A control sequence generator for fuzzy gain schedulers

Rainer Palm[1,3] Christiane Stutz[2] Thomas Runkler[1]

[1] Siemens AG Corporate Technology
Information and Communications
81730 Munich, Germany

[2] software design & management
Thomas-Dehler-Str. 27
81737 Munich, Germany

Abstract

The generation of the sequence of control inputs along a given state trajectory for a nonlinear system is described. A nonlinear system is linearized at predefined points in the product space of the states and control inputs and then approximated by local linear fuzzy models. Based on this approximation the system is controlled by a set of local linear Takagi-Sugeno fuzzy controllers. The local control laws designed for the error system incorporate both the desired and the actual state as well as the corresponding control input. Normally, the desired state is defined by the user but the related control input cannot always be calculated in a unique way especially for a non-square system. The proposed method generates the desired control inputs on the basis of the states and its derivatives using inverse fuzzy models of the system. In an optimization loop the control inputs are corrected by the analytical forward model of the nonlinear system.

Keywords Takagi Sugeno fuzzy systems, fuzzy gain scheduling, clustering, optimization, control sequences

1 Introduction

Gain scheduling is used for the control of nonlinear systems like airplanes or ships at certain operating regimes or along predefined reference trajectories. Gain scheduling along reference trajectories requires the choice of a finite number of operating points belonging to the reference state trajectory. For each of the operating points a local linear autonomous approximation of the original nonlinear system is constructed. Furthermore, a local linear controller is designed for each local linearized subsystem. For intermediate operating points the local linear controllers are scheduled (interpolated) which results in a global gain scheduling control law. Because of stability reasons the operating points on the reference trajectory are considered to be time-frozen and, thus, the nonlinear system is assumed to be a slowly time varying system [9, 11]. One efficient way of gain scheduling is to use a Takagi Sugeno (TS) fuzzy approximation of the nonlinear system [13, 14]. TS fuzzy gain schedulers provide a general and computationally efficient method for the interpolation of in advance calculated local control laws [8]. The fuzzy approximation is done on the basis of the system linearized around the desired state trajectory and the related control inputs. The local control laws are therefore designed for the error system which includes the desired and actual state on the one hand and the corresponding control input on the other hand. Since control tasks are mostly goal oriented, the desired state is normally given by the user but the related control input is not and has still to be calculated. This calculation cannot always be performed in a unique way because of

i) a smaller number of control input components than states,

ii) the implicit representation of the control input vector in the nonlinear state equation.

This problem has not been sufficiently addressed in the literature. Several authors deal with the desired output (or state) trajectory together with corresponding inputs but without showing the way of how to generate the corresponding control inputs [5, 6, 1, 11, 9]. The method proposed in this paper uses an inverse fuzzy model to generate the control input on the basis of the state and its derivative. In an optimization loop the so obtained control input is corrected by the analytical forward model of the nonlinear system.

2 The Fuzzy Gain Scheduling Problem

Given a nonlinear autonomous system

$$\dot{x} = f(x, u), \tag{1}$$

where $x \in R^n$ is the state vector, and $u \in R^m$ is the vector of control inputs. Let, furthermore, $x_d(t)$ be a known state reference trajectory and $u_d(t)$ the sequence of corresponding control inputs. Then the tracking control problem for (1) can be transformed into a stabilization control problem for the nonlinear non-autonomous error system

$$\dot{e}_x = f(e_x + x_d, e_u + u_d) - f(x_d, u_d), \tag{2}$$

---

[3] Corresponding author. Rainer.Palm@mchp.siemens.de

where $e_x = x(t) - x_d(t)$, $e_u = u(t) - u_d(t)$, and $\dot{e}_x = \dot{x}(t) - \dot{x}_d(t)$. Observe here that (2) became now non-autonomous because of the time dependence of the reference trajectory involved. The control problem of local stabilization can be solved by Lyapunov-linearization of (2) around $(0,0)$

$$\dot{e}_x = A(x_d,u_d)e_x + B(x_d,u_d)e_u, \tag{3}$$

where $A(x_d,u_d) = \left.\frac{\partial f(x,u)}{\partial x}\right|_{x_d,u_d}$ and $B(x_d,u_d) = \left.\frac{\partial f(x,u)}{\partial u}\right|_{x_d,u_d}$ are Jacobians. Equation (3) is equivalent to

$$\dot{x} = \dot{x}_d + A(x_d,u_d)(x - x_d) + B(x_d,u_d)(u - u_d), \tag{4}$$

where the equilibrium point of the latter system is now the trajectory $x_d(t)$ together with the related control input $u_d(t)$. An appropriate control law is

$$u = u_d + K(x_d,u_d)e_x \tag{5}$$

The goal of gain scheduling is to generate the gain $K(x_d,u_d)$ in a efficient way. Existing crisp gain scheduling design methods are described in [9], [11], [7]. Another option is to design a fuzzy gain scheduler which is based on TS fuzzy models [8]. The open loop fuzzy model of (1) is given as a set of c fuzzy rules of the form

$$\begin{aligned} \text{IF} \quad x_d &= LX^i \quad AND \quad u_d = LU^i \\ \text{THEN} \quad \dot{x} &= \dot{x}_d + A_i(x - x_d) + B_i(u - u_d). \end{aligned} \tag{6}$$

where $A_i = A(x^i,u^i)$, $B_i = B(x^i,u^i)$, (x_d,u_d) is an operating point , LX^i is the term set (e.g. SMALL, MEDIUM, etc.) of the i-th fuzzy region of the state space, LU^i is the term set of the i-th fuzzy region of the space of control inputs. The THEN-part of the above fuzzy rule contains a linear autonomous open loop model describing the open loop dynamics at the center of the i-th fuzzy region defined by the term sets LX^i, LU^i $(i = 1,\ldots,c)$.

The analytical expression corresponding to the set of fuzzy rules (6) is

$$\dot{x} = \dot{x}_d + \sum_i^c w^i(x_d,u_d)\cdot(A_i(x - x_d) + B_i(u - u_d)). \tag{7}$$

where $w^i(x_d,u_d) = LX^i(x_d) \wedge LU^i(u_d)$ is the firing strength of the i-th rule $(i = 1.\ldots,c)$, $\sum_i^c w^i(x_d,u_d) = 1$ and c is the number of fuzzy regions (submodels or clusters). Equation (7) approximates the "missing" linear autonomous open loop system (4). Observe here that the weights w^i are functions of the desired values x_d, u_d. The fuzzy gain scheduler is given as a set of c fuzzy rules

$$\begin{aligned} \text{IF} \quad x_d &= LX^j \quad AND \quad u_d = LU^j \\ \text{THEN} \quad u &= K_j(x - x_d) + u_d \end{aligned} \tag{8}$$

where $K_j = K(x^j,u^j)$ is the gain matrix for the j-th fuzzy region to be designed $(j = 1,\ldots,c)$. The analytical expression corresponding to the above rules (8) reads

$$u = \sum_j^c w^j(x_d,u_d)\cdot K_j(x - x_d) + u_d. \tag{9}$$

that is intended to stabilize (7) at any arbitrary operating point (x_d,u_d).

From (7) and (9) one obtains the linear but non-autonomous closed loop fuzzy model

$$\dot{x} = \dot{x}_d + \sum_i^c \sum_j^c w^i(x_d,u_d)w^j(x_d,u_d)\cdot(A_i + B_iK_j)(x - x_d). \tag{10}$$

while the K_j's have to be designed so that the closed loop system is stable.
In the following, however, we are not interested in designing the individual controller gains of (9) but in finding the appropriate $u_d(t)$ sequence.

3 Calculation of the direct and the inverse TS fuzzy model of the plant

If the fuzzy model is generated on the basis of input/output data the nonlinear system (1) is locally approximated at a point (x,u) by the affine fuzzy model

$$\dot{x} = A(x,u)x + B(x,u)u + con(x,u). \tag{11}$$

where $A(x,u) = \sum_i^c w^i(x,u)A_i$, $B(x,u) = \sum_i^c w^i(x,u)B_i$, $con(x,u) = \sum_i^c w^i(x,u)con_i$, and $A_i \in R^{n\times n}$, $B_i \in R^{n\times m}$, and $con_i \in R^n$. Observe here that the weights w^i are functions of the actual states x and their corresponding control values u.

The approximation of (1) at some operating point (x_d,u_d) is

$$\dot{x}_d = A(x_d,u_d)x_d + B(x_d,u_d)u_d + con(x_d,u_d). \tag{12}$$

where $A(x_d,u_d) = \sum_i^c w^i(x_d,u_d)A_i$, $B(x_d,u_d) = \sum_i^c w^i(x_d,u_d)B_i$, $con(x_d,u_d) = \sum_i^c w^i(x_d,u_d)con_i$.

For a given $\hat{x}_d = x_d$ and a measured $\hat{u}_d$ we obtain the approximation

$$\dot{\hat{x}}_d = A(x_d,\hat{u}_d)x_d + B(x_d,\hat{u}_d)\hat{u}_d + con(x_d,\hat{u}_d). \tag{13}$$

where $A(x_d,\hat{u}_d) = \sum_i^c w^i(x_d,\hat{u}_d)A_i$, $B(x_d,\hat{u}_d) = \sum_i^c w^i(x_d,\hat{u}_d)B_i$, $con(x_d,\hat{u}_d) = \sum_i^c w^i(x_d,\hat{u}_d)con_i$.

Although the number of inputs m is normally smaller than or equal to the number of states n there is in principle a unique solution for the *inverse* of (1) within a restricted region of the product space of $x, u, \dot{x}$

$$u = g(x, \dot{x}) \tag{14}$$

However, in many cases the explicit representation of the nonlinear inverse model (14) is not possible. Therefore u is approximated by the local linear fuzzy model

$$u = \tilde{A}(x, \dot{x})x + \tilde{B}(x, \dot{x})\dot{x} + c\tilde{o}n(x, \dot{x}) \tag{15}$$

where $\tilde{A}(x, \dot{x}) = \sum_i^c w^i(x, \dot{x})\tilde{A}_i$, $\tilde{B}(x, \dot{x}) = \sum_i^c w^i(x, \dot{x})\tilde{B}_i$, $c\tilde{o}n(x, \dot{x}) = \sum_i^c w^i(x, \dot{x})c\tilde{o}n_i$ and $\tilde{A}_i \in R^{m\times n}$, $\tilde{B}_i \in R^{m\times n}$, and $c\tilde{o}n_i \in R^m$ have to be determined by means of selected training data sets. For a given $\hat{x}_d = x_d$ and a measured $\hat{\dot{x}}_d$ the corresponding $\hat{u}_d$ is approximated by

$$\hat{u}_d = \tilde{A}(x_d, \hat{\dot{x}}_d)x_d + \tilde{B}(x_d, \hat{\dot{x}}_d)\hat{\dot{x}}_d + c\tilde{o}n(x_d, \hat{\dot{x}}_d) \tag{16}$$

where $\tilde{A}(x_d, \hat{\dot{x}}_d) = \sum_i^c w^i(x_d, \hat{\dot{x}}_d)\tilde{A}_i$, $\tilde{B}(x_d, \hat{\dot{x}}_d) = \sum_i^c w^i(x_d, \hat{\dot{x}}_d)\tilde{B}_i$, $c\tilde{o}n(x_d, \hat{\dot{x}}_d) = \sum_i^c w^i(x_d, \hat{\dot{x}}_d)c\tilde{o}n_i$.

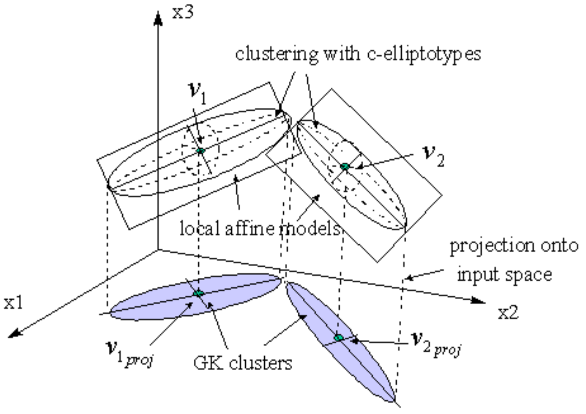

Figure 1: Clustering and model computation

The principle modeling steps are

1. Choose an appropriate number of data clusters c
2. Cluster the $(x, \dot{x}, u)$ data tripels in the product space using c-elliptotypes clusters [10, 4]
3. Project the c-elliptotype clusters onto the $(x, \dot{x})$ input-space and change the projected clusters into Gustafson-Kessel clusters (GK) [3]
4. Calculate the local linear (affine) models in the product space by using the GK clusters from step 2.

Figure 1 shows a 3-dimensional example where the product space and the input space are defined by $(x, \dot{x}, u) = (x1, x2, x3)$, and $(x, \dot{x}) = (x1, x2)$, respectively. In this example two clusters are shown with their centers (v_1, v_2) in the product space and (v_{1proj}, v_{2proj}) in the input space.

The detailed algorithms are described in [12].
The final result of the steps described above is the fuzzy model (15) which is evaluated by means of test data sets.

4 Correction of $\hat{u}_d$ by the forward analytical nonlinear model

Recall that we need to calculate a control value u_d for a given state x_d and its derivative $\dot{x}_d$. From the inverse fuzzy model (16) only a rough approximation of the desired control valued $\hat{u}_d$ can be obtained. The size of the approximation error $\Delta u_d = \hat{u}_d - u_d$ strongly depends on the quality of the inverse fuzzy system (16). A too rough approximation may lead to a large error of the state x. This error could be eliminated by an integrator in the control loop but at the expense of an increasing order of the system to be controlled. The method presented here introduces an optimization loop to correct $\hat{u}_d$ with the help of the known analytical nonlinear forward model of the plant.

In most applications the trajectory x_d is given and its derivative $\dot{x}_d$ can be calculated but the corresponding u_d can not. Let the estimate $\hat{u}_d$ calculated by (16) be sufficiently close to u_d. The regarding estimate $\hat{\dot{x}}_d$ is computed from the given x_d and $\hat{u}_d$

$$\hat{\dot{x}}_d = f(x_d, \hat{u}_d) \tag{17}$$

In an optimization loop the norm of the difference $\Delta\dot{x}_d = \hat{\dot{x}}_d - \dot{x}_d$ is minimized and, with this, the norm of the difference $\hat{u}_d - u_d$. If the norm of the difference $\Delta\dot{x}_d = \hat{\dot{x}}_d - \dot{x}_d$ is smaller than a defined threshold ϵ or if a maximum number of optimization steps is reached then the optimization is stopped. The optimization procedure converges for

$$\frac{d}{d\tau}\hat{u}_d = -\gamma\tilde{B}(x_d, \hat{\dot{x}}_d)\Delta\dot{x}_d \tag{18}$$

where $\gamma > 0$ is the optimization rate and τ is the "virtual" time parameter. To prove this we define the Lyapunov function

$$V = \frac{1}{2}\Delta\dot{x}_d^T\Delta\dot{x}_d \tag{19}$$

and the change of V

$$\frac{d}{d\tau}V = \Delta\dot{x}_d^T\frac{d}{d\tau}\hat{\dot{x}}_d \tag{20}$$

Since x_d is fixed so $\frac{d}{d\tau}x_d = 0$. Furthermore we assume the matrices $A(x_d, \hat{u}_d)$, $B(x_d, \hat{u}_d)$, and $con(x_d, \hat{u}_d)$ to be constant during the optimization process. Therefore, we make the following assumptions

$$\begin{aligned} \frac{d}{d\tau}A(x_d, \hat{u}_d) &= 0 \\ \frac{d}{d\tau}B(x_d, \hat{u}_d) &= 0 \\ \frac{d}{d\tau}con(x_d, \hat{u}_d) &= 0 \end{aligned} \tag{21}$$

Then we obtain from (13)

$$\frac{d}{d\tau}\hat{\dot{x}}_d = B(x_d, \hat{u}_d)\frac{d}{d\tau}\hat{u}_d \tag{22}$$

In order to guarantee stability in the optimization process we choose

$$\frac{d}{d\tau}\hat{u}_d = -\gamma \tilde{B}(x_d, \hat{\dot{x}}_d)\Delta\dot{x}_d \tag{23}$$

where $\gamma > 0$.

Substitution of (23) into (20) yields

$$\frac{d}{d\tau}V = -\gamma\Delta\dot{x}_d^T B(x_d, \hat{u}_d)\tilde{B}(x_d, \hat{\dot{x}}_d)\Delta\dot{x}_d \tag{24}$$

Comparing (16) with (13) we find that $\tilde{B}(x_d, \hat{\dot{x}}_d)$ is the pseudoinverse of $B(x_d, \hat{u}_d)$ and so $B(x_d, \hat{u}_d)\tilde{B}(x_d, \hat{\dot{x}}_d) \geq 0$. With this $dV/d\tau$ is semidefinite. Incorporating the analytical forward model $\hat{\dot{x}}_d = f(x_d, \hat{u}_d)$ we obtain for the change of $\hat{u}_d$ in terms of a fuzzy approximation

$$\frac{d}{d\tau}\hat{u}_d = -\gamma\sum_j^c \tilde{w}^j(x_d, \hat{\dot{x}}_d)\tilde{B}_j(f(x_d, \hat{u}_d) - \dot{x}_d). \tag{25}$$

The correction scheme for u_d is shown in Fig. 2 It has to be emphasized that (15) approximates the inverse system (14) only well enough in the regions of training data $(x, \dot{x}, u)$. A unique representation $\dot{x} = f(x, u) \leftrightarrow u = g(x, \dot{x})$ may therefore only be possible in regions covered by these data. Even in this case $V > 0$ may hold as $dV/d\tau \to 0$. That is, $\Delta\dot{x}$ does not go to zero as $d\hat{u}_d/d\tau \to 0$ but the components of $\Delta\dot{x}$ reach some level of compromise.

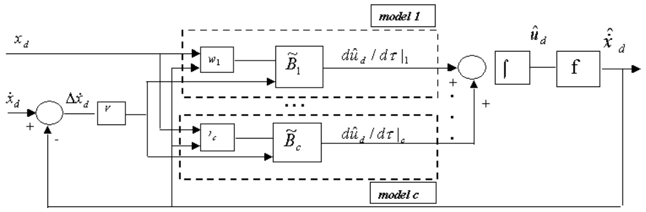

Figure 2: Correction scheme for u_d

5 Simulation results

SISO case The first example deals with the SISO system

$$\ddot{x} = -0.5\cdot\dot{x}^2 - 0.1\cdot x + e^u + \sin u \tag{26}$$

from which it is clear that the control value u cannot be explicitly calculated. In order to obtain a corresponding TS fuzzy model the system (26) is driven by

$$u = 0.5\cdot\sin(1.5t + 2) - 0.2\cdot\cos(2t) \tag{27}$$

from which 1000 data matrices $(x, \dot{x}, \ddot{x}, u)$ are gathered. Both the direct fuzzy model (11) and the inverse fuzzy model (15) are built on the basis of 30 clusters (submodels).

In this experiment the desired trajectory is chosen as

$$x_d = 3\cdot\sin(0.3t - 6) \tag{28}$$

for which the corresponding u_d has to be generated. Figure 3 shows the evolution of $(x_d, \dot{x}_d, \ddot{x}_d)$ and the acceleration errors $\Delta\ddot{x}_d = \ddot{x}_d - \hat{\ddot{x}}_d$. The error $\Delta\ddot{x}_d$ is in the range of the $\ddot{x}_d$ and is therefore inacceptably high. Figure 4 depicts the corresponding $\hat{u}_d$ values.

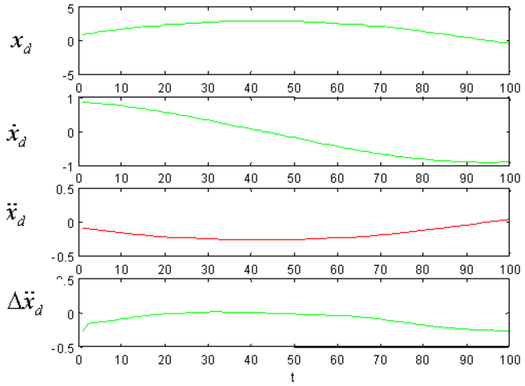

Figure 3: Accelleration errors, no correction, SISO case

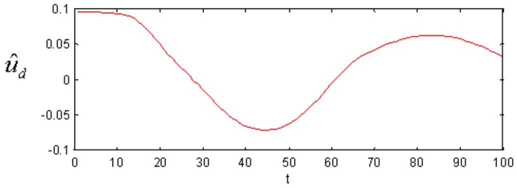

Figure 4: Control values, no correction, SISO case

Figure 5 shows the evolution of $(x_d, \dot{x}_d, \ddot{x}_d)$ and the accelleration errors $\Delta\ddot{x}_d = \ddot{x}_d - \hat{\ddot{x}}_d$ in the correction case. Observe here that the accelleration error $|\Delta\ddot{x}_d|$ is restricted to $\epsilon = 0.01$. That is, the generated $\hat{u}_d$ is very close to its ideal value. The chattering effect is due to

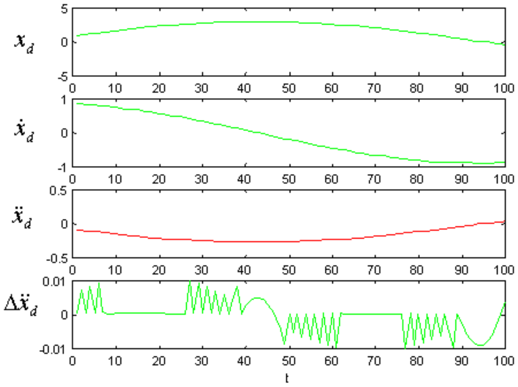

Figure 5: Accelleration errors, with correction, SISO case

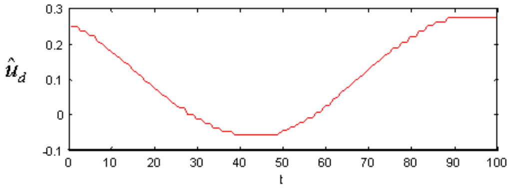

Figure 6: Control values, with correction, SISO case

the fact that inside the ϵ layer the error $\Delta\ddot{x}_d$ may increase without any correction. As $|\Delta\ddot{x}_d| > \epsilon$ then the correction algorithm starts and, with the present optimization rate $\gamma = 1$, reduces $|\Delta\ddot{x}_d|$ very fast. Chattering can be damped by reducing the optimization rate. Figure 6 depicts the corresponding control input $\hat{u}_d$.

MIMO case The second example deals with a two-link robot arm described by the following nonlinear equations [8]

$$M(q) \cdot \ddot{q} = -N(q, \dot{q}) + u. \tag{29}$$

In this equation

$$M(q) = \begin{pmatrix} (m_1 + m_2)l_1^2 & m_2 l_1 l_2 \cos(q_1 - q_2) \\ m_2 l_1 l_2 \cos(q_1 - q_2) & m_2 l_2^2 \end{pmatrix}$$

and

$$N(q, \dot{q}) = (n_1, n_2)^T ,$$

where

$$n_1 = \dot{q}_2^2 m_2 l_1 l_2 \sin(q_1 - q_2) - (m_1 + m_2) g l_1 \sin q_1 + K_{q1}\dot{q}_1$$

$$n_2 = -\dot{q}_1^2 m_2 l_1 l_2 \sin(q_1 - q_2) - m_2 g l_2 \sin q_2 + K_{q2}\dot{q}_2$$

K_{q1} and K_{q2} are damping coefficients, $q = (q_1, q_2)^T$ is the vector of joint angles, $u = (u_1, u_2)^T$ is the control input vector. The mechanical parameters of the system are

$$m_1 = 3kg,\ m_2 = 3kg,\ l_1 = 0.2m,\ l_2 = 0.2m$$

$$K_{q1} = 2\frac{kgm^2}{s},\ K_{q2} = 3\frac{kgm^2}{s}$$

To obtain a corresponding TS fuzzy model the system (29) is driven by

$$u_1 = 20 \cdot \sin 10t; \quad u_2 = -30 \cdot \sin 25t \tag{30}$$

from which 1000 data points $(q_1, \dot{q}_1, \ddot{q}_1, q_2, \dot{q}_2, \ddot{q}_2, u_1, u_2)$ are gathered. The direct and the inverse fuzzy model, respectively, are built on the basis of 30 clusters. In the experiment the desired trajectory $q(t)$ is chosen as

$$q_1 = \sin 10t; \quad q_2 = \sin 10t \tag{31}$$

for which the corresponding u_d has to be generated.

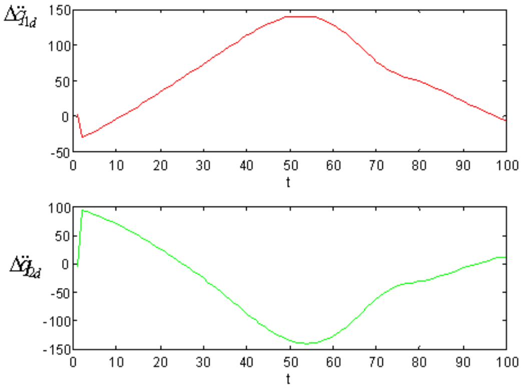

Figure 7: Accelleration errors, no correction, MIMO case

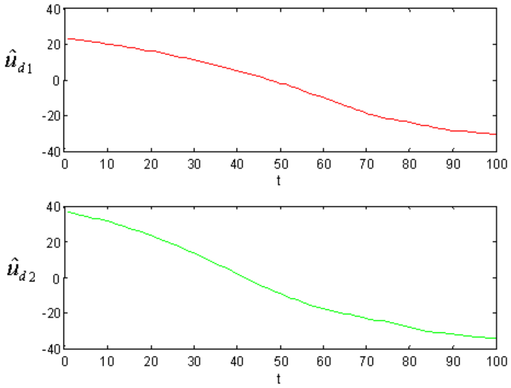

Figure 8: Control values, no correction, MIMO case

Figure 7 shows very high acceleration errors for the non correction case. The corresponding control values are depicted in Fig. 8. With the help of the optimization correction loop the acceleration errors are reduced significantly (see Fig. 9). Observe here that the corresponding control values (Fig. 10) do not deviate that much from those of the non correction case. That is, small deviations in control values may cause very large acceleration errors. Hence, an accurate generation of u_d may inrease the control performance considerable.

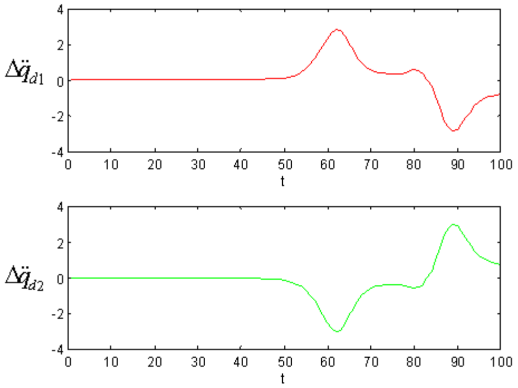

Figure 9: Accelleration errors, with correction, MIMO case

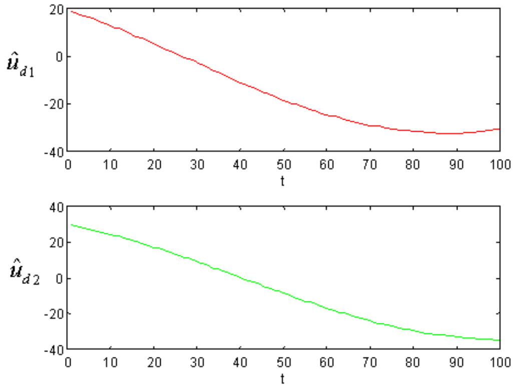

Figure 10: Control values, with correction,MIMO case

6 Conclusions

The generation of control sequences for a fuzzy gain scheduler on the basis of a Takagi-Sugeno (TS) fuzzy model was discussed. The modeling of the system was done by a combination of c-elliptotypes clustering and Gustafson-Kessel clustering of input output data and a subsequent identification of local linear and affine TS models. The control input to be applied incorporates (i) an feedback control input part that stabilizes the system around the desired state trajectory and (ii) a feedforward control input part that brings the system to the vicinity of the desired state trajectory. The method presented here used an inverse TS fuzzy model to generate the feedforward control input on the basis of the state and its derivative. In an optimization loop the feedforward control input is corrected by the analytical forward model of the nonlinear system. Experimental results with a SISO and a MIMO system show the high performance of the optimization method.

References

[1] Babuška R. *Fuzzy Modeling and Identification.* PhD thesis, Technische Universiteit Delft, 1997.

[2] Bezdek J.C. Pattern recognition with Fuzzy Objective Function Algorithms. Plenum Press, New York, 1981

[3] Gustafson D.E., Kessel W.C. Fuzzy Clustering with a Fuzzy Covariance Matrix Proc. IEEE CDC, Jan. 1979, pp.761-766

[4] Hoeppner F., Klawonn F., Kruse R., Runkler T.A. Fuzzy Cluster Analysis - Methods for Image Recognition, Classification, and Data Analysis. John Wiley & Sons, 1999

[5] Johansen T.A., Murray-Smith R. The Operating Approach to Nonlinear Modelling and Control. In Multiple Model Approaches to Modelling and Control ed. by Murray-Smith and Johansen. Taylor and Francis London 1997, pp. 3-72

[6] Johansen T.A., Hunt K.J., Gawthrop P.J. Transient Performance, Robustness and Off-Equilibrium Linearisation in Fuzzy Gain Scheduled Control. In Advances in Fuzzy Control, ed. by Driankov D. and Palm R., Physika Verlag Heidelberg, New York 1998, pp. 357-375

[7] Nichols R.A. Reichert R.T., Rugh W.J. Gain Scheduling for H-Infinity Controllers: A Flight Control Example *IEEE Transactions on Control Systems Technology* **1**(2)(1993)69–79.

[8] Palm R., Driankov D., Hellendoorn H. Model Based Fuzzy Control. Springer-Verlag, Berlin New York Heidelberg, 1996

[9] Rugh W.J. Analytical Framework for Gain Scheduling *IEEE Control Systems Magazine* **11**(1)(1991)79–84.

[10] Runkler T.A., Palm R. Identification of Nonlinear Systems using regular fuzzy c-elliptotypes clustering. IEEE Conference on Fuzzy Systems, New Orleans, Sept. 1996, pp. 1026-1030

[11] Shamma J.S. *Analysis and Design of Gain Scheduled Control Systems* PhD Thesis No. LIDS-TH-1770 Lab. for Information and Decision Sciences MIT Cambridge MA 02139 1988.

[12] Stutz C. Fuzzy Clustering for Functional Approximations. Internal report, Siemens AG Munich 1998

[13] Takagi T., Sugeno M. Fuzzy Identification of Systems and Its Applications to Modeling and Control. *IEEE Trans. on Syst., Man, and Cyb.*Vol. SMC-15. No.1 January/February 1985, pp.116-132

[14] Tanaka K., Sugeno M. Stability Analysis and Design of Fuzzy Control Systems. *Fuzzy Sets and Systems* 45 (1992) North-Holland, pp. 135-156