LOGICAL DESIGN OF RELATIONAL FUZZY CONTROLLERS

VOLKER KREBS
University of Karlsruhe, Institut für Regelungs- und Steuerungssysteme; Kaiserstraße 12, D - 76131 Karlsruhe, Germany

ROLF BÖHM
Robert Bosch GmbH. D - 77815 Bühl, Germany

ABSTRACT

A new direct design method for fuzzy controllers will be presented in this paper. It combines the lucid strategy of compensation used in conventional control systems and the straight forward determination of a fuzzy controller by means of the chain rule of propositional logic. The quality and potential of the proposed method is demonstrated by the design of a fuzzy controller for an automatic clutch management system for motor-vehicles.

KEYWORDS: design of fuzzy controllers, qualitative models, fuzzy relations, relational fuzzy controller, propositional logic, fuzzy automatic clutch

INTRODUCTION

Differing from the usual way of fuzzy controller design by mapping the control strategy of a human operator into a set of fuzzy rules, in this contribution a direct design method is proposed that shows a certain relationship to conventional compensation control. It is assumed that the process to be controlled can be qualitatively described by a set of linguistic rules that corresponds to a fuzzy relation **S** (fig. 1). Next, the desired behaviour of the closed-loop system is represented in terms of fuzzy logic, too. This yields a relation **G** containing the series connection of the unknown fuzzy controller **R** and the plant **S** which is

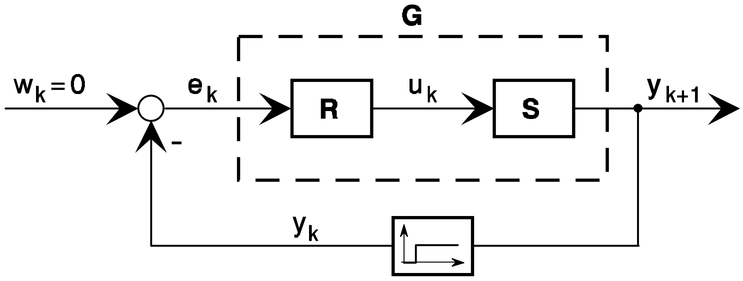

Figure 1. Qualitative model of the closed-loop system

represented by the relation equation $\mathbf{G} = \mathbf{R} \circ_T \mathbf{S}$ with $\circ_T$ as supremum-t-operator [1]. Although the direct solution of this relation equation is possible in special cases [2], there are some practical drawbacks due to the fact that the resulting controller relation **R** frequently cannot be directly represented by a set of rules with continuous membership functions.

Thus another new method is proposed that is based on the chain-rule of propositional logic. It comprises the possibility either to calculate **R** by means of the rule sets for **G** and **S** or their respective relations. This controller design method will be illustrated by a simple example.

In the last section the design of a fuzzy controller for an automatic clutch will illustrate the potential of this approach for industrial applications.

2. FUZZY RELATIONS FOR KNOWLEDGE REPRESENTATION

In this section the method which is used in this article for knowledge representation by means of qualitative rules is briefly outlined. Assume that the relation between the input variable X and the corresponding output variable Y of a system is given by a set of *n* fuzzy rules

$$\mathbf{R}_i\text{: If X is } \mathbf{A}_i \text{ then Y is } \mathbf{B}_i \qquad i = 1, ..., n. \tag{1}$$

Fuzzy sets **A**, defined on the multidimensional input domain *X* and fuzzy sets **B**, defined on the multidimensional output domain *Y* are assigned to the input variable X and the output variable Y, respectively. A relation **R** has to be found that represents this set of rules and enables the determination of a fuzzy set $\mathbf{B}_0$, assigned to the output variable Y according to the particularly given set $\mathbf{A}_0$, assigned to the input variable X. Each rule $\mathbf{R}_i$ is to be represented by a relation which in boolean logic embodies the situation: Premise **a** is fulfilled (true) and **a** implies **b** $(\mathbf{a} \Rightarrow \mathbf{b})$. That is exactly the inference rule *modus ponens* of propositional logic; it can easily be transformed in the following way:

$$\mathbf{r}\text{: } \mathbf{a} \wedge [\mathbf{a} \Rightarrow \mathbf{b}] = \mathbf{a} \wedge \left[(\mathbf{a} \wedge \mathbf{b}) \vee (\bar{\mathbf{a}} \wedge 1)\right] = (\mathbf{a} \wedge \mathbf{b}). \tag{2}$$

In fuzzy logic this idea is generalized and therefore the resulting inference rule is called *generalized modus ponens* [3]. Using the corresponding fuzzy operator in (2) each rule in (1) may finally be represented in fuzzy terms by the relation

$$\mathbf{R}_i\text{: } \mathbf{A}_i \rightarrow \mathbf{B}_i := \mathbf{A}_i \otimes_T \mathbf{B}_i, \tag{3}$$

with $\otimes_T$ as the cartesian product. Hence, a set of rules is mapped into the union of all relations to express the union of all matching rules. Connecting an input $\mathbf{A}_0$ with the union of all relations of the rule set,

$$\mathbf{B}_0(y_1,...,y_t) = \mathbf{A}_0(x_1,...,x_r) \circ_T \left(\bigcup_{\substack{\perp \\ i=1,n}} \mathbf{A}_i \otimes_T \mathbf{B}_i \right) = \mathbf{A}_0 \circ_T \mathbf{R} \tag{4}$$

is obtained. $\mathbf{B}_0$ is the fuzzy output of those rules that are activated by $\mathbf{A}_0$. It is guaranteed that this output does not contradict the rule set. Since all fuzzy sets are continuous, the relation **R** is continuous, too. It can be shown that $\mathbf{B}_0$ may also be considered as the upper set of the fuzzy output of all rules that are activated by $\mathbf{A}_0$ [4]. The activation of a rule $\mathbf{R}_i$ is the level of matching between the input variable $\mathbf{A}_0$ and the corresponding premise $\mathbf{A}_i$.

3. CONTROLLER DESIGN

In the problem of controller design there are three rule sets to be represented by fuzzy relations, namely **G**, **S**, and **R** (see fig.1). It is our goal to determine the controller relation **R** by using **G** and **S**. First the expression $\mathbf{G} = \mathbf{R} \circ_T \mathbf{S}$ has to be explained. As shown in fig.1 $\mathbf{R} \circ_T \mathbf{S}$ represents the series connection of the controller and the process, both mapped into fuzzy relations according to the union expression in (4); the operator $\circ_T$ connects the relations like the chain rule of propositional logic does. Thus two rules, one representing the controller behaviour and the other the plant´s one

$$\begin{aligned} &\mathbf{R}_i\!: \mathbf{E}_i \rightarrow \mathbf{U}_i := \mathbf{E}_i \otimes_T \mathbf{U}_i\,, \\ &\mathbf{S}_i\!: \mathbf{U}_i \rightarrow \mathbf{Y}_i := \mathbf{U}_i \otimes_T \mathbf{Y}_i \end{aligned} \tag{5}$$

lead to the following relation **G** of their series connection in the closed loop

$$\mathbf{G}_i\!: \mathbf{E}_i \rightarrow \mathbf{Y}_i := \mathbf{E}_i \otimes_T \mathbf{Y}_i\,. \tag{6}$$

3.1 Synthesis of R by propositional logic

In this section the method to design a fuzzy controller based on propositional logic is outlined. From (5) and (6) the meaning of the series connection of $\mathbf{R}_i$ and $\mathbf{S}_i$ as well as the way to obtain the rule $\mathbf{R}_i$ appears clear. A very simple example will demonstrate the way this method works. Suppose there are two rules:

$\mathbf{S}_i$**:** If **BRAKE** is **APPLIED** then **CAR STOPS** (7)

as the description of a process which is assumed to be a car and

$\mathbf{G}_i$**:** If **LIGHT** is **RED** then **CAR STOPS** (8)

as a desription of the desired behaviour of the car approaching a traffic light. The control rule $\mathbf{R}_i$ is to be determined. For experienced drivers this won't be a problem, but for unexperienced operators of a plant it could be a problem to get the right control rule from (7) and (8) in particular when several rules are involved. If (8) is fulfilled, in the same way the inverse

$\mathbf{G}_i^{-1}$**:** If **CAR STOPS** then **LIGHT** is **RED** (9)

should be true in case of red light. Applying the chain rule to (7) and (9) the resulting rule is obviously given by

$\mathbf{S}_i \circ \mathbf{G}_i^{-1}$: If **BRAKE** is **APPLIED** then **LIGHT** is **RED** (10) which describes the inverse behaviour of a driver acting as the car´s controller compared to a usual traffic situation. However, to get a control rule that maps the driver´s behaviour, the input and output variables have to be interchanged; thus we obtain

$$\mathbf{R}_i = \left(\mathbf{S}_i \circ \mathbf{G}_i^{-1}\right)^{-1} : \text{If } \mathbf{LIGHT} \text{ is } \mathbf{RED} \text{ then } \mathbf{BRAKE} \text{ is } \mathbf{APPLIED}, \tag{11}$$

which really represents the behaviour of many drivers and as such is a fitting control rule for this particular simple system.

In a more general practical situation, assume a set of *n* rules $\mathbf{S}_i$ for the qualitative modelling of the plant is given as well as *m* rules $\mathbf{G}_j$ for the desired behaviour of the controlled system. Now, not more than n x m rules $\mathbf{R}_{ij}$ for the controller have to be determined. They are obtained using the chain rule by the premises of **G** and the premises of **S**; the latter constitute the conclusions of the controller rule set. It can be proven [4] that this chain rule approach results in a controller which warrants the desired behaviour **G** as a subset of the calculated set $\mathbf{G}_{\text{chain}} = \mathbf{R}_{\text{chain}} \circ_T \mathbf{S}$. In case of no intersection of the membership functions in the premises of **S** the controller even leads to $\mathbf{G}_{\text{chain}} = \mathbf{G}$.

It should be noticed that using this method no relations and no calculations with relations are required, because the whole job is done based on rules. Therefore the result is a rule set, which can directly be implemented as a fuzzy controller. However, for computerized calculation in cases of larger rule sets nevertheless it is more efficient to deal with relations. The way how the controller relation **R** can be calculated is already indicated in (10) and (11).

3.2 Synthesis of R by relational equations

A computationally efficient way to determine a proper rule set **R** from **S** and **G** which lead to the same result as in section 3.1 is constituted as follows:

In order to obtain relational matrices for **S** and **G** which are easy to handle, a rough discretization of the respective membership functions of **S** and **G** by means of the peak pattern method is proposed. Then the set of rules $\mathbf{S}_i$ and $\mathbf{G}_j$ can be represented by relational matrices **S** and **G** where the linguistic input and the corresponding output variable of each rule is represented by the number "1". All other elements of the matrices are zeros. Once the relational matrices **S** and **G** are given it is comparatively easy to calculate the controller relation **R** by

$$\mathbf{R} = \left(\mathbf{S} \circ \mathbf{G}^{-1}\right)^{-1}. \tag{12}$$

The resulting relation **R** has to be retransformed to a valid rule set; as there is a given order of linguistic values for its in- and output variables, once defined in **S** and **G**, it is easy to obtain this rule set. Each "1" in the relation **R** corresponds to a valid rule.

4. DESIGN OF A FUZZY CONTROLLER FOR AN AUTOMATED CLUTCH

In automobiles the degree of automation is increased on the one hand to reach a higher level of security and on the other hand to make driving more comfortable. The system for automatic clutch management, which will be presented in this section has been developed to enable driving without using the clutch-pedal. The driver only has to select the gear with the shift lever. The controller controls the clutch travel. The car works with a conventional shift transmission and the dryplate clutch which is engaged or disengaged by an electro-hydraulic system (fig. 2).

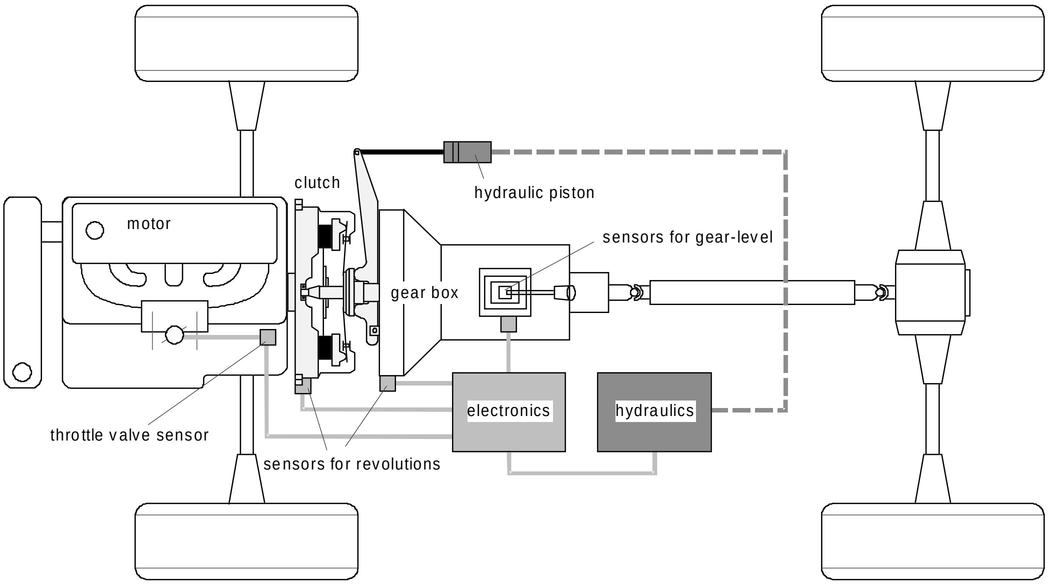

Figure 2. automobile with sensors and actuators for the automated clutch

In this special application a fuzzy controller only for the start of the car has been developed. Shifting up and down already has been solved by means of open loop control.
There are several restrictions to pay attention to: The accelerations to the driver have to remain under a certain level and the car may not move jerky. The most important item in designing an appropriate controller is to take the driver´s starting strategy into consideration, because he should always feel save and should not have any problems using a car with such an equipment. For the development of the system the following steps are executed:

1. quantitative analysis of the process to be controlled (as far as possible)
2. collecting data to get qualitative knowledge about the process (driving tests)
3. specification of the rule base for the process behaviour
4. specification of the rule base for the desired system behaviour
5. calculation of the controller rule base
6. simulation of the closed loop system
7. implementation and driving tests

In this paper we will deal in particular with the topics 2 (process behaviour) and 3 (desired system behaviour).

First, for knowledge acquisition purposes several start-ups using a normally equipped car (mechanic clutch) were carried out. The tests showed considerable differences of the starting strategies depending on the respective driver. However, the results given in fig. 3 allow a clear assignment of the driver´s intention to the throttle position; this figure serves as one key information for obtaining the qualitative process model.
The start-up in fig. 3 develops as follows: The driver opens the throttle valve and starts closing the clutch slowly. Thus, the motor runs up to a certain revolution ω_{Mot} which

depends on α. Then the driver leaves s_K at this position until the revolution ω_{Mot} equals the revolution ω_{Gea} of the gear unit. At this point the clutch sticks and the start-up is finished; the clutch is engaged whitch means that s_K is turned to zero. A deeper process analysis shows that closing the clutch at constant throttle valve angle leads to a decreasing change of ω_{Mot}.

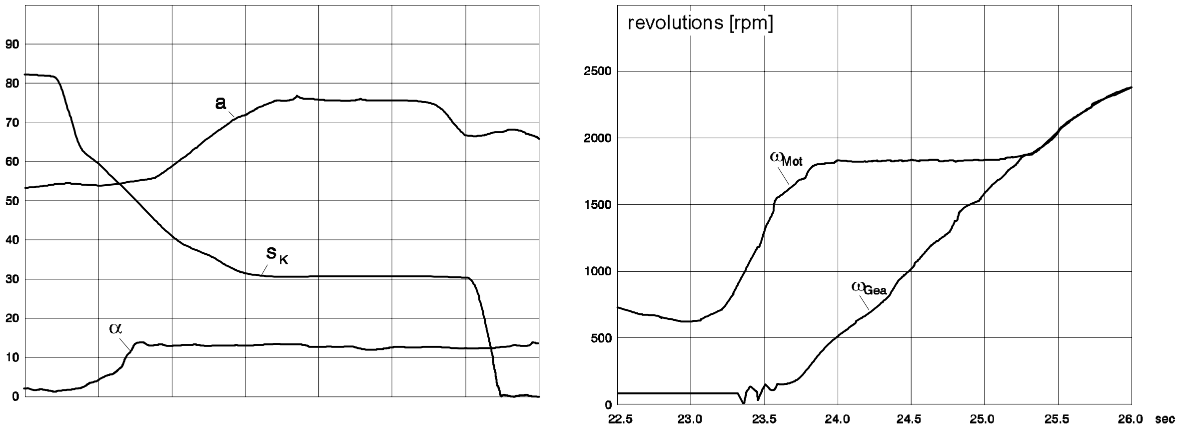

Figure 3. Start-up: 1. acceleration **a** of the car, clutch travel s_K, throttle valve angle α
2. angular frequency of the motor ω_{Mot} and of the gear ω_{Gea}

The second task is building the rule base of the desired system behaviour where the following specifications are to be met:

1. There should not be any jumps in the characteristic of the acceleration a.
2. ω_{Mot} has to run up without overshoot; thus $\dot{\omega}_{Mot}$ has to be taken into account, too.
3. The stationary value $\omega_{Mot_{stat}}$ depends on the position of the throttle valve angle α.

Therefore the ideal start-up should look as shown in fig .4: The continuous characteristic of the clutch torque M_{cl} guarantees a smooth acceleration and the motor runs from the no-load-speed ω_{nl} up to $\omega_{Mot_{stat}}$ without any overshoot.

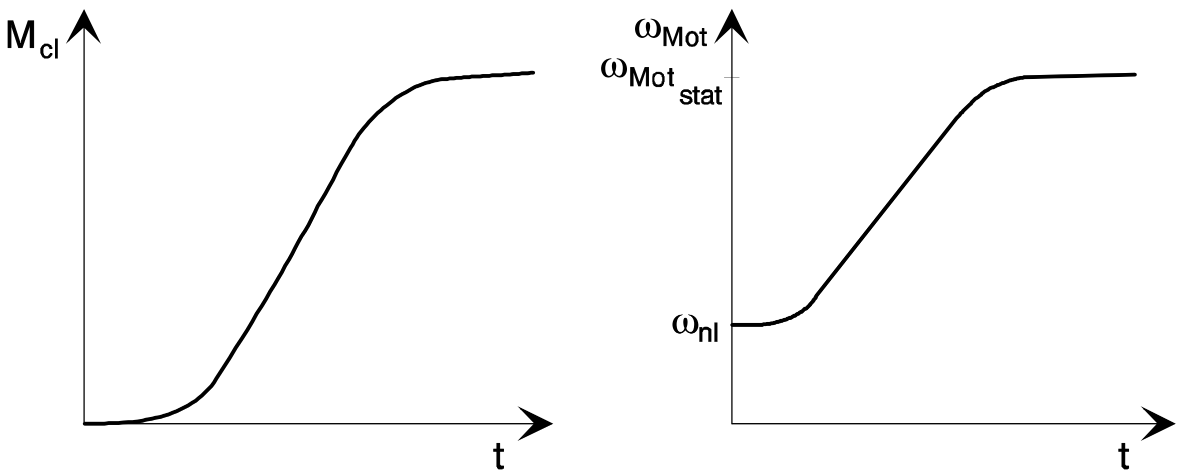

Figure 4. ideal start-up

From this desired behaviour the following structure of the closed-loop system will be obtained (fig. 5): The system's setpoint is $\omega_{\mathrm{Mot\,stat}}(\alpha)$, the system output is ω_{Mot} and the difference between these two revolutions represents the input of the fuzzy controller. Its output is $\dot{M}_{\mathrm{cl}}$ which is also the input to the process.

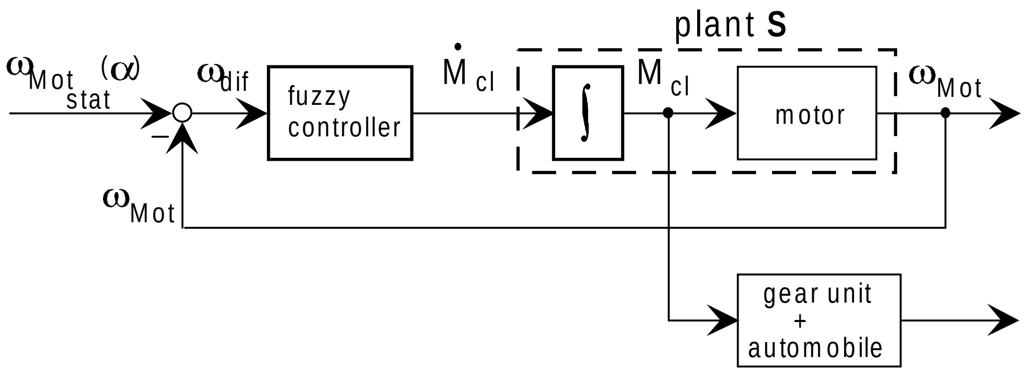

Figure 5. Structure of the closed-loop system

Thus the relational equation governing the controlled system is represented by

$$\mathbf{G}\left(\omega_{\mathrm{dif}}, \dot{\omega}_{\mathrm{Mot}}; \ddot{\omega}_{\mathrm{Mot}}\right) = \mathbf{R}\left(\omega_{\mathrm{dif}}, \dot{\omega}_{\mathrm{Mot}}; \dot{M}_{\mathrm{cl}}\right) \circ \mathbf{S}\left(\dot{M}_{\mathrm{cl}}; \ddot{\omega}_{\mathrm{Mot}}\right). \tag{13}$$

The behaviour of the plant and the desired behaviour of the system are mapped into 15 and 55 rules, respectively. The 55 rules for the desired behaviour are structured in 5 times 11 rules. As an illustrative example two rules of each of them are

plant:

S1: If $\dot{M}_{\mathrm{cl}}$ is *positive small* then $\ddot{\omega}_{\mathrm{Mot}}$ is *negative small.*

S15: If $\dot{M}_{\mathrm{cl}}$ is *negative huge* then $\ddot{\omega}_{\mathrm{Mot}}$ is *positive very big*

desired behaviour:

G1: If ω_{dif} is *negative small* and $\dot{\omega}_{\mathrm{Mot}}$ is *positive very small* then $\ddot{\omega}_{\mathrm{Mot}}$ is *negative small.*

G55: If ω_{dif} is *positive big* and $\dot{\omega}_{\mathrm{Mot}}$ is *negative small* then $\ddot{\omega}_{\mathrm{Mot}}$ is *positive very big.*

From the whole set of rules we get two relations **S** and **G** and are able to solve equation (13) by

$$\mathbf{R} = \left(\mathbf{S}\,\mathrm{o}\,\mathbf{G}^{-1}\right)^{-1}. \tag{14}$$

The relation **R** can be represented by 63 rules for the fuzzy controller. Two rules of the controller are given by

R1: If ω_{dif} is *negative small* and $\dot{\omega}_{\mathrm{Mot}}$ is *positive very small* then $\dot{M}_{\mathrm{cl}}$ is *positive small.*

R63: If ω_{dif} is *positive big* and $\dot{\omega}_{\mathrm{Mot}}$ is *negative small* then $\dot{M}_{\mathrm{cl}}$ is *negative big.*

The characteristics of the closed-loop system are shown in fig.6 which is based on a simulation using diffential equations and the motor nonlinear characteristics.

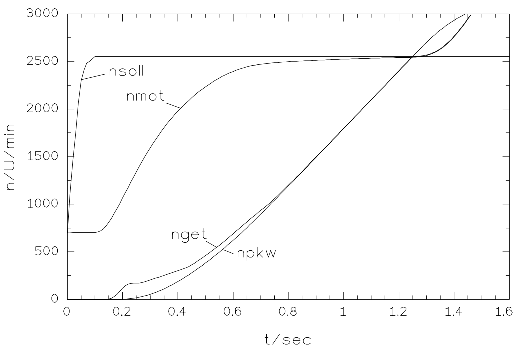

Figure 6.1. Speeds in cloosed-loop control, throttle angle α from 0° to 50°

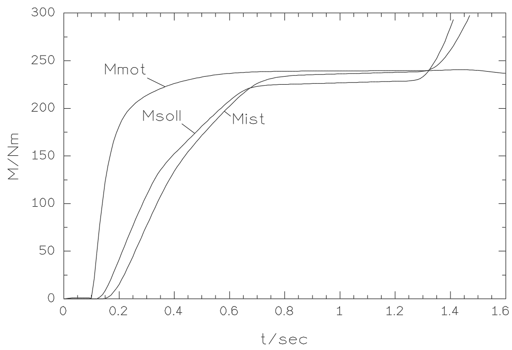

Figure 6.2. Torques, throttle angle α from 0° to 50°

$$n_{soll} := \frac{\omega_{Mot_{stat}}}{2\pi}, n_{mot} := \frac{\omega_{Mot}}{2\pi}, n_{get} := \frac{\omega_{gea}}{2\pi} \qquad M_{soll} := \text{desired torque}, \ M_{ist} := M_{cl}$$

The characteristics show the steady increase of the torque and the speed without overshoot. In the car a fuzzy controller with nearly the same characteristic has been implemented and lead to good results, essentially comparable to the simulations.

5. CONCLUSIONS

A new approach for the design of fuzzy controllers was outlined which is based on the chain rule of propositional logic. This approach makes use of a qualitative process model as well as a qualitative model of the desired behaviour of the closed loop system. The method works well as long as the rough discretization of the membership functions of the linguistic variables involved by means of the peak pattern method results in qualitative models of satisfying accuracy. The application of this design for the control of an automated clutch for automobiles demonstrated the practical benefits of this method.

REFERENCES

1. Bandemer, H. and Näther, W.:"Fuzzy Data Analysis." *Kluwer Academic Publishers*, Dordrecht 1992.
2. Sanchez, E.:"Resolution of Composite Fuzzy Relation Equations". *Inf. and Contr*. 30 (1976),pp. 38-48.
3. Zadeh, L. A.:"The Concept of a Linguistic Variable and its Application to Approximate Reasoning, Part 1". *Information Sciences* 9 (1976), pp. 43 - 80.
4. Böhm, R.: "Fuzzy-Regelungen: Entwurf auf Basis von Relationen und Stabilitätsanalyse." Fortschr.-Ber. VDI Reihe 8 Nr. 501 Düsseldorf: VDI-Verlag 1995.