# TEACHING FUZZY LOGIC CONTROL DESIGN THROUGH LABORATORY EXPERIMENTS AND STUDENT PROJECTS

**G. Schmidt and U. Hanebeck**

*Lehstuhl für Steuerungs- und Regelungstechnik, Technische Universität München, Postfach 202420, D-8000 München 2, Germany*

**Abstract.** This article reports on laboratory experiments that were developed to complement students' theoretical basis on FUZZY LOGIC techniques acquired in a course on linear and nonlinear controller design. A first application example is concerned with an "advanced cruise control system" for automobiles. Described in the second example is a laboratory experiment "magnetically levitated mass" during which various types of FUZZY controllers are developed and tested. Both experiments let students gain a deeper understanding of procedures, expenditures and performance linked to fundamentally different controller design techniques. For convenient experimentation FUZZY–software tools were developed which are also briefly described in this paper.

**Key Words.** Fuzzy Logic control;education;laboratory experiments;student projects.

## 1. INTRODUCTION

To familiarize control engineering students with the potential and limitations of the FUZZY LOGIC control approach a balanced combination of theoretical instruction and practical hands-on experience seems to be in place. For this reason our existing rich structure of laboratory set-ups was extended by adding a few real-world FUZZY LOGIC control experiments which are both challenging and simple enough for a student to understand in limited time. To avoid the need for every student to write his own code for FUZZY LOGIC control, a precompiler was developed that accepts high-level FUZZY control specifications and generates legal C-code. The precompiler syntax is discussed in the appendix.

## 2. ADVANCED CRUISE CONTROL

### 2.1. *Control Task*

The first laboratory experiment is concerned with an advanced cruise control system, i.e. the combined control of a follower's car velocity and its distance to a leader car. A realistic vehicle model provides the basis of the car simulation. It includes among other nonlinearities an automatic transmission (hydrodynamic converter/automatic gear–box) with hysteresis and the motion resistance. Fig. 1 shows the control concept consisting of two cascaded controllers. The inner loop takes care of the follower car velocity while the outer loop controls the distance between follower and leader car. Without a leader the outer loop is opened (switch in position 1), resulting in pure velocity control. If a car enters the field of view of the range sensor, the outer loop is closed (switch in position 2). To avoid switching between both control modes a hysteresis–type switching law is formulated. The desired distance is a constant $C = (0.5 \cdot \mathrm{m} \cdot \mathrm{h})/\mathrm{km}$ times the velocity of the controlled follower car. The velocity signal generated by the distance controller is limited to 1.1 times the velocity given by the driver. The measured distance is corrupted by uncorrelated gaussian noise and a signal that simulates erronous measurements of the range sensor. These errors typically appear when the sensor misses the leader car. For both the leader and the follower car the same dynamic model is used. It is also possible to simulate on and off turning cars. Sampling intervals are 10 ms for car simulation and 100 ms for evaluation of control algorithms.

### 2.2. *Linear Controller*

Conventional controllers for this problem have been designed based on a linearized car model, where parameters $K$ and $T$ of the linear model

$$F(s) = \frac{\Delta V}{\Delta gas} = \frac{K(V_0)}{1 + T(V_0)s}$$

depend on current velocity $V_0$. For both velocity and distance control we assume PI controllers. Minimization of a quadratic performance index yields analytic expressions for the parameters of the velocity controller and their dependency on $V_0$. Since the velocity control loop linearizes the car's behaviour, it suffices to choose a set of constant parameters for the distance controller in order to shift the closed loop poles into a desired region. Both linear controllers provide a reference for similar experiments performed with FUZZY LOGIC controllers.

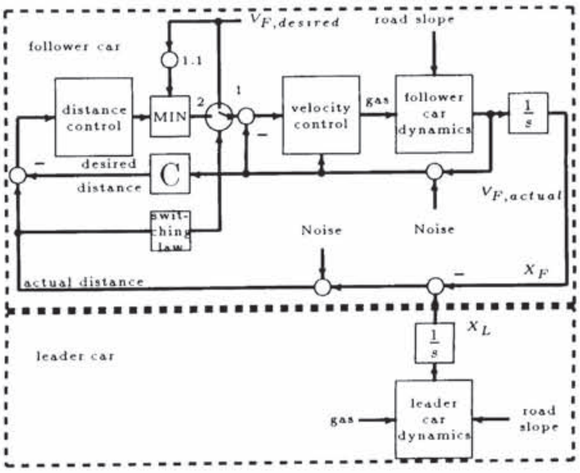

Fig. 1: Block diagram of cascaded control system.

### 2.3. *Laboratory task*

A team of students is supposed to design a velocity and a distance controller one after the other with Fuzzy Logic techniques and to compare their performance against results obtained with the optimal controllers. The structures of the Fuzzy Logic controllers are identical. Their input variables are the error in velocity / distance and their rate of change between consecutive samples. Output variables are the change in gas / velocity. For comparison purposes energy consumption, driving comfort and integral performance indices are used.

### 2.4. *Results*

At the beginning of the laboratory session students have a doubting attitude if Fuzzy Logic theory will work in practice. They first try to familiarize themselves with the dynamics of the car model and start reasoning about a rule base for a velocity controller. Two design strategies are typically used: In the first strategy students assign appropriate actions to typical situations. In the second they list all actions and define triggering situations. The first controller design generally works but does not behave satisfactorily. The students discuss the problems by means of graphically presented membership functions and control surfaces. They alter shape/overlap of the membership functions and change the rule base. After typically two or three iterations the controller works as desired. Its performance is comparable to the reference PI controller. However, the Fuzzy Logic controller shows less overshoot and is more tolerant to measurement noise and plant parameter changes. Fig. 2 shows the actual and desired velocity of the follower car for classical and Fuzzy Logic control for an additional load of 600 kg. A typical rule base for the velocity controller is shown in the appendix. Similar experience is collected during distance controller design. At this stage students are a bit more experienced, so the design process needs less time. In general, the resulting controller shows comparable performance to the reference but it is more robust. It even masters situations where the reference controller produces a crash. A typical plot of classical and Fuzzy Logic distance control is shown in Fig. 3. Peaks are due to simulated target misses of the range sensor. The conventional controller almost produces a crash in this situation. The laboratory experiment is performed by 80 students per semester. Documents for the experiments are provided beforehand. The laboratory session is limited to four hours. Within this time students are supposed to design both controllers and test them with respect to performance and robustness. They are expected to submit a report explaining their design strategy and to give a critical assessment of the results achieved.

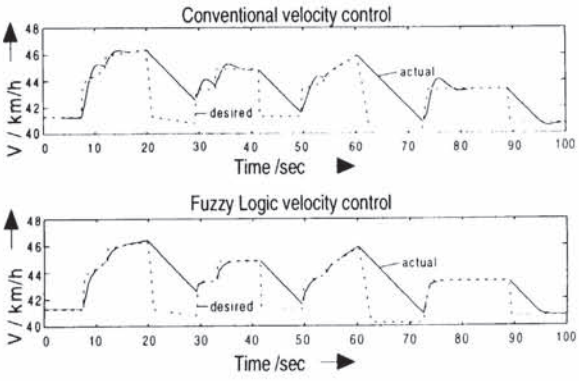

Fig. 2: Typical plot of actual and desired velocity of the follower car for classical and Fuzzy Logic control. (Cars additional load is set to 600 kg.)

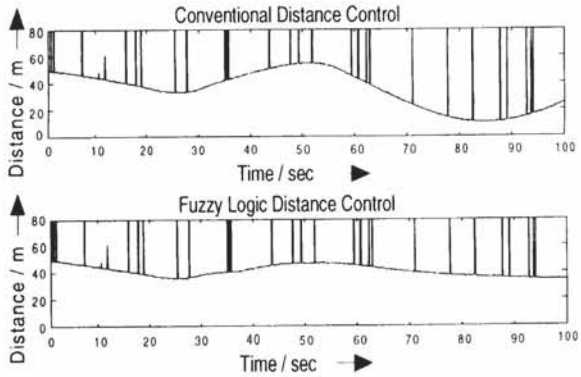

Fig. 3: A typical plot of classical and Fuzzy Logic distance control.

## 3. CONTROL OF A LEVITATED MASS

### 3.1. *Control Task*

The second example is concerned with a real physical setup. An iron mass is to be held free-flying at a variable set point $r$ from the normal operating point by means of a magnetic field. The corresponding coil is driven by a current $I$. The plant is unstable and nonlinear. A sensor measures the actual distance $x$ of the body from the normal operating point. The plant is connected to a PC/XT equipped with appropriate interfac-

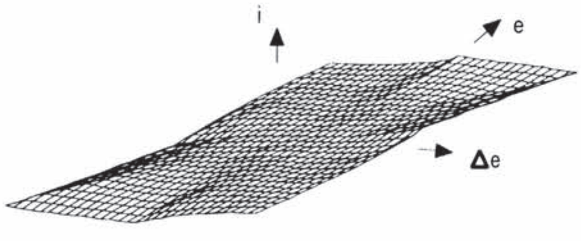

Fig. 4: Look up table for FL controller.

ing. The control algorithms are implemented in software and work with a sampling interval of 1 ms. Around the normal operating point the plant may be linearized leading to

$$\ddot{x} - c_1\, x = c_2\, i\,,\; c_1 = 854.4\,\frac{1}{s^2}\,, c_2 = 6.84\,\frac{mm}{mA\, s^2}\,.$$

The plant may successfully be stabilized by conventional PD or PID controllers.

## 3.2. *Project Task*

At the beginning of this project the student studied this system in detail in order to understand its dynamics and the behaviour of different linear control laws. He observed that for large deviations $x$ from the operating point and fast changes of the set point $r$ linear controllers do not work adequately. They often tend to loose the iron mass in these cases (Fig. 7 a) since a linear design is only valid around the normal operating point. For stabilizing the nonlinear system it is useful to note that the simplest stabilizing linear controller is of PD type. This leads to the conclusion that appropriate input variables for the simplest stabilizing FUZZY controller should be the positional error and change in error. Consequently the objective is now to generate with FUZZY LOGIC techniques a nonlinear controller mapping from $e$ and $\Delta e$ to $i$. For this purpose the student described linguistically his ideas of how a stabilizing controller should work. This protocol is fed into the precompiler, which produces a C-routine that implements the desired control actions. Since it was indispensable to run the FUZZY LOGIC controller in the same environment (PC/XT) as the classical controllers, FUZZY control rules could not be evaluated at runtime. A table with 256 times 256 entries had to be produced beforehand and was looked up at runtime. The table is visualized in Fig. 4. This is the fastest way to implement those types of FUZZY LOGIC controllers where the rule base remains fixed. The resulting controller works highly satisfactory. It even allows larger set point changes than the linear controller does (Fig. 7 b). However, it shows limit cycles around the normal operating point. To avoid this problem the student used the facts that a low gain linear PD controller ($K_p \approx 300\frac{mA}{mm}$) does not oscillate around the normal operating point but does not allow large set point changes. On the other hand, a high-gain

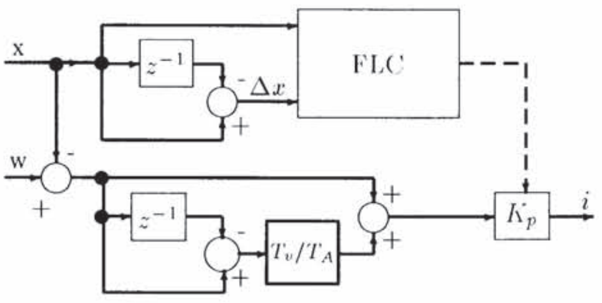

Fig. 5: Structure of the FUZZY adapted PD controller for a magnetically levitated mass.

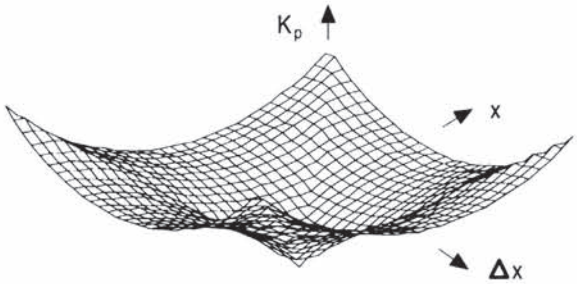

Fig. 6: Look up table for $K_p$.

PD controller ($K_p \approx 3000\frac{mA}{mm}$) limitcycles around the operating point but allows somewhat larger jumps of the set point. This experience led to the conclusion that PD controller parameters should be appropriately adapted by FUZZY LOGIC techniques. The easiest way to handle this problem is by adjusting $K_p$ depending on $x$ and $\Delta x$. A block diagram of this controller is shown in Fig. 5. The output value of the FUZZY LOGIC part has to be scaled to the range $300\frac{mA}{mm}$ .. $3000\frac{mA}{mm}$. A moderate $K_p$ is needed around the operating point such that noise is not amplified while a large $K_p$ is applied if $|x|$ and $|\Delta x|$ are large. This idea is easily translated into a linguistic protocol for an adaption law. $K_p$ becomes a function of $x$ and $\Delta x$ as shown in Fig. 6. The resulting controller allows to command large set point jumps (Fig. 7 c) and it produces only very small limit cycle amplitudes around the operating point. The steady-state error exhibited by this FUZZY adapted PD controller is much smaller than that of a constant gain PD controller. Steady-state errors were completely eliminated by adding an integral action.

## 3.3. *Results*

The objective of this project was to design and implement a stabilizing controller for a nonlinear, unstable real physical plant with FUZZY LOGIC techniques. It was performed during a 100 hours student project over a period of two months. While familiarizing with the existing plant the student gained experience on the plant's behaviour. FUZZY LOGIC techniques allowed him to cast this experience into various types of FUZZY and hybrid FUZZY/classical control algorithms that outperformed existing linear structures.

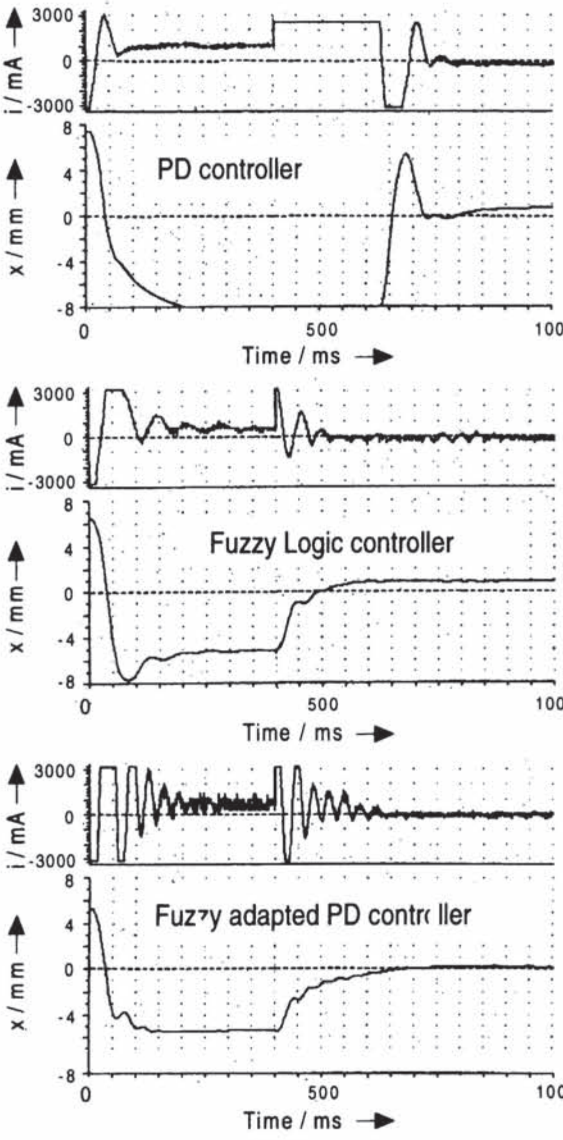

Fig. 7: Plot of current and distance for the magnetically levitated mass with a set point change from 5 mm to -5 mm and from -5 mm to 0 mm.

## 4. CONCLUSIONS

The experiments described in this paper make students understand Fuzzy Logic as an interesting extension of the great variety of well-established approaches used in control. They experience the user-benefits of the Fuzzy Logic approach as well as the value of a strong background in classical control theory and practice for the design of high-performance fuzzy controllers for real-world applications. They learn that the often heard assertion that Fuzzy Logic design does not require a process model is misleading. Obviously, the Fuzzy Logic designer needs certain model knowledge, however in a non-standard representation. This fact limits the application of Fuzzy Logic control techniques to those processes where causalities and dynamics are sufficiently well understood.

## 5. APPENDIX

The syntax of the developed precompiler will be explained via the rule base for the velocity controller of section 2.4.

```
CONTROLLER Tempo;
IMPLICATION MINIMPLY;
DEFUZZYFICATION COG;
IN Vd RANGE -10.0 ... 10.0
  MEMBER positive TRIANGLE 0.0 10.0 10.0
  MEMBER zero     TRIANGLE -1.0 0.0 1.0
  MEMBER negative TRIANGLE -10.0 -10.0 0.0;
IN DVd RANGE -1.0 ... 1.0
  MEMBER positive TRIANGLE 0.0 1.0 1.0
  MEMBER zero     TRIANGLE -0.1 0.0 0.1
  MEMBER negative TRIANGLE -1.0 -1.0 0.0;
OUT Gas RANGE -1.0 ... 1.0
  MEMBER muchmore TRIANGLE 0.5 1.0 1.0
  MEMBER more     TRIANGLE 0.0 0.5 1.0
  MEMBER equal    TRIANGLE -0.1 0.0 0.1
  MEMBER less     TRIANGLE -1.0 -0.5 0.0
  MEMBER muchless TRIANGLE -1.0 -1.0 -0.5;
RULE 1      Vd positive AND DVd zero
         OR Vd zero AND DVd positive
         -> Gas more;
RULE 2      Vd zero AND DVd zero
         OR Vd positive AND DVd negative
         OR Vd negative AND DVd positive
         -> Gas equal;
RULE 3      Vd negative AND DVd zero
         OR Vd zero AND DVd negative
         -> Gas less;
RULE 4      Vd positive AND DVd positive
         -> Gas muchmore;
RULE 5      Vd negative AND DVd negative
         -> Gas muchless;
```

Keywords are written in capitals. A # instructs the precompiler to ignore the rest of the line. A comment may be marked with /* */ as in the C language. The following order of instructions has to be maintained:

1. controller's name
2. implication used: **MINIMPLY** minimum implication, **PRODIMPLY** product implication.
3. defuzzyfication used: **COG** center of gravity, **MOM** mean of maxima.
4. linguistic input- and output variables with ranges and various types of membership functions, for instance **TRIANGLE a b c** yields

$$\mu(x) = \begin{cases} \frac{x-a}{b-a}, & a \le x \le b \\ \frac{c-x}{c-b}, & b \le x \le c \\ 0, & \text{else} \end{cases}$$

   Input values that are not in the predefined ranges will be mapped to the edge values.
5. the rulebase with precedence of **AND** over **OR**.

This protocol is translated into C-code by the precompiler. The C-routine evaluates the Fuzzy Logic control rules with a precision of 8 bits.