleicht können sogar die kompletten Histogramme als Inputdaten verwendet werden. In dem Fall müsste jedoch das KNN wesentlich erweitert und optimiert werden.

9 Danksagung

Das interdisziplinäre Projekt entstand am Institut für Computer Science, Vision and Computational Intelligence in Zusammenarbeit mit dem Labor für Biotechnologie an der Fachhochschule Südwestfalen. Das Institut wird vom Ministerium für Innovation, Wissenschaft, Forschung und Technologie des Landes Nordrhein-Westfalen gefördert.

10 Literatur

- [1] Rainer Gradinger, Hans Gollwitzer: *Ossäre Integration*, Springer Berlin Heidelberg 2006
- [2] M. Tscherepanow, F.Zöllner, F.Kummert: *Aktive Konturen für die robuste Lokalisation von Zellen*; Springer Verlag; Berlin Heidelberg; 2005
- [3] Lars Dornheim, Jana Dornheim.: *Automatische Detektion von Lymphknoten in CT-Datensätzen des Halses*; Springer Verlag; Berlin Heidelberg; 2008
- [4] Giuliana Ramella, Gabriella Sanniti di Baja: *Image Segmentation by Non-Topological Erosion and Topological Expansion*; Advances in Mass Data Analysis of Signals and Images in Medicine Biotechnology and Chemistry – International Conferences MDA 2006/2007
- [5] MVTec Halcon Referenzhandbuch Version 9.0

Graph Mining: An Overview

Christian Borgelt

European Centre for Soft Computing
Campus Mieres, Edificio Científico-Tecnológico
c/ Gonzalo Gutiérrez Quirós s/n, E-33600 Mieres (Asturias), Spain
Tel.: +34 985 456545, Fax: +34 985 456699
E-Mail: christian.borgelt@softcomputing.es

1 Introduction

In the early years of data mining and knowledge discovery in databases, method development focused on rigidly and plainly structured data. Most often efforts were even confined to data that can be represented as a simple table, which describes a set of sample cases by attribute-value pairs. Recent years, however, have seen a constantly growing interest in the analysis of more complex data, with a less rigid and/or more sophisticated structure.

In this development, two foci of attention can be identified: *text document analysis* and *graph mining*. The former tries to extract the rich information available in text documents like news articles, scientific papers, or blog posts, in order to make this information more easily accessible, to group related documents and rank them, and even to generate digests for more effective and efficient browsing [17]. The main problems in this area stem from the need to deal with natural language, which, due to its amazing variability and flexibility of expressing similar or even the same meaning in vastly differing ways and with different words, poses severe obstacles to linking the subjects of different documents automatically. Nevertheless, statistical approaches, which model texts as a so-called *bags of words* and rely mainly on term frequencies and their distribution in a collection of documents, have led to impressive results. More sophisticated approaches exploit lexical databases like WordNet [18] to resolve ambiguities and to deal with hyponyms and hypernyms.

The other area, graph mining, enjoys an increasing popularity due to two reasons: in the first place, a lot of data naturally comes in the form of graphs. Whether we deal with molecules, protein interaction networks, metabolic networks, the Internet and world wide web, co-author networks, social networks, etc., the corresponding data clearly describes a graph with nodes and edges. Secondly, basically all n to m relationships in a relational database can fairly naturally be represented or interpreted as a graph, thus making graphs a natural tool to analyze single- as well as multi-relational data. Examples are customer and product databases (containing information about who bought what), which are easily cast as a bipartite graph, or collaborative (or social) tagging systems, the data of which can conveniently be modeled as tripartite graphs of users, resources, and tags.

In this paper I try to give an overview of some of the main tasks and techniques of different subareas of graph mining. However, although I try to cover the core ideas in each of the subareas, I will provide a somewhat more detailed treatment only for the special task of frequent subgraph mining, which is my personal area of expertise. Apart from frequent subgraph mining (Section 6), I review the tasks of identifying global graph patterns (Section 2), classifying graphs (Section 3), predicting the labels of nodes and edges (Section 4), and predicting new or yet unknown links (Section 5).

2 Global Graph Patterns

Global graph patterns are similar in spirit to the characteristic measures used in descriptive statistics (like mean, variance, skew, etc.) or to the state variables used in thermodynamics (like pressure, temperature, energy, entropy, etc.). That is, they serve the purpose of characterizing complex graphs with few quantities, mainly with one of the following three goals: (1) find properties that distinguish real-world graphs from random graphs, (2) detect anomalies in a given graph, and (3) generate synthetic, but realistic graphs.

The seminal work in this area is the small-world network model [49], which introduced formal properties (specifically average path length and clustering coefficient, see below) to capture the small-world phenomenon that was observed in social networks [38, 45]. These properties made it possible to contrast real-world graphs to random graphs built with the Erdős–Rényi model [16] (edges occur independently with equal probability). The most common global graph properties in use nowadays are the following [8]:

Power laws. In many real-world graphs, many nodes have few connections, while only few nodes have a large number of connections. This can be expressed formally with a power law distribution for the degree x of the nodes: $p(x) = ax^{-b}$ where $a > 0$, $b > 1$ (needed so that the density can be normalized), and $x \geq x_{\min}$. Graphs exhibiting a power law degree distribution are often called *scale-free*, because the power law $y(x) = ax^{-b}$ is invariant up to a multiplicative factor, that is, $y(c_1x) = c_2y(x)$. The goal of a power law analysis is usually to find the value of b in order to characterize the graph with a single number. Power laws have been found to be empirically valid for the degree distribution of the Internet graph and the in- and out-degree distribution of the World Wide Web. However, checking for a power law distribution is often done sloppily, e.g., by looking at a scatter plot. More rigorous testing methods have only recently been developed.

Diameter / average path length. In real-world graphs, two nodes are usually not far apart in the sense that one can reach the one from the other on a fairly short path. This property can be captured by several closely related measures: (1) The *effective diameter* or *eccentricity* is the minimum number of hops (hop: following an edge to an adjacent node) in which some fraction (e.g. 90%) of all connected pairs of nodes can reach each other. (2) The *characteristic path length* is computed by taking the average length of the shortest paths from a given node to every other node, and then taking the median of these values over all nodes. (3) The *average diameter* is defined in basically the same way, only that the mean is computed instead of the median. If the graph is not connected, the characteristic path length and average diameter are computed only for the largest component. All of these measures yield fairly small values for a large variety of real-world graphs.

Community structure. A *community* is a set of nodes where each node is closer to the other nodes in the community than to nodes outside it. A community exhibits transitivity in a graph: if a and b are adjacent and b and c are adjacent, then a and c are likely also connected. This property is formally measured with the *clustering coefficient*. For a single node u with n_u neighbors, between which k_u edges exist, the clustering coefficient $C(u)$ is defined as k_u/n_u if $n_u > 1$ and 0 otherwise. A global clustering coefficient is computed by simple averaging: $C = \frac{1}{|V|} \sum_{u \in V} C(u)$. Alternatively, the clustering coefficient can be defined by drawing on the insight that transitivity occurs if *triangles* exist in the graph. Therefore one may define $C = (3 \cdot \text{number of triangles}) / \text{number of connected triples}$, where a *connected triple* is any subgraph with three nodes and two edges. In this form C measures the fraction of connected triples that actually form triangles.

3 Graph Classification

In machine learning, classification is generally the task of predicting the label of an input object, and therefore graph classification is the task of predicting the label of an input graph. For example, we may want to predict the function of a protein [6] or whether a given molecule will be bioactive (that is, for example, will inhibit the growth of cancer cells). In statistics, however, classification is the task of placing given objects into groups—a task commonly called *clustering* in machine learning—and thus graph classification may also refer to graph clustering. Here we consider both tasks.

The most straightforward approach to graph classification (in either of the abovementioned meanings) is to reduce it by feature extraction to the task of classifying or grouping vectors of attribute-value pairs. That is, rather than capturing the actual structure of a graph, it is described by attributes that are derived from its structure and/or from the labels that may be associated with its nodes or its edges or with both. For example, a graph may be described by how many triangles it contains, by how many edges between nodes having a given label exist in it, of how many connected components it consists etc. Depending on how many different node and/or edge labels as well as specific substructures are considered, a graph may thus be described a huge number of attributes. This is also the main drawback of this approach: it is usually difficult to find the right features, so that a classical attribute-value based classifier can yield good results.

As a consequence, an alternative approach has gained increasing popularity, which focuses on graph similarity as a central problem for clustering and classification. The core idea is that classifiers like, for example, support vector machines require only a similarity measure that satisfies certain properties, and many clustering algorithms are able to work on a mere distance or similarity matrix as input. In such an approach the similarity of two graphs is computed with so-called *graph kernels*, where a *kernel* is generally a measure of similarity that is symmetric and positive semi-definite. The basic idea of graph kernels is to compare substructures that are traversable in polynomial time (like walks, paths, cyclic patterns, trees, etc.), where this restriction is meant to ensure that the kernel function can be evaluated in reasonable time. In contrast to this, more rigid approaches based, for instance, on maximum common subgraphs suffer from being NP-hard.

Since the seminal work [21], many different types of graph kernels have been designed, including so-called marginalized graph kernels [28, 29], subtree kernels [46], and rational kernels [10]. Good overviews, providing also a generalized view, can be found in [23, 48]. Here I focus on the concepts underlying random walk kernels [22], which are based on the following idea: if we consider the possible walks (paths) between two nodes, it is plausible that the two graphs are similar if, for most pairs of nodes, many walks are matching, so we simply count the number of matching walks. However, in order to ensure convergence, it is advisable to discount longer paths (i.e., give them a lower weight).

Technically, not all possible walks (paths) are considered, but a random sample, hence the name *random walk kernel*. Technically, the kernel is computed on the product graph of two given graphs G_1 and G_2 , which consists of pairs of identically labeled nodes and edges from G_1 and G_2 . The reason is that a random walk on the product graph is equivalent to simultaneous random walks on the input graphs. The kernel is then defined as

$$k(G_1, G_2) = \frac{1}{|G_1||G_2|} \sum_k \frac{\lambda^k}{k!} \vec{e}^\top A_\times^k \vec{e},$$

where $|G_i|$ is the number of nodes of graph G_i , \vec{e} encodes a probability distribution on the nodes, which may be chosen as the uniform distribution, A_\times is the adjacency matrix

of the product graph, and $\lambda^k/k!$ is the discounting factor. The main problem with such an approach is that the product graph, and thus its adjacency matrix, is huge. However, with certain technical tricks the computation can be made feasible and efficient [47].

4 Label Prediction

Label prediction is a semi-supervised learning problem, also called *within-network classification*, which consists in the task to classify the nodes (and maybe also the edges) of a partly labeled graph (that is, to assign labels to them). Solutions to this problem have applications in image processing (i.e. classify the nodes of an image graph), document and web page classification, classifying protein interaction and gene expression data, part-of-speech tagging, fraud detection, customer suggestions and many other areas.

Many existing methods for label prediction are proximity-based: missing labels are inferred based on the hypothesis that linked or nearby nodes are likely to have the same labels [7, 20]. This hypothesis is also known as *homophily* (i.e., love of the same) and describes the tendency of individuals to associate and bond with similar others. Thus it is not surprising that these approaches work fairly well in social networks.

A similar approach is based on random walk kernels, similar to those for graphs [31]: it is plausible that two nodes are similar if they are connected by many walks. Long walks are discounted (receive a lower weight) in order to deal with the fact that the number of walks goes to infinity if the graph has cycles. Formally, walks are computed by taking powers of adjacency matrix A : $[A^k]_{ij} = c$ means that there are c paths of length k between nodes i and j . The kernel can then be defined as

$$k(i, j) = \left[\sum_k \frac{\lambda^k}{k!} A^k \right]_{ij},$$

where $\lambda^k/k!$ is the discounting factor with which a lower weight is assigned to longer walks. As an alternative, the kernel may be computed from the Laplacian matrix instead of the adjacency matrix. In this case it is known as a *diffusion kernel* and has been shown to be a valid positive semi-definite kernel. In both cases, however, the kernel is not evaluated for all paths, but only for a random sample; hence the name *random walk kernel*.

Unfortunately, the *homophily* assumption fails for many types of graphs, including trading networks, molecules, metabolic networks, etc. As an alternative to a *homophily*-based approach, such graphs can be handled with approaches based on the similarity of nodes, which can also be computed with kernel-like approaches, for example, with a marginalized similarity kernel [14]. In this approach random walks from two given nodes are compared in order to determine their similarity. Similar to the approach to graph kernels, two nodes are considered to be the more similar, the more matching walks start at them.

In all of these approaches it is important that the node labels should be inferred simultaneously instead of individually—a technique known as *collective classification*. This can be achieved in different ways, for example by iterative classification, which crisply assigns labels, or by learning the joint probability distribution of the labels. The latter can be achieved exactly using Markov Random fields and their extensions, or approximately e.g. by Gibbs sampling, but also by relaxation labeling or loopy belief propagation.

5 Link Prediction

Link prediction is the problem to predict between which nodes, which are unconnected in the current state of a graph, a new link (edge) will emerge or will be discovered. Solutions to this problem have applications in social network analysis (e.g.: which authors are likely to write a paper together soon?), web page improvement (e.g.: which references between web pages—for example: in Wikipedia—are missing?), and biological networks (e.g.: which protein interaction is not yet known, but may be discovered soon?).

The seminal work in this area considered co-author networks and tried to predict from a snapshot of a co-author graph which people, who did not work together in the past, would co-author a paper soon after the snapshot was taken [35]. More specifically, it was investigated how well topological and proximity measures for the nodes of a co-author graph are suited to predict emerging links between authors. Formally, each measure produces a ranked list of edges. Given the number n of edges that actually emerged, it is determined how many of these edges are among the top n edges of the ranked list.

The employed measures draw on ideas that already showed up in previous sections: they are either based on the direct neighbors and specifically the *common neighbors* of two unconnected nodes under consideration, or on the existing *paths (walks) between nodes*. Among the former are: (1) the simple *number of common neighbors* $|N(x) \cap N(y)|$, where $N(\cdot)$ denotes the set of neighbors of a node; (2) the *Jaccard coefficient* of the common neighbors, that is, $J(x, y) = |N(x) \cap N(y)| / |N(x) \cup N(y)|$; (3) the *Adamic/Adar measure* [1], in which common neighbors z are weighted with $1/\log(|N(z)|)$, that is, common neighbors with many neighbors contribute little weight; and (4) *preferential attachment*, which encodes the principle that nodes with many neighbors are more likely to get another neighbor than nodes with few neighbors, formalized as $|N(x)||N(y)|$. Among the latter (i.e. measures based on paths between the nodes) are: (1) the (negated) *length of the shortest path*; (2) the *unweighted Katz measure* [30], which sums over all paths between two nodes, discounting or dampening longer paths with a factor β^k where k is the path length; (3) the *weighted Katz measure* [30], which instead of simply counting the edges to determine the path length weights them with the number of existing co-authorships; (4) the *hitting time*, which is the expected number of steps of a random walk from one node to the other; (5) the *commute time*, which is the symmetrized hitting time; (6) the *rooted PageRank* [41], which is the stationary distribution weight of the node y under the following random walk: with probability α , jump to node x and with probability $1 - \alpha$, go to random neighbor of the current node; and (7) the *SimRank* [27], which is recursively defined as two nodes being similar to the extent that they are joined to similar neighbors.

Experiments showed that basically all measures improve significantly over a random prediction, in which each non-existent edge has the same probability of emerging. Especially the Katz measures perform consistently well, but also simple measures like the number of common neighbors and the Adamic/Adar measure yield surprisingly good results.

The idea has been transferred to biological networks, where the task is to predict links that exist, but are not yet known [43]. The prediction is evaluated by comparing the predicted links to links that were discovered (with other, i.e., biological means) later.

Newer approaches to link prediction are based on the algebraic spectrum of a graph, which generalizes many graph kernels [33]. Note also that label prediction approaches that are based on proximity or node similarity can easily be transferred to link prediction.

6 Frequent Subgraph Mining

In analogy to the well-known task of frequent item set mining, with which item sets are found that are contained in a sufficiently large number of transactions of a given database (as specified by a user-provided minimum support), frequent subgraph mining tries to find (sub)graphs that are contained in a sufficiently large number of (attributed or labeled) graphs of a given graph database. Since the advent of this research area around the turn of the millennium, several clever algorithms for frequent subgraph mining have been developed. Some of them rely on principles from inductive logic programming and describe the graph structure by logical expressions [19]. However, the vast majority transfers techniques developed originally for frequent item set mining. Examples include MolFea [32], FSG [34], MoSS/MoFa [3], gSpan [50], CloseGraph [51], FFMS [26], and Gaston [39, 40]. A related, but slightly different approach is used in Subdue [11], which is geared towards graph compression with common subgraphs rather than frequent subgraph mining. An overview of several methods and related problems can be found in [12].

6.1 Motivation: Molecular Fragment Mining

Developing a new drug can take ten to twelve years (from the choice of the target to the introduction into the market) and the duration of the development process even increases continuously. At the same time the number of substances under development has gone down drastically. The reasons for these trends are raised safety standards for (new) drugs and the fact that due to rising research investments pharmaceutical companies must secure their market position and competitiveness by only a few, highly successful drugs (like Aspirin). As a consequence the chances for the development of drugs for target groups with rare diseases or with special diseases that are most frequent in developing countries (like AIDS) are considerably reduced, since the expected revenue from such drugs is low. In order to improve this situation, considerable efforts are devoted to significantly reduce the development time, by which one hopes to mitigate or even reverse this trend.

One way in which one tries to reduce the drug development time is to try to improve the discovery and optimization of candidate substances (so-called *pharmacophores*) and in particular to enhance the evaluation of high-throughput screening experiments. In such experiments a large number of potentially useful molecules are tested for activity w.r.t. some chosen target, for example, whether they are able to protect a human cell against a certain virus. The result is a database of chemical compounds together with activity information. An example of such a database is the DTP AIDS Antiviral Screen Database of the National Cancer Institute [15], in which about 40,000 chemical compounds are recorded together with a measurement of their ability to protect human CEM cells against an HIV-1 infection. An (adapted) excerpt from this database is shown in Table 1, where CI means “confirmed inactive”, CM means “moderately active” (provided reproducibly 50% protection), and CA means “confirmed active” (provided reproducibly 100% protection). Molecules are described in the SMILES language (Simplified Molecular Input Line Entry System), which is a fairly popular description languages for molecules.

Molecular fragment mining is a very useful analysis method for such databases, which is based on frequent subgraph mining. The main goal is to identify molecular substructures that are frequent in the active, but rare in the inactive molecules, hoping that such fragments provide insights into what causes the activity. As an example, Figure 1 shows four confirmed active molecules from the DTP AIDS Antiviral Screen Database as well as a fragment that is common to all of them. This fragment is the characteristic substructure

```

737, CI, CN(C)C1=[S+] [Zn]2(S1)SC(=[S+]2)N(C)C
2018, CI, N#CC(=CC1=CC=CC=C1)C2=CC=CC=C2
19110, CI, OC1=C2N=C(NC3=CC=CC=C3)SC2=NC=N1
20625, CA, NC(=N)NC1=C(SSC2=C(NC(N)=N)C=CC=C2)C=CC=C1.OS(O)(=O)=O
22318, CI, CCCC(N(CCCC)C1=[S+] [Cu]2(S1)SC(=[S+]2)N(CCCC)CCCC
24479, CI, C[N+](C)(C)C1=CC2=C(NC3=CC=CC=C3S2)N=N1
55917, CI, O=C(N1CCCC[CH]1C2=CC=CN=C2)C3=CC=CC=C3
64054, CA, CC1=C(SC[C-]2N=C3C=CC=CC3=C(C)[N+]2=O)C=CC=C1
64055, CM, CC1=CC=CC(=C1)SC[C-]2N=C3C=CC=CC3=C(C)[N+]2=O
64057, CA, CC1=C2C=CC=CC2=N[C-](CSC3=NC4=CC=CC=C4S3)[N+]1=O
66151, CI, [O-][N+](=O)C1=CC2=C(C=NN=C2C=C1)N3CC3
...

```

Table 1: A fraction of the NCI DTP AIDS Antiviral Screen Database in a format in which the molecules are described in SMILES (Simplified Molecular Input Line Entry System). Each row consists of an identifier, an activity indicator, and a molecule description.

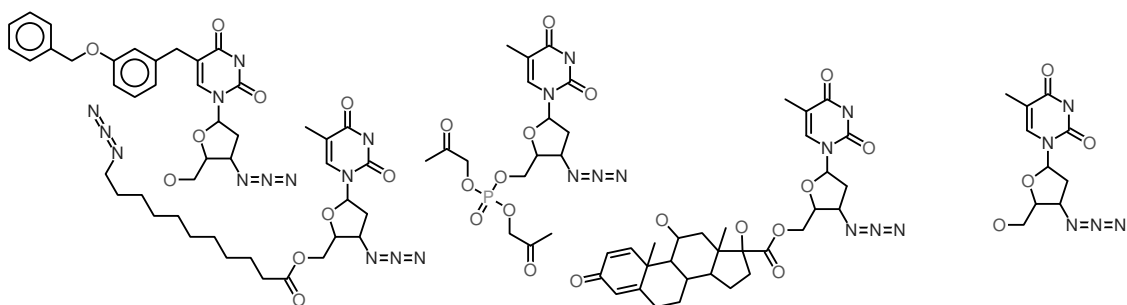


Abbildung 1: Four confirmed active molecules from the NCI DTP AIDS Antiviral Screen Database (left) and a molecular fragment that is part of all four of them (right).

of a class of substances known as AZT, which are currently used as AIDS medication.

6.2 Notation, Presuppositions, and Problem Definition

Formally, frequent subgraph mining works on a database of *labeled graphs* (also called *attributed graphs*). A labeled graph is a triple $G = (V, E, l)$, where V is the set of vertices, $E \subseteq V \times V - \{(v, v) \mid v \in V\}$ is the set of edges, and $l : V \cup E \rightarrow L$ assigns labels from some label set L to vertices and edges. The graphs we consider are undirected and simple (that is, there is at most one edge between two given vertices) and contain no loops (that is, there are no edges connecting a vertex to itself). However, graphs without these restrictions (that is, directed graphs, graphs with loops and/or multiple edges) can be handled as well with properly adapted methods. Note also that several vertices and edges may carry the same label, which is one core reason for the complexity of the problem.

The *support* $s_G(S)$ of a (sub)graph S w.r.t. a given graph database \mathcal{G} is the number of graphs $G \in \mathcal{G}$ it is contained in. What is meant by a graph being contained in another is made formally precise by the notion of a *subgraph isomorphism*. Given two labeled graphs $G = (V_G, E_G, l_G)$ and $S = (V_S, E_S, l_S)$, a subgraph isomorphism of S to G is an injective function $f : V_S \rightarrow V_G$ satisfying (1) $\forall v \in V_S : l_S(v) = l_G(f(v))$ and (2) $\forall (u, v) \in E_S : (f(u), f(v)) \in E_G \wedge l_S((u, v)) = l_G((f(u), f(v)))$. That is, the mapping f preserves the connections between vertices and the labels of both vertices and edges.

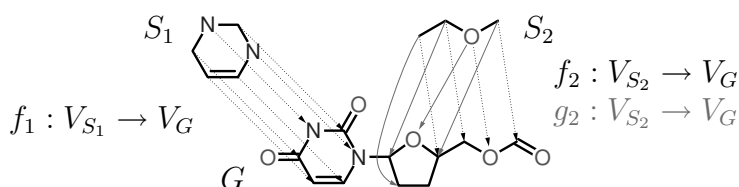


Abbildung 2: Examples of subgraph isomorphisms with two fragments and a molecule that contains them in different ways.

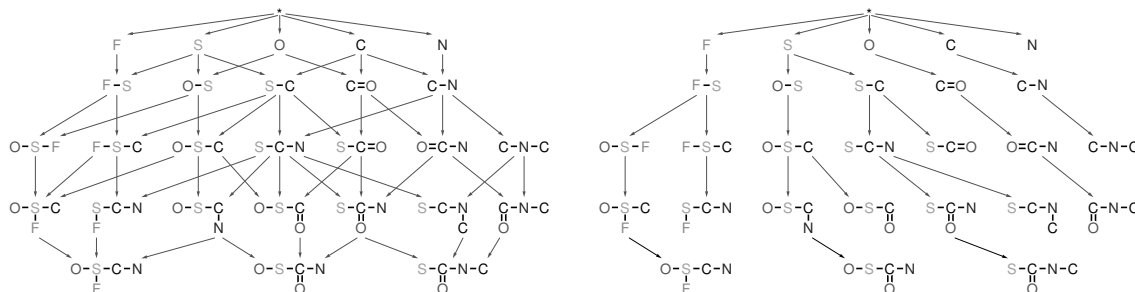


Abbildung 3: A semi-lattice of subgraphs for three molecule graphs, to be seen at the bottom (left) and an assignment of unique parents, which turns the semi-lattice into a tree (right).

Note that there may be several ways to map a labeled graph S to a labeled graph G , so that connections and labels are preserved. For example, G may possess several subgraphs that are isomorphic to S . It may even be that S can be mapped in several different ways to the same subgraph of G . This is the case if there exists a subgraph isomorphism of S to itself (a *graph automorphism*) that is not the identity. Examples of subgraph isomorphism w.r.t. a molecule G and two molecular fragments S_1 and S_2 , all of which are modeled as labeled graphs, are shown in Figure 2. Fragment S_2 is contained in several different ways in molecule G , two of which are indicated. Here, however, we neglect that there may be several subgraph isomorphism and consider only whether there exists at least one or not.

Given a database \mathcal{G} of labeled graphs and a user-specified minimum support $s_{\min} \in \mathbb{N}$, a (sub)graph S is called *frequent* in \mathcal{G} iff $s_{\mathcal{G}}(S) \geq s_{\min}$. The task of frequent subgraph mining is to identify all subgraphs that are frequent in a given graph database \mathcal{G} . However, the output is usually restricted to *connected subgraphs* for two reasons: in the first place, connected subgraphs suffice for most applications. Secondly, restricting the result to connected subgraphs considerably reduces the search space, which otherwise is so huge that searching it becomes infeasible even for small graph databases.

6.3 Search Space

In order to search for frequent subgraphs, we consider, in analogy to frequent item set mining, the semi-lattice of subgraphs of the graphs of a given database. That it is only semi-lattice is due to the fact that there is no natural largest element, since there is no largest labeled graph for any given label set L . Therefore we use the database graphs as maximal elements, which, however, are usually not comparable. A *Hasse diagram* of an example semi-lattice, for a simple database of three molecule-like graphs (no chemical meaning attached, to be seen at the bottom) is shown in Figure 3 on the left: two subgraphs are connected if one possesses an additional edge (and an additional vertex) compared to the other. The frequent subgraphs are located at the top of this semi-lattice, thus suggesting a top-down search, just as it is used for frequent item set mining.

Therefore the search for frequent subgraphs consists in growing subgraphs into the graphs of the given database, step by step adding edges and vertices, until the support falls below the threshold. Like in frequent item set mining, a core problem of the search is that the same graph can be reached on different paths, as is easily visible in the subgraph semi-lattice shown in Figure 3. Hence we face the task to avoid redundant search. The solution principle is to assign a unique parent to each subgraph, which turns the subgraph semi-lattice into a subgraph tree. This is illustrated in Figure 3 on the right (the principle underlying the assignment of unique parents is discussed below, see Section 6.4).

With a unique parent for each subgraph, we can carry out the search for frequent subgraphs according to the following simple recursive scheme: in a base loop, all possible vertex labels are traversed (their unique parent is the empty graph). All vertex labels (and thus all single vertex subgraphs) that are frequent are processed recursively. A given frequent subgraph S is processed recursively by forming all possible extensions R of S by a single edge (also adding a vertex except when the edge closes a cycle), for which S is the chosen unique parent. All such extensions R that are frequent (that is, for which $s_G(R) \geq s_{\min}$) are processed recursively, while infrequent extensions are discarded.

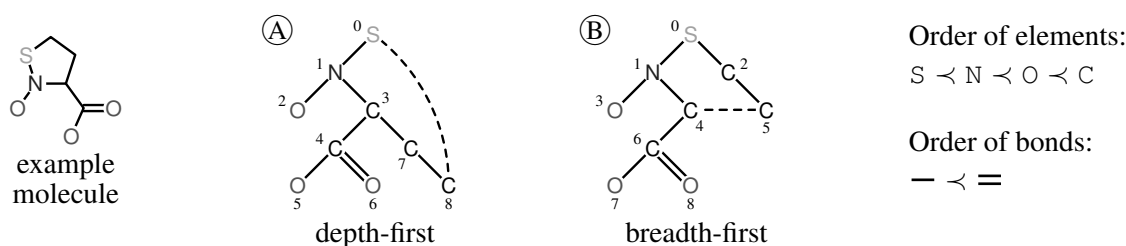
6.4 Canonical Forms of Graphs

In the search for frequent item sets it is trivial to assign unique parents, namely by choosing an (arbitrary, but fixed) order of the items and defining the parent of an item set as the set that results if its maximum element w.r.t. this chosen order is removed. Unfortunately, in the search for frequent subgraphs ordering the labels in the set L , though also necessary, is not enough. The reason is mainly that several vertices (and several edges) may carry the same label. Hence the labels do not uniquely identify a vertex, thus rendering it impossible to describe the graph structure solely with these labels. We rather have to endow each vertex with a unique identifier (usually a number), so that we can unambiguously specify the edges of the graph and which of them has to be removed to obtain the parent.

Given an assignment of unique vertex identifiers, we can describe a graph with a *code word*, which specifies the vertex and edge labels and the connection structure, and from which the graph can be reconstructed. Of course, the form of this code word depends on the employed numbering of the vertices: each numbering leads to a different code word. In order to single out one of these code words as the so-called *canonical code word*, we simply select the lexicographically smallest (or largest) code word.

With canonical code words, we can easily define unique parents: the canonical code word of a (sub)graph S is obtained from a specific numbering of its vertices and thus also fixes (maybe with some additional stipulation) an order of its edges. By removing the last edge in this order, which is not a bridge or which is incident to at least one vertex with degree 1 (which is then also removed), we obtain a connected graph that is exactly one level up in the semi-lattice of subgraphs and thus may be chosen as the unique parent of S .

Technically, canonical code words for graphs can either be based on a systematic way of constructing a spanning tree [50, 4] or on (extended) adjacency matrices [36]. Here we consider only the former: starting at an arbitrary vertex, a spanning tree of a given graph is formed either by a depth-first or by a breadth-first search. The vertices of the graph are numbered in the order in which they are visited and are thus endowed with unique identifiers. A code word is then formed by appending edge descriptions to a single letter stating the label of the root vertex, thus yielding the following general code words:



Code words: A: S 10-N 21-O 31-C 43-C 54-O 64=O 73-C 87-C 80-C
 B: S 0-N1 0-C2 1-O3 1-C4 2-C5 4-C5 4-C6 6-O7 6=O8

Abbildung 4: Spanning trees for an example graph, vertex numberings that have been derived from them with depth-first or breadth-first traversal, and the corresponding code words.

$$\begin{aligned} \text{depth-first:} & \quad a (i_d \underline{i_s} b a)^m \\ \text{breadth-first:} & \quad a (i_s b a i_d)^m \quad (\text{or } a (i_s i_d b a)^m), \end{aligned}$$

where i_s and i_d are the source and destination index of the incident vertices (using the convention that the incident vertex with the smaller index is the source), b is an edge label, a is a vertex label, and m is the number of edges. The edge descriptions are generally listed in the order in which the edges are visited in the spanning tree traversal. However, for both depth-first and breadth-first this can also be described by a sorting order on the edge descriptions, specified by a precedence order of their elements for comparisons, which is given in the above regular expressions by the order in which the elements are listed. All elements are compared ascendingly, with the exception of the source index in the depth-first case, which is compared descendingly (indicated by the underscore).

As an illustration, Figure 4 shows an example molecule and two spanning trees, in which the vertices are numbered in the order in which they are visited by a depth-first (A) or breadth-first search (B). The resulting code words are shown at the bottom of the figure. It is not difficult to verify that these code words are actually the lexicographically smallest code words that can result from different choices of the root vertex and different ways of forming the spanning trees. (Even though the general traversal order is fixed, there is no a-priori rule in which order to visit the neighbors of a vertex.) Therefore they are the canonical code words for these graphs, for these two specific code word construction methods. As a consequence, we see that the parent of the example molecule w.r.t. a depth-first code word is the graph in which the edge between sulfur and carbon is missing, while the parent w.r.t. a breadth-first code word is the graph in which the double bond is missing.

Constructing a canonical code word, or testing whether a given code word is canonical, is done by recursively enumerating (prefixes of) all possible code words (see [4] for details). However, the above canonical forms possess the *prefix property*: any prefix of a canonical code word is a canonical code word itself. This property simplifies the search: code words of extensions can be formed by simply appending the description of the added edge to the canonical code word of the (sub)graph that is extended. If the result is a canonical code word, the extension was created from the correct parent, otherwise it is to be discarded. Furthermore, this possibility enables us to determine in many cases with very simple checks whether the result of an extension is canonical and thus whether it has been created from the correct parent [4]. Unfortunately, though, these *simple rules* are reliable only if they state the the result is not canonical. Otherwise a full canonical form test is needed in order to be sure that an extension needs to be processed.

6.5 Search Algorithms

The basic search scheme was already outlined in Section 6.3. Nevertheless many different concrete approaches are possible, for example, depending on the order in which the two properties are tested that an extended graph must have in order to be processed recursively: it must have been created from the correct parent (which is determined with the help of a canonical form test) and it must have at least the minimum support. Which property is easier or faster to test may determine the order in which they are tested.

In addition, one can consider two possible ways of forming possible extensions of a given graph. The first method records all occurrences of a subgraph to extend and checks what extensions are actually possible in the database graphs. In this case the support of each extension is readily available and thus checked first against the minimum support. Only for those extensions that satisfy the minimum support requirement a canonical form test is carried out. Alternatively, the possible extensions may be determined from general information about the database of graphs, like which edge labels occur together with which vertex labels. In this case the support is not readily available, as one must check in which database graphs a created extension occurs. As a consequence, it may be advantageous to check for canonical form first and to compute the support (by subgraph isomorphism tests on the database graphs) only for those extensions that are canonical. Generally the first scheme is advantageous if the database graphs have many different labels, because in this case using only general database information yields too many extensions. In database with few or even no labels, however, the second scheme is preferable, because it avoids the storage and computation overhead of maintaining lists of all occurrences [13].

6.6 Other Techniques

Since they are for general graphs, the methods described above are, of course, also applicable to trees and sequences (modeled as chains). However, for these restricted classes of graphs there exist specialized canonical forms that allow for perfectly reliable *simple rules*, so that a costly backtracking check for canonical form can be avoided [9, 40]. Such a canonical form is even known for the more general class of outerplanar graphs [25].

In the specific area of molecular fragment mining, special methods have been developed to deal with rings [24, 5] (which chemists often treat as irreducible building blocks), carbon chains [37] and wild cards [24]. The goal of these approaches is to make the output of a frequent subgraph mining algorithm easier to interpret and more useful for a chemist.

An interesting alternative to a canonical form based search is an enumeration scheme that stores part of the subgraph lattice together with certain transformations in order to avoid redundant search and to achieve polynomial delay in the enumeration [42].

7 Outlook

Similar to recent developments in other areas of data mining, there is currently a strong shift of focus towards dynamic aspects of graphs, that is, towards change mining on graph databases, which are observed over time. This includes, for example, global graph patterns in evolving graphs like the densification power law and shrinking diameters [8] as well as graph evolution rules [2], but also link prediction (see Section 5).

For applications, it appears to be mandatory to include (more) background knowledge into the mining process, especially in frequent subgraph mining. Without problem-specific chemical, biological, social etc. background knowledge, it seems infeasible to

restrict the huge search space to the relevant part and to reduce the output from simply frequent subgraphs to the actually interesting and useful ones.

Literatur

- [1] L.A. Adamic and E. Adar. Friends and Neighbors on the Web. *Social Networks* 25(3):211–230. Elsevier Science, Amsterdam, Netherlands 2003
- [2] M. Berlingerio, F. Bonchi, B. Bringmann, and A. Gionis. Mining Graph Evolution Rules. *European Conference on Machine Learning and Knowledge Discovery in Databases (ECML/PKDD 2009, Bled, Slovenia)*, LNAI 5781:115–130. Springer, Heidelberg, Germany 2009
- [3] C. Borgelt and M.R. Berthold. Mining Molecular Fragments: Finding Relevant Substructures of Molecules. *Proc. IEEE Int. Conf. on Data Mining (ICDM 2002, Maebashi, Japan)*, 51–58. IEEE Press, Piscataway, NJ, USA 2002
- [4] C. Borgelt. On Canonical Forms for Frequent Graph Mining. *Proc. 3rd Int. Workshop on Mining Graphs, Trees and Sequences (MGTS'05, Porto, Portugal)*, 1–12. ECML/PKDD 2005 Organization Committee, Porto, Portugal 2005
- [5] C. Borgelt. Combining Ring Extensions and Canonical Form Pruning. *Proc. 4th Int. Workshop on Mining and Learning with Graphs (MLG 2006, Berlin, Germany)*, 109–116. ECML/PKDD Organization Committee, Berlin, Germany 2006
- [6] K. Borgwardt, C. Ong, S. Schönauer, S.V.N. Vishwanathan, A. Smola, and H.-P. Kriegel. Protein Function Prediction via Graph Kernels. *Bioinformatics* 21(1):47–56. Oxford University Press, Oxford, United Kingdom 2005
- [7] J. Callut, K. Francoise, M. Saerens, and P. Dupont. Semi-supervised Classification from Discriminative Random Walks. *European Conference on Machine Learning and Knowledge Discovery in Databases (ECML/PKDD 2008, Antwerp, Belgium)*, LNAI 5211:162-177. Springer, Heidelberg, Germany 2008
- [8] D. Chakrabarti and C. Faloutsos. Graph Mining: Laws, Generators and Algorithms. *ACM Computing Surveys* 36(1):article 2. ACM Press, New York, NY, USA 2006
- [9] Y. Chi, S. Nijssen, R.R. Muntz, and J.N. Kok. Frequent Subtree Mining - An Overview. *Fundamenta Informaticae* XXI:1001-1038. IOS Press, Amsterdam, Netherlands 2001
- [10] C. Cortes, P. Haffner, and M. Mohri. Rational Kernels: Theory and Algorithms. *Journal of Machine Learning Research* 5:1035–1062. Massachusetts Institute of Technology, Cambridge, MA, USA 2004
- [11] D.J. Cook and L.B. Holder. Graph-Based Data Mining. *IEEE Trans. on Intelligent Systems* 15(2):32–41. IEEE Press, Piscataway, NJ, USA 2000
- [12] D.J. Cook and L.B. Holder. *Mining Graph Data*. J. Wiley & Sons, Chichester, United Kingdom 2007

- [13] C. Desrosiers, P. Garnier, P. Hansen, and A. Hertz. Improving Frequent Subgraph Mining in the Presence of Symmetry. *Proc. 5th Int. Workshop on Mining and Learning with Graphs (MLG 2007, Florence, Italy)*, 25–30. MLG 2007 Organization Committee, Florence, Italy 2007
- [14] C. Desrosiers and G. Karypis. Within-Network Classification Using Local Structure Similarity. *European Conference on Machine Learning and Knowledge Discovery in Databases (ECML/PKDD 2009, Bled, Slovenia)*, LNAI 5781:260–275. Springer, Heidelberg, Germany 2009
- [15] *DTP AIDS Antiviral Screen (HIV Data Set) — Subset from 2001*. Developmental Therapeutics Program (DTP), National Cancer Institute, USA 2001
http://dtp.nci.nih.gov/docs/aids/aids_data.html
- [16] P. Erdős and A. Rényi. On Random Graphs. *Publ. Math. Debrecen* 6:290–297. Institute of Mathematics, University of Debrecen, Hungary, 1959
- [17] R. Feldman and J. Sanger. *The Text Mining Handbook*. Cambridge University Press, Cambridge, United Kingdom 2006
- [18] C. Fellbaum (ed.) *WordNet — An Electronic Lexical Database*. MIT Press, Cambridge, MA, USA 1998
- [19] P.W. Finn, S. Muggleton, D. Page, and A. Srinivasan. Pharmacore Discovery Using the Inductive Logic Programming System PROGOL. *Machine Learning*, 30(2-3):241–270. Kluwer, Amsterdam, Netherlands 1998
- [20] B. Gallagher, H. Tong, T. Eliassi-Rad, and C. Faloutsos. Using Ghost Edges for Classification in Sparsely Labeled Networks. *Proc. 14th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining*, 256-264. ACM Press, New York, NY, USA 2008
- [21] T. Gärtner. Exponential and Geometric Kernels for Graphs. *Proc. NIPS*2002 Workshop on Unreal Data: Principles of Modeling Nonvectorial Data*. NIPS*2002 organization committee, Vancouver, Canada 2002
- [22] T. Gärtner, P. Flach, and S. Wrobel. On Graph Kernels: Hardness Results and Efficient Alternatives. *Proc. Annual Conf. Computational Learning Theory*, 129–143. Springer, New York, NY, USA 2003
- [23] T. Gärtner. *Kernels for Structured Data*. World Scientific, Hackensack, NJ, USA 2008
- [24] H. Hofer, C. Borgelt, and M.R. Berthold. Large Scale Mining of Molecular Fragments with Wildcards. *Intelligent Data Analysis*, 8:495–504. IOS Press, Amsterdam, Netherlands 2004
- [25] T. Horvath, J. Ramon, and S. Wrobel. Frequent Subgraph Mining in Outerplanar Graphs. *Proc. 12th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining*, 197–206. ACM Press, New York, NY, USA 2006

- [26] J. Huan, W. Wang, and J. Prins. Efficient Mining of Frequent Subgraphs in the Presence of Isomorphism. *Proc. 3rd IEEE Int. Conf. on Data Mining (ICDM 2003)*, 549–552. IEEE Press, Piscataway, NJ, USA 2003
- [27] G. Jeh and J. Widom. SimRank: A Measure of Structural Context Similarity. *Proc. ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining (KDD 2002, Edmonton, Canada)*, 538–543. ACM Press, New York, NY, USA 2002
- [28] H. Kashima, K. Tsuda, and A. Inokuchi. Marginalized Kernels between Labeled Graphs. *Proc. Int. Conf. on Machine Learning*, 321–328. Morgan Kaufmann, San Mateo, CA, USA 2003
- [29] H. Kashima, K. Tsuda, and A. Inokuchi. Kernels on Graphs. In: K. Tsuda, B. Schölkopf, and J. Vert (eds.) *Kernels and Bioinformatics*, 155–170. MIT Press, Cambridge, MA, USA 2004
- [30] L. Katz. A New Status Index Derived from Sociometric Analysis. *Psychometrika* 18(1):39–43. Psychometric Society, Greensboro, NC, USA 1953
- [31] I.R. Kondor and J.D. Lafferty. Diffusion Kernels on Graphs and Other Discrete Structures. *Proc. Int. Conf. on Machine Learning*, 315–322. Morgan Kaufmann, San Mateo, CA, USA 2002
- [32] S. Kramer, L. de Raedt, and C. Helma. Molecular Feature Mining in HIV Data. *Proc. 7th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining (KDD 2001, San Francisco, CA)*, 136–143. ACM Press, New York, NY, USA 2001
- [33] J. Kunegis and A. Lommatzsch. Learning Spectral Graph Transformations for Link Prediction. *Proc. 26th Int. Conference on Machine Learning (ICML'09, Montreal, Canada)*, 1–8. ACM Press, New York, NY, USA 2009
- [34] M. Kuramochi and G. Karypis. Frequent Subgraph Discovery. *Proc. 1st IEEE Int. Conf. on Data Mining (ICDM 2001, San Jose, CA)*, 313–320. IEEE Press, Piscataway, NJ, USA 2001
- [35] Liben-Nowell and J. Kleinberg. The Link Prediction Problem for Social Networks. *Proc. 12th Annual ACM Int. Conf. on Information and Knowledge Management (CIKM'03)*, 556–559. ACM Press, New York, NY, USA 2003
- [36] B. McKay. Practical Graph Isomorphism. *Congressus Numerantium* 30:45-87. Utilitas Mathematica Publishing, Winnipeg, Canada 1981
- [37] T. Meinl, C. Borgelt, and M.R. Berthold. Mining Fragments with Fuzzy Chains in Molecular Databases. *Proc. 2nd Int. Workshop on Mining Graphs, Trees, and Sequences (MGTS 2004 at PKDD 2004, Pisa, Italy)*, 49–60. ECML/PKDD Organization Committee, Pisa, Italy 2004
- [38] S. Milgram. The Small World Problem. *Psychology Today* 1:61–67. Sussex Publishers, New York, NY, USA 1967
- [39] S. Nijssen and J.N. Kok. A Quickstart in Frequent Structure Mining Can Make a Difference. *Proc. 10th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining (KDD2004, Seattle, WA)*, 647–652. ACM Press, New York, NY, USA 2004

- [40] S. Nijssen and J.N. Kok. The Gaston Tool for Frequent Subgraph Mining. *Electronic Notes in Theoretical Computer Science* 127(1):77-87. Elsevier Science, Amsterdam, Netherlands 2005
- [41] L. Page, S. Brin, R. Motwani, and T. Winograd. *The PageRank Citation Ranking: Bringing Order to the Web*. Technical Report, Stanford University, Palo Alto, CA, USA 1999
- [42] J. Ramon and S. Nijssen. Polynomial-Delay Enumeration of Monotonic Graph Classes. *Journal of Machine Learning Research* 10:907–929. MIT Press, Cambridge, MA, USA 2009
- [43] P. Sevon, L. Eronen, P. Hintsanen, K. Kulovesi, and H. Toivonen. Link Discovery in Graphs Derived from Biological Databases. *Proc. 3rd Int. Workshop on Data Integration in the Life Sciences (DILS'06, Hinxton, United Kingdom)*, LNBI 4705:35–49. Springer, Heidelberg, Germany 2006
- [44] L.T. Thomas, S.R. Valluri, and K. Karlapalem. MARGIN: Maximal Frequent Subgraph Mining. *Proc. 6th IEEE Int. Conf. on Data Mining (ICDM 2006)*, 1097–1101. IEEE Press, Piscataway, NJ, USA 2006
- [45] J. Travers and S. Milgram. An experimental study of the Small World Problem. *Sociometry* 32(4):425-443. American Sociological Association, Washington, DC, USA 1969
- [46] S.V.N. Vishwanathan and A. Smola. Fast Kernels for String and Tree Matching. *Advances in Neural Information Processing Systems 15*, 569–576. MIT Press, Cambridge, MA, USA 2003
- [47] S.V.N. Vishwanathan, K.M. Borgwardt, and N.N. Schraudolph. Fast Computation of Graph Kernels. *Advances in Neural Information Processing Systems 19*, 1449–1456. MIT Press, Cambridge, MA, USA 2007
- [48] S.V.N. Vishwanathan, K.M. Borgwardt, I.R. Kondor, and N.N. Schraudolph. Graph Kernels. *Journal of Machine Learning Research* (submitted). Massachusetts Institute of Technology, Cambridge, MA, USA 2008 (available at arXiv:0807.0093)
- [49] D.J. Watts and S.H. Strogatz. Collective Dynamics of ‘Small-World’ Networks. *Nature* 393(6684):440-442. Nature Publishing Group, New York, NY, USA 1998
- [50] X. Yan and J. Han. gSpan: Graph-Based Substructure Pattern Mining. *Proc. 2nd IEEE Int. Conf. on Data Mining (ICDM 2003, Maebashi, Japan)*, 721–724. IEEE Press, Piscataway, NJ, USA 2002
- [51] X. Yan and J. Han. Closegraph: Mining Closed Frequent Graph Patterns. *Proc. 9th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining (KDD 2003, Washington, DC)*, 286–295. ACM Press, New York, NY, USA 2003

Simulation von Neuronalen Netzen für E-Learning

Viktor Gottfried, Ulrich Lehmann, Uwe Klug, Burkhard Neumann,
Michael Schneider, Johannes Brenig, Sven Buhl

Institut für Computer Science, Vision und Computational Intelligence
Fachhochschule Südwestfalen, Frauenstuhlweg 31, 58644 Iserlohn
www.cvci.fh-swf.de Tel. (02371) 566-303 Fax (02371) 566-209
E-Mail: info-cv-ci@listen.fh-swf.de

Kurzfassung: Die Methoden der Computational Intelligence haben viele Anwendungsbereiche erschlossen. Für eine nachhaltige Breitenwirkung ist jedoch auch ein breiter Kenntnisstand und angehenden Fachleuten eine wichtige Voraussetzung. Dazu gehört eine möglichst gute und jederzeit abrufbare Wissensbasis. Künstliche Neuronale Netze (KNN) stellen für Einsteiger und Quereinsteiger aus Informatik und Ingenieurwissenschaften ein abstraktes Thema dar. Um die Verhaltensweise von Neuronalen Netzen auf einfache und spielerische Weise zu vermitteln, wurde eine e-Simulation für die E-Learningplattform Moodle entwickelt.

1 Einleitung

Mit der E-Learningplattform Moodle [4] [5] bietet die Fachhochschule Südwestfalen für die Studenten eine Ergänzung zu den normalen Lernveranstaltungen im Bereich der Computational Intelligence. Dabei wird viel Wert darauf gelegt, die Theorie durch dynamische Beispiele leicht verständlich und zugänglich zu machen. Studierende aus Bachelor- (BA) und Masterstudiengängen (MA), Angewandte Informatik (BA), Bio- und Nanotechnologie (BA) und Computer Vision and Computational Intelligence (MA) sollen spielerisch in die Wissensgebiete eingeführt werden. Dies soll zu Beginn der Lehrveranstaltung ohne tiefere Kenntnis von Simulationssystemen durch einfaches Probieren ermöglicht werden.

Im Rahmen dieses Projektes sollte eine Anwendung entwickelt werden, mit der die Mustererkennung mit Künstlichen Neuronalen Netzen simuliert werden kann. Da die Künstlichen Neuronalen Netze ein sehr abstraktes Thema darstellen und die Zielgruppe in den meisten Fällen über sehr wenig Kenntnisse in diesem Bereich verfügt, sollte der Nutzer vor allem die Funktionalitäten und die Abläufe im Hintergrund erkennen und leicht verstehen können.

2 Entwicklung

Die Anwendung sollte so aufgebaut werden, dass drei Trainingspattern und ein Testpattern in Form einer 6x5 Matrix zu Verfügung stehen, die durch den Benutzer vorgegeben und verändert werden können. Die Pattern dienen als Eingabe für ein 30-30 Netzes. Dabei werden die binären Werte (1,-1) der 30 Felder eines Patterns auf das Netz gelegt. Ein weiteres Pattern (Response Pattern) stellt die Ausgabe des Netzes dar. Der Benutzer kann zwei verschiedene Netze trainieren lassen. Als Trainingsverfahren wurde die Hebbsche-Regel (Netz A) sowie der Backpropagation Trainingsalgorithmus (Netz B) implementiert.

$$\Delta W_{ij} = n * a_i * o_j$$

Formel 1: Hebbsche-Regel [1]

$$\Delta W_{ij} = n * o_i * \delta_j$$

Formel 2: Backpropagation [1]

Als Aktivierungsfunktion wird für das Netz A die Schwellwertfunktion verwendet und für das Netz B die logistische Funktion.

$$a_j = \begin{cases} 1 & \text{falls } net_j(t) \geq 0 \\ -1 & \text{sonst} \end{cases}$$

Formel 3: Schwellwertfunktion [1]

$$a_i = 1 / (1 + \exp(-net_j))$$

Formel 4: logistische Funktion [1]

Die Unterschiede von Aktivierungsfunktionen werden anhand des Ausgabepatterns gezeigt. Die Schwellwertfunktion liefert binäre Werte und das Ausgabemuster zeigt nur zwei unterschiedliche Farben an. Die logistische Aktivierungsfunktion liefert Werte zwischen 0 und 1, die Farben für das Ausgabemuster werden demensprechend skaliert.

Während des Trainings kann der Ablauf und das Verhalten des Netzes anhand einer Lernkurve beobachtet werden. Desweiteren hat der Benutzer die Möglichkeit optional einen Blick auf einen Ausschnitt aus der Architektur des Neuronalen Netzes zu erhalten

sowie die Gewichtsmatrix zu visualisieren. Durch die zwei Grafiken erhält man eine Übersicht über die Abläufe im Hintergrund.

Die Gewichtsmatrix (Abb. 1) besteht aus 30x30 Feldern, jedes Feld stellt eine Verbindung zwischen zwei Neuronen dar. Ein Feld wird mit einer Farbe, die vom jeweiligen Gewicht abhängig ist, gefüllt. Die Farbskalierung ist im positiven Bereich von hell grün bis dunkel grün. Im negativen Bereich von hell rot bis dunkel rot. Bei 0 ist die Farbe grau.

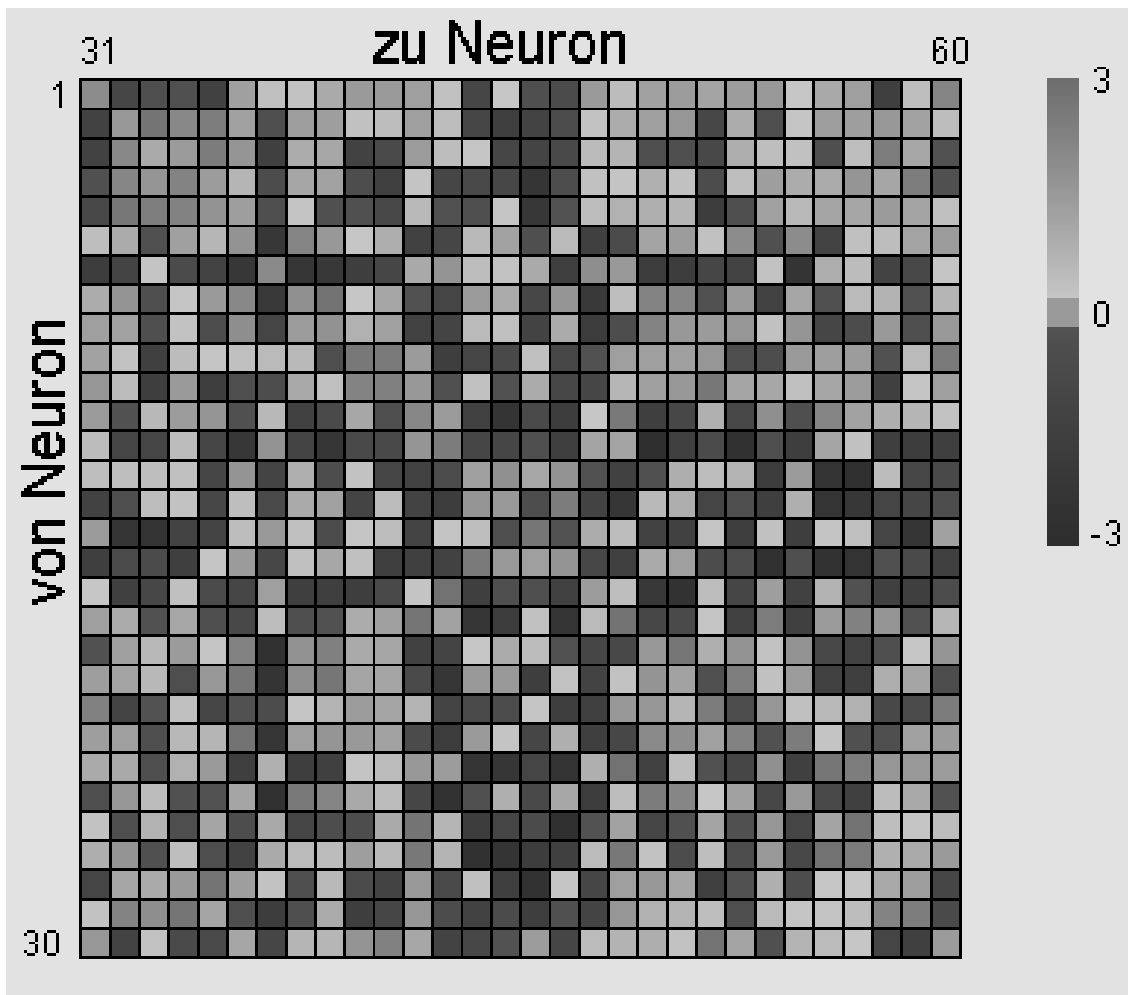


Abbildung 1: Gewichtsmatrix

Die Architekturanzeige (Abb. 2) stellt einen Ausschnitt des Künstlichen Neuronales Netzes dar. Dabei wird ein vorgefertigtes Bild als Hintergrundgrafik auf das Panel gesetzt. Die Werte werden dynamisch während des Trainings und der Musterberechnung auf die Grafik gezeichnet. Um die Übersicht zu wahren werden nur die ersten sowie die letzten zwei Neuronen angezeigt.

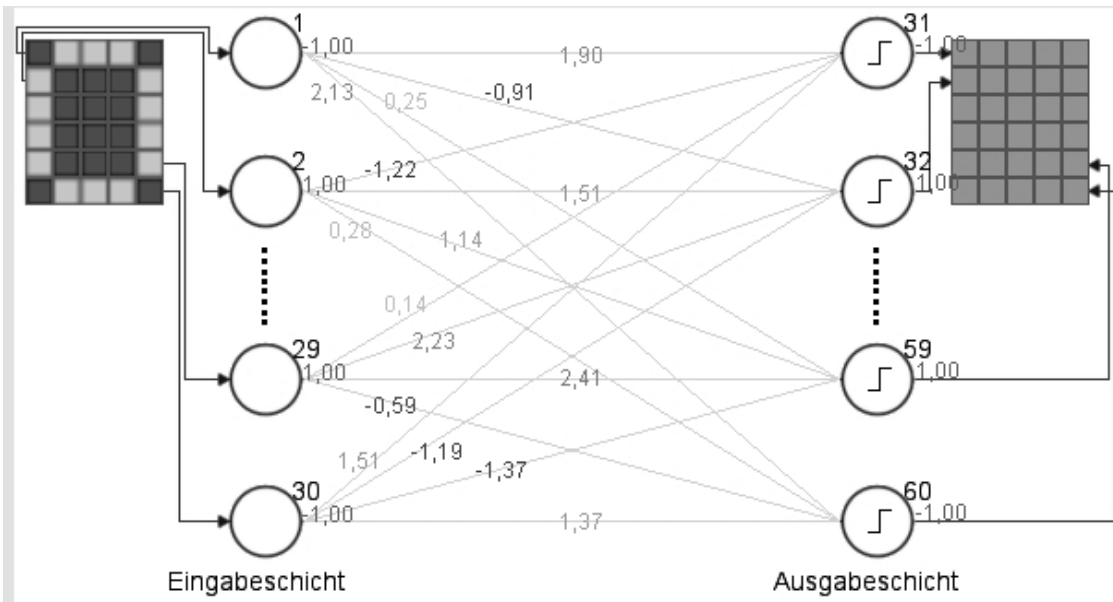


Abbildung 2: Architektur

3 Das Applet

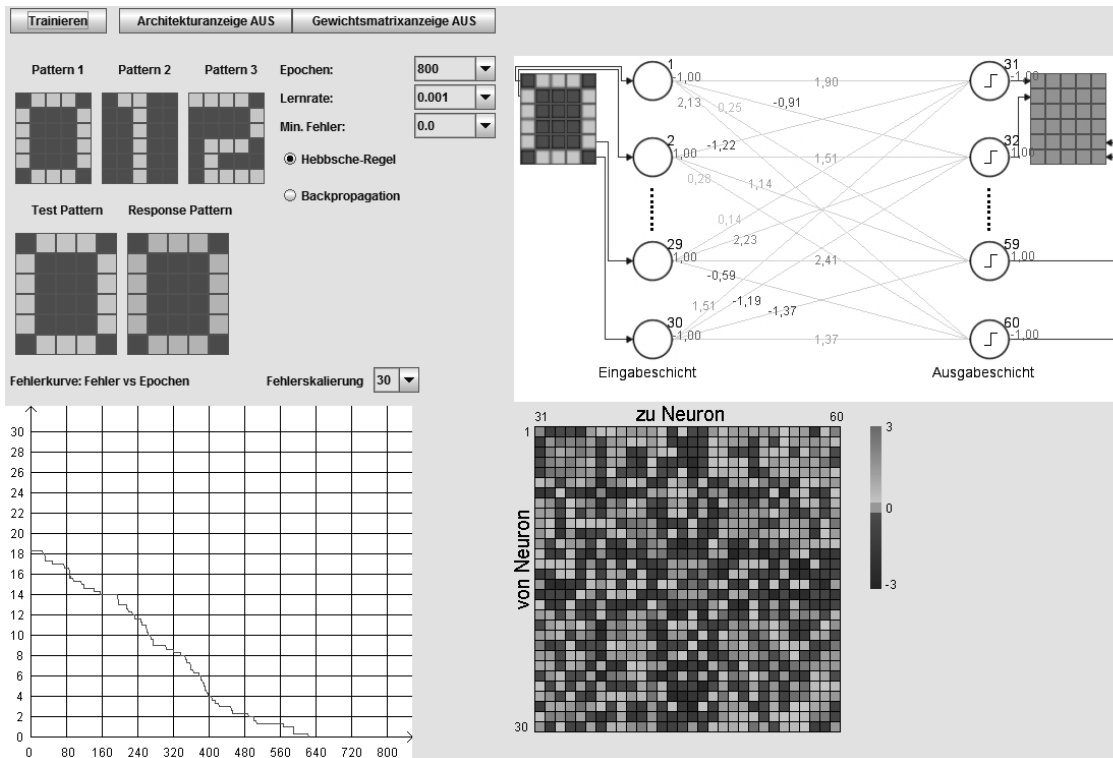


Abbildung 3: Applet

Die Abbildung 3 zeigt das Java-Applet in der vollständigen Form. Der Benutzer hat die Möglichkeit drei Trainingsmuster einzustellen. Desweiteren kann der Benutzer die folgenden Trainingseinstellungen verändern:

Epochen: Gibt die Anzahl der Trainingszyklen vor.

Lernrate: Konstante für die Größe der Gewichtsänderung

Min. Fehler: Gibt an, nach welchem Trainingserfolg das Training beendet wird.

Durch die zwei Radiobuttons hat der Benutzer die Möglichkeit zwischen zwei verschiedenen Netzen zu wählen, welche mit unterschiedlichen Verfahren (Hebbsche-Regel und Backpropagation) trainiert werden.

Wird das Training vom Benutzer gestartet, so werden die drei Trainingspattern vom KNN trainiert. Während des Trainings wird die Lernkurve gezeichnet und die Veränderungen des Netzes anhand der Architekturgrafik und der Gewichtsmatrix visuell dargestellt. Nach dem Training hat der Benutzer die Möglichkeit das trainierte KNN zu evaluieren, indem er das Testmuster verändert. Nach jeder Änderung wird ein neues Ausgabemuster vom KNN berechnet und in dem Feld Response Pattern dargestellt.

4 Danksagung

Diese Arbeit wird am Institut für Computer Science, Vision und Computational Intelligence an der Fachhochschule Südwestfalen im Rahmen der vom Ministerium für Innovation, Wissenschaft, Forschung und Technologie des Landes Nordrhein-Westfalen geförderten Kompetenzplattform Computer Vision und Computational Intelligence (KOPF CV&CI) bearbeitet.

5 Literaturverzeichnis

1. **Zell, Andreas.** *Simulation neuronaler Netze.* München : Oldenbourg Wissenschaftsverlag, 2003. ISBN 3-486-24350-0.
2. **VDI/VDE-Gesellschaft Mess- und Automatisierungstechnik.** *Computational Intelligence: Künstliche Neuronale Netze in der Automatisierungstechnik Begriff und Definition.* Berlin : Beuth Verlag GmbH, 2001.
3. **Ullenboom, Christian.** *Java ist auch eine Insel.* s.l. : Galileo Computing, 2009. 978-3-8362-1371-4.
4. <http://www.moodle.de/>
5. Moodle-Plattform der Fachhochschule Südwestfalen <http://193.174.71.93/>

Entscheidungshilfe zur Umgebungsklimatisierung von Kulturgütern mit Methoden der Fuzzy-Theorie

Christian Arnold und Bernd Cuno

Hochschule Fulda, Fachbereich Elektrotechnik und Informationstechnik
Marquardstr. 35, 36039 Fulda
Tel.: (0661) 9640-557 und -570; Fax: (0661) 9640-559
Email: christian.arnold@et.hs-fulda.de, bernd.cuno@et.hs-fulda.de

1 Einleitung und Problemstellung

Im Sinne des *Übereinkommens zum Schutz des Kultur- und Naturerbes der Welt* [1] zählen das Kultur- und Naturerbe zu den Objekten, deren Wert als unschätzbar einzu-stufen ist und welche zu den unersetzbaren Besitztümern der ganzen Menschheit gehören. Kulturgüter sind von Menschenhand geschaffen, dies können in Gebäuden oder Räumen (wie z.B. Museen, Bibliotheken oder Archiven) gelagerte Objekte sein (mobiles Kulturgut) oder Gebäude und Orte selbst (ortsfestes Kulturgut wie zum Beispiel historische Gebäude, Denkmäler, Kirchen oder Gärten). Diese sind der natürlichen Alterung unterworfen, welche zwar unaufhaltsam ist, aber durch konservatorische Bemühungen verzögert werden kann. Diese Aufgabe wird jedoch durch den derzeitigen Wandel im Museumsbetrieb erschwert. Während früher der Genuss von Kunst das Privileg eines kleinen Personenkreises war, müssen kulturelle Einrichtungen und Exponate heute wachsende Besuchermassen und klimatische Belastungen ertragen: Bibliotheken werden als Museumsbibliotheken betrieben, Kirchen dienen als Ausrichtungsorte für Konzerte und historische Gebäude werden für Wechselausstellungen genutzt (mit der Notwendigkeit von Kunsttransporten). Durch den gestiegenen Anspruch an die Authentizität des Originals soll die Objektoberfläche als Zeugnisebene bewahrt werden [2]. Dies bedeutet, dass der Erhaltung von Kulturgut mit dessen historischen Veränderungen eindeutig der Vorzug vor der Wiederherstellung des originalen Zustands gegeben wird (Konservierung vor Restaurierung bzw. *non toccare* vor *toccare*). Die präventive Konservierung umfasst Maßnahmen zum Schutz eines Objekts ohne Eingriffe am Objekt selbst [2]. Eine geeignete Maßnahme der präventiven Konservierung zur Verlangsamung der Alterungs- und vor allem der Zerstörungsprozesse ist die Stabilisierung der Umgebungsverhältnisse am Kulturobjekt innerhalb eines konservatorisch vorgegebenen Toleranzbereichs. Neben den Anforderungen an Lichtverhältnisse, Staub, usw. spielen die Anforderungen an das Umgebungsklima der Kulturobjekte eine entscheidende Rolle.

Ein Klima, welches für ein Material geeignet ist, kann andere zerstören. Eine niedrige relative Luftfeuchte kann zum Verspröden und Schrumpfen von Materialien führen, eine hohe relative Luftfeuchte dagegen zum Quellen, zur Oxidation von Metallen sowie zum Befall durch Schimmel und Mikroorganismen. Schrumpfen bzw. Quellen verursachen wiederum Zerstörungen in Form von Rissbildung bzw. Spannungen im Materialverbund. Nach [2] führt "eine Reduktion der Temperatur um 5°C bei chemisch wenig stabilen Objekten zu einer ungefähren Halbierung der Zerfallsgeschwindigkeit und somit zu einer ungefähren Verdoppelung der Lebensdauer der Objekte". Andererseits kann unterhalb von 13,2°C die Zinnpest auftreten, indem sich das silberweiße, metallische β -Zinn in das schwarze pulverförmige α -Zinn umwandelt ([2]).

Neben den absoluten Klimawerten spielen bei den Zerstörungsprozessen deren Änderungsgeschwindigkeiten eine wesentliche Rolle. Während Materialien in vielen Fällen die Jahreszyklen von Temperatur und Feuchte unbeschadet überstehen, können sich unter Umständen durch Tageszyklen oder sonstige schnelle Änderungen (zum Beispiel Lüften durch Fensteröffnen oder Stellgrößeneingriffe von Klimaanlage) interne Spannungen anstauen, welche zu irreversiblen Materialalterungen führen. Haben sich aber durch die Zyklen interne Spannungen und in ihrer Folge Risse gebildet, so resultiert daraus wiederum eine geringere Empfindlichkeit gegenüber Temperatur- und Feuchteschwankungen, d.h. das Material hat sich „akklimatisiert“, bleibt jedoch weiterhin geschädigt [3]. Da Kulturobjekte im Allgemeinen aus unterschiedlichen Materialien zusammengesetzt sind und diese sowohl auf bestimmte Klimawerte als auch Klimagradienten unterschiedlich reagieren, müssen bei der Festlegung eines geeigneten Objekt-, Vitrinen- bzw. Raumklimas komplexe Zusammenhänge berücksichtigt werden [4]. Das ideale Umgebungsklima kann nur auf Basis eines Kompromisses gefunden werden [3]. Dies gilt besonders, wenn unterschiedliche Objekte in einem Raum aufbewahrt werden und zusätzlich der Erhalt der unter Umständen denkmalwerten Gebäudehülle sowie Nutzeranforderungen (wie der Behaglichkeit) berücksichtigen werden sollen. Zudem sind die Kosten zur Klimatisierung (Investitions- und Betriebskosten von Heizungen bzw. Klimaanlage) zu berücksichtigen. Ferner sind aus Gründen des Gesundheits- sowie Bestandsschutzes Schimmelpilz- sowie Schädlingsbefall zu verhindern. Die wissenschaftlich fundierte Ermittlung von Klimaanforderungen für Archive, Museen, Bibliotheken und denkmalwerte Gebäude wird somit zu einer komplexen Aufgabe, deren Lösung die Zusammenarbeit von Restauratoren, Denkmalpflegern, Bauphysikern, Materialkundlern, Mikrobiologen, Museumsbetreibern und Bibliothekaren erfordert.

In der Museumspraxis wurden im Lauf der Zeit unterschiedliche Klimaempfehlungen aufgestellt, zum Standard erklärt und zum Teil hart verteidigt. [5] zeigt jedoch, dass sich die zum Teil vehement vorgetragenen Standards über Jahre hinweg hauptsächlich an Kosten und Performance von Klimaanlage und nicht an wissenschaftlichen Erkenntnissen zu Anforderungen des Kulturguts orientierten. Die hauptsächlich in angelsächsischen Ländern geführte Klimadiskussion bevorzugte relativ enge Toleranzgrenzen. Die 1960 in Paris aufgestellten Forderungen des *Conseil International des Musées (ICOM)* nach einer Temperatur von $20^{\circ}\text{C} \pm 4^{\circ}\text{C}$ und einer relativen Feuchte von $55\% \pm 5\%$ ([6]) galten lange als anzustrebender Kompromiss. Allerdings kam es durch das starre Festhalten an diese Vorgaben auf verschiedene Gebäude und Nutzungsintensitäten zu umfangreichen Schäden am Sammlungsgut [3]. 1982 wurde ein Paper veröffentlicht, in dem auf Basis der Finiten Elemente Analyse Schädigungsprozesse an Gemälden wissenschaftlich untersucht wurden ([5]). Wissenschaftler der *Smithsonian Institution (SCMRE)* lösten 1994 durch eine Pressemitteilung zur Zulässigkeit von Klimaschwankungen eine heftige Debatte aus ([5]), welche zu einem Wandel in den Klimaanforderungen im Depot- und Ausstellungsbereich führte. Ausführliche Angaben zum Stand der Anforderungen finden sich in [2], [7], [10] und [11]. Derzeit konzentriert sich die Richtlinienarbeit auch unter dem Aspekt der Energieeinsparung - auf weitere Anpassungen der Vorgaben mit Orientierungen an Jahreszeiten und Klimazonen, wie DIN EN 15757 (siehe [12]) exemplarisch zu entnehmen ist. Die Vorgaben optimaler Umgebungsbedingungen für Kulturgut sind nach wie vor umstritten, besonders für ein aus unterschiedlichen Materialien bestehendes und unter Umständen vorgeschädigtes Kulturobjekt. Im Rahmen der vorliegenden Arbeit soll ein Verfahren zur wissenschaftlich fundierten Festlegung der Umgebungsbedingungen beschrieben werden, wobei zunächst die Klimabedingungen betrachtet werden.

2 Lösungsansatz

Neben den Zielvorgaben an die Umgebungsbedingungen stellt sich die Frage, wie diese hergestellt und eingehalten sowie kritische Situationen bereits im Vorfeld vermieden werden können. Die Auswirkungen von Sollwerten und Handlungsfolgen wie zum Beispiel Temperatur- und Feuchtesollwerte, Betrieb von Lüftungsanlagen oder Abschatten der Sonneneinstrahlung sind oft nicht vollständig überschaubar, sondern lediglich abschätzbar. Hierzu sind prädiktive Strategien erforderlich. Zudem ist zu berücksichtigen, dass das Klima räumlich verteilt ist und die Erfassung nur an bestimmten Referenzpunkten erfolgt. Da die Klimadynamik in einem Gebäude, Raum oder Kulturobjekt sehr komplex ist, soll das lokale Klima am Standort des zu betrachtenden Kulturobjekts bzw. im Kulturobjekt selbst durch vereinfachte Beobachter abgeschätzt werden.

Idealerweise werden die gewünschten klimatischen Verhältnisse über das gesamte Jahr hinweg durch die natürliche Klimatisierung, d.h. ohne Eingriff gewährleistet. In vielen Fällen ist jedoch ein aktiver Eingriff in das Raumklima erforderlich, um die Zielvorgaben einzuhalten. Je nach technischer Ausstattung stehen hierfür unterschiedliche Methoden und Verfahren zur Verfügung. Das im Folgenden beschriebene Klimaführungssystem soll geeignete Maßnahmen zur Klimatisierung empfehlen bzw. selbst einleiten. Es soll für ein breites Anwendungsspektrum entwickelt werden und sowohl in technisch gut ausgestatteten Gebäuden wie repräsentativen Museen oder Galerien als auch in Gebäuden ohne Klimatechnik wie kleinere Kirchen oder Heimatmuseen einsetzbar sein.

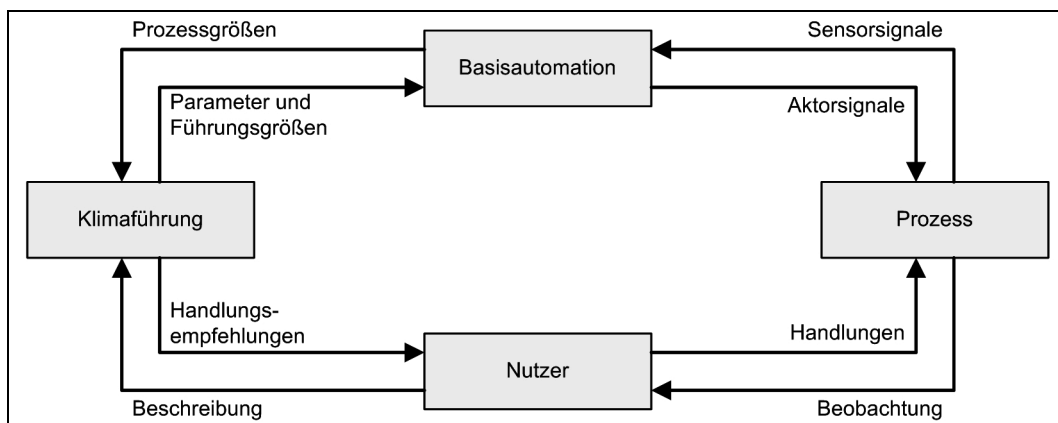


Abbildung 1: Integration der Klimaführung in den Prozess über Basisautomation und Nutzer

Im offline-Betrieb kann das Klimaführungssystem aufgrund der (über Messwerte oder Benutzereingaben) verfügbar gemachten Informationen Handlungsempfehlungen ausgeben, wie zum Beispiel Fenster- oder Türlüftung, Beschattung, Besucherbeschränkung, Be- oder Entfeuchtereinsatz, usw.. Die vorhandene technische Ausstattung legt dabei die Einbeziehung des Nutzers fest. Sind technische Komponenten wie Heizungs-, Klima-, Abschattungs-, Be- und Entfeuchtungsanlagen vorhanden, können diese je nach Automatisierungsgrad durch den Benutzer bedient oder durch den Austausch von Sensor- und Aktorsignalen über eine Basisautomatisierung betrieben werden. Im Falle des vollautomatischen Betriebs kommuniziert die Klimaführung nicht mehr mit dem Benutzer, sondern nur noch mit der Basisautomatisierung. Installationen technischer Komponenten in historischen und denkmalwerten Gebäuden werden aus Platz- und Kostengründen nur bedingt akzeptiert. Weiterhin bestehen häufig Bedenken gegenüber dem Einsatz der Klimatechnik in Museen, Archiven und Galerien, da Fehlfunktionen und Ausfälle mit ihren oftmals katastrophalen Folgen für das Kulturgut befürchtet werden. In den meisten Anwendungen ist daher eine Mischform der Ansteuerungen zu erwarten und der vollautomatische oder der komplett manuelle Betrieb stellt eher die Ausnahme

dar. Die Klimaführung soll durch die Vorhersage von Prozessgrößen künftige Handlungen so planen, dass definierte Zielvorgaben eingehalten werden. Hierzu eingesetzte Methoden können gleichzeitig zur Diagnose und Überwachung der Umgebungsbedingungen verwendet werden, wobei dann nicht künftige sondern vergangene und aktuelle Daten betrachtet werden. Aufgrund der flexiblen Einsatzmöglichkeiten muss vor der oben beschriebenen Betriebsphase eine Projektierung der Klimaführungssysteme (Abbildung 2) erfolgen. Hier ist zunächst eine Schnittstellendefinition zu Nutzer, Basisautomation und externen Datenquellen wie etwa der Wettervorhersage festzulegen.

Weiterhin erfordert die Vorhersage von Prozessgrößen ein Modell, welches den Zusammenhang von Referenzklimawerten unter Berücksichtigung bekannter zukünftiger Einflüsse berechnet. Raumklimaprozesse haben in aller Regel eine hohe Komplexität. Um den Aufwand in der Modellierung zu reduzieren wird ein vereinfachter Modellansatz angestrebt, dessen Parametrierung durch eine Raumcharakterisierung (wie Gebäudeart, Beschaffenheit, Aufbau der Gebäudehülle, Geometrie, usw.) und einem hinterlegten Regelwerk erfolgt. Nicht erfassbare Prozessgrößen können durch eine Beschreibung wissensbasierter Zusammenhänge des Nutzers geschätzt werden, so dass diese anschließend nach dem Beobachterprinzip aus erfass- und prognostizierbaren Prozessgrößen abgeleitet werden können. Um die Umgebungsbedingungen einer Situation einschätzen zu können, ist eine Bewertung gemessener, geschätzter und vorhergesagter Prozessgrößen erforderlich. Es wurde bereits geschildert, dass die Anforderungen an die Umgebungsbedingung von Kulturgütern von unterschiedlichen Kriterien bestimmt werden, welche untereinander zu gewichten sind. Zudem sind zu beobachtende und vorherzusagende Prozessgrößen mit Unsicherheiten behaftet, welche entsprechend berücksichtigt werden sollten. Die Konfiguration der Bewertungsmethode erfordert somit ebenfalls eine Beschreibung und Charakterisierung durch den Nutzer in der Projektierung. Da sowohl Anforderungen als auch Prozessgrößen unscharf formuliert werden, liegt die Anwendung der Fuzzy-Theorie als Bewertungsmethode nahe.

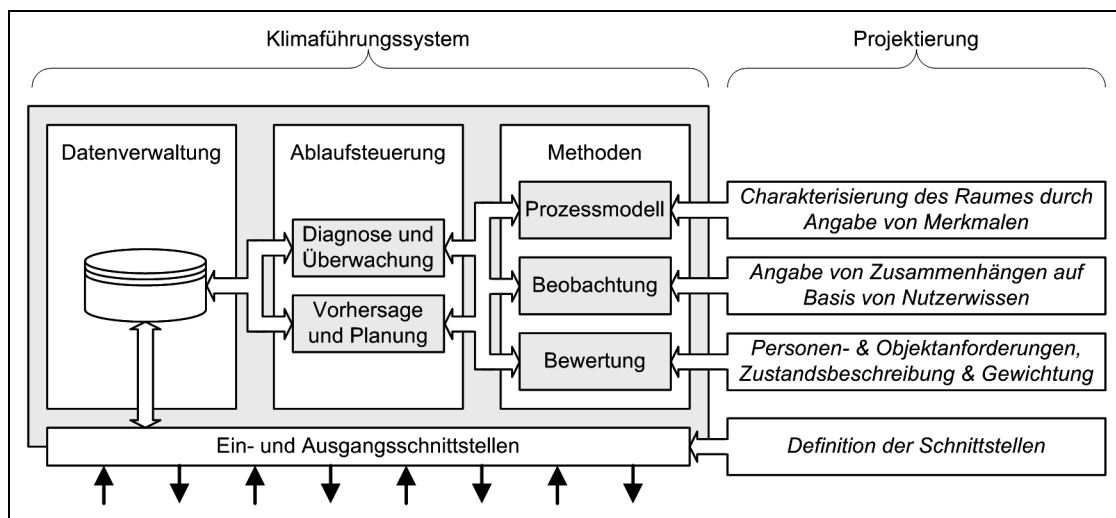


Abbildung 2: Projektierung der Klimaführung an den Anwendungsfall

3 Bewertung über Methoden der Fuzzy-Theorie

Da im vorliegenden Anwendungsbeispiel mehrere Kriterien bewertet werden müssen, wird eine multikriterielle Bewertungsmethode benötigt. Üblicherweise können mehrere Kriterien durch einen Gütevektor dargestellt werden. Sind die Gewichtungen einzelner Gütemaße untereinander bekannt, so kann zudem ein aggregiertes Gütemaß, wie etwa

ein Summenkriterium aufgestellt werden. Häufig führt das Aufstellen der Kriterien und deren Gewichtungen zu einer nur schwer nachzuvollziehenden mathematischen Formulierung, so dass die intuitive Möglichkeit zur Parametrierung der Gütefunktionen verloren geht. Im Folgenden wird ein Verfahren vorgestellt, mit welchem Prozessgrößen durch unscharfe Gütekriterien anschaulich aufgestellt, untereinander gewichtet und unter Berücksichtigung der Unschärfe der Prozessgrößen bewertet werden können.

3.1 Unscharfe Formulierung von Zielvorgaben (bzw. Sollwerten)

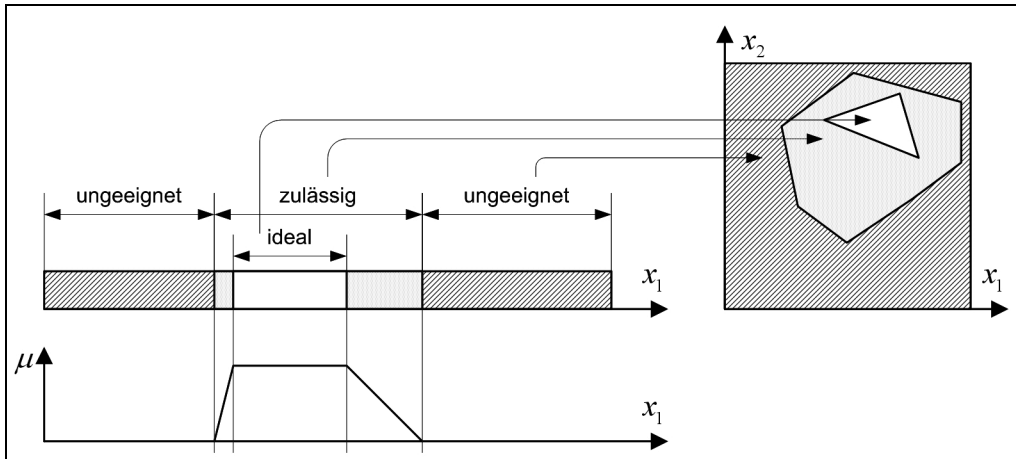


Abbildung 3: Aufstellung idealer, zulässiger und ungeeigneter Bereiche zur Bewertung von Prozessgrößen und Darstellung durch eine Zugehörigkeitsfunktion

In der Anwendung wird das Aufstellen von Gütefunktionen erschwert, da klimatische Anforderungen nur unscharf formuliert werden können. Häufig wird hier lediglich zwischen „idealen“, „zulässigen“ und „ungeeigneten“ Bereichen unterschieden. Diese Zuordnung kann in Abhängigkeit einer oder mehrerer Größen erfolgen (Abbildung 3). Auf eine solche Darstellung wird beispielsweise bei der Angabe von Bereichen der relativen Luftfeuchtigkeit für die Lagerung von Materialien und Kulturgütern zurückgegriffen. Andere Beispiele sind unscharfe Angaben von Temperatur- und Feuchtigkeitsbereichen zur Beschreibung der Behaglichkeit (Grad des Wohlbefindens von Personen) oder des Schimmelpilzwachstums, siehe Abbildung 4.

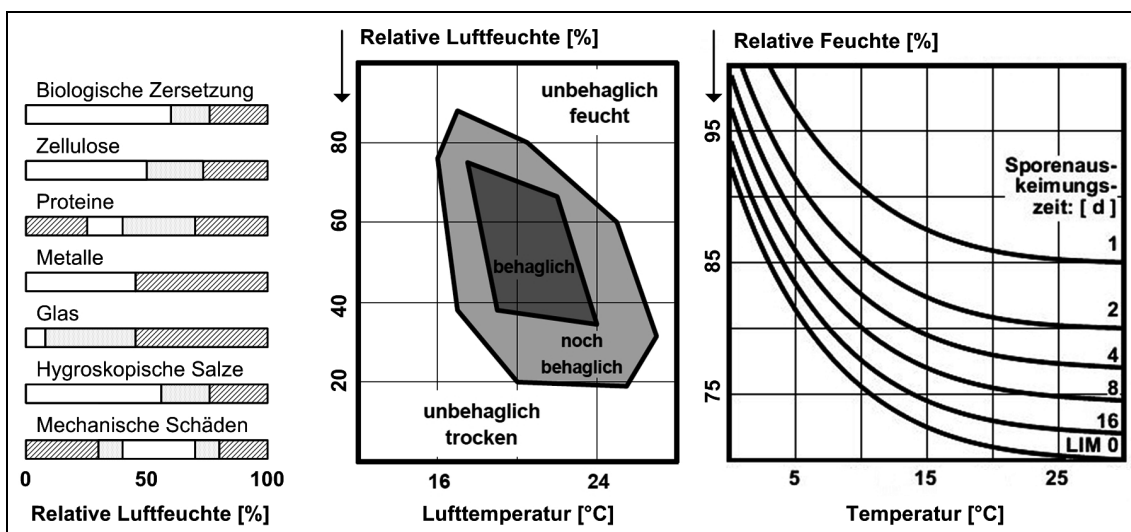


Abbildung 4: Material- (links in Anlehnung an [4], und [13]), Behaglichkeitsanforderungen (mitte, Quelle [14]) und Schimmelpilz-Isoplethen (Quelle [15])

Unter dem Begriff *Fuzzy-Decision-Making* versteht man die Entscheidungsfindung mit Methoden der Fuzzy-Theorie. Hierbei werden Gütekriterien durch Zugehörigkeitsfunktionen ausgedrückt, so dass der Zugehörigkeitsgrad zum oben erwähnten „idealen Bereich“ ausgedrückt wird. Demnach können durch Zugehörigkeitsfunktionen Fuzzy-Ziele und Fuzzy-Restriktionen ausgedrückt werden ([16], [17], [18]). Die Vereinigungsmenge der Zugehörigkeitsfunktionen spaltet dabei den zulässigen Lösungsraum auf. Dabei können Situationen mit einer Zugehörigkeit zur optimalen Situation bewertet werden, wobei das Optimum das Maximum der Vereinigungsmenge ist [19], da dieses alle Gütekriterien und Restriktionen gleichermaßen berücksichtigt, Abbildung 5.

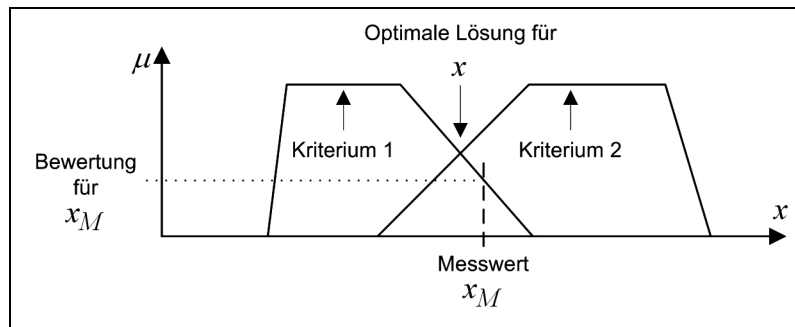


Abbildung 5: Zugehörigkeitsfunktionen als unscharfe Zielvorgabe

Bei dieser Art der Bewertung handelt es sich aufgrund der Unabhängigkeit von der Zeit um eine Bewertungsmöglichkeit für eine statische im Allgemeinen nichtlineare multikriterielle Optimierung. Wird die Zeitabhängigkeit der Prozessgrößen berücksichtigt, so ist es erforderlich, die Bewertung innerhalb eines betrachteten Zeitabschnittes durchzuführen, beispielsweise zu jedem Prädiktionsschritt innerhalb des Prädiktionshorizontes mit entsprechender Aggregation der Einzelbewertungen zu einem Gütewert. Hierbei wird dann ein Bewertungsmaß für eine dynamische Optimierung, also einen mehrstufigen Entscheidungsprozess, bereitgestellt [17].

3.2 Unscharfe Formulierung der Prozessgrößen (bzw. Istwerte)

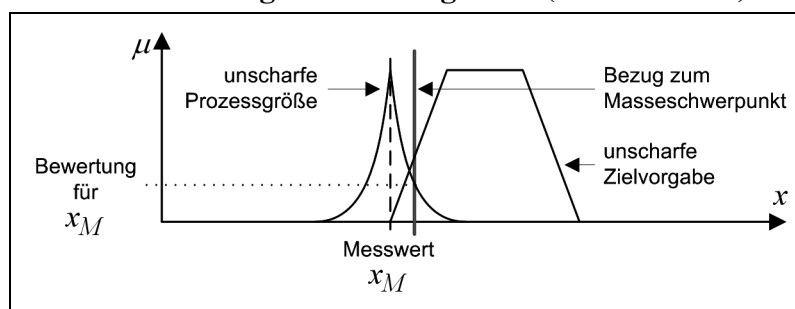


Abbildung 6: Unscharfe Zielvorgabe und unscharfe Prozessgröße

Neben der Unschärfe der Zielanforderung kann auch die Erfassung von Prozessgrößen - wie bereits oben erläutert - unscharf sein, was durch Prognoseverfahren, Beobachtungen oder Messungenauigkeiten zu begründen ist. Die Darstellung von Prozessgrößen über Fuzzy-Zahlen oder Fuzzy-Intervalle ermöglicht eine Formulierungsart, durch welche die Unschärfe in weiteren Berechnungen effizient berücksichtigt werden kann. Eine Zugehörigkeitsfunktion drückt hier die Zugehörigkeit von Prozessgröße zu realen möglichen Werten aus. Je kleiner die Fläche der Zugehörigkeitsfunktion, desto geringer wird die Möglichkeit der Annahme alternativer realer Werte, eine ideale Erfassung wäre somit die Zuordnung durch einen Singleton. Mit dieser Methode können auch statistische Unsicherheiten berücksichtigt werden, wobei die Zuordnung durch Wahrscheinlich-

keitswerte die untere Grenze für die Zuordnung durch Möglichkeitswerte festlegt. Eine Weiterführung dieser Betrachtung ist in [16] und [20] zu finden. Um unscharfe Prozessgrößen mit unscharfen Zielvorgaben zu bewerten, wird die Zugehörigkeit des Massechwerpunktes der Vereinigungsmenge beider Zugehörigkeitsfunktionen vorgeschlagen. Auf diese Weise werden nicht nur erfasste, sondern auch mögliche Werte der Prozessgrößen in der Bewertung berücksichtigt, Abbildung 6 verdeutlicht dies grafisch.

4 Fuzzy-Modifikatoren

Die bisherigen Ausführungen zur Definition unscharfer Ziele und unscharfer Formulierungen von Prozessgrößen basieren auf unveränderlichen Lagen idealer, zulässiger und ungeeigneter Gebiete sowie einem linearen Übergang innerhalb des zulässigen Gebietes. Durch eine Operation auf eine Fuzzy-Menge kann diese in eine neue Fuzzy-Menge transformiert werden. Eine solche Operation wird als Fuzzy-Modifikator bezeichnet [21]. Eine der Hauptanwendungen für den Einsatz von Fuzzy-Modifikatoren ist die Berücksichtigung linguistischer Hecken in der Fuzzyfizierung. Nach [22] sind linguistische Hecken sprachliche Ausdrücke, welche andere sprachliche Ausdrücke unscharf machen oder den Unschärfegrad eines sprachlichen Ausdrucks verändern.

4.1 Umsetzung von Modifikationen

Sei eine Fuzzy-Menge definiert durch:

$$M = \{(x, \mu(x, \bar{p})) | (x \in X)\} \quad (1)$$

Eine Modifikation der Fuzzy-Menge kann erfolgen, indem die Grundmenge der Fuzzy-Menge, die Parameter der Zugehörigkeitsfunktion oder die Fuzzy-Menge selbst verändert werden (Abbildung 7). Die Modifikation der Parameter einer Zugehörigkeitsfunktion wird auch als typgebunden bezeichnet [22], da hier im Gegensatz zu den anderen Modifikationsrealisierungen eine mathematische Beschreibung der Zugehörigkeitsfunktion erforderlich ist. Je nach ausgewähltem Modifikatortyp stehen somit mehrere Möglichkeiten der Umsetzung zur Verfügung.

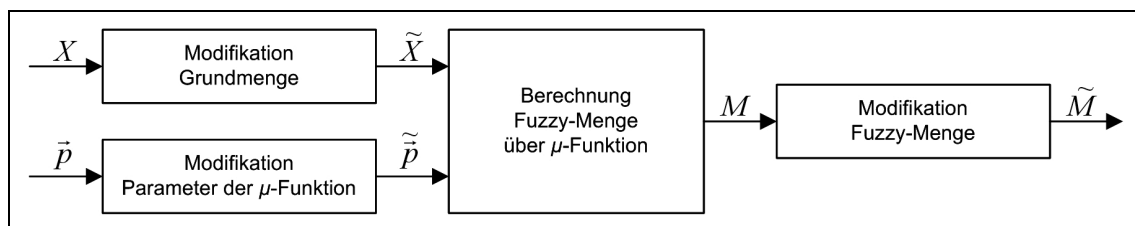


Abbildung 7: Möglichkeiten zur Realisierung der Modifikation

4.2 Modifikatortypen

In der Literatur (wie [18], [21], [22], [23]) aufgeführte Modifikatoren lassen sich im Wesentlichen in die drei Klassen *Shifting*, *Powering* und *Intensification* unterteilen. Beim *Shifting* werden die Elemente einer Fuzzy-Menge teilweise oder vollständig auf der Abszisse verschoben. Meist wird die gesamte Fuzzy-Menge verschoben, indem eine Verschiebung der Grundmenge erfolgt oder der Parameter der Zugehörigkeitsfunktion um den entsprechenden Wert verschoben werden (Abbildung 8 links, Gleichung (2)).

$$\tilde{M} = \{(x, \mu(x, a, b)) | (x \in X + \Delta x)\} = \{(x, \mu(x, a + \Delta x, b)) | (x \in X)\} \quad (2)$$

Eine *Powering*-Modifikation verändert die Zugehörigkeitswerte der Fuzzy-Menge im Übergangsbereich, so dass entweder eine Dehnung in Richtung der scharfen Trägermenge (Abbildung 8 mitte \tilde{A}_1) oder eine Konzentration in Richtung der scharfen Kernmenge (Abbildung 8 mitte \tilde{A}_2) erfolgt. Meist wird diese Modifikation durch das Potenzieren der Fuzzy-Mengenelemente realisiert, wobei für Exponenten kleiner 1 eine Dehnung und für Exponenten größer 1 eine Konzentration erfolgt. Bei mehreren Kriterien kann dieser Modifikator auch zur Vergabe von Gewichtungen verwendet werden: hierbei werden Kriterien mit hoher Gewichtung mit hohen Exponenten und Kriterien mit geringen Gewichtungen entsprechend mit kleinen Exponenten modifiziert.

$$\tilde{M} = \left\{ \left\langle x, \mu(x)^\alpha \right\rangle \mid x \in X \right\} \quad (3)$$

Bei der *Intensification* wird der Bereich unterhalb einer Zugehörigkeitsgrenze (eines Levels) auf eine andere Art modifiziert wie der oberhalb der Grenze, so dass der Übergang zwischen den Zugehörigkeiten von 0 und 1 schärfer verläuft: der Kontrast zwischen „Element der Menge“ oder „nicht Element der Menge“ wird verschärft. Zur Realisierung wird in der Literatur (z.B. [22]) häufig die Kontrastintensivierung vorgeschlagen, wobei der Level auf 0,5 definiert ist (Gleichung (4)). Prinzipiell kann jedoch auch jeder andere Wert zwischen 0 und 1 durch entsprechende Anpassung erfolgen.

$$\tilde{\mu}(x) = \begin{cases} 2 \cdot \mu(x)^2 & ; 0 \leq \mu(x) \leq 0,5 \\ 1 - 2 \cdot (1 - \mu(x))^2 & ; 1 \geq \mu(x) \geq 0,5 \end{cases} \quad (4)$$

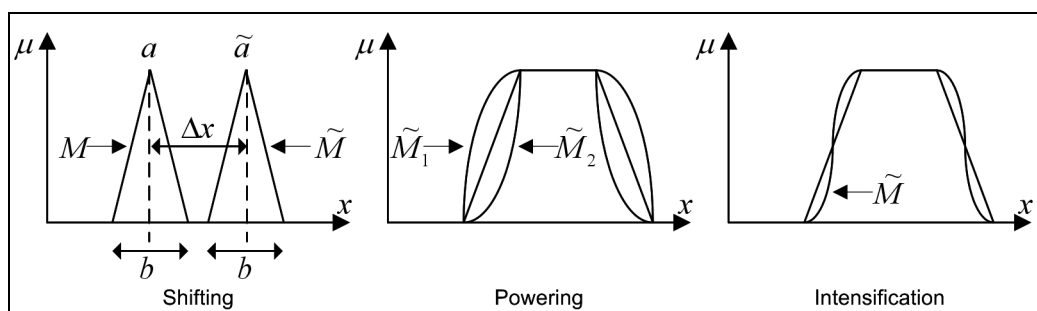


Abbildung 8: Hauptklassen von Modifikatoren

4.3 Anwendung der Modifikatoren im Klimaführungssystem

Ausgangslage für die Betrachtungen zur Bewertung von Prozessgrößen war die Ablage und Verarbeitung von Beschreibungen des Nutzers zum Prozess in der Projektierungsphase des Klimaführungssystems. Neben der Auswahl verschiedener Zielvorgaben können diese durch die Zustandsbeschreibung modifiziert werden. Beispielsweise macht eine Modifikation einer Zielvorgabe über den *Powering*-Modifikator Sinn, wenn sich Zustand oder Umgebung eines Materials verändert haben, so dass die linguistischen Hecken „robuster“ (z.B. nach einer Restauration) oder „empfindlicher“ (nach einer Vorschädigung) berücksichtigt werden könnten. Ebenfalls ist die Anwendung des *Shifting*-Operators denkbar, wenn linguistische Hecken vom Typ „eher feuchter“ oder „eher trockener“ zu berücksichtigen sind. Gleiches gilt für die Modifikation unscharfer Prozessgrößen: etwa kann diese in der Vorhersage bei steigendem Prognosehorizont durch Angaben wie „unsicherer“ umschrieben werden. Es ist ersichtlich, dass durch die Anwendung von Fuzzy-Modifikatoren eine Anpassung an herrschende Bedingungen und Zustände erfolgen kann, die Parameter der Modifikation allerdings nicht vereinheitlicht ermittelt werden können. Der Grad einer Modifikation hängt von mehreren Kriterien ab,

welche zum Teil kompensatorische Wirkungen haben können, so dass die Ermittlung des anzuwendenden Modifikators und dessen Parametrierung über ein separates Regelwerk ermittelt werden sollte. Abbildung 9 zeigt schematisch die Verarbeitung von Nutzerangaben, abgelegtem Expertenwissen und den Prozessgrößen zu dessen Bewertung.

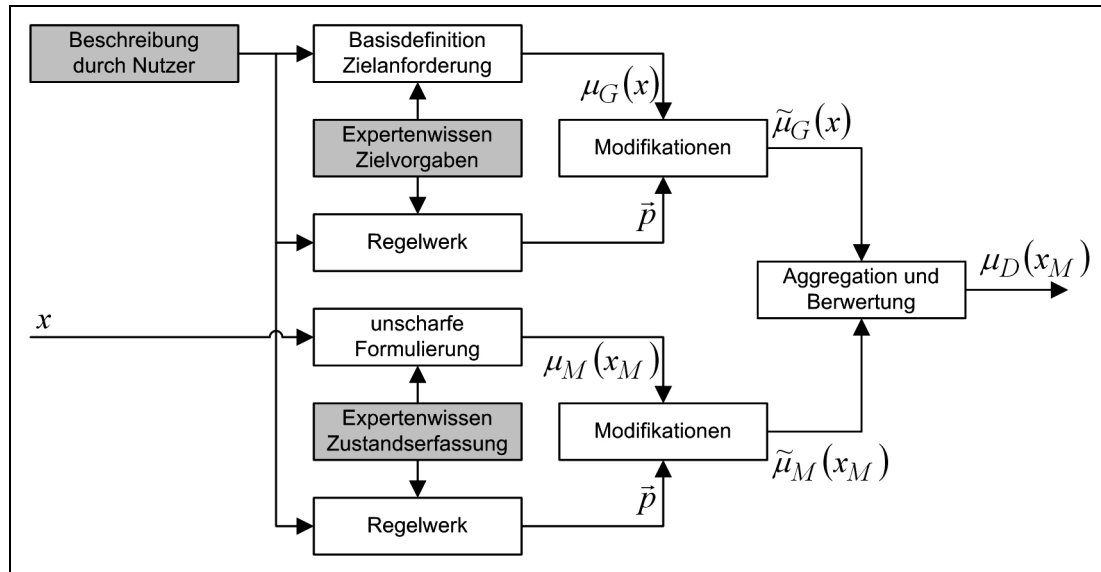


Abbildung 9: Struktur zur Modifikation von Zielvorgabe und Prozessgrößenformulierung

5 Anwendungsbeispiel zur Bewertung der relativen Luftfeuchte

Im Beispiel wird die relative Luftfeuchte der Umgebung von Büchern bewertet. Hierzu werden Messdaten analysiert, welche in einer Bibliothek gesammelt wurden. Um die Zielvorgabe abzuleiten, werden die einzelnen Materialien betrachtet: der Buchblock der Bücher sei aus Papier und der Einband mit Leder oder Pergament überzogen (Materialien anderer Bestandteile werden hier vernachlässigt). Angaben verschiedener Literaturquellen (wie [3], [7], [10], [11]) grenzen günstige Feuchtebereiche ein, unterscheiden sich jedoch. Der ideale Bereich wird als der angenommen, welcher sich in allen Angaben überdeckt und der zulässige als dieser bei welchem mindestens eine Angabe in der Literatur zu finden ist. Abbildung 10 verdeutlicht die Grenzen resultierender Bereiche.

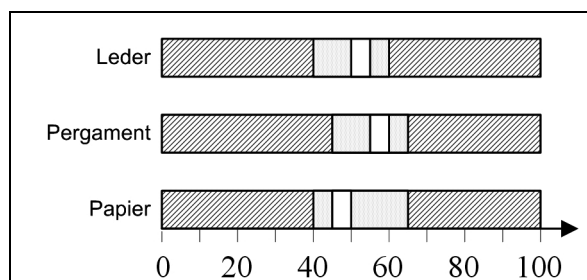


Abbildung 10: Anforderungen der Materialien an die relative Feuchte

Der ideale Bereich kann nicht gleichzeitig für alle Materialien erreicht werden, so dass ein Kompromiss geschlossen werden muss. Die Kriterien werden in Zugehörigkeitsfunktionen mit zunächst linearem Übergang im zulässigen Bereich transformiert und die Vereinigungsmenge als Zielvorgabe gebildet. Die Unschärfe im Übergangsbereich wird durch *Powering*-Modifikationen genauer spezifiziert. Bei einem wichtigeren Kriterium (hier die Erhaltung des Papiers) werden die Elemente der Fuzzy-Menge quadriert und bei einem eher weniger wichtigen Kriterium (hier die Erhaltung von Leder und Pergament) wird aus ihnen die Wurzel gezogen.

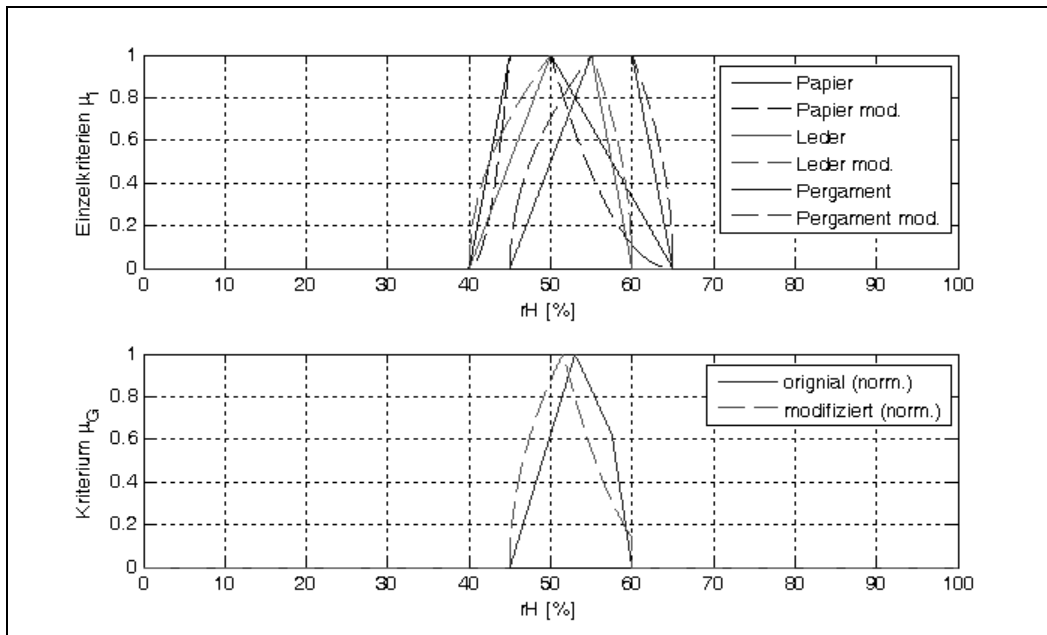


Abbildung 11: Unscharfe Formulierung von Zielvorgaben

Abbildung 11 zeigt die einzelnen Kriterien der Materialien (oben) und die resultierende Zielvorgabe (unten), jeweils in originaler und modifizierter Form. Abbildung 12 oben zeigt den Verlauf der relativen Feuchte innerhalb des betrachteten Zeitraums für eine Messstelle. Im unteren Teil der Abbildung wurde das Szenario mit den beiden Zielvorgaben (Abbildung 11 unten) bewertet. Die durchgezogene Linie repräsentiert die Bewertung für den ursprünglichen und die gestrichelte Linie für den modifizierten Kompromiss. Es wird deutlich, dass die Bewertungen je nach Kriterium unterschiedlich ausfallen, so dass die zunächst trivial scheinende Modifikation einen signifikanten Einfluss auf die Bewertung der Situation hat. Entsprechend ist dann auch die Betriebsführung zur Klimatisierung des Raumes anzupassen.

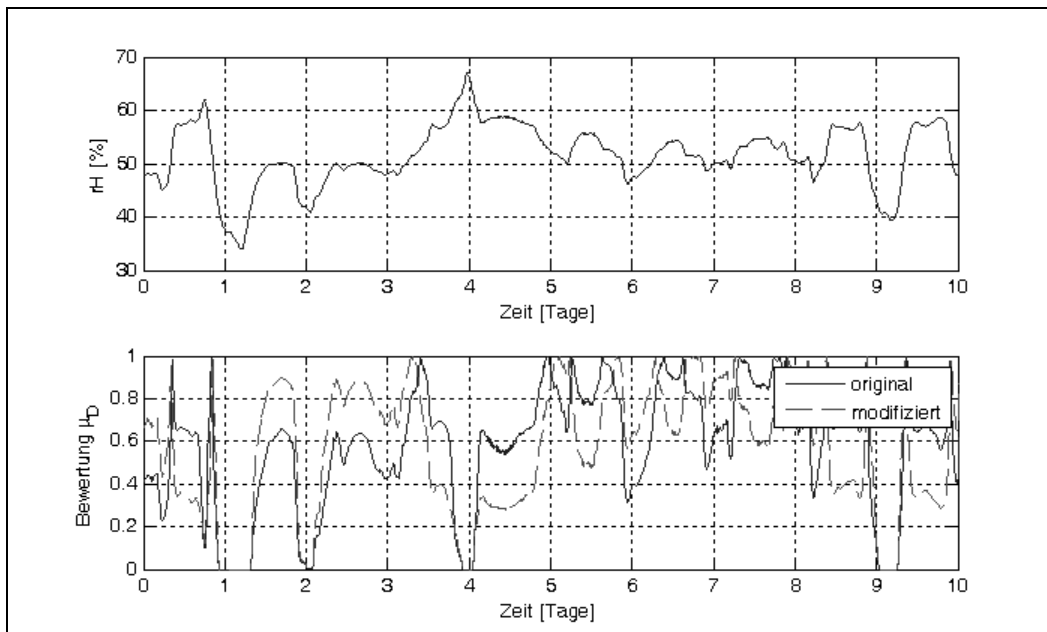


Abbildung 12: Bewertung der relativen Luftfeuchte in Szenario

6 Ausblick und Zusammenfassung

Die Erhaltung von Kulturgütern erfordert die Schaffung entsprechender Umgebungsbedingungen, wobei diese nicht mit anderen Anforderungen, wie die von Besuchern oder bezüglich der Betriebskosten, übereinstimmen müssen. Hierbei wird eine Kompromissfindung erforderlich, welche durch unscharfe Zielvorgaben erschwert wird. Weiterhin sind Prozessgrößen im Allgemeinen mit Unschärfen behaftet. Mit der Fuzzy-Theorie ist es möglich unscharfe Zielvorgaben zu formulieren und die Unschärfe von Prozessgrößen in der Bewertung zu berücksichtigen. Modifikatoren können dabei zur Einbeziehung linguistischer Hecken aus der Zustandsbeschreibung, zur Gewichtung von Kriterien und zur Anpassung der Unschärfe von Prozessgrößen eingesetzt werden.

Derzeit laufende Arbeiten am Bewertungsmodul befassen sich mit der Entwicklung einer strukturieren Methode zur intuitiven Systemkonfiguration durch den Benutzer. Im Beitrag konzentrierte sich die Bewertung auf aktuelle Werte der Prozessgrößen, jedoch nicht auf dessen Änderungsgeschwindigkeiten. Zur Erhaltung von Kulturgut sind diese jedoch zu berücksichtigen, zu deren Bewertung wird an Methoden gleichartiger Ansätze gearbeitet. Parallel dazu wird die Akzeptanz des beschriebenen Klimaführungssystems in entsprechenden Anwenderkreisen (wie Restauratoren, Betreiber von Bibliotheken und Museen) geprüft und eine Wissensakquise bezüglich der Zielvorgaben und den beschriebenen Regelwerken zur Modifikation bei betreffenden Experten (Restauratoren, Bauphysiker, etc.) durchgeführt. Wie der Bericht zeigt, können zur Lösung des zunächst recht trivial scheinenden Problems (der Schaffung idealer Umgebungsbedingungen) die Methoden der *computational intelligence* hilfreiche Unterstützung bieten: nicht nur zur Bewertung unscharfer Prozessgrößen bei unscharfen Zielvorgaben, sondern auch zur Modellierung und Lösung von Optimierungsproblemen ist dessen Einsatz denkbar.

Literatur

- [1] N.N.: *Convention concerning the protection of the world cultural and natural heritage*. Welterbekonvention der 17. Generalkonferenz der UNESCO, Paris, 1972
- [2] Huber, J.; von Lerber, K.: *Handhabung und Lagerung von mobilem Kulturgut: Ein Handbuch für Museen, kirchliche Institutionen, Sammler und Archive*; transcript Verlag, Bielefeld, 2003
- [3] Brechtold, W.; Burmester A.; Eckermann, W.; Großes Schmidt, H.; Hass, J.; Kaluza, J.; Puttfarcken, G.; Schimmler, C.; Wolf, H.-G.: *Raumklima in Museen*; Fachinstitut Gebäude-Klima e.V., Bietigheim-Bissingen, 1999
- [4] Waller, C.: *Wissenswertes zur „Präventiven Konservierung“*; Long Life of Art, Unterrichtseinheiten als Webversion, http://cwaller.de/fr_didaktik.htm, letzter Aufruf am 03.06.2009
- [5] Erhardt, D.; Tumosa, C. S.; Mecklenburg, M. F.: *Applying Science to the Question of Museum Climate*; in Padfield, T.; Borchersen, K. (Eds.): *Museum Microclimates*; National Museum of Denmark, Kopenhagen, 2007
- [6] Kilian, R.; Sedlbauer, K.; Krus, M.: *Klimaanforderungen für Kunstwerke und Ausstattung historischer Gebäude*; Fraunhofer Institut für Bauphysik, IBP-Mitteilung 462, 2005

- [7] Kühn, H.: *Erhaltung und Pflege von Kunstwerken - Material und Technik, Konservierung und Restaurierung*; Klinkhardt & Biermann Verlag, München, 2001, 3. Auflage
- [10] Hilbert, G.-S.: *Sammlungsgut in Sicherheit*; Institut für Museumskunde, Staatliche Museen zu Berlin – Preußischer Kulturbesitz, Gebr. Mann Verlag, Berlin, 2002, 3. Auflage
- [11] Jeberien, A.; Lindhorst, C.: *Achtung Kunst- & Kulturgut - Ein Leitfaden zum Umgang mit Ausstellungsobjekten*; Dokument auf Homepage, 2003, download am 26.05.2009: <http://www.2care.org/Exponatleitfaden1.0.pdf>
- [12] *Erhaltung des kulturellen Erbes - Vorgaben für Temperatur und relative Feuchte zur Reduzierung von klimabedingter mechanischer Beschädigung in organischen hygroskopischen Materialien*; DIN EN 15757, Normenausschuss Bauwesen im DIN, Beuth Verlag, 2008
- [13] Erhardt, D.; Mecklenburg, M.: *Relative humidity re-examined*; in: *Preventive Conservation*; International Institute for Conservation, preprints, Ottawa, 1994, Seite 32-38
- [14] Sedlbauer, K.; Breuer, K.; Kaufmann, A.: *Der Mensch in Räumen - Risiko oder Behaglichkeit?*; in *Bauen mit Holz in kompakter Umsetzung*, 9. Internationales Holzbau-Forum, Fraunhofer IRB Verlag, Garmisch-Partenkirchen, 2003
- [15] Sedlbauer, K.; Zilling, W.; Krus, M.: *Isoplethensysteme ermöglichen eine Abschätzung von Schimmelpilzbildung*; Fraunhofer Institut für Bauphysik, IBP-Mitteilung 388, 2001
- [16] Bandemer, H.; Gottwald, S.: *Einführung in die Fuzzy-Methoden*; Akademie Verlag, Berlin, 1993, 4. Auflage
- [17] Kacprzyk, J.: *Multistage fuzzy control*; John Wiley & Sons Verlag, Chichester, 1997
- [18] Zimmermann, H.-J.: *Fuzzy set theory - and its applications*; Kluwer Academic Publishers, Boston, 1994, second edition
- [19] Bernard, T.: *Ein Beitrag zur gewichteten multikriteriellen Optimierung von Heizungs- und Lüftungsregelkreisen auf Grundlage des Fuzzy Decision Making*; Dissertation, Uni Karlsruhe, Fakultät für Maschinenbau, 2000
- [20] Rommelfanger, H. J. : *Fuzzy Decision Support-Systeme - Entscheiden bei Unschärfe*; Springer Verlag, Berlin, 1994, 2. Auflage
- [21] De Cock, M; Kerre, E. E.: *Fuzzy modifiers based on fuzzy relations*; Information Sciences, Vol. 160, Seite 173–199, 2004
- [22] Biewer, B.: *Fuzzy-Methoden – Praxisrelevante Rechenmodelle und Fuzzy-Programmiersprachen*; Springer Verlag, Berlin, 1997
- [23] Bosc, P.; HadjAli, A.; Pivert, O.: *Modeling Fuzzy Modifiers by Means of a Proximity Relation*; IEEE Annual meeting of the North American Fuzzy Information Processing Society 2006 (NAFIPS 2006), Seite 200-205

Imitation Learning for Visual Robotic Behaviors

**Krishna Kumar Narayanan, Felipe Posada, Frank Hoffmann,
Torsten Bertram**

Lehrstuhl für Regelungssystemtechnik, Technische Universität Dortmund

Otto-Hahn-Str. 4, 44227 Dortmund

Tel.: (0231) 755-3592

Fax: (0231) 755-2752

E-Mail: krishna.narayanan@tu-dortmund.de

Abstract

This paper proposes learning from imitation for visual obstacle avoidance and corridor following in indoor environments. Supervised learning from images becomes feasible by generating and selecting appropriate global visual features that capture the robots local environment. A learning by demonstration approach is utilized to learn the demonstrated trajectories as a mapping between visual features computed with respect to a panoramic view onto a corresponding robot motion. The mobile robot is equipped with a 3D PMD, a 2D monocular and an omni-directional camera. Floor, wall and obstacles are classified in the 3D frontal image according to the orientation and size of surface regions. The classification is further augmented by propagating the segmentation on the panoramic view reconstructed from the omni-directional image. Several visual features computed from the segmented image capture the appearance of the robots local environment. These candidate features reflect the position of objects similar to proximity sensors, but also capture other relevant aspects such as corridor orientation or tempo-spatial relationships of the perception. A training set is generated from a sonar based wandering behavior, for which during execution the visual perception are recorded in conjunction with executed motor action. An instance based learning scheme is employed to imitate the sonar based behavior with a corresponding relationship between perceptions and actions. From the set of candidate features the ones that best correlate with the executed actions are identified in a wrapper approach via feature selection with forward chaining. The experimental results on a mobile robot indicate that the acquired visual behavior is robust and able to generalize to novel scenarios that include situations and environments not presented during training.

1 Introduction

The progress in computer vision methods in conjunction with the increased computational power at decreasing costs of computer vision systems stimulated research and development in vision based mobile robot navigation during the last decade [1, 2, 3, 4, 5, 6]. Nevertheless in spite of the advancements, robust reactive visual navigation still remains a substantial challenge as

- 2D images provide no depth information to reconstruct the 3D scene directly,
- images contain a large amount of information which requires substantial data aggregation and fusion prior to learning relationships between visual perception and robot actions,

- indoor environments vary substantially in geometry, visual appearance and lighting conditions and
- the relevance of alternative visual cues such as intensity, texture, optical flow is context dependent.

These problems aggravate the conventional behavior based approach to identify prototypical scenarios and to design context specific albeit general reactive behaviors for them. One feasible approach is to reconstruct the geometry of the local environment from single or multiple views. The vision system more or less mimics the conventional functionality of proximity sensors, such that existing navigation behaviors can be again utilized. However, such an approach is prone to ignore information in the image that might be useful to improve the robots navigational capacities and skills. A vision system in contrast to a proximity sensor is able to distinguish between objects such as obstacles, walls or doors based on their visual appearance, for example a door passing behavior that relies on the detection of the door frame and leaf is more robust [6] or color based cues to perform object detection and obstacle avoidance [4].

Imitation learning provides a means to bypass the design of vision based navigation in that a behavior only needs to be demonstrated rather than explicitly designed [2, 7]. A robot learns reactive behaviors either by interacting with hundreds of sensor based rules which requires no internal model or adhering and approximating to the prior knowledge about the environment fed by the experimenter or demonstrator. A vision system for such a behavior learning process has been the focus of many robotic research such as object manipulation [8] and mobile navigation [9, 10, 11]. A variety of applications such as manipulation task learning from demonstration using vision and range sensing is performed by [12] to generalize an abstract behavior whereas [13] relies on depth cues obtained from the monocular images to design obstacle avoidance behaviors by means of genetic programming. The most recent work by [10] performs case-based reasoning on the demonstrated examples and vision cues are extracted to execute a pure reactive learned behavior on a 4-legged robot.

Here, we propose a novel integration of 3D range measurements with 2D image cameras to learn an effective corridor navigation behavior arising from the fusion of demonstrated obstacle avoidance and corridor following examples. In learning from demonstration [14], a control policy as a mapping from perceptions to actions, is learned from examples provided by either a human teacher demonstrating or an existing behavior that relies on a different sensor modality. This utilizes the generated training instances to obtain a behavior with a novel sensor modality that mimics the human control or the demonstrated behavior thereby avoiding the need of programming the robot. It allows both non-robotic and robotic experts to instruct the robot in an intuitive manner. An equivalent of human vision or local perception of a sonar used for demonstration is an omnidirectional camera in vision systems. Nevertheless, the critical challenge in computer vision is to condense the complex raw image into appropriate features that capture the perceptual aspects relevant in the demonstration.

In this contribution, an instance based model [15] is trained on a subset of the recorded perception action pairs. Instance based learners approximate the target function for each distinct query instance thereby allowing it to describe complex target functions with less complex local approximations [16]. This model is different from case-based reasoning

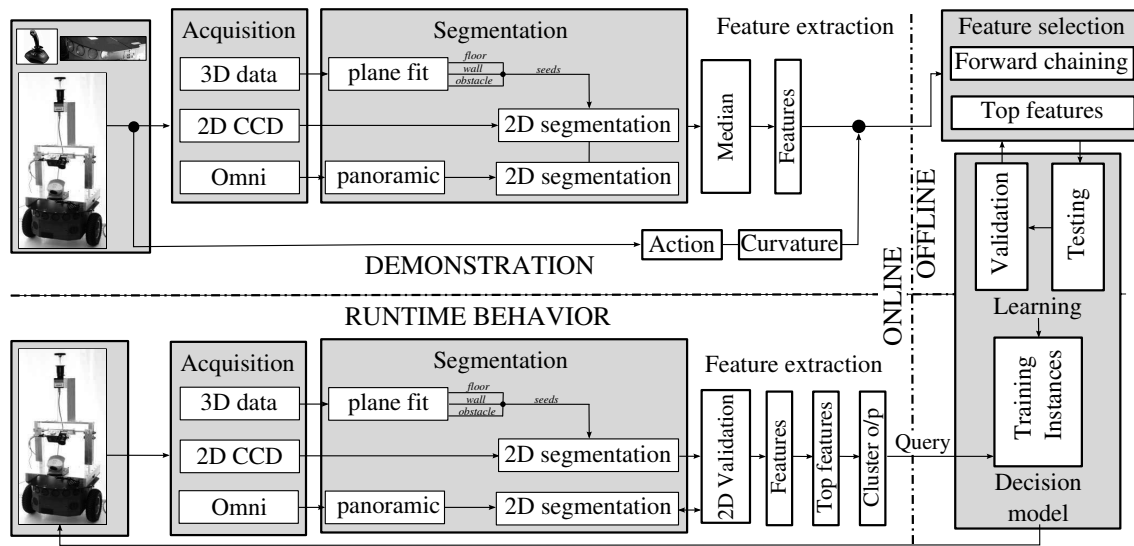


Figure 1: System architecture: demonstration, learning and run-time behavior

[10] in which instances are represented by complex logical descriptions rather than similarity of the image features as in our approach.

The overall architecture of the system is shown in Fig. 1. A 3D-2D camera system acquires images of the robots central front view. A catadioptric omnidirectional camera provides a panoramic view of the robots local environment. Planes are extracted from the 3D scan data by means RANSAC (RANDOM SAMPLE CONSENSUS) and are classified as either floor, walls or obstacles according to their surface normal orientation and size. [17] propose an adaptive texture based floor model for free space segmentation. In our case the appearance based segmentation in the monocular and panoramic images is initialized and validated by means of markers provided by the ground truth 3D segmentation. The demonstration is performed using a robust sonar based wandering behavior during which the 3D-2D perception images are captured together with the performed action. The curvature of the traversed trajectory is recorded from the absolute translational and rotational velocities of the robot thereby decoupling the learned behavior from the robot dynamics [18]. The feature extraction module aggregates a collection of geometric candidate features that capture a diverse range of local features pertaining to specific regions of interest in the image and global features that describe the larger perspective by covering the entire robot local perceptual environment accounting to a grand total of 53 features. This high dimensional collection of local and global features are further tested, validated and reduced by means of forward chaining. The underlying instance based learning schemes are distance weighted constant and linear local regression models. The final set of 13 features provide the input from which the associated action is inferred from the trained model during run time.

The paper is organized as follows. Section 2 describes the utilized 3D and 2D segmentation methods followed by their validation, and Section 3 explain the extraction of the relevant features for learning. Validation of extracted training data and the learning model is introduced in Section 4 with the experimental results and conclusion in Section 5 and 6.

2 Segmentation

In the following we introduce a mixture of experts [19] approach for robust floor segmentation in the robots monocular and panoramic view. The key idea is to use the reliable depth information of the 3D camera in the frontal view, to monitor, select and adapt the appearance based segmentation. The proposed mixture of experts approach facilitates a robust appearance based robotic navigation. The weighted aggregation of multiple visual experts that operate with different cues, is based on the 2D image information itself rather than the 3D ground truth. The starting point of our segmentation and feature extraction scheme is the segmentation of the depth image of the robots front view. From the 3D point cloud planar surfaces that belong to either walls, floor or obstacles are extracted by means of RANSAC which provides the ground truth for the monitoring and adaptation of the 2D segmentation.

2.1 3D segmentation

The underlying assumption that floors in an indoor office environment are planar motivates the use of RANSAC algorithm to fit and estimate planar surfaces from the 3D scan data. Depth calibration of the 3D camera is first carried out in order to remove inherent systematic errors associated with the distance measurement [20]. RANSAC generates multiple candidate planes from hypothetical inliers and evaluates the quality of the hypothesis on the remaining data. Points that fit well to the estimated plane are selected as inliers. RANSAC is applied iteratively, in that the inliers of the best model are removed from the dataset and the next plane is estimated with the remaining points. The orientation of the surface normal and the area of the plane determine the classification of a plane as either floor, wall or obstacle. Pixels that belong to a large horizontal plane are considered as floor, vertical planes as walls and small planes or remaining isolated nonplanar patches as obstacles. Fig. 2 shows one such classification in which the identified plane inliers are visualized. A minimum of 200 inliers and a normal separation threshold of 0.05m between a 3D point and the plane is set as a prerequisite to identify a plane and qualify an inlier.

2.2 2D segmentation

The inherent low resolution intensity image of 64×48 pixels presents itself unusable for appearance based segmentation methods. This limitation is overcome by employing a monocular camera with standard resolution to work in tandem with the 3D camera. The advantage of this system is the facility to fuse depth and high resolution intensity images to enhance and obtain a detailed picture of the environment. This fact is exploited by transferring the identified 3D floor points onto the monocular image plane by homography obtained by the intrinsic camera parameters. Fig. 3(a) shows one instance of the floor points projected onto the 2D monocular image. *Region growing* and *watershed* are standard algorithms that perform well with good initialization [21]. Region growing or flood fill [22] use predefined markers to isolate portions of images to find regions of similar appearance pertaining to a similarity criteria. The difference between the intensities of the input seeds and their neighboring pixel serves as a similarity measure. Watershed algorithm [23] relies on the gradient of the intensity image to form valleys or low points in

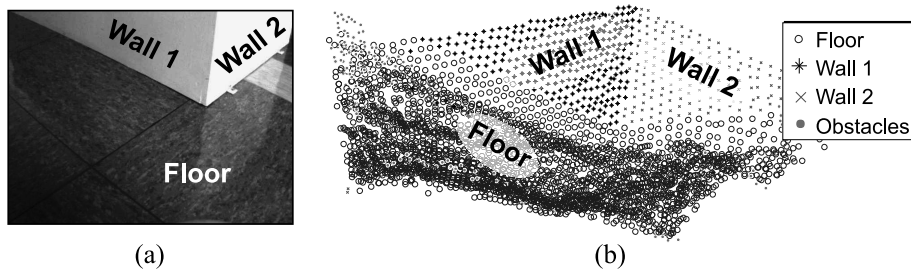


Figure 2: RANSAC plane fit (a) Monocular image of the scenario (b) 3D coordinates of the identified planes

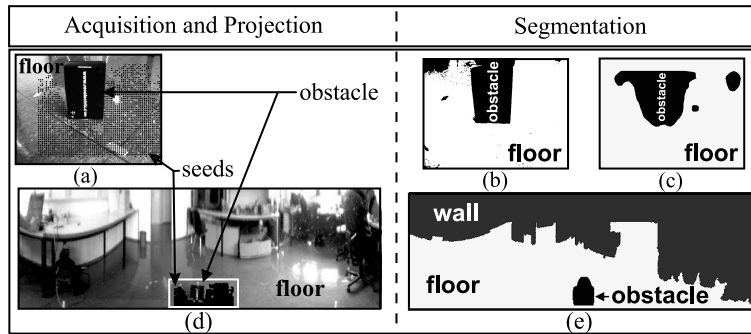


Figure 3: 2D segmentation (a) floor seeds projected onto monocular image, (b) region growing, (c) watershed, (d) floor seeds projected onto panoramic image, (e) panoramic segmentation

homogeneous regions and employs dominant lines or edges in the image to form mountains or high ridges. The valleys are then flooded with seeds as sources thereby merging the low regions in the landscape to form flooded segments. The difference between flood fill and watershed lies in the definition of seeds. Flood fill only takes the set of seeds belonging to a single foreground segment whereas watershed algorithm can be initialized with separate seeds for multiple alternative regions. Fig. 3(b) and (c) show one example of both the segmentations in a typical indoor scene.

The robots perception is enhanced by an omnidirectional view which is converted into a panoramic image to obtain a perspective, undistorted image. The panoramic view enables the robot to perceive lateral obstacles and to infer its pose with respect to the corridor walls. This information is crucial for corridor following as well as passing of obstacles. The panoramic view has a coarse resolution as apparent in Fig. 3(d), in which 3D floor points are projected on into the monocular front view (a) and onto the panoramic image (d). Similar to human vision, the robot exhibits a high resolution in the central vision and a low resolution in the peripheral vision. As will be shown the acquired visual behaviors will rely on features from both central and peripheral vision.

2.3 Mixture of experts

The quality of the marker based segmentation depends on the initial seeds and the texture and illumination of the environment [21]. No segmentation algorithm performs robust across all conditions and situations. Fig. 3(b) and (c) shows an example in which region growing (b) and watershed (c) segment floor and obstacle in a different manner.

Therefore, we propose a mixture of experts approach in which the optimal segmentation algorithm, its parameters and the visual cues are selected and adapted with respect to the visual context. In our case, the quality of segmentation is evaluated based on 3D floor segmentation with RANSAC that provides the ground truth classification in the frontal view. The accuracy of both segmentation schemes is validated with respect to the fraction of false positives and false negatives pixel classifications in the frontal view. False positives correspond to pixels in which an obstacle or wall is incorrectly classified as a floor and similarly false negatives correspond to floor pixels that are actually obstacles or walls. With respect to the task of obstacle avoidance it is apparent that false positive classification are potentially more severe for the robot than a conservative estimate of the robots free space. Considering the significance of both types of errors the overall weighted classification rate (J) is defined by:

$$J = \frac{(2f_p + f_n)}{3},$$

where f_p denotes the ratio of false positives and f_n the ratio of false negatives. Region growing is only seeded with positive examples of floor pixels whereas watershed is initialized with markers of all the available classes namely floor, wall and obstacles. The algorithm with better classification rate on the front view on is elected as the more reliable expert for panoramic segmentation. In order to reduce the overhead computational costs, validation in our experiments are performed for every fifth iteration and the winning algorithm is performed until the next validation.

3 Visual features

This section is concerned with the identification of relevant aggregated visual features extracted from the segmented frontal and peripheral view. The features are supposed to capture those aspects relevant for the demonstrated behavior, such as proximity and direction to objects and orientation and pose within the corridor. Hence, a conglomeration of potential candidate features which capture the larger perceptual space as well as local detailed regions of interests are extracted. These features are then further reduced via forward chaining within an instance based learning framework.

Training examples are generated by demonstrating and guiding the robot through different corridor environments. In our case, the demonstrator is a wandering behavior emerging from the behavior fusion of obstacle avoidance and corridor following based on sonar information that provides the training examples in terms of recorded perception-action pairs. A total of 8876 instances are generated in five separate scenarios, two scenarios with different corridors and three scenario in an open room and foyer environments with isolated obstacles emphasizing obstacle avoidance.

In each scenario examples are generated from recording visual percepts and actions during a wandering behavior executed over a time span of 10 to 15 minutes. In order to increase the diversity of robot poses and perceptions within the environment, random changes in heading directions are executed at regular time intervals. In other words, the wandering behavior is temporarily interrupted, a random heading direction is executed before the wandering behavior resumes control over the robot. Examples are recorded only during the wandering process and are ignored during the disruptive execution of random robot

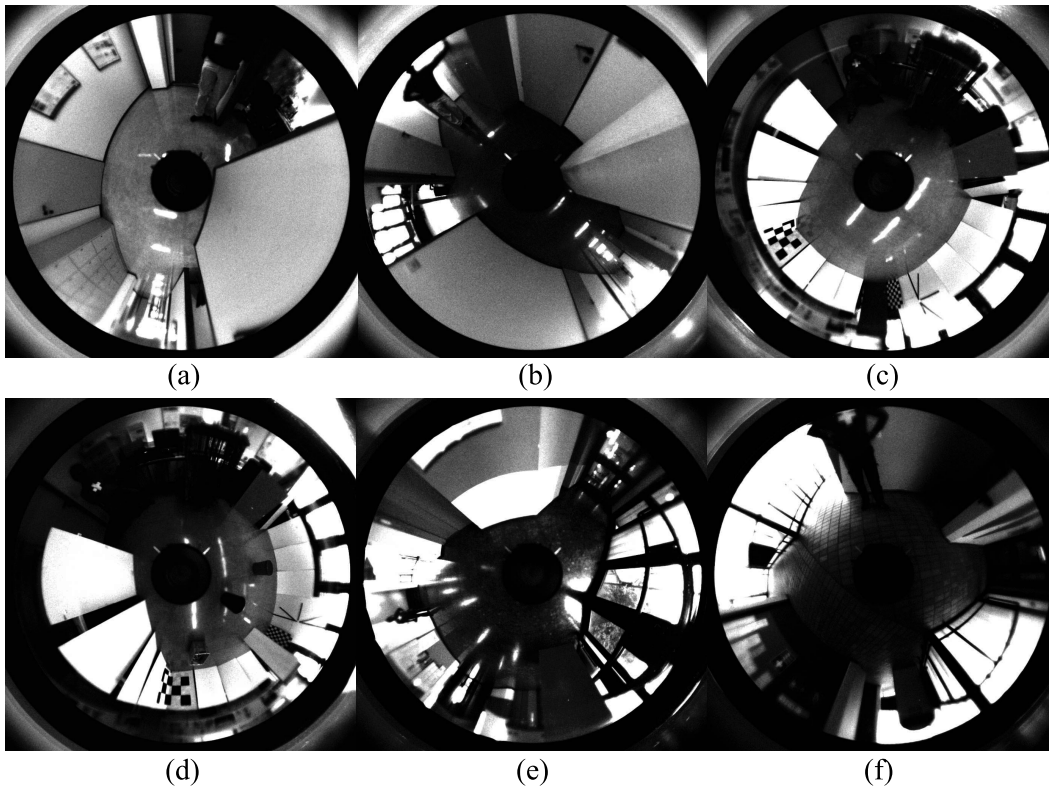


Figure 4: Training scenarios.(a) Corridor 1,(b) Corridor 2,(c) Open room 1,(d) Open room 2,(e) Foyer 1,(f) Foyer 2

heading directions. This systematic reinitialization guarantees a diverse range of robot poses within the corridor. Without these disturbances the training set would be dominated by perceptual states in which the robot is already aligned and centered with respect to the corridor. Both corridors are approximately 2.5 m wide with pillars at regular intervals but are distinguished by the unique floor texture and contrast between the floor, wall and obstacles as shown in Fig. 4.

The recorded monocular and the panoramic views are partitioned into angular segments, in each segment the lowest non-floor pixel indicates the proximity to obstacles or walls thereby mimicking the functionality of a proximity sensor. Fig. 5(a) and (b) shows the partition of one such monocular and panoramic image in which the height of the shaded column indicates the extension of free space along that direction. These proximity features form an initial set of twelve features for the frontal view and additional eight features of the panoramic view. Furthermore the relative difference of floor area in opposite column pairs (1-8, 2-7, 3-6, 4-5) which indicate the lateral alignment and orientation within the corridor are also recorded as shown in Fig. 5(b). Similar difference in monocular image relates to the amount of relative free space available in the robots front view (Fig. 5(a)). Similarly global features that capture the total area of the floor, wall and obstacles in both images, the slope and intercept of the line fitting the two extreme columns and the ratio of floor area between the right and left hemispheres are also accumulated. One important aspect is to ensure continuity of avoiding motions in corners or dead end situations. In order to bias the turning direction the orientation of the column with the most free space to the robots left or right side is identified. In order to capture tempo-spatial aspects, the height and orientation of the free column at the current and previous time in-

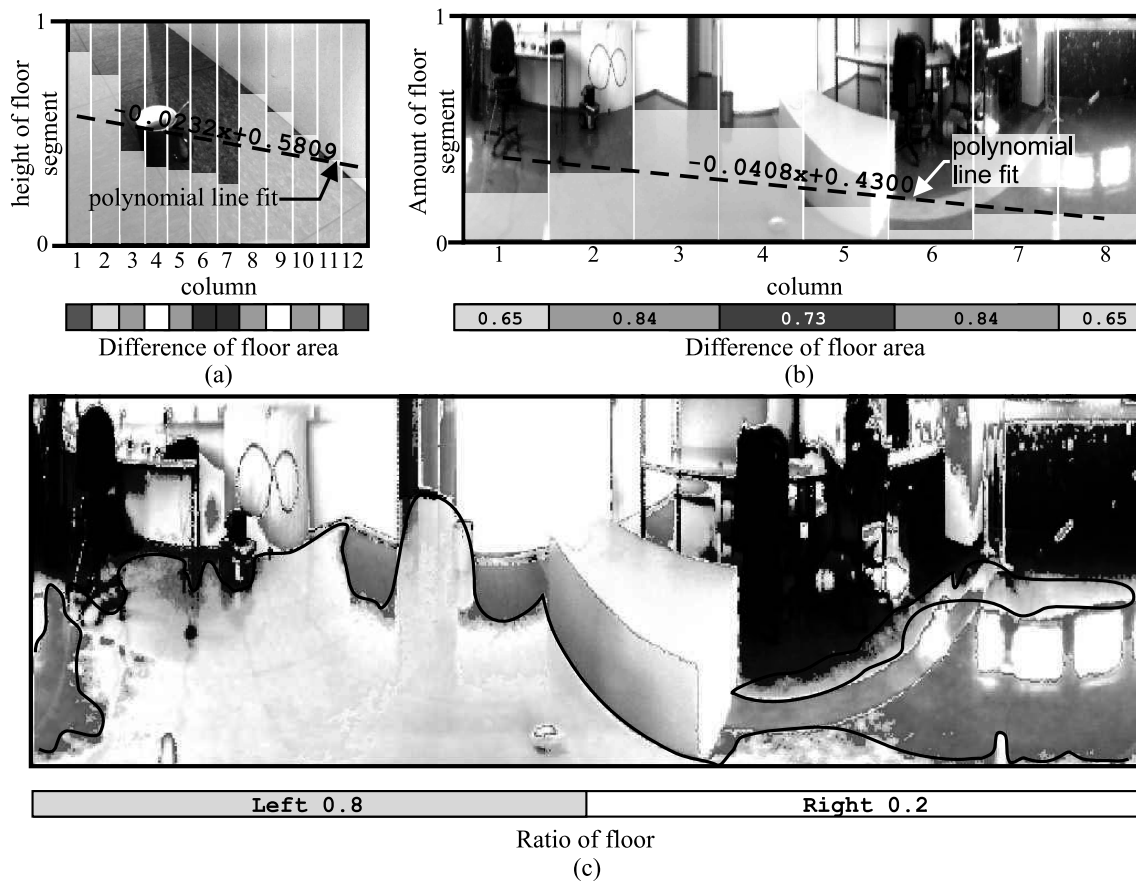


Figure 5: Extracted visual features. (a) Monocular image features. (Height of floor segment in every column, polynomial line fit, ratio of the floor area between column pairs 1-12, 2-11, 3-10, 4-9, 5-8, 6-7), (b) Panoramic image features. (Height and amount of floor segment in every column, polynomial line fit, ratio of the floor area between column pairs 1-8, 2-7, 3-6, 4-5), (c) Panoramic segmentation result and ratio of the floor area between right and left sides

stance are recorded. These four features are only computed in case all the central columns in front of the robot fall below a minimal distance threshold. This feature is relevant in corner and deadend situations to perform and complete an ongoing turn in one direction until again the robot is heading towards free space. To promote continuation of turns a flag that indicates the previous turning direction is also registered. It attains the value -1 and 1 for left and right turns and 0 in the event of no turn. Thereby a total of 53 features are extracted of which 27 features are related to the frontal and 25 features to the panoramic view. Some of these features are illustrated in Fig. 5 and Fig. 6. The demonstrated motion is recorded and encoded in terms of the curvature of the robots trajectory expressed as the ratio between rotational and translational velocity. The representation merely describes the robots path rather than its trajectory and thereby render the learning and recall independent of the robots dynamics. The translational velocity is controlled separately and adjusted to the frame rate of the visual feedback. In the current implementation not optimized for minimal run time the frame rate of the visual processing is about 1.5Hz, such that the robot operates at a slightly lower translational velocity than in the original behavior.

From the set of 53 candidate features, the ones most relevant for the learning problem

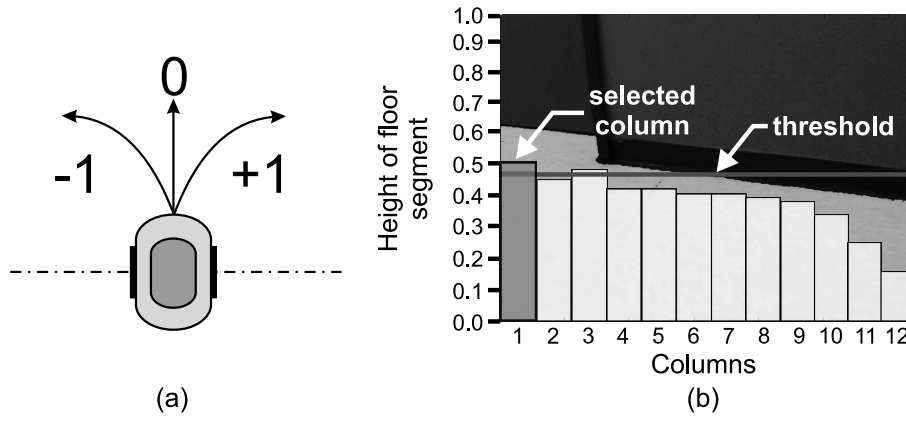


Figure 6: (a) Robot centric feature: Previous turn made, (b) The column with the highest floor segment to the immediate right or left of the robots front view

are systematically determined by feature selection using forward chaining. The instance based learner with local linear or local constant model is employed to determine the training and generalization error of the training data. Instance based learner is a memory based technique that once queried with input features predicts the output by locally interpolating the neighbors of the query considered relevant according to a distance measure [15]. Nearest neighbors are determined with the help of an exhaustive search using either Manhattan or Euclidean distance. With a wrapper approach and forward chaining, relevant features are incrementally included to the input representation, until the test set generalized mean squared error of the trained model no longer decreases with additional features. The selected features and the generalization error exhibit similar trends for both Manhattan and Euclidean distance based neighbors search. Hence, we only present the results based on Manhattan distance metrics. The nearest neighbors of a query point with m input features is computed according to the Manhattan distance

$$D(x_i, x_q) = \frac{\sum_{j=1}^m W_j |x_i(j) - x_q(j)|}{\sum_{j=1}^m W(j)}, \quad (1)$$

where W_j , $x_{i(j)}$ and $x_{q(j)}$ denote the j^{th} components of vectors $W [m, 1]$ weights, i^{th} input example x_i and query point x_q . The scaling factors indicate the relative contribution of a feature to the distance function and is set to one by default. By setting scaling factors inversely proportional to the squared distance between its maximum and minimum feature value, the distribution of feature values in the training set is also considered. Training data generated from different scenarios are validated against each other to identify the feature combination that reduces the generalization error. This ensures the selection of relevant features which best generalize on the unseen inter- and intra- scenario examples. Figures 7 and 8 show the evolution of the mean squared error (mse) between the prediction and the validation output against the number of selected features for both scaled and unscaled models. From the graph it is apparent that the constant model for unscaled distance outperforms the linear instance based model. The reason for the inferior performance of the linear model is due to the relatively sparse density of training examples within a high-dimensional feature space. The prediction y_q for the query point with a constant model is given by,

$$y_q = \frac{\sum_1^k D(x_i, x_q) y_i}{\sum_1^k D(x_i, x_q)},$$

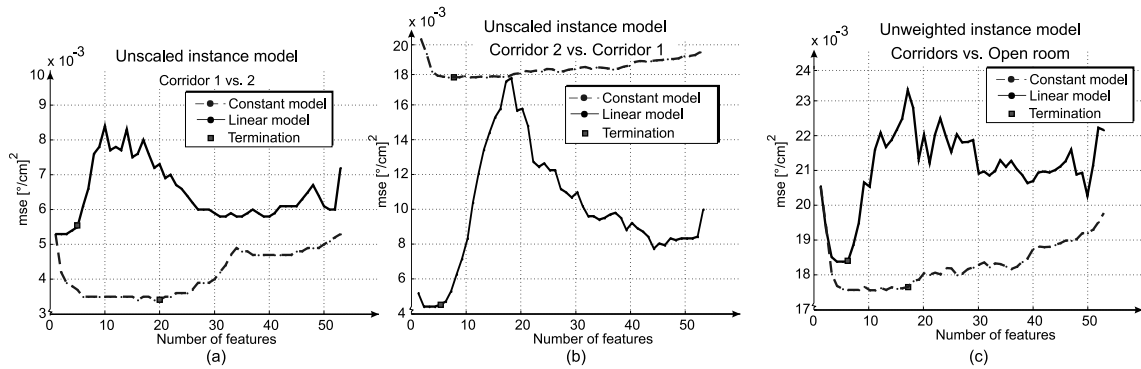


Figure 7: Forward chaining error evolution. (a) Unscaled instance based model. Training: corridor 1, validation: corridor 2, (b) Training: corridor 2, validation: Corridor 1, (c) Training: corridors 1,2, validation: open room. Total number of Instances: 8876. Corridor 1: 3312 instances, corridor 2: 2302 instances, open room: 3326 instances

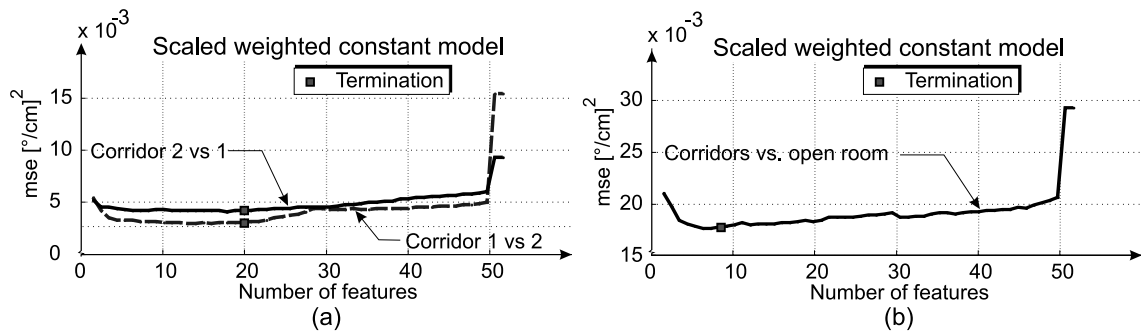


Figure 8: Forward chaining error evolution. (a) scaled constant model. Training and validation between corridor 1,2 and vice versa, (b) Training: corridors 1,2, validation: open room. Total number of Instances: 8876. Corridor 1: 3312 instances, corridor 2: 2302 instances, open room: 3326 instances.

where k is the number of nearest neighbors for the query point x_q , x_i is the i^{th} input example, y_i is the corresponding output and $D(x_i, x_q)$ is the Manhattan distance shown in Eq. 1. With a set of twenty features and only about few thousands of training instances a local linear model is subject to overfitting, in particular as the training instances within a demonstration tend to be correlated. Hence, further feature selection is only performed with respect to the constant instance based model. From the figure it is also apparent that the distance scaled model presents is more robust with respect to inclusion of additional features compared to the unscaled constant model for which the generalization error is less robust. The lower generalization error is attributed to the fact that features with a large data range contribute possess more influence than features with low variation. Hence, the results of the scaled model are used for forward chained feature selection. The inclusion of additional features terminates once the generalization error increases by more than $0.001 \frac{\text{cm}}{\text{cm}}$ or if the number of selected features exceeds twenty. Training on obstacle avoidance examples and validating on corridor following data exhibits substantially larger generalization error and terminates within only 3-4 features. This failure to generalize from one scenario onto another is explicable by the fundamentally different structure or open rooms and corridors. In the former the robot circumnavigates isolated obstacles, thus avoiding proximity to objects, whereas in the later the robot balances proximity by alignment and centering with respect to the corridor. The futile attempt to generalize

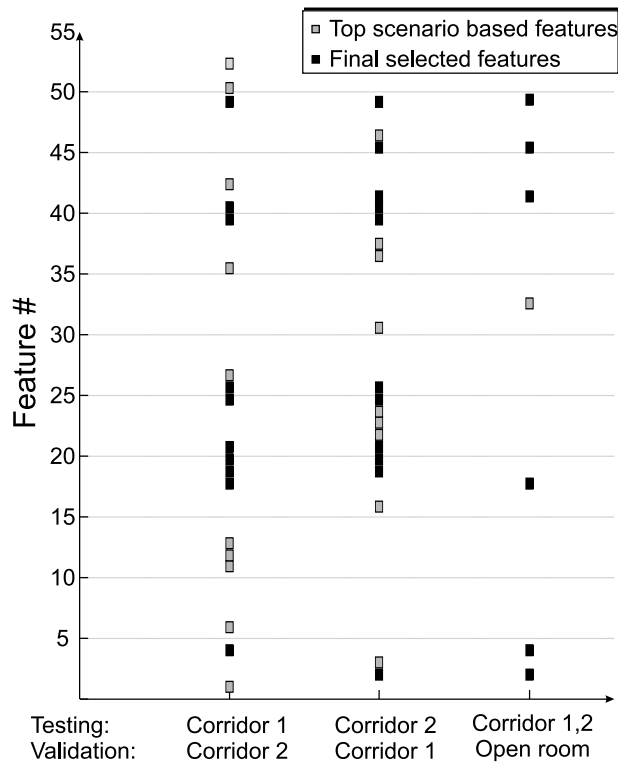


Figure 9: Table of selected features and their corresponding testing scenario

isolated obstacle avoidance to corridor following results in an overly aggressive behavior that merely bounces the robot between left and right wall.

Fig. 9 illustrates the selected features for the three aforementioned scenarios. Those features that are selected at least in two out of three scenarios are included in the final feature set for learning. The black squares in the plot correspond to the thirteen finally selected features listed in Table 1. The aggregated image features (2,4,41,42,49) that provide information about the robots global perceptual space represent almost half of all the features. The remaining features capture local properties such as direct proximity or relative proximity between symmetrically opposite headings. A subset of diverse, but carefully selected features is obviously sufficient establishes to generalize a visual control from the demonstrated behavior even though it relies on completely different sensor modality. The selected feature set represents a balanced mixture of monocular and panoramic image features emphasizing the importance of both coarse and detailed image features. The generalized behavior selected is validated for its robustness in the following section.

4 Behavior learning and validation

We propose instance based learning to generalize from demonstrated examples onto a visual behavior. For learning we employ a local constant model. It constitutes a regression of the nearest neighbors found relevant for the query. The generalization capability of the learned behavior is validated using *k-fold* validation [16] and an inter-scenario testing. The number of nearest neighbors are varied between one and fifteen. The validation of the training set is performed on the unseen subset data. The inter-scenario validation

Table 1: Relevant features selected for training with scaled constant instance based model

Feature #	Category	Feature
2	Monocular	Percentage of wall
4	Monocular	Height of column 1
18	Monocular	Ratio of floor to the left and right
19,20,21	Monocular	Difference of floor area column pair 1-12,2-11,3-10
25,26	Panoramic	Height of column 1,2
40	Panoramic	Amount of floor in column 8
41	Panoramic	Slope of the line
42	Panoramic	Intercept of the line
46	Panoramic	Difference of floor area column pair 2-7
49	Robot centric	Previous turn flag

Table 2: Inter-scenario and inter-behavioral validation

Training scenario	Testing scenario	<i>rmse</i> [°/cm]	Maximum squared error[°/cm]
k^{th} block set	rest $k - 1$ sets	0.0858	0.6028
k-fold C1	C2	0.0695	0.2350
k-fold C2	C1	0.0845	0.2422
C1	C2	0.0736	0.2785
C2	C1	0.0728	0.3183
C1&C2	OA	0.1407	1.07
OA	C1&C2	0.0958	0.5264

tests the ability to generalize from training in one corridor onto performance in the second unseen, geometric similar corridor with albeit different texture. The scenario analysis also investigates as to whether the learner is able to generalize from the two corridor scenarios onto the fundamentally different open room scenario. Since the final features for training are in fact generated using generalization error with forward chaining, the combination of scenario specific image features for the entire data set would further establish the final robustness of the training set.

Table 2 shows the validation results of *k-fold*, inter-scenario and inter-behavioral testing. The generalization root mean squared error (*rmse*) between corridor 1 and 2 (C1 and C2 in Table 2) is robust, due to the similarity of training and test instances. This characteristic robustness of the training data is also seen from the *4-fold* validation between the two corridor data where the k^{th} block of C1 is tested against the entire C2 dataset and vice versa. The validation error between the corridor following and obstacle avoidance (OA in Table 2) and vice versa is substantially larger mainly attributing to the lack of similar examples.

In learning from demonstration the training examples exhibit a substantial amount of ambiguity of the actions as well as the perceptions. This is particular true for mobile robots with complex visual perceptions in unstructured environments. This perceptual

aliasing is first compensated by clustering the nearest neighbors of the output curvature. Regression is then limited to those subsets of neighbors that form the largest cluster in order to achieve robustness. It also helps to resolve problems arising from ambiguous training instances and noise.

This clustering in output space improves the robustness of the acquired behaviors even though it increases the mean regression error from a machine learning point of view. However, from the perspective of robotic behaviors, the compromise action is often inadequate. With the clustering, matching of queries with training instances occurs in two stages. Clustering identifies the general similar scenario, e.g. passing left or right, and regression the most similar instances within the same situation.

5 Experimental results

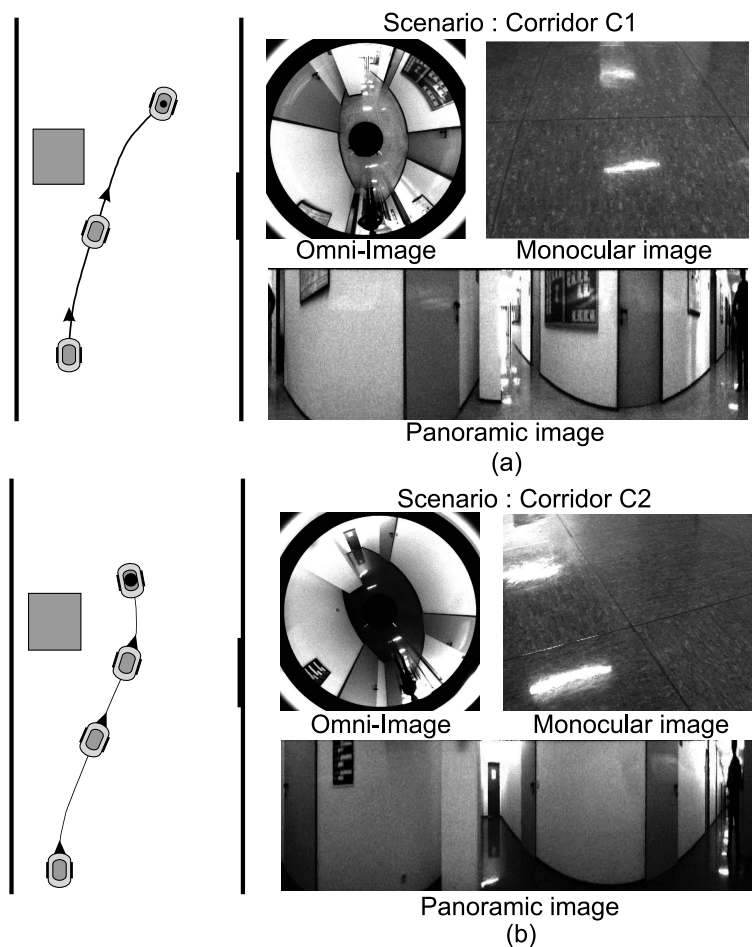


Figure 10: Obstacle avoidance traversed path. (a) Corridor 1, (b) Corridor 2.

A variety of experiments is performed to validate the performance and robustness of acquired behaviors and in particular their ability to generalize to previously unseen scenarios. The experimental platform is a Pioneer 3DX mobile robot equipped with a 3D-2D camera rig and an omnidirectional camera. Fig. 10(a) and (b) show a prototypical experimental scenario. The robot's ability to avoid obstacles within a corridor is tested in corridors C1 and C2 with different floor texture and illumination. In both cases the robot

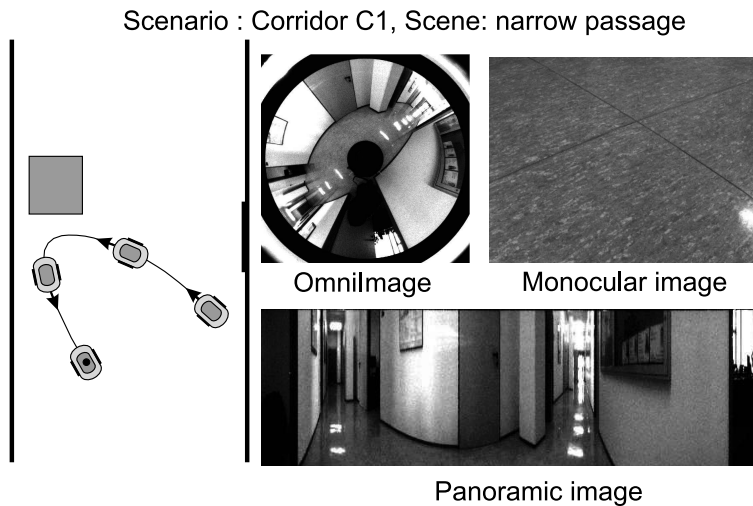


Figure 11: Corner scenario

is positioned in front of a pillar alongside the left wall of the corridor. The depicted trajectories show that the robot successfully evades the obstacle and centers itself in the corridor. The experiment is repeated in the second corridor for the same environmental configuration. The difference in the resulting path is attributed to the different floor texture and contrast between floor, wall and obstacles. Nevertheless the general emerging behavior is similar which confirms the ability to generalize behaviors across different visual appearances of geometrically similar environment. Fig. 11 shows another test scenario in which the robot starts in a confined space with a frontal obstacle adjacent to the wall. This configuration represents a dead end or a narrow passage situation. The traversed trajectory shows that the learned behavior is robust enough to cope with corrupted segmentation that results from the occlusion of floor seed points by the nearby wall and obstacle.

The robustness of the acquired behavior is validated by having the robot wandering in the environment over a prolonged period of about one hour. During that period the robot successfully avoided static obstacles and people that intruded the corridor. The narrow passages and corridor dead end situations are resolved by maintaining the initial turning direction until again the robot is heading towards the free corridor space. From the results, it is seen that the wandering behavior accomplishes a successful supervised learning of autonomous visual robotic behavior emanating from an appropriate fusion of range and appearance based features.

6 Conclusion

This paper presented a novel framework for imitation learning of visual reactive robotic behaviors from examples generated from a sonar based behavior. Perception to action mapping is performed by extracting global and local image features that capture the robots local perceptual space. The instance based model generalizes the relationship between visual appearance and trajectory curvature from the training examples. The method seamlessly transfers established mobile robot behaviors that rely on a different sensor modality onto a vision based behavior. The method is also applicable to learn robotic behaviors from demonstrations in which a human controls the robot by a joystick. The approach

provides a possible solution to the problem of finding a suitable record mapping in the context of vision based learning from demonstration, such that the visual features used for learning capture those aspects relevant for the decisions of the demonstrator. Demonstration examples were generated using a robust sonar based behavior emanating from the behavior fusion of corridor following and obstacle avoidance with random starting positions to cover all diverse robot poses in the corridor. From the accumulated training data, relevant features were extracted by testing for their generalization capability against potentially unseen scenario. The results establish that a proper subset of global and local visual features suffices to mimic the demonstrated behavior. The quality of the learned model is validated with standard statistical methods. Finally the experimental results confirm that a robust behavior that generalizes beyond known environments and scenarios can be learned from a limited set of visual perceptions and associated actions.

References

- [1] DeSouza, G. N.; Kak, A. C.: Vision for Mobile Robot Navigation: A Survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 24 (2002) 2, S. 237–267.
- [2] Bonin-Font, F.; Ortiz, A.; Oliver, G.: Visual Navigation for Mobile Robots: A Survey. *Journal of Intelligent Robotics System* 53 (2008) 3, S. 263–296.
- [3] Gaspar, J.; Winters, N.; Santos-Victor, J.: Vision-based navigation and environmental representations with an omnidirectional camera. *IEEE Transactions on Robotics and Automation* 16 (2000), S. 890–898.
- [4] Lenser, S.; Veloso, M.: Visual sonar: fast obstacle avoidance using monocular vision. In: *Proc. of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2003)*, Bd. 1, S. 886–891. 2003.
- [5] Wolf, J.; Burgard, W.; Burkhardt, H.: Robust Vision-based Localization for Mobile Robots using an Image Retrieval System Based on Invariant Features. In: *Proc. of the IEEE International Conference on Robotics & Automation (ICRA)*. 2002.
- [6] Posada, L.; Nierobisch, T.; Hoffmann, F.; Bertram, T.: Image Signal Processing for Visual Door Passing with an Omnidirectional Camera. In: *VISSAPP (1)*, S. 472–479. 2009.
- [7] Nicolescu, M.; Jenkins, O.; Olenderski, A.: Learning Behavior Fusion Estimation from Demonstration. In: *Robot and Human Interactive Communication, 2006. RO-MAN 2006. The 15th IEEE International Symposium on*, S. 340–345. 2006.
- [8] Ekvall, S.; Kragic, D.: Robot learning from demonstration: a task-level planning approach. *International Journal of Advanced Robotic Systems* (2008).
- [9] Dillmann, R.; Rogalla, O.; Ehrenmann, M.; R., Z.; M., B.: Learning robot behaviour and skills based on human demonstration and advice: the machine learning paradigm. In: *9th International Symposium of Robotics Research (ISRR)*, S. 229–238. 1999.

- [10] Peula, J.; Urdiales, C.; Herrero, I.; SÆnchez-Tato, I.; Sandoval, F.: Pure reactive behavior learning using Case Based Reasoning for a vision based 4-legged robot. *Robotics and Autonomous System* 57 (2009) 6-7, S. 688–699.
- [11] Sofman, B.; Lin, E.; Bagnell, J.; Vandapel, N.; Stentz, A.: Improving Robot Navigation Through Self-Supervised Online Learning. In: *Proceedings of Robotics: Science and Systems*. Philadelphia, USA. 2006.
- [12] Nicolescu, M.; Mataric, M.: Natural Methods for Robot Task Learning: Instructive Demonstrations, Generalization and Practice. In: *In Proceedings of the Second International Joint Conference on Autonomous Agents and Multi-Agent Systems*, S. 241–248. 2003.
- [13] Martin, C.: *The Simulated Evolution of Robot Perception*. Dissertation, Robotics Institute, Carnegie Mellon University, Pittsburgh, PA. 2001.
- [14] Argall, B.; Chernova, S.; Veloso, M.; Browning, B.: A survey of robot learning from demonstration. *Robot. Auton. Syst.* 57 (2009) 5, S. 469–483.
- [15] Birattari, B.; Bersini, M.: Lazy learning for local modeling and control design. *International Journal of Control* 72 (1997), S. 643–658.
- [16] Mitchell, T.: *Machine Learning*. New York: McGraw-Hill. 1997.
- [17] Ulrich, I.; Nourbakhsh, I.: Appearance-Based Obstacle Detection with Monocular Color Vision. In: *Proceedings of the Seventeenth National Conference on Artificial Intelligence and Twelfth Conference on Innovative Applications of Artificial Intelligence*, S. 866–871. AAAI Press / The MIT Press. ISBN 0-262-51112-6. 2000.
- [18] Ward, K.; Zelinsky, A.: Acquiring Mobile Robot Behaviors by Learning Trajectory Velocities. *Journal of Autonomous Robots* 9 (2000) 2, S. 113–133.
- [19] Polikar, R.: Ensemble Based Systems in Decision Making. *IEEE Circuits and Systems Magazine* 6 (2006) 3, S. 21–45.
- [20] Lindner, M.; Kolb, A.: Calibration of the intensity-related distance error of the PMD TOF-Camera. In: *SPIE: Intelligent Robots and Computer Vision XXV*, Bd. 6764, S. 6764–35. 2007.
- [21] Cheng, H.; Jiang, X.; Sun, Y.; Wang, J.: Color image segmentation: advances and prospects. *Journal of Pattern Recognition* 34 (2001) 12, S. 2259–2281.
- [22] Heckbert, P.: *A seed fill algorithm*. San Diego, CA, USA: Academic Press Professional, Inc. ISBN 0-12-286169-5. 1990.
- [23] Meyer, F.: Color image segmentation. In: *Image Processing and its Applications, 1992., International Conference on*, S. 303–306. 1992.

Multiple Geometrical Alignments for the Analysis of Protein Active Sites

Thomas Fober and Eyke Hüllermeier

Department of Mathematics and Computer Science
University of Marburg, Germany
{thomas, eyke}@informatik.uni-marburg.de

Abstract

Geometric objects are often represented approximately in terms of a finite set of points in three-dimensional Euclidean space. In this paper, we extend this representation to what we call labeled point clouds. A labeled point cloud is a finite set of points, where each point is not only associated with a position in three-dimensional space, but also with a discrete class label that represents a specific property. This type of model is especially suitable for modeling biomolecules such as proteins and protein binding sites, where a label may represent an atom type or a physico-chemical property. Proceeding from this representation, we address the question of how to compare two labeled points clouds in terms of similarity. Using fuzzy modeling techniques, we develop a suitable similarity measure as well as an efficient evolutionary algorithm to compute it. Having calculated the optimal superposition it is easy to establish an *alignment* in the sense of a one-to-one correspondence between the basic units of two or more protein structures. From a biological point of view, alignments of this kind are of great interest, as they offer important information about evolution, heredity, and the mutual correspondence between molecular constituents. In this paper, we therefore additionally developed a method for computing pairwise or multiple alignments of protein structures on the basis of labeled point cloud superpositions.

1 Introduction

Geometric objects are often represented in terms of a set of points in three-dimensional Euclidean space. This type of representation is finite and hence approximate (even though the number of points can become very large, as for example in laser range scanning), focusing on the most important characteristics of the object while ignoring less important details. A well-known example of a representation of this kind is the *Molfile* format [10], where molecules are described in terms of the spatial coordinates of all atoms. However, since not only the position but also the type of an atom is of interest, this representation is not a simple point cloud. Likewise, other biomolecular structures, such as proteins and protein binding sites, are not only characterized by their geometry but also by additional features, such as physico-chemical properties. In this paper, we therefore introduce the concept of a *labeled point cloud*. A labeled point cloud is a finite set of points, where each point is not only associated with a position in three-dimensional space, but also with a discrete class label that represents a specific property. Formally, a labeled point cloud P is a set of points $\{p_1, \dots, p_n\}$ with two associated functions: $c : P \rightarrow \mathbb{R}^3$ maps points to coordinates in the Euclidean space, and $\ell : P \rightarrow \mathcal{L}$ assigns a label to each point.

Since theory formation in the biological sciences is largely founded on similarity-based and analogical reasoning principles, the comparison of two (or more) objects with each

other is a fundamental problem in bioinformatics. To compare two point clouds, the authors in [14] make use of a measure based on the *Gromov-Hausdorff distance* of sets. This approach is limited to unlabeled point clouds, however. Another possibility is to transform a labeled point cloud into a (labeled) graph first, capturing, in one way or the other, geometrical information in terms of edges, and to apply graph matching techniques afterward. This strategy was recently proposed in [1], where the use of *graph kernels* as similarity measures [5, 6, 9] has been especially advocated. At first sight, this idea looks appealing, especially since methods for comparing graphs abound in the literature. Nevertheless, it also comes with a number of disadvantages. For example, many techniques for matching and comparing graphs capture aspects of similarity which are reasonable for graphs but not necessarily for geometric objects. Besides, graph matching techniques are typically quite complex from a computational point of view.

Perhaps most importantly, however, a graph representation captures the geometrical information only in an *implicit* way, namely through the presence, absence, and possibly the labeling of edges. Moreover, the transformation is often not even lossless. Matching objects while obeying geometrical constraints can then become troublesome, since the geometrical information is not explicitly available. Instead, it must be reconstructed from the graph representation whenever needed.

As an alternative to an indirect approach of that kind, we therefore propose the method of *labeled point cloud superposition* (LPCS), which operates on labeled point clouds directly. Thus, it preserves as much geometrical information as possible and facilitates the exploitation thereof. Related to the concept of an LPCS, we introduce a similarity measure which makes use of modeling techniques from fuzzy set theory. This measure proceeds from the idea of equivalence (inclusion) of point clouds in a set-theoretic sense, but is tolerant toward exceptions (on the level of label information) and geometric deformations.

Yet, in contrast to methods for *multiple graph alignment* as recently introduced in [17], LPCS does not establish a one-to-one correspondence between the basic units of two or more protein structures. From a biological point of view, alignments of this kind are of great interest, as they offer important information about evolution, heredity, and the mutual correspondence between molecular constituents. Additionally, we therefore develop a method for computing pairwise or multiple alignments of protein structures on the basis of labeled point cloud superpositions.

The remainder of the paper is organized as follows. Subsequent to a brief introduction to protein binding sites and their representation in Section 2, we introduce the concept of LPCS in Section 3. The problem of computing an LPCS is then addressed in Section 4, where an evolution strategy is proposed for this purpose. Section 5 introduces the concept of multiple geometrical alignments. Section 6 is devoted to the experimental validation of the approach, and Section 7 concludes the paper.

2 Modeling Protein Binding Sites

In this paper, our special interest concerns the modeling of protein binding sites. More specifically, our work builds upon CavBase [15], a database for the automated detection, extraction, and storing of protein cavities (hypothetical binding sites) from experimentally

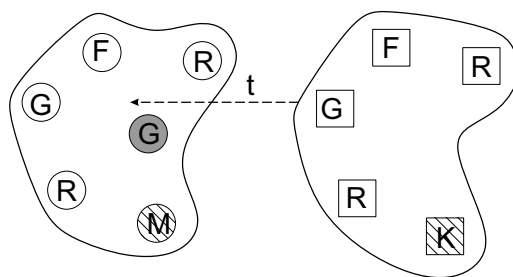


Figure 1: Two point clouds A (left, points as circle) and B (right, points as squares): The intra-point distances are the same in both point clouds, except for the additional gray point in A . Labels are depicted as letters within the circles and boxes, respectively.

determined protein structures (available through the PDB). In CavBase, a set of points is used as a first approximation to describe a binding pocket. The database currently contains 113,718 hypothetical binding sites that have been extracted from 23,780 publicly available protein structures using the LIGSITE algorithm [11].

The geometrical arrangement of the pocket and its physicochemical properties are first represented by predefined *pseudocenters* – spatial points that represent the center of a particular property. The type and the spatial position of the centers depend on the amino acids that border the binding pocket and expose their functional groups. They are derived from the protein structure using a set of predefined rules [15]. As possible types for pseudocenters, hydrogen-bond donor, acceptor, mixed donor/acceptor, hydrophobic aliphatic, metal ion, pi (accounts for the ability to form π - π interactions) and aromatic properties are considered.

Pseudocenters can be regarded as a compressed representation of areas on the cavity surface where certain protein-ligand interactions are experienced. Consequently, a set of pseudocenters is an approximate representation of a spatial distribution of physicochemical properties. Obviously, just like in the case of Molfile, this representation is already in the form of a labeled point cloud: pseudocenters are given with their coordinates and labels, so that no further transformation is needed.

3 Labeled Point Cloud Superposition

Intuitively, two labeled point clouds are similar if they can be spatially superimposed. That is, by fixing the first and “moving” the second one (as a whole, i.e., without changing the internal arrangement of points) in a proper way, an approximate superposition of the two structures is obtained. More specifically, we will say that two point clouds are well superimposed if, for each point in one of the structures, there exists a point in the other cloud which is spatially close and has the same label. As an illustration, the example in Fig. 1 shows two point clouds A and B , for simplicity only in two dimensions. By moving B to the left (or A to the right), a superposition can be found so that, except for the hatched and gray nodes, all points in A spatially coincide with a corresponding point in B having the same label, and vice versa. So, A and B can be considered as being similar, at least to some extent.

More formally, let

$$A = \{(x_1, \ell(x_1)), \dots, (x_m, \ell(x_m))\}$$

be a point cloud consisting of m points $x_i = (x_{i1}, x_{i2}, x_{i3}) \in \mathbb{R}^3$ with associated label $\ell(x_i) \in \mathcal{L}$, where \mathcal{L} is a discrete set of labels (in the context of modeling protein binding sites, as discussed in the previous section, \mathcal{L} is given by the seven types of pseudocenters). Moreover, let

$$B = \{(y_1, \ell(y_1)), \dots, (y_n, \ell(y_n))\}$$

be a second point cloud to be compared with A . In the following, we define a function $\text{SIM}(\cdot, \cdot)$ that returns a degree of similarity between two such structures A and B .

Roughly speaking, we consider similarity as a generalized (fuzzy) equivalence, which we in turn reduce to two inclusion relations, namely the inclusion of A in B and, vice versa, of B in A . Thus, we are first of all interested in whether each point $y \in B$ is also present in A (and each point $x \in A$ also present in B). For a fixed $y \in B$, we define the membership degree of this point in A by

$$\mu_A(y) = \exp(-\gamma \cdot d(y, A)) \quad , \quad (1)$$

where

$$d(y, A) = \min_{\substack{x \in A \\ \ell(x) = \ell(y)}} \|y - x\|_1$$

is the distance between a point $y \in B$ and the closest point $x \in A$ having the same label ($d(y, A) = \infty$ and hence $\mu_A(y) = 0$ if no such point exists); for $x \in A$, $\mu_B(x)$ and $d(x, B)$ are defined analogously.

In its proper sense, the inclusion of a set B in a set A means that *each* point $y \in B$ is also contained in A or, stated differently, if a point y is in B , then it is also present in A . If membership is a matter of degree, i.e., if A and B are fuzzy sets, this condition is often formalized in terms of a fuzzy implication [16]:

$$\min_{y \in B} (\mu_B(y) \rightarrow \mu_A(y)) \quad .$$

Here, the minimum operator plays the role of a generalization of the universal quantifier. In our case, $\mu_B(y) \equiv 1$, so that the above expression can be simplified as follows:

$$\text{inc}(B, A) = \min_{y \in B} \mu_A(y) \quad . \quad (2)$$

However, a universal quantification (modeled by the min operator) is too strict in our biological context, where data is typically inexact and noisy. To relax this definition of fuzzy inclusion, we replace the minimum by a fuzzy quantifier Q , which is specified in the form of a non-decreasing $[0, 1] \rightarrow [0, 1]$ mapping [19, 8]. This leads to

$$\text{inc}(B, A) = \min_{i=1 \dots |B|} \max\{Q(i/|B|), m_i\} \quad ,$$

where m_i is the i -th largest membership degree in the fuzzy set $\{\mu_A(y) \mid y \in B\}$. (Note that we recover (2) for Q defined by $Q(1) = 1$ and $Q(t) = 0$ for $0 \leq t < 1$.) Here, we simply take Q as the identical mapping $t \mapsto t$. Roughly speaking, $\text{inc}(B, A)$ thus defined can be interpreted as the generalized truth degree of the proposition that A is *almost* contained in B . The degree of inclusion of A in B , $\text{inc}(A, B)$, is defined analogously.

As mentioned above, the idea of our approach is to define the similarity between two labeled point clouds in terms of the best superposition of these two clouds. Therefore, let

$\text{TF}(\cdot, t)$ be a function that moves a point cloud via rotation and translation, as specified by the six-dimensional vector $t = (\theta_1, \theta_2, \theta_3, \delta_1, \delta_2, \delta_3) \in [0, 2\pi]^3 \times \mathbb{R}^3$. Thus,

$$B^* = \text{TF}(B, t) = \{(y_1^*, \ell(y_1^*)), \dots, (y_n^*, \ell(y_n^*))\}$$

is the point cloud obtained by translating the point cloud B by $\delta = (\delta_1, \delta_2, \delta_3)$ (which means adding δ to each point $y \in B$) and rotating the result thus obtained by the angles θ_1 , θ_2 , and θ_3 . Note that this operation leaves the label information unchanged (i.e., $\ell(y_i) = \ell(y_i^*)$). The position-invariant degree of inclusion of B in A is then given by

$$\text{INC}(B, A) = \max_{t \in [0, 2\pi]^3 \times \mathbb{R}^3} \text{inc}(\text{TF}(B, t), A) , \quad (3)$$

and $\text{INC}(A, B)$ is defined analogously.

Based on these degrees, the similarity between A and B , in the sense of a generalized equivalence, can be defined as

$$\text{SIM}(A, B) = \min\{\text{INC}(A, B), \text{INC}(B, A)\} . \quad (4)$$

It is worth mentioning, however, that (4) is not always appropriate, especially if A and B greatly differ in size. In some applications, it makes sense to have a high similarity degree even if A is only a substructure of B , for example if A is a subpocket of B containing the most important catalytic residues (while the rest of the binding site B is functionally less important). Obviously, this is not guaranteed by (4). An interesting generalization, therefore, is to let

$$\begin{aligned} \text{SIM}(A, B) = & \alpha \cdot \min\{\text{INC}(A, B), \text{INC}(B, A)\} \\ & + (1 - \alpha) \cdot \max\{\text{INC}(A, B), \text{INC}(B, A)\} . \end{aligned} \quad (5)$$

Formally, this similarity measure can be motivated from a fuzzy logical point of view as follows. Considering the min (max) operator as a generalized conjunction (disjunction), the first (second) combination of the two inclusion degrees is the truth degree of the proposition that A is contained in B AND (OR) B is contained in A . A conjunctive combination of the two degrees of inclusion is obviously more demanding than a disjunctive one, as the former requires equality between A and B while the latter only requires inclusion of A in B or B in A . The measure (5), which formally corresponds to an OWA (ordered weighted average) combination of the two degrees of inclusion [18], achieves a trade-off between these two extreme aggregation modes, which is controlled by the parameter $\alpha \in [0, 1]$: The closer α is to 0, the closer the aggregation is to the maximum, i.e., the less demanding it becomes. The optimal α is application-specific and depends on the purpose of the similarity measure.

4 Solving the LPCS Problem

The computation of the similarity (5) involves the solution of a real-valued optimization problem, namely the problem of finding an optimal vector t in (3) and, thus, an optimal point cloud superposition. The objective function to be maximized here is highly non-linear and multimodal. As an illustration, Fig. 2 shows the objective function obtained for the superposition of a randomly generated two-dimensional point cloud A (in which

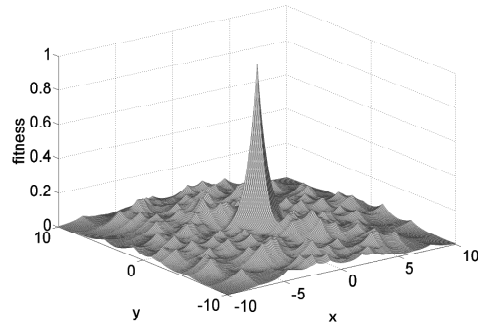


Figure 2: Example of an LPCS objective function.

all points have the same label) with itself. This function maps each two-dimensional translation vector $t = (x, y)$ to the corresponding similarity degree between $TF(A)$ and A (where we used $\alpha = 0.5$ in (5) and did not consider rotation). As can be seen, there is a sharp peak at $t = (0, 0)$, which corresponds to the optimal superposition. Surrounding this solution, however, there are also many local optima.

The problem of local optima also becomes clear from the small example in Fig. 1. Moving the point cloud A from left to right, into the direction of B , has the following effect: First, a good superposition of two sub-clouds will be found, namely the right part of cloud A and the left part of cloud B . This results in a local maximum. Moving A further to the right leads to a larger local maximum (sub-clouds are growing), until the global maximum will eventually be reached.

4.1 Evolution Strategies

To solve the LPCS problem, we resort to *evolution strategies* (ES), a population-based, stochastic optimization method inspired by biological evolution and specifically developed for real-valued optimization problems [3]. An evolution strategy is based on a population, a set of μ (sub-optimal) candidate solutions that are initially spread randomly over the search space. In each generation, new solutions are generated by applying the genetic operators *recombination* and *mutation*. Recombination randomly selects ρ individuals from the current population and combines them to a new solution. Mutation takes this solution and shifts it randomly in the search space. An ES produces $\lambda = \lceil \mu \cdot \nu \rceil$ offsprings per iteration, so that this procedure has to be repeated λ times. A selection operator implements the “survival of the fittest” principle by picking the best individuals for the new population. There are two kinds of selection: The *plus*-selection chooses the best μ individuals among the offsprings plus the parents, while the *comma*-selection ignores the parent generation (this requires $\nu > 1$). A main advantage of the ES is its self-adaptation mechanism that controls the step sizes used in the mutation operator. One property of this mechanism (the advantage during optimization is obvious) is that step sizes decrease dramatically if the optimization reached a maximum. This property can be used as a qualitative termination criterion (stop when the largest step size falls below a given threshold).

Population-based optimization methods are especially advantageous for highly multi-modal problems. Using a large population leads to an increased probability to generate

a candidate solution in a region where the direction of descent points to the global maximum. Choosing the membership function (1) as a strictly monotone decreasing function which converges to zero ensures to have this direction in each point $t \in [0, 2\pi]^3 \times \mathbb{R}^3$ and thus greatly simplifies the maximization problem. However, our experiments indicated that the solution we found was most often only a local maximum. Therefore, we propose to use *fast restarts* of the ES. This means that the ES is started n times using comma-selection and weak termination criteria to achieve a large and quick but inexact exploration of our search space. We thus obtain n results in total. In a last step, we use the ES with plus-selection and strong termination criterion. Additionally, we include the best solution so far in the start population. The last run of the ES usually yields a globally optimal degree of similarity.

4.2 Complexity

Even though evolution strategies are generally known to be quite efficient solvers, the concrete complexity does of course depend on the application at hand. The application-specific part is the fitness function, i.e., the objective function to be optimized. This function has to be evaluated frequently and, therefore, is an important factor for the runtime. In our case, this function is given by the similarity measure (5), and its evaluation is strongly dominated by the nearest neighbor search which has to be conducted for each single point in both structures (recall that, according to (1), membership degrees are determined by the distance to closest points with the same label).

There exist a lot of data structures for supporting nearest neighbor search; see e.g. [7]. The most efficient among them need time $\mathcal{O}(n \log^2 n)$ for construction and $\mathcal{O}(\log^3 n)$ for answering a query. Unfortunately, we are not aware of an approach that allows for updating a data structure in an efficient and dynamic way. This would be desirable for our problem, in which the point clouds permanently change (the point cloud associated with an individual changes in each iteration). Instead, conventional approaches necessitate a construction from scratch in every iteration.

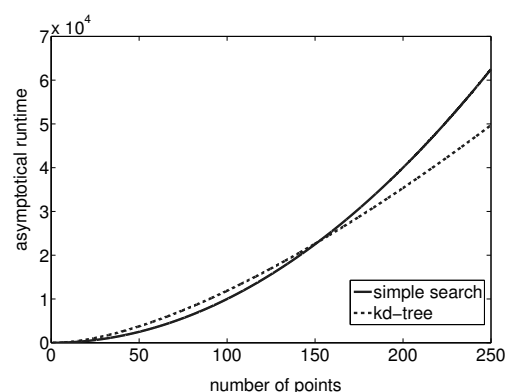


Figure 3: Runtime of a simple procedure and a more complex data structure as a function of the number of points.

Fig. 3 compares the runtimes, as a function of the number of points, for two approaches: (1) The use of a kd-tree data structure, which is reconstructed in each iteration and then used for query processing. (2) The use of a simple linear data structure, in which the

points are stored in a fixed order. It needs linear instead of logarithmic time to answer a query but, on the other hand, does not cause additional costs for reconstruction. As can be seen, the use of a more complex approach pays off only for sufficiently large point clouds: The kd-tree reaches a break-even point at approximately 150 points.

In our application, we are mainly concerned with protein binding sites, which are characterized by around 180 points on average (even though much larger structures do of course exist). The use of a complex data structure did therefore not pay off. Nevertheless, we increased efficiency by hashing the points x_i of a point cloud, using the label $\ell(x_i) \in \mathcal{L}$ as a key. Since nearest neighbors are only searched among points having the same label, this obviously reduces runtime by a factor of approximately $|\mathcal{L}|$.

5 Multiple Geometrical Alignment

When comparing homologs from different species in protein cavity space, one has to deal with the same mutations that are also given in sequence space. Corresponding mutations, in conjunction with conformational variability, strongly affect the spatial structure of a binding site as well as its physicochemical properties and, therefore, its point cloud descriptor. For example, a pseudocenter can be deleted or introduced due to a mutation in sequence space. Likewise, if a mutation replaces a certain functional group by another type of group at the same position, the physicochemical property of a pseudocenter can change. Finally, the distance between two pseudocenters can change due to conformational differences.

Due to the above reasons, one cannot expect that point clouds of two related binding pockets match exactly. When looking for an alignment of two structures in the form of a one-to-one correspondence between pseudocenters, it is therefore necessary to allow for mismatches as well as pseudocenters for which no matching partner is defined. This situation is quite similar to sequence alignment, where mismatches between symbols and the insertion of blanks (to compensate for non-existing matching partners) is also allowed.

In this paper, we derive alignments from labeled point cloud superpositions and, therefore, refer to the latter as *geometric alignments*.

Definition 1 (Multiple Geometrical Alignment) *Let \mathcal{P} be a set of m point clouds $P_i = \{p_1^i, \dots, p_{n_i}^i\}$, $i = 1, \dots, m$. A multiple geometrical alignment of these point clouds is a subset $\mathcal{A} \subseteq (P_1 \cup \{\perp\}) \times \dots \times (P_m \cup \{\perp\})$ with the following properties:*

1. *for all $i = 1 \dots m$ and for each $p \in P_i$ there exists exactly one $a = (a_1 \dots a_m) \in \mathcal{A}$ such that $p = a_i$;*
2. *for each $a = (a_1 \dots a_m) \in \mathcal{A}$ there exists at least one $1 \leq i \leq n$ such that $a_i \neq \perp$.*

Here, the symbol \perp denotes a “dummy point” which is needed to compensate for non-existing matching partners.

Each tuple in the alignment represents a mutual assignment of m points, one from each point cloud P_i (possibly a dummy). Thus, the second property in the above definition

requires that each tuple of the alignment contains at least one non-dummy point, and the first property means that each point of each point cloud occurs exactly once in the alignment. While these properties can be satisfied by a large number of alignments, we are of course looking for an alignment in which mutually assigned points have the same label and nearby spatial positions.

5.1 Construction of pairwise alignments

To construct a pairwise alignment of two point clouds P_1 and P_2 , we reduce the alignment problem to a problem of optimal assignment. To this end, we need a square matrix $M = (m_{i,j})$, where $m_{i,j} \in \mathbb{R}$ defines the costs for assigning point $p_i \in P_1$ to point $p_j \in P_2$. According to definition 1, the maximal length of a pairwise alignment is $n = n_1 + n_2 = |P_1| + |P_2|$. Therefore, to consider all possible alignments, the matrix M has size $n \times n$.

The entries $m_{i,j}$ are derived from the optimal superposition of point clouds P_1 and P_2 as produced by our LPCS method. Since this approach calculates in sum two independent t -vectors (one for each INC function) we had to modify this approach slightly. Instead of using eq. (4) we define the similarity as

$$\text{SIM}_{3\text{DA}}(A, B) = \max_{t \in [0, 2\pi]^3 \times \mathbb{R}^3} \frac{1}{2} \text{inc}(\text{TF}(B, t), A) + \frac{1}{2} \text{inc}(A, \text{TF}(B, t)) \quad (6)$$

and search now for exactly one t -vector.

Given such an optimal spatial superposition, it makes sense to define $m_{i,j}$ by the distance between point $p_i \in P_1$ and $p_j \in P_2$ in the superimposed point clouds. To account for point-to-dummy mappings, the distance between a point and a dummy is specified by a parameter k . Finally, dummy-dummy assignments are scored by zero, so that these mappings will not influence the construction of the alignment. As an illustration, Table 1 shows a matrix M for two point clouds $P_1 = \{a, b, c, d\}$ and $P_2 = \{a', b', c'\}$.

Table 1: Matrix representation of the optimal assignment problem.

	a'	b'	c'	\perp	\perp	\perp	\perp
a	$d(a, a')$	$d(a, b')$	$d(a, c')$	k	k	k	k
b	$d(b, a')$	$d(b, b')$	$d(b, c')$	k	k	k	k
c	$d(c, a')$	$d(c, b')$	$d(c, c')$	k	k	k	k
d	$d(d, a')$	$d(d, b')$	$d(d, c')$	k	k	k	k
\perp	k	k	k	0	0	0	0
\perp	k	k	k	0	0	0	0
\perp	k	k	k	0	0	0	0

Formally, an assignment (weighted bipartite matching) problem is specified by a graph $G = (V, E)$ with $V = V_1 \cup V_2$ ($V_1 \cap V_2 = \emptyset$) and $E = \{\{u, v\} \mid u \in V_1, v \in V_2\}$. The problem is to find a subset of edges $M \subseteq E$ such that $e \cap e' = \emptyset$ for all $e, e' \in M$ (i.e., one point has exactly one mapping partner),

$$\bigcup_{(v_1, v_2) \in M} \{v_1\} = V_1, \quad \bigcup_{(v_1, v_2) \in M} \{v_2\} = V_2,$$

and

$$\sum_{e \in M} c(e) \rightarrow \min,$$

where $c(e)$ is the cost associated with edge e . In our case, the sets V_1 and V_2 represent, respectively, the points in point cloud P_1 with additional $|P_2|$ dummy points and the points in cloud P_2 with additional $|P_1|$ dummy points. Moreover, the costs $c(e)$ are given by the corresponding matrix entries $m_{i,j}$. See Figure 4 for an illustration.

To solve the weighted bipartite matching problem, we use the Hungarian algorithm [13] that needs time $\mathcal{O}(n^3)$. Once a cost-minimal assignment has been found, the geometric alignment is defined by the corresponding node-to-node and node-to-dummy assignments, while dummy-to-dummy assignments are ignored.

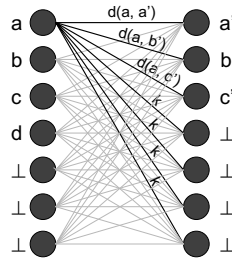


Figure 4: Illustration of the weighted bipartite graph matching problem.

5.2 Construction of Multiple Alignments

Pairwise alignments can be used, for example, to derive a measure of similarity between two objects. From a biological point of view, however, it is even more interesting to look for a *multiple* alignment, that is, the simultaneous alignment of a set of $m > 2$ structures. Alignments of this type are of interest, for example, to discover conserved patterns in a family of evolutionary related proteins.

To derive a multiple geometrical alignment (3DA) of m point clouds, we resort to the star alignment approach [17]: One of the point clouds, say, P_1 , is selected and aligned in a pairwise way with all other clouds P_i , $i = 2, \dots, m$. The pairwise alignments are then “merged” by using P_1 as a pivot structure. Thus, if $p_{ij} \in P_i$ denotes the point (possibly a dummy) aligned with $p_j \in P_1$ in the alignment of P_1 and P_i , then a single assignment in the multiple alignment is of the form

$$(p_j, p_{2j}, p_{3j}, \dots, p_{mj}).$$

Since the quality of a multiple alignment is strongly influenced by the choice of the pivot structure, we try each point cloud as a pivot and adopt the best result. Thus, $m(m-1)/2$ pairwise alignments have to be computed in total.

5.3 Conserved Patterns

As already mentioned each $a \in \mathcal{A}$ corresponds to a vector of mutually assigned points from the point clouds P_1, \dots, P_m . Note that, by matching points, a mutual assignment

of distances is determined in an implicit way. Once a 3DA has been established, it can be used to derive approximately conserved patterns. This can be done in different ways, we propose to use fuzzy consensus graphs [17] originally introduced for *Multiple Graph Alignments (MGA)*. In a first step we generate a *fuzzy consensus graph* $\tilde{G} = (V, E)$. V contains a node for each tuple $a \in \mathcal{A}$ and all pairs of nodes are connected by an edge $e \in E = V \times V$. Each node is labeled with the distribution of the mutually assigned points. Additionally, a degree of conservation $cons(v)$ is calculated, which is defined by the relative number of point clouds in which this point is present. The edges of the consensus graph are defined accordingly; see [17] for details. For given thresholds $\omega, \xi \in (0, 1]$, a conserved pattern can then be defined in terms of the subgraph of G consisting of all nodes v with $cons(v) \geq \omega$ and $maj(v) \geq \xi$, where $maj(v)$ is the relative frequency of the most frequent label in a .

6 Experimental Results

In our experimental study, we perform two types of experiments. Both have in common that we compare the introduced geometrical approach with graph-based approaches. In the first study we will only consider the similarity scores and use them for classification. In the second study we will consider alignments and again compare the geometrical and a graph-based approach [17].

6.1 Data

For the experimental study different data sets are needed. The first type of experiment require a data set consisting of at least two classes so that a classification can be performed. For the second type of experiment we need a data set that consists of many structures that share a common fragment for that we can search using the multiple alignment approaches.

6.1.1 NADH/ATP

One important problem in pharmaceutical chemistry is the identification of protein binding sites that bind a certain ligand. We selected two classes of binding sites that bind, respectively, to NADH or ATP. This gives rise to a binary classification problem: Given a protein binding site, predict whether it binds NADH or ATP.

More concretely, we compiled a set of 355 protein binding pockets representing two classes of proteins that share, respectively, ATP and NADH as a cofactor. To this end, we used CavBase to retrieve all known ATP and NADH binding pockets that were co-crystallized with the respective ligand. Subsequently, we reduced the set to one cavity per protein, thus representing the enzymes by a single binding pocket. As protein ligands adopt different conformations due to their structural flexibility, it is likely that the ligands in our data set are bound in completely different ways, hence the corresponding binding pocket does not necessarily share much structural similarity. We thus had to ensure the selection of binding pockets with ligands bound in similar conformation. To achieve this, we used the Kabsch algorithm [12] to calculate the root mean square deviation (RMSD)

between pairs of ligand structures. Subsequently, we combined all proteins whose ligands yielded a RMSD value below a threshold of 0.2, thereby ensuring a certain degree of similarity. This value was chosen as a trade-off between data set size and similarity. Eventually, we thus obtained a two-class data set comprising 214 NADH-binding proteins and 141 ATP-binding proteins.

6.1.2 Benzamidine

For a first proof-of-concept of the 3DA approach, we analyzed a data set consisting of 87 compounds that belong to a series of selective thrombin inhibitors and were taken from a 3D-QSAR study [4]. The data set is suitable for conducting experiments in a systematic way, as it is quite homogeneous and relatively small (the descriptors contain 47 – 100 points, where each point corresponds to an atom). Moreover, as the 87 compounds all share a common core fragment (which is distributed over two different regions with a variety of substituents), the data set contains a clear and unambiguous target pattern.

6.1.3 Thermolysin

Additionally, we used a data set consisting of 74 structures derived from the Cavbase database. Each structure represents a protein cavity belonging to the protein family of thermolysin, bacterial proteases frequently used in structural protein analysis and annotated with the E.C. number 3.4.24.27 in the ENZYME database. The data set is well-suited for our purpose, as all cavities belong to the same enzyme family and, therefore, evolutionary related, highly conserved substructures ought to be present. On the other hand, with cavities (hypothetical binding pockets) ranging from about 30 to 90 pseudo-centers and not all of them being real binding pockets, the data set is also diverse enough to present a real challenge for matching techniques.

6.2 Classification

In our experiments, first we compared our novel method (LPCS) with existing graph-based approaches, namely the random walk (RW) kernel [9], the shortest path (SP) kernel [6], and the method of multiple graph alignment (MGA) recently introduced in [17]. Given two labeled points clouds as input, all these methods produce a degree of similarity as an output. Yet, for the graph-based approaches, it is of course necessary to transform a point cloud into a graph representation in a preprocessing step. This was done as proposed in [17]:

1. each point is transformed into a node with corresponding node label
2. for each pair of nodes:
 - (a) the Euclidean distance between both nodes is calculated
 - (b) if the distance is below a certain threshold (here 11 \AA to ensure connected graphs), an edge with weight equal to this distance is added

Our ES was restarted $n = 5$ times. The parameterization was optimized with the *sequential parameter optimization toolbox* [2] and was chosen as follows:

- inexact ES: $\mu = 30, \nu = 4, \rho = 2$, comma-selection, termination criteria: largest step size < 0.05 , discrete recombination for strategy- and object-component.
- exact ES: $\mu = 30, \nu = 4, \rho = 6$, plus-selection, termination criteria: largest step size < 0.00001 , intermediate recombination for object and discrete recombination for strategy-component.

A comprehensive explanation of the different ES parameters and operators can be found in [3].

For both variants we initialized the object-component in $[-150, 150]^3$ for translation and $[0, 2\pi]^3$ for rotation: The step sizes were initialized in $[5, 15]^3$ and $[1, \pi]^3$, respectively. The SP-kernel is parameter-free, the RW-kernel expects a parameter λ that is set to the largest degree of a node in the data set to ensure a geometric series during calculation, which results in a simpler evaluation [5]. Since the geometric information of real-world data is noisy, we also need a tolerance parameter ϵ to decide whether two edges have equal length (difference $\leq \epsilon$) or not; in our experiments, we used $\epsilon = 0.2$. For MGA, we chose the parameterization proposed in [17].

The assessment of a similarity measure for biomolecular structures, such as protein binding sites, is clearly a non-trivial problem. In particular, since the concept of similarity by itself is rather vague and subjective, it is difficult to evaluate corresponding measures in an objective way. To circumvent this problem, we propose to evaluate similarity measures in an indirect way, namely by means of their performance in the context of nearest neighbor (NN) classification. The underlying idea is that, the better a similarity measure is, the better should be the predictive performance of an NN classifier using this measure for determining similar cases.

6.2.1 Results

The results of a leave-one-out cross validation, using the simple 1-NN classifier for prediction, are summarized in Table 2. As can be seen, the kernel-based methods (SP and RW) perform very poorly and are hardly better than random guessing. In terms of accuracy, MGA is much better, though still significantly worse than LPCS. In fact, LPCS performs clearly best on this problem.

Table 2: Accuracy and runtimes (in seconds with standard deviation, referring to a single comparison) of LPCS ($\alpha = 0.5$, with restarts like described above), MGA, RW, and SP on the NADH/APT data set.

Method	Accuracy	Runtime
MGA	0.7662	121.74 \pm 418.02
SP	0.6056	9.75 \pm 97.77
RW	0.5972	65.51 \pm 89.07
LPCS	0.9352	20.04 \pm 24.65

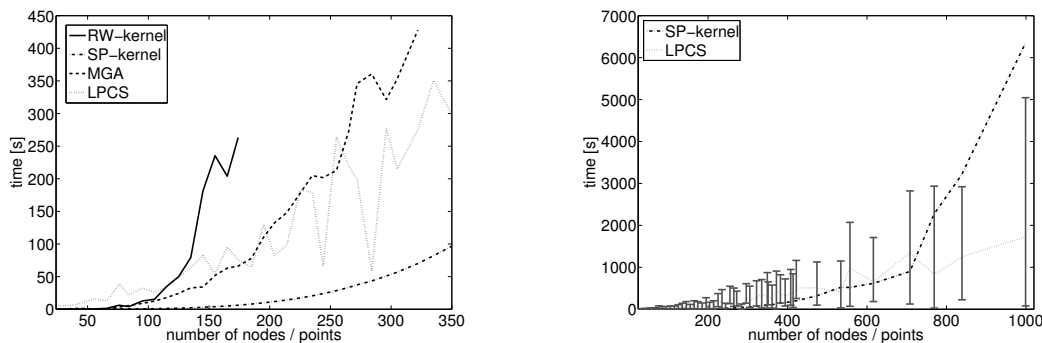
Table 3 furthermore shows how the performance of LPCS depends on the choice of the trade-off parameter α in (5). As can be seen, this parameter does indeed have an influence, even though the differences are not extreme. For this data set, α -values around 0.5 yield better results than extreme values close to 0 or 1; the optimal choice would be $\alpha = 0.7$. In practice, α can be considered as a tuning parameter to be adapted to the problem at hand (e.g., by means of a cross-validation on the training data).

Table 3: Accuracy of LPCS for different values of α in (5).

α	accuracy	α	accuracy
0	0.9042	0.6	0.9352
0.1	0.9183	0.7	0.9380
0.2	0.9126	0.8	0.9239
0.3	0.9154	0.9	0.9267
0.4	0.9267	1	0.9183
0.5	0.9352		

6.2.2 Runtime

To investigate the behavior regarding runtime of the approaches applied in this paper we used again the NADH/ATP data set and chose protein binding sites of size approximately 25, 35, ..., 985, 995. For a size s this was done by selecting the largest binding site that is smaller than s and a smallest binding site that is larger s . Doing this has the advantages that first the size of the problem to solve is in mean s , and second that both selected protein binding sites are different, so that side effects due to equivalence of both binding sites can be avoided. For MGA, SP- and RW-kernel the runtime for each size s was evaluated once since these methods are deterministic and have always same runtime. Since LPCS is based on a stochastic optimizer we repeated this experiment for each s 10 times using the same point clouds. The results are summarized in figure 5. As can be seen, from a



(a) runtimes of all methods in the range [25, 350], for LPCS only median was plotted

(b) runtimes of SP-kernel and LPCS (min, median, max) in the range [25; 1000]

Figure 5: Runtimes of LPCS, MGA, SP-, and RW-kernel w.r.t. problem size; for RW-kernel and MGA a calculation was possible to a certain size of the problem since the memory requirement was becoming too high

certain problem size for MGA and RW-kernel a calculation is not possible since these methods works with the product graph that is growing quadratically with the size of the

input graphs so that even modern computers (2 GB RAM) cannot offer sufficient memory. The LPCS approach has for very small problems the highest runtime though the runtime is growing very slow w.r.t. problem size, so that LPCS is already for point clouds of size 200 faster than MGA and RW-kernel, approached that are appropriate only for small structures. It sticks out that the LPCS runtime fluctuate strongly. The reason is quite simple. Since we use a real world data set the distribution of point labels vary. As already mentioned for the nearest neighbor search we hash points with equal label. So, if the labels are distributed uniformly the search is more efficient than if there exists a label that dominates the point cloud. The SP-kernel has cubic runtime, so that this method is for $s < 600$ the most efficient of all alternatives. However, it completely fails in terms of predictive accuracy. That LPCS is becoming the most efficient approach for $s > 600$ is hardly surprising, since the dimensionality of the LPCS optimization problem is constant (six parameters have to be optimized) and does not depend on the number of data points. It is true that the size of the point clouds does have an influence on the evaluation of the objective function, which involves a nearest neighbor search for each point. The increase in runtime is at most quadratic, however.

6.3 Alignment Quality

In the second study, we compared the quality of the alignments calculated, respectively, by 3DA and MGA. To this end, 100 alignments of size 2 were calculated for randomly chosen structures. Restricting to pairwise alignments is justified since both 3DA and MGA use the star alignment procedure to derive multiple alignments. The quality of a pairwise alignment \mathcal{A} is evaluated in terms of two criteria. The first criterion is the fraction of assignments of pseudocenters preserving the label information:

$$s_1 = \frac{1}{|\mathcal{A}|} \sum_{(a_1, a_2) \in \mathcal{A}} \begin{cases} 1, & \ell(a_1) = \ell(a_2) \\ 0, & \ell(a_1) \neq \ell(a_2) \end{cases} ,$$

where $\ell(a_1)$ is the label of the pseudocenter a_1 . Similarly, the second criterion evaluates to what extent the geometry of the structures is preserved. Since an MGA does not include information about the position of single pseudocenters, this has to be done by looking at distances between pairs of pseudocenters in each structure:

$$s_2 = \frac{1}{N} \sum_{(a_1, a_2), (b_1, b_2) \in \mathcal{A}} \begin{cases} 1, & |d(a_1, b_1) - d(a_2, b_2)| \leq \epsilon \\ 0, & |d(a_1, b_1) - d(a_2, b_2)| > \epsilon \end{cases} ,$$

where $d(a_1, b_1) = |c(a_1) - c(b_1)|$ and $N = |\mathcal{A}|(|\mathcal{A}| - 1)/2$. We summarize the evaluation by the vector

$$\mathbf{s} = (s_1, s_2) \in [0, 1] \times [0, 1] .$$

To measure the improvement of our method, we calculate the relative improvement

$$\mathbf{ri} = \begin{pmatrix} \frac{[\mathbf{s}_{3DA}]_1 - [\mathbf{s}_{MGA}]_1}{[\mathbf{s}_{MGA}]_1} \\ \frac{[\mathbf{s}_{3DA}]_2 - [\mathbf{s}_{MGA}]_2}{[\mathbf{s}_{MGA}]_2} \end{pmatrix} \quad (7)$$

where \mathbf{s}_{3DA} and \mathbf{s}_{MGA} denote, respectively, the evaluations of 3DA and MGA and where $[s]_i$ gives the i -th element of a vector \mathbf{s} .

6.3.1 Results

For our calculations we parameterized MGA as proposed in [17], for 3DA we set $k = 6$ and performed experiments like described above. The results for the benzamidine data set are shown in Figure 6, where the relative improvement vectors are plotted. As one can see, most of the ri vectors are lying in the first quadrant, indicating a positive improvement for both criteria.

The corresponding results for the thermolysin data set are depicted in Figure 6. Here, the picture is not as clear, and the number of negative improvements is even slightly higher than the number of positive ones. Apparently, 3DA performs especially good on highly similar structures while not improving on structures that are more diverse. This is hardly surprising, since 3DA strongly exploits information about the geometry of the structures.

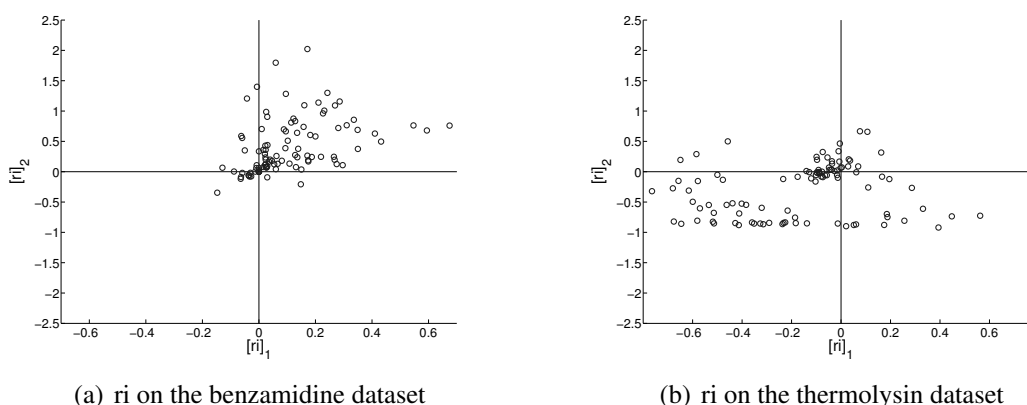


Figure 6: Relative improvements (ri) obtained by substituting the MGA approach in 3DA

6.3.2 Parametrization

As an important advantage of 3DA, it deserves mentioning that it only has a single parameter, while MGA has six parameters. In spite of this, we found that it often produces better results, even when trying to parameterize MGA in an optimal way. For example, Figure 7 shows a set of solutions for the benzamidine data that we found by varying the parameters in 3DA and MGA. For ease of exposition, we only plotted the solutions that are Pareto optimal in the two respective sets of solutions; in total, 7776 result vectors s were computed for MGA by variation its 5 parameters in a systematic way. This was done by varying penalties from -5 to 0 and awards from 0 to 5 and considering all possible combinations (see [17] for an explanation of these parameters). For 3DA there was only one parameter (threshold k) to vary, so that here only 12 results were calculated by considering $k = 0, \dots, 11$. To have a readable plot we removed results that are not Pareto optimal¹ and plot only the remaining Pareto optimal points. The resulting plot is illustrated in figure 7. As one can see the 3DA solutions were independent of parameterization always better than the MGA results, so that we can claim that our novel method is easy to adjust and will lead to results that are better, even for an optimal adjusted MGA approach.

¹Given a set of results S only such results $s \in S$ are called Pareto optimal that are not dominated by other solutions. A vector x dominates another vector y if $x[i] \geq y[i]$ for all i and $x[i] > y[i]$ for some i .

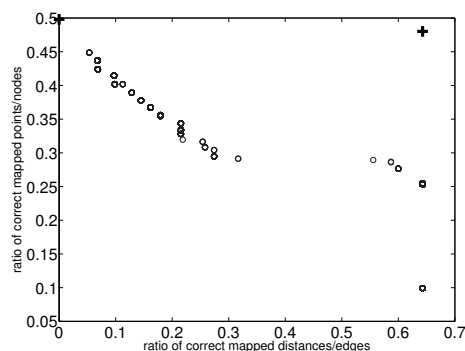


Figure 7: Pareto optimal solutions found by MGA (circles) and 3DA (crosses)

6.4 Structure Retrieval

The focus of the third study is on the ability to detect common substructures in a set of biochemical structures. We randomly selected 100 subsets of c compounds from the benzamidine data set and used 3DA and MGA to calculate an alignment. Then, we checked whether the aforementioned benzamidine core fragment, an amide derivative of benzol which consists of 25 atoms (11 hydrogens), was fully conserved in the alignment, which means that all pseudocenters belonging to the core were mutually assigned in a correct way. The results, shown in Table 4 for different numbers c , clearly show that 3DA is able to retrieve the core fragment much more reliably than MGA.

Table 4: Percent of alignments in which the benzamidine core fragment was fully conserved in the alignment of $c = \{2, 4, 8, 16\}$ structures.

c	2	4	8	16
MGA	0.85	0.38	0.14	0.04
3DA	0.96	0.92	0.80	0.76

For detecting the core fragment we searched for conserved patterns in the alignment and used the parameter $\omega = 1$ and $\xi = 0.9$.

7 Conclusions

In this paper, we have introduced labeled point cloud superposition (LPCS) as a novel tool for structural bioinformatics, namely as a method for comparing biomolecules on a structural level. Besides, using fuzzy modeling techniques, we have defined a related similarity measure. The concept of a labeled point cloud appears to be a quite natural representation for biological structures, especially since it is closely leaned on existing database formats. In comparison to other approaches, such as the prevalent graph-based methods, the modeling is hence simplified and does not involve any complex transformations. More importantly, a labeled point cloud preserves the full geometric information and makes it easily accessible to computational procedures.

A labeled point cloud superposition is a spatial “alignment” of two point clouds which is optimal in the sense of a given scoring (similarity) function. As for related problems

in bioinformatics, such as sequence alignment, the computation of the similarity between two objects hence involves the solution of an optimization problem. To this end, we have proposed the use of an evolution strategy, an approach from the family of evolutionary algorithms, which appears to be especially suitable for this problem.

First experimental results with classification data are quite promising and suggest that our approach is able to compare protein binding sites in a reasonable way. In terms of classification accuracy, LPCS turned out to be significantly better than existing (graph-based) methods used for comparison. Moreover, even though it is computationally more complex than these methods for small data sets, it scales much better and becomes more efficient for larger data sets. This is due to the fact that, in contrast to graph-based methods, the search space does not depend on the size of the point clouds and remains low-dimensional.

In this paper, we proposed an extension of the method of labeled point cloud superposition (LPCS), too. Motivated by applications in structural bioinformatics, we extended LPCS for the calculation of multiple geometric alignment which, based on a given superposition, computes an one-to-one correspondence between the points. First experiments carried out in the context of protein structure comparison are quite promising and show that our method is competitive, if not even superior, to state-of-the-art graph-based methods for multiple structure alignment. All things considered, multiple geometric alignment is therefore a viable option for protein structure comparison and might even be of interest beyond the field of structural bioinformatics.

References

- [1] Francis R. Bach. Graph kernels between point clouds. In *International Conference on Machine Learning*, pages 25–32, Helsinki, Finland, 2008.
- [2] Thomas Bartz-Beielstein. *Experimental research in evolutionary computation: The new experimentalism*. Springer, 2006.
- [3] Hans-Georg Beyer and Hans-Paul Schwefel. Evolution strategies: A comprehensive introduction. *Natural Computing*, 1(1):3–52, 2002.
- [4] M. Böhm, J. Stürzebecher, and G. Klebe. Three-dimensional quantitative structure-activity relationship analyses using comparative molecular field analysis and comparative molecular similarity indices analysis to elucidate selectivity differences of inhibitors binding to trypsin, thrombin, and factor xa. *Journal of Medicinal Chemistry*, 42(3):458–477, 1999.
- [5] K. M. Borgwardt. *Graph Kernels*. PhD thesis, Ludwig-Maximilians-Universität München, Germany, 2007.
- [6] K. M. Borgwardt and H. P. Kriegel. Shortest-path kernels on graphs. In *International Conference on Data Mining*, pages 74–81, Houston, Texas, 2005.
- [7] M. de Berg, M. van Kreveld, M. Overmars, and O. Schwarzkopf. *Computational Geometry*. Springer, New York, 2000.
- [8] J. Fodor and R.R. Yager. Fuzzy set-theoretic operators and quantifiers. In D. Dubois and H. Prade, editors, *Fundamentals of Fuzzy Sets*, pages 125–194. Kluwer Academic Publishers, Boston/London/Dordrecht, 2002.
- [9] Thomas Gärtner. A survey of kernels for structured data. *SIGKDD Explorations*, 5(1):49 – 58, 2003.

- [10] Johann Gasteiger and Thomas Engel. *Chemoinformatics*. Wiley-Vch, Weinheim, 2003.
- [11] M. Hendlich, F. Rippmann, and G. Barnickel. LIGSITE: Automatic and efficient detection of potential small molecule-binding sites in proteins. *Journal of Molecular Graphics and Modelling*, 15:359–363, 1997.
- [12] Wolfgang Kabsch. A solution of the best rotation to relate two sets of vectors. *Acta Crystallographica*, 32:922–923, 1976.
- [13] H.W. Kuhn. The Hungarian Method for the Assignment Problem. *Naval Research Logistics*, 52(1):7–21, 2005.
- [14] F. Mémoli and G. Sapiro. Comparing point clouds. In *Eurographics / ACM SIGGRAPH symposium on Geometry processing*, pages 32–40, Nice, France, 2004.
- [15] S. Schmitt, D. Kuhn, and G. Klebe. A new method to detect related function among proteins independent of sequence and fold homology. *Journal of Molecular Biology*, 323(2):387–406, 2002.
- [16] D. Sinha and E.R. Dougherty. Fuzzification of set inclusion: theory and applications. *Fuzzy Sets and Systems*, 55(1):15–42, 1993.
- [17] N. Weskamp, E. Hüllermeier, D. Kuhn, and G. Klebe. Multiple graph alignment for the structural analysis of protein active sites. *IEEE Transactions on Computational Biology and Bioinformatics*, 4(2):310–320, 2007.
- [18] R.R. Yager. On ordered weighted averaging aggregation operators in multicriteria decision making. 18(1):183–190, 1988.
- [19] L.A. Zadeh. A computational approach to fuzzy quantifiers in natural languages. *Comput. Math. Appl.*, 9:149–184, 1983.

OCD: Ein Online Konvergenzkriterium für die evolutionäre, mehrkriterielle Optimierung

Boris Naujoks¹, Heike Trautmann² und Tobias Wagner³

¹ Log!n GmbH, Schwelm; E-Mail: Boris.Naujoks@login-online.de

² Technische Universität Dortmund, Fakultät Statistik;

E-Mail: trautmann@statistik.uni-dortmund.de

³ Technische Universität Dortmund, Institut für spanende Fertigung;

E-Mail: wagner@isf.maschinenbau.uni-dortmund.de

Abstract

Die Bestimmung des optimalen Abbruchzeitpunktes bei der Anwendung eines mehrkriteriellen, evolutionären Algorithmus hängt von vielen Faktoren ab. Derzeit wird zumeist eine feste Anzahl von Zielfunktionsauswertungen vorgegeben. Das vorgestellte Online Konvergenzkriterium (Online Convergence Detection, OCD) stützt sich auf die Auswertung von Performance-Indikatoren im Verlauf des Algorithmus, die die Güte der aktuellen Pareto-Front-Approximation widerspiegeln. Es wird untersucht, ob die Varianz dieser Werte klein genug und/oder keine signifikante Verbesserung mehr erkennbar ist. Die Anwendung auf mathematischen Testfunktionen sowie Optimierproblemen aus der Aerodynamik signalisiert eine hohe Effektivität von OCD. Ohne signifikanten Performance-Verlust lassen sich im Vergleich zu den Empfehlungen aus der Literatur viele Zielfunktionsauswertungen einsparen.

1 Einleitung

Der Abbruch eines stochastischen Optimierverfahrens ist eine kritische Entscheidung. Einerseits kann die Entscheidung, ob ein globales Optimum gefunden wurde, nicht mit Sicherheit getroffen werden. Eine Fortsetzung könnte sich also lohnen. Andererseits dürfen keine unnötigen Ressourcen verschwendet werden, wenn eine Fortführung des Optimierlaufs nicht vielversprechend ist, d.h. nur mit einer sehr geringen Wahrscheinlichkeit zu Verbesserungen führen wird.

Das Einsatzgebiet des Online Konvergenzkriteriums sind evolutionäre, mehrkriterielle Optimierverfahren (Evolutionary Multi-objective Optimisation Algorithms, EMOA), da diese sich in den letzten Jahren gerade für relevante, praxisnahe Anwendungen bewährt haben (vgl. die Bücher von Deb [1] und Coello Coello et al [2, 3]). Speziell bei diesen Anwendungen ist der Abbruch des Optimierlaufs zum richtigen Zeitpunkt von besonderem Interesse, da Zielfunktionsauswertungen in der Regel sehr zeitaufwändig und somit kostenintensiv sind. Normalerweise wird hier nach einer festen, vordefinierten Anzahl von Zielfunktionsauswertungen abgebrochen.

OCD basiert auf statistischen Tests bzgl. der Varianz und dem Trend der zuletzt erzielten Ergebnisse. Dazu wird ein Zeitfenster definiert, in dessen Rahmen ein χ^2 -Varianztest und

ein t -Test für den Regressionskoeffizienten auf einer Menge von Qualitätsindikatoren durchgeführt wird.

Der folgende Abschnitt dieses Beitrags stellt OCD detaillierter vor. Zudem wird Pseudocode zu dem eigentlichen Verfahren und den statistischen Tests angegeben. Abschnitt 3 stellt die Verfahren vor, mit denen OCD gekoppelt wurde und beschreibt die zugehörigen Funktionen und Testfälle. Die Ergebnisse dieser Experimente sind in Abschnitt 4 aufgeführt. Abschnitt 5 fasst diese zusammen und gibt einen Ausblick auf weitere Forschungsarbeiten.

2 Das Online Konvergenzkriterium OCD

Zur Beantwortung der Frage, wann der Lauf eines stochastischen Optimierverfahrens abgebrochen werden kann, sollten mehrere Aspekte berücksichtigt werden. Wie bereits dargelegt, beschränken wir uns hier auf EMOA. Die folgenden Aspekte lassen sich aber in der Regel direkt auf andere Optimierverfahren übertragen:

- Verbesserungen während der letzten Generationen
- Art der Performance-Messung bei den eingesetzten EMOA
- Nähe zur wahren Pareto-Front
- Verteilung der Individuen etc.

Algorithmus 1 fasst das OCD Verfahren kompakt zusammen. Er kann als Grundlage für eine eigene Implementierung dienen.

Um die Entscheidung, ob ein EMOA zu einem bestimmten Zeitpunkt abgebrochen werden soll, zu treffen, wird die Güte der Population zu eben diesem Zeitpunkt benötigt. Als Referenzmenge für Vergleiche wird die Pareto-Front aus der aktuellen Population verwendet. Mit Hilfe von minimalen und maximalen Zielfunktionswerten dieser werden die Werte normalisiert. Für die normalisierte Front werden Indikatorfunktionen für die Pareto-Fronten einer festen Anzahl vorangegangener Generationen berechnet, deren Verläufe für die spätere Entscheidung über den Abbruch herangezogen werden. Basierend auf der Empfehlung von Knowles et al. [4] sind dies im vorliegenden Fall das Hypervolumen („HV“), der ε -Indikator („Eps“) und der R2-Indikator („R2“). Das Zeitfenster, für das die Indikatoren rückwirkend von der aktuellen Generation berechnet werden, ist ein Parameter (n_{pregen}) des Verfahrens OCD.

Auf Basis der Werte der Indikatoren im Zeitfenster werden dann die erwähnten statistischen Tests durchgeführt. Zuerst wird mit Hilfe eines χ^2 -Varianztests geprüft, ob die Varianz der Indikatorwerte unterhalb einer vorgegebenen Grenze liegt (vgl. Algorithmus 2). Der zweite verwendete Test ist ein t -Test bzgl. des Regressionskoeffizienten einer linearen Regression auf den Indikatorwerten innerhalb des gegebenen Zeitfensters. Hier wird überprüft, ob noch von einem fallenden Trend der Werte ausgegangen werden kann (vgl. Algorithmus 3). Konvergenz wird von OCD angezeigt, wenn mindestens einer der beiden oben genannten Tests für alle drei Indikatoren gleichzeitig Konvergenz signalisiert. In diesem Fall wird der Optimierlauf gestoppt. Sollte dies nicht bis zu einer vorgegebenen Generationsgrenze $MaxGen$ passiert sein, wird der EMOA dort beendet.

Algorithmus 1 OCD: Algorithmus für das Online Konvergenzkriterium

Vorausgesetzt: $VarLimit$ /* Varianzlimit */
 $npregen$ /* Anzahl vorangegangener Generationen (Zeitfenster) */
 α /* Signifikanzniveau der Tests */
 $MaxGen$ /* Maximale Generationsanzahl */
 (PI_1, \dots, PI_n) /* Vektor von Performance-Indikatoren, z.B. (HV, ϵ , R2) */

1: $i = 0$ /* Initialisierung der Generationsanzahl */
2: **for all** $i \in \{1, \dots, npregen\}, j \in \{1, \dots, n\}$ **do**
3: $pChi2(j, i) = 1$ /* Initialisierung der p-Werte des χ^2 - Varianztests */
4: $pReg(i) = 0$ /* Initialisierung der p-Werte des t-Tests */
5: **end for**
6: $\vec{l}b = []$ /* Initialisierung der unteren Schranke für die Normalisierung */
7: $\vec{u}b = []$ /* Initialisierung der oberen Schranke für die Normalisierung */
8: **repeat**
9: $i = i + 1$
10: Berechne die d -dimensionale Pareto front PF_i der i -ten EMOA-Generation
11: $\vec{l}b = \min(\vec{l}b \cup PF_i)$ /* Aktualisiere untere Schranken */
12: $\vec{u}b = \max(\vec{u}b \cup PF_i)$ /* Aktualisiere obere Schranken */
13: **if** $(i > npregen)$ **then**
14: $PF_i = 1 + (PF_i - \vec{l}b) / (\vec{u}b - \vec{l}b)$ /* Normalisiere PF_i auf $[1, 2]^d$ */
15: **for all** $k \in \{i - npregen, \dots, i - 1\}$ **do**
16: Berechne die Pareto front PF_k der k -ten EMOA-Generation
17: $PF_k = 1 + (PF_k - \vec{l}b) / (\vec{u}b - \vec{l}b)$ /* Normalisiere PF_k auf $[1, 2]^d$ */
18: **end for**
19: **for all** $j \in \{1, \dots, n\}$ **do**
20: $\vec{P}I_{j,i} = (PI_j(PF_{i-npregen}, PF_i, \vec{1}, \vec{2}\cdot\vec{1}), \dots, (PI_j(PF_{i-1}, PF_i, \vec{1}, \vec{2}\cdot\vec{1})))$
/* Berechne PI_j für $PF_{i-npregen}, \dots, PF_{i-1}$ mit PF_i als Referenzmenge, $\vec{1}$ als idealer, und $\vec{2}\cdot\vec{1}$ als Nadir-Punkt */
21: $pChi2(j, i) = \text{call } Chi2(\vec{P}I_{j,i}, VarLimit)$ /* p-Wert des χ^2 -Tests */
22: **end for**
23: $pReg(i) = \text{call } Reg(\vec{P}I_{1,i}, \dots, \vec{P}I_{n,i})$ /* p-Wert des t-Tests auf $\vec{P}I_{j,i}$ */
24: **end if**
25: **until** $\forall j \in \{1, \dots, n\} :$
 $(pChi2(j, i) \leq \alpha/n) \wedge (pChi2(j, i-1) \leq \alpha/n)$
 $\vee (pReg(i) > \alpha) \wedge (pReg(i-1) > \alpha)$
 $\vee i = MaxGen$
26: Beende EMOA
27: **return** $\{MaxGen, Chi2, Reg\}$ /* Abbruchkriterium des EMOA */
 i /* Stoppgeneration */

Algorithmus 2 $Chi2$: Einseitiger χ^2 -Varianztest für

$$H_0 : \text{var}(\vec{P}I) \geq VarLimit \text{ vs. } H_1 : \text{var}(\vec{P}I) < VarLimit$$

Vorausgesetzt: $\vec{P}I$ /* Vektor von Indikatorwerten */
 $VarLimit$ /* Varianzlimit */

1: $N = \text{length}(\vec{P}I) - 1$ /* Anzahl Freiheitsgrade */
2: $Chi = [\text{var}(\vec{P}I) * N] / VarLimit$ /* Berechnung der Teststatistik */
3: $p = \chi^2(Chi, N)$ /* χ^2 - Verteilungsfunktion mit N Freiheitsgraden */
4: **return** p

Tabelle 1: Parametereinstellungen der Experimente (I)

Testfunktion	$MaxGen$	μ
Fonseca	66	60
ZDT1, ZDT2	200	100
ZDT4	100	200
DTLZ2	300	100
NACA	$1000/\mu$	8, 20
RAE	$1000/\mu$	8, 20

Tabelle 2: Parametereinstellungen der Experimente (II)

Alg.	Implementierung	p_c	p_m	η_c	η_m	p_{swap}
NSGA-II	R [14]	0.7	0.2	20	20	0
SMSEMOA	PISA [15]	0.9	$1/\text{length}(\vec{x})$	15	20	0.5

3 Experimente

OCD wurde sowohl auf mathematischen Testfunktionen (Wagner et al. [5]), Trautmann et al. [6]) als auch auf Anwendungsfällen aus dem Bereich der Aerodynamik mit zwei EMOA getestet, dem NSGA-II¹ (Deb et al. [7]) und dem SMS-EMOA (Beume et al. [8, 9]). Zur Sicherung der Vergleichbarkeit mit dem NSGA-II werden beim SMS-EMOA μ Zielfunktionsauswertungen als eine Generation aufgefasst, wobei μ die Populationsgröße bezeichnet.

Als mathematische Testfunktionen wurden überwiegend ZDT-Funktionen (Zitzler, Deb und Thiele [10]) herangezogen. Es handelt sich dabei um die bi-kriteriellen Funktionen ZDT1 bis ZDT4. Zusätzlich wurde die Fonseca-Funktion [11] und die Funktion DTLZ2 (Deb, Thiele, Laumanns und Zitzler [12]) betrachtet; letztere in der Ausprägung mit 3 Zielfunktionskriterien.

Neben den mathematischen Testfunktionen wurde OCD mit den beiden Algorithmen auf zwei Anwendungsfällen aus der Aerodynamik getestet (Naujoks und Trautmann [13]). Hierbei handelt es sich um den bi-kriteriellen Fall eines Redesigns von Tragflächenschnitten (im folgenden auch NACA Testfall genannt) und einer Aufgabenstellung aus der reinen Widerstandsminimierung mit drei Zielfunktionskriterien (RAE Testfall).

Für jede Kombination von Algorithmus und Testfunktion wurden 10 unabhängige Läufe durchgeführt. Die konkreten Parametereinstellungen der Algorithmen und der Experimente inklusive der empfohlenen Anzahl der Generationen $MaxGen$ aus der Literatur sind in den Tabellen 1 und 2 zusammengefasst.

Für das Varianzlimit wurde ein Wert von $VarLimit = 0.001^2$ verwendet, das Signifikanzniveau beider Tests ist $\alpha = 0.05$. Für die mathematischen Testfunktionen wurde die Länge des Zeitfensters durch $npregen = 10$ beschrieben, für die aerodynamischen Testfälle wurde auch der Effekt von $npregen = 15$ untersucht. Als Performance-Indikatoren wurde das Hypervolumen (Zitzler und Thiele [16]) für die aerodynamischen Testfälle betrachtet, für die mathematischen Testfunktionen zusätzlich sowohl der ε - (Zitzler et al. [17])- als auch der $R2$ -Indikator (Hansen und Jaszkiewicz [18]).

¹R-Version des NSGA-II aus dem Paket *mco* (<http://cran.r-project.org/web/packages/mco/index.html>)

Algorithmus 3 *Reg: Zweiseitiger t-Test auf Signifikanz des linearen Trends*

$$H_0 : \beta = 0 \text{ vs. } H_1 : \beta \neq 0$$

Vorausgesetzt: $\vec{P}I_j$, $j = (1, \dots, n)$ /* Vektor von Indikatorwerten */
1: $N = n \cdot \text{length}(\vec{P}I^*) - 1$ /* Anzahl Freiheitsgrade */
2: **for all** $j \in \{1, \dots, n\}$ **do**
3: $\vec{P}I^*_j = (\vec{P}I_j - \vec{P}I_j) / \sigma_{\vec{P}I_j}$ /* Standardisierung */
4: **end for**
5: $\vec{P}I^* := \text{concatenate}(\vec{P}I^*_1, \dots, \vec{P}I^*_n)$ /* Zeilenvektor aller $\vec{P}I_j$ */
6: $X = \underbrace{(1, \dots, \text{length}(\vec{P}I^*), \dots, 1, \dots, \text{length}(\vec{P}I^*))}_{n \text{ mal}}$

7: $\hat{\beta} = (X * X^T)^{-1} * X * (\vec{P}I^*)^T$ /* Zeilenvektor der Generationen entsprechend $\vec{P}I^*$ */
8: $\varepsilon = \vec{P}I^* - X * \hat{\beta}$ /* Schätzung des Regressionskoeffizienten */
9: $s^2 = (\varepsilon * \varepsilon^T) / N$ /* Berechnung der Residuen */
10: $t = \frac{\hat{\beta}}{\sqrt{s^2(X * X^T)^{-1}}}$ /* mittlerer quadratischer Fehler der Regression */
11: $p = 2 \cdot \min(t_N(t), 1 - t_N(t))$ /* Berechnung der Teststatistik */
12: **return** p /* t-Verteilung mit N Freiheitsgraden */

Bei RAE wurden für die Zielfunktionskriterien drei Strömungsbedingungen vorgegeben (vgl. Tabelle 3), zu denen jeweils ein Tragflächenschnitt mit minimalem Widerstandsbeiwert generiert werden sollte. Zusätzliche Nebenbedingungen sowohl geometrischer, als auch aerodynamischer Natur wurden für den Test des OCD Verfahrens nicht weiter betrachtet.

Tabelle 3: Strömungsbedingungen, Anwendungsfall RAE

	Bedingung 1	Bedingung 2	Bedingung 3
Mach Zahl	0.734	0.754	0.680
Reynolds Zahl	$6.5 \cdot 10^6$	$6.2 \cdot 10^6$	$5.7 \cdot 10^6$
Anstellwinkel	2.8	2.8	1.8
Transition	3%	3%	11%

Im NACA Testfall wurden zwei nahezu optimale Tragflächenschnitte zu zwei Strömungsbedingungen *A* und *B* (vgl. Tabelle 4) vorgegeben. Die beiden Zielfunktionskriterien ergeben sich aus den Abweichungen in der Druckverteilung entlang der jeweiligen Tragflächenschnitte.

4 Ergebnisse

Die Ergebnisse zu den mathematischen Testfunktionen sind in Tabelle 5 dargestellt. Diese zeigt zu den jeweiligen Algorithmen in der rechten Spalte, wie viele Generationen (im Kasten jeweils oben) und Zielfunktionsauswertungen (Mitte) mit OCD eingespart werden konnten. Dazu wurden die Ergebnisse von OCD mit den in der Literatur empfohlenen Anzahlen von Generationen bzw. Zielfunktionsauswertungen verglichen. Der untere Wert der jeweiligen Kästen gibt die prozentuale Einsparung an.

Tabelle 4: Strömungsbedingungen, Anwendungsfall NACA

	A	B
Mach Zahl	0.20	0.77
Reynolds Zahl	$5 \cdot 10^6$	10^7
Anstellwinkel	10.8	1.0

Tabelle 5: Ergebnisse für die mathematischen Testfunktionen, PI: Performance-Indikator

Testfunktion	PI	NSGA-II		SMS-EMOA	
		med(PIDiff)	med(GenDiff)	med(PIDiff)	med(GenDiff)
ZDT1	HV	2.07e-03	124	0.96e-03	112
	Eps	2.08e-03	12400 FE	1.31e-03	11200 FE
	R2	0.93e-03	62%	0.38e-03	56%
ZDT2	HV	2.56e-03	104	1.01e-03	101
	Eps	3.13e-03	10400 FE	0.91e-03	10100 FE
	R2	1.46e-03	52%	0.63e-03	51%
ZDT4	HV	0.26e-03	18	21.72e-03	63
	Eps	0.28e-03	3600 FE	19.75e-03	12600 FE
	R2	0.06e-03	18%	9.07e-03	63%
DTLZ2	HV	-0.39e-03	287	0.72e-03	256
	Eps	-14.76e-03	28700 FE	3.37e-03	25600 FE
	R2	0.06e-03	96%	0.02e-03	85%
Fonseca	HV	0.97e-03	50	2.49e-03	49
	Eps	-0.51e-03	3000 FE	5.14e-03	2940 FE
	R2	0.14e-03	76%	0.21e-03	74%

Die linke Spalte für den jeweiligen Algorithmus gibt den Qualitätsverlust für die jeweiligen Indikatoren an, wenn der Lauf entsprechend den Werten der rechten Spalte gestoppt wurde. Dieser Verlust liegt für alle Indikatoren in der Regel in der Größenordnung von 10^{-3} , teils sogar darunter (bis 10^{-5}). Seltener treten Werte aus dem Bereich 10^{-2} auf.

Mit diesen geringen Qualitätsverlusten lassen sich mindestens 18% der Zielfunktionsauswertungen im Vergleich zu den Literaturempfehlungen einsparen. Von den 18% von NSGA-II auf ZDT4 aber abgesehen, sind alle prozentualen Werte für die Einsparung größer als 50%. Es lassen sich also fast immer mehr als die Hälfte der Auswertungen einsparen. Dies geht sogar hoch bis 96% der Auswertungen im Vergleich zur Literaturempfehlung für NSGA-II auf DTLZ2.

Tabelle 6 gibt für verschiedene Populationsgrößen im Algorithmus und Größen des Zeitfensters n_{pregen} an, welches Konvergenzkriterium in den Anwendungsfällen aus der Aerodynamik verantwortlich für den Abbruch des Optimierlaufs ist. Sowohl für den NSGA-II, als auch den SMS-EMOA war das auf dem NACA Testfall ausschließlich das Varianzkriterium. Die Experimente auf dem RAE Testfall zeigen aber die Bedeutung des Regressionskriteriums. Insbesondere der SMS-EMOA wird hier fast ausschließlich durch dieses Kriterium terminiert.

Die Tabellen 7 und 8 geben die konkreten Einsparungen und Verluste für alle oben bereits betrachteten Parameter an. Für den NACA Testfall (vgl. Tabelle 7) sind Einsparungen

Tabelle 6: Anzahl von Läufen, die durch das Varianz- (Var) bzw. das Regressionskriterium (Reg) beendet wurden.

Alg.	μ	n_{pregen}	NACA		RAE	
			Var	Reg	Var	Reg
NSGA-II	8	10	0	10	5	5
	8	15	0	10	8	2
	20	10	0	10	9	1
	20	15	0	9	10	0
SMS-EMOA	8	10	0	10	5	5
	8	15	0	3	10	0
	20	10	0	10	10	0
	20	15	0	0	10	0

von bis zu 80% der Auswertungen möglich, wenn Verluste bis 4% im Hypervolumen akzeptabel sind (NSGA-II, $\mu = 8$, $n_{pregen} = 10$). Soll der Verlust auf 1% beschränkt werden, sind Einsparungen von bis zu 40% für den NSGA-II und 22% für den SMS-EMOA möglich. Bei zwei Parametereinstellungen terminierte OCD den SMS-EMOA auf diesem Testfall nicht (jeweils $n_{pregen} = 15$).

Tabelle 7: NACA: Absolute und prozentuale mediane Differenzen im Hypervolumen und bzgl. des Vergleiches der OCD Stoppgeneration und der Literaturempfehlung $MaxGen$.

Alg.	μ	n_{pregen}	NACA	
			med(Gen_{Diff})	med(HV_{Diff})
NSGA-II	8	10	100.0 (80.0 %)	0.0031 (3.97 %)
	8	15	93.0 (74.4 %)	0.0026 (3.29 %)
	20	10	20.0 (40.0 %)	0.0008 (0.99 %)
	20	15	5.5 (11.0 %)	0.0000 (0.01 %)
SMS-EMOA	8	10	87.0 (69.6 %)	0.0047 (5.97 %)
	8	15	0.0 (0.0 %)	0.0000 (0.00 %)
	20	10	11.0 (22.0 %)	0.0007 (0.81 %)
	20	15	0.0 (0.0 %)	0.0000 (0.00 %)

Für den RAE Testfall (vgl. Tabelle 8) sind die Verluste im Hypervolumen immer kleiner als 1%, und Einsparungen bis zu 71% der Zielfunktionsauswertungen sind möglich ($\mu = 8$, $n_{pregen} = 10$). Hier greift OCD bei allen Parametrisierungen und allen Algorithmen.

5 Zusammenfassung und Ausblick

Das Online Konvergenzkriterium OCD erweist sich als eine verlässliche und effiziente Methode zur Bestimmung des optimalen Abbruchzeitpunktes von evolutionären, mehrkriteriellen Algorithmen. Sowohl der χ^2 -Varianztest als auch der t -Test auf Performance-Indikatoren vergangener Generationen signalisieren sehr exakt, wann die Konvergenz des Algorithmus zu erkennen ist. Die Art und Anzahl der verwendeten Performance-Indikatoren zur Messung der Güte der Approximation der Pareto-Front sind variabel. OCD wurde unter Verwendung dreier State-Of-the-Art-Indikatoren mit mehrkriteriellen,

Tabelle 8: Absolute und prozentuale mediane Differenzen im Hypervolumen und bzgl. des Vergleiches der OCD Stoppgeneration und der Literaturempfehlung *MaxGen*.

Alg.	μ	n_{pregen}	RAE	
			med(Gen_{Diff})	med(HV_{Diff})
NSGA-II	8	10	87.0 (69.6%)	5.29e-05 (0.23%)
	8	15	78.0 (62.4%)	2.58e-05 (0.11%)
	20	10	26.0 (52.0%)	8.06e-05 (0.35%)
	20	15	21.5 (43.0%)	4.36e-05 (0.19%)
SMS-EMOA	8	10	89.5 (71.6%)	8.33e-05 (0.36%)
	8	15	81.0 (64.8%)	4.38e-05 (0.19%)
	20	10	24.0 (48.0%)	7.41e-05 (0.32%)
	20	15	19.5 (39.0%)	3.58e-05 (0.16%)

evolutionären Algorithmen auf mathematischen Testfunktionen als auch Optimierproblemen aus der Aerodynamik gekoppelt. In beiden Fällen konnten mit Hilfe von OCD bemerkenswert viele Zielfunktionsauswertungen eingespart werden ohne dabei signifikant an Qualität bzgl. der Indikatoren zu verlieren.

Ferner kann OCD zu einem fundierten und effizienten Performance-Vergleich mehrkriterieller evolutionärer Algorithmen verwendet werden. Werden diese vergleichbar parametrisiert und eine ausreichende Anzahl an Wiederholungen der Läufe durchgeführt, können die Verteilungen der Stoppgenerationen zum statistischen Vergleich herangezogen werden.

Vielversprechende weitere Forschungsfelder sind zum einen die Übertragbarkeit auf ein-kriterielle, evolutionäre Algorithmen als auch eine Untersuchung der optimalen Parametrisierung von OCD für verschiedene Anwendungsfälle und Klassen von mehrkriteriellen Optimierproblemen.

Literatur

- [1] Deb, K.: *Multi-objective Optimization using Evolutionary Algorithms*. Chichester, UK: Wiley. ISBN 047187339X. 2001.
- [2] Coello, C. C.; Lamont, G. B.: *Applications Of Multi-Objective Evolutionary Algorithms*. World Scientific Publishing Co., Singapore. 2004.
- [3] Coello, C. C.; Veldhuizen, D. V.; Lamont, G. B.: *Evolutionary Algorithms for Solving Multi-Objective Problems, 2nd ed.*. Springer, Berlin. 2007.
- [4] Knowles, J.; Thiele, L.; Zitzler, E.: A Tutorial on the Performance Assessment of Stochastic Multiobjective Optimizers. 214, Computer Engineering and Networks Laboratory (TIK), Swiss Federal Institute of Technology (ETH) Zurich. 2005.
- [5] Wagner, T.; Trautmann, H.; Naujoks, B.: OCD: Online Convergence Detection for Evolutionary Multi-Objective Algorithms Based on Statistical Testing. In: *Evolutionary Multi-Criterion Optimization (EMO 2009)*. Springer, Berlin. (in print). 2009.

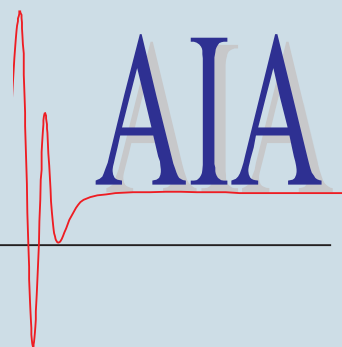
- [6] Trautmann, H.; Wagner, T.; Naujoks, B.; Preuss, M.; Mehnen, J.: Statistical Methods for Convergence Detection of Multiobjective Evolutionary Algorithms. *Evolutionary Computation Journal* (erscheint 2009).
- [7] Deb, K.; Pratap, A.; Agarwal, S.: A Fast And Elitist Multi-objective Genetic Algorithm: NSGA-II. *IEEE Trans. on Evolutionary Computation* 6 (2002) 8.
- [8] Beume, N.; Naujoks, B.; Rudolph, G.: Mehrkriterielle Optimierung durch evolutionäre Algorithmen mit S-Metrik-Selektion. In: *16. Workshop Computational Intelligence* (Mikut, R.; Reischl, M., Hg.), S. 1–10. Universitätsverlag Karlsruhe. 2006.
- [9] Beume, N.; Naujoks, B.; Rudolph, G.: SMS-EMOA: Effektive evolutionäre Mehrzieloptimierung. *at-Automatisierungstechnik* 56 (2008) 7, S. 357–364.
- [10] Zitzler, E.; Deb, K.; Thiele, L.: Comparison of Multiobjective Evolutionary Algorithms: Empirical Results. *Evolutionary Computation* 8 (2000) 2, S. 173–195.
- [11] Fonseca, C. M.; Fleming, P. J.: Multiobjective Genetic Algorithms Made Easy: Selection, Sharing, and Mating Restriction. In: *Genetic Algorithms in Engineering Systems: Innovations and Applications*, S. 42–52. 1995.
- [12] Deb, K.; Thiele, L.; Laumanns, M.; Zitzler, E.: Scalable Multi-Objective Optimization Test Problems. In: *Congress on Evolutionary Computation (CEC)*, Bd. 1, S. 825–830. IEEE Press, Piscataway NJ. 2002.
- [13] Naujoks, B.; Trautmann, H.: Online Convergence Detection for Aerodynamic Applications. In: *IEEE Congress on Evolutionary Computation 2009, CEC 2009* (Tyrrell, A.; et al., Hg.), S. 332–339. IEEE press, US. 2009.
- [14] Ihaka, R.; Gentleman, R.: R: A language for data analysis and graphics. *Journal of Computational and Graphical Statistics* 5 (1996), S. 299–314.
- [15] Bleuler, S.; Laumanns, M.; Thiele, L.; Zitzler, E.: PISA — A Platform and Programming Language Independent Interface for Search Algorithms. In: *Evolutionary Multi-Criterion Optimization (EMO)* (Fonseca, C. M.; et al., Hg.), S. 494 – 508. Springer, Berlin. 2003.
- [16] Zitzler, E.; Thiele, L.: Multiobjective Optimization Using Evolutionary Algorithms — A Comparative Case Study. *Lecture Notes in Computer Science* 1498 (1998), S. 292–301.
- [17] Zitzler, E.; Thiele, L.; Laumanns, M.; Fonseca, C.; Fonseca, V.: Performance Assessment of Multiobjective Optimizers: An Analysis and Review. *IEEE Trans. on Evolutionary Computation* 8 (2003) 2, S. 117–132.
- [18] Hansen, M. P.; Jaszkiewicz, A.: Evaluating the quality of approximations to the non-dominated set. *Techn. Ber. IMM-REP-1998-7*. 1998.

**Bereits veröffentlicht wurden in der Schriftenreihe des
Instituts für Angewandte Informatik / Automatisierungstechnik im
Universitätsverlag Karlsruhe:**

- Nr. 1: BECK, S.: Ein Konzept zur automatischen Lösung von Entscheidungsproblemen bei Unsicherheit mittels der Theorie der unscharfen Mengen und der Evidenztheorie, 2005
- Nr. 2: MARTIN, J.: Ein Beitrag zur Integration von Sensoren in eine anthropomorphe künstliche Hand mit flexiblen Fluidaktoren, 2004
- Nr. 3: TRACHEL, A.: Neue Verfahren zur Modellierung nichtlinearer thermodynamischer Prozesse in einem Druckbehälter mit siedendem Wasser-Dampf Gemisch bei negativen Drucktransienten, 2005
- Nr. 4: LOOSE, T.: Konzept für eine modellgestützte Diagnostik mittels Data Mining am Beispiel der Bewegungsanalyse, 2004
- Nr. 5: MATTHES, J.: Eine neue Methode zur Quellenlokalisierung auf der Basis räumlich verteilter, punktwiser Konzentrationsmessungen, 2004
- Nr. 6: MIKUT, R.; REISCHL, M.: Proceedings – 14. Workshop Fuzzy-Systeme und Computational Intelligence: Dortmund, 10. - 12. November 2004, 2004
- Nr. 7: ZIPSER, S.: Beitrag zur modellbasierten Regelung von Verbrennungsprozessen, 2004
- Nr. 8: STADLER, A.: Ein Beitrag zur Ableitung regelbasierter Modelle aus Zeitreihen, 2005
- Nr. 9: MIKUT, R.; REISCHL, M.: Proceedings – 15. Workshop Computational Intelligence: Dortmund, 16. - 18. November 2005, 2005
- Nr. 10: BÄR, M.: μ FEMOS – Mikro-Fertigungstechniken für hybride mikrooptische Sensoren, 2005
- Nr. 11: SCHAUDEL, F.: Entropie- und Störungssensitivität als neues Kriterium zum Vergleich verschiedener Entscheidungskalküle, 2006
- Nr. 12: SCHABLOWSKI-TRAUTMANN, M.: Konzept zur Analyse der Lokomotion auf dem Laufband bei inkompletter Querschnittlähmung mit Verfahren der nichtlinearen Dynamik, 2006
- Nr. 13: REISCHL, M.: Ein Verfahren zum automatischen Entwurf von Mensch-Maschine-Schnittstellen am Beispiel myoelektrischer Handprothesen, 2006
- Nr. 14: KOKER, T.: Konzeption und Realisierung einer neuen Prozesskette zur Integration von Kohlenstoff-Nanoröhren über Handhabung in technische Anwendungen, 2007
- Nr. 15: MIKUT, R.; REISCHL, M.: Proceedings – 16. Workshop Computational Intelligence: Dortmund, 29. November - 1. Dezember 2006
- Nr. 16: LI, S.: Entwicklung eines Verfahrens zur Automatisierung der CAD/CAM-Kette in der Einzelfertigung am Beispiel von Mauerwerksteinen, 2007
- Nr. 17: BERGEMANN, M.: Neues mechatronisches System für die Wiederherstellung der Akkommodationsfähigkeit des menschlichen Auges, 2007
- Nr. 18: HEINTZ, R.: Neues Verfahren zur invarianten Objekterkennung und -lokalisierung auf der Basis lokaler Merkmale, 2007
- Nr. 19: RUCHTER, M.: A New Concept for Mobile Environmental Education, 2007
- Nr. 20: MIKUT, R.; REISCHL, M.: Proceedings – 17. Workshop Computational Intelligence: Dortmund, 5. - 7. Dezember 2007
- Nr. 21: LEHMANN, A.: Neues Konzept zur Planung, Ausführung und Überwachung von Roboteraufgaben mit hierarchischen Petri-Netzen, 2008
- Nr. 22: MIKUT, R.: Data Mining in der Medizin und Medizintechnik, 2008
- Nr. 23: KLINK, S.: Neues System zur Erfassung des Akkommodationsbedarfs im menschlichen Auge, 2008

- Nr. 24: MIKUT, R.; REISCHL, M.: Proceedings – 18. Workshop Computational Intelligence: Dortmund, 3. - 5. Dezember 2008
- Nr. 25: WANG, L.: Virtual environments for grid computing, 2009
- Nr. 26: BURMEISTER, O.: Entwicklung von Klassifikatoren zur Analyse und Interpretation zeitvarianter Signale und deren Anwendung auf Biosignale, 2009
- Nr. 27: DICKERHOF, M.: Ein neues Konzept für das bedarfsgerechte Informations- und Wissensmanagement in Unternehmenskooperationen der Multimaterial-Mikrosystemtechnik, 2009
- Nr. 28: MACK, G.: Eine neue Methodik zur modellbasierten Bestimmung dynamischer Betriebslasten im mechatronischen Fahrwerkentwicklungsprozess, 2009

Die Schriften sind als PDF frei verfügbar, eine Nachbestellung der Printversion ist möglich.
Nähere Informationen unter www.uvka.de.



Dieser Tagungsband enthält die Beiträge des 19. Workshops „Computational Intelligence“ des Fachausschusses 5.14 der VDI/VDE-Gesellschaft für Mess- und Automatisierungstechnik (GMA) und der Fachgruppe „Fuzzy-Systeme und Soft-Computing“ der Gesellschaft für Informatik (GI), der vom 2.-4. Dezember 2009 im Haus Bommerholz bei Dortmund stattfindet.

Der GMA-Fachausschuss 5.14 „Computational Intelligence“ entstand 2005 aus den bisherigen Fachausschüssen „Neuronale Netze und Evolutionäre Algorithmen“ (FA 5.21) sowie „Fuzzy Control“ (FA 5.22). Der Workshop steht in der Tradition der bisherigen Fuzzy-Workshops, hat aber seinen Fokus in den letzten Jahren schrittweise erweitert.

Die Schwerpunkte sind Methoden, Anwendungen und Tools für

- Fuzzy-Systeme,
- Künstliche Neuronale Netze,
- Evolutionäre Algorithmen und
- Data-Mining-Verfahren

sowie der Methodenvergleich anhand von industriellen und Benchmark-Problemen.

Die Ergebnisse werden von Teilnehmern aus Hochschulen, Forschungseinrichtungen und der Industrie in einer offenen Atmosphäre 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.

ISSN 1614-5267

ISBN 978-3-86644-434-8

