Fuzzy Sets and Systems 33 (1989) 275–290
North-Holland

# FUZZY CONTROL THEORY: THE LINEAR CASE

William SILER* and Hao YING
*Kemp-Carraway Heart Institute, Birmingham, AL 35234, U.S.A.*

Received February 1988
Revised June 1988

*Abstract:* A linear fuzzy proportional-integral (PI) controller with one input and one output is defined in terms of piecewise linear membership functions for fuzzification; control rules; and defuzzification algorithm. It is shown that a linear fuzzy controller is not equivalent to a linear non-fuzzy PI controller if the rules are evaluated using the Zadeh, probability or Lukasiewicz fuzzy logic alone. However, the linear fuzzy controller is precisely equivalent to a linear non-fuzzy PI controller if mixed fuzzy logic is used to evaluate the control rules, when the fuzzy logics used are selected with due regard to prior associations implied by the control rule operands themselves. This finding has relevance to the selection of fuzzy logics in fuzzy expert systems and for fuzzy logics in general.

*Keywords:* Control theory; logic; expert systems.

## 1. Introduction

Fuzzy controllers have achieved some popularity [7] due to the convenient way in which desired nonlinearities may be introduced, especially when no model of the controlled process is available but operator's experience may be used as a guide to formulation of the control rules. Procyk and Mamdani [3] have developed a method for automatic construction of fuzzy control rules. However, although a substantial body of theory for non-fuzzy controllers is available [2], there is no corresponding theory for fuzzy controllers. The work here reported was undertaken in an effort to improve this lack of theory. Since the theory of linear non-fuzzy controllers is especially well developed, we began by seeking to develop a theory for linear fuzzy controllers. Such a theory might permit isolation and description of nonlinearities present in fuzzy controllers, and perhaps to enable fuzzy controller designers better to utilize non-fuzzy controller design theory.

A simple nonfuzzy linear proportional-integral (PI) controller may have one input $y$ (the process output), setpoint $s$ (desired process output) and one controller output $u$. Such a controller can be described by the following differential equation:

$$\frac{du}{dt} = -a(y-s) - b\frac{dy}{dt}$$

* Present address: Mote Marine Laboratory, Sarasota, FL 34236, U.S.A.

in which $a$ and $b$ are constants. $y$ may be differentiated, and $\mathrm{d}u/\mathrm{d}t$ integrated numerically within the controller to yield the controller output $u$.

Denote the difference between actual process output and desired process output $y - s$ by error, denote the rate of change of process output $\mathrm{d}y/\mathrm{d}t$ by rate, denote the rate of change of controller output $\mathrm{d}u/\mathrm{d}t$ by deriv and scale so that $y$, $\mathrm{d}y/\mathrm{d}t$ and $\mathrm{d}u/\mathrm{d}t$ fall within the interval $[-1, 1]$. We may then write Eq. (1) as

$$\text{deriv} = -(\text{error} + \text{rate})/2. \tag{2}$$

In an analogous fuzzy controller, we fuzzify error and rate into discrete fuzzy sets ERROR and RATE, compute a fuzzy set DERIV through rules which describe our controller, defuzzify DERIV into deriv, and integrate deriv numerically to obtain our controller output. Typical fuzzy set members for DERIV, ERROR and RATE are 'positive_large', 'zero', 'negative_small' and the like. Typical controller rules are of the type

(ERROR is 'positive') AND (RATE is 'zero') OR

(ERROR is 'zero') AND (RATE is 'positive')

→ DERIV is 'negative_small'.

In this paper we will define a linear fuzzy controller in terms of fuzzification, rules and defuzzification. We will evaluate the rules by various fuzzy logics, and will compare our result to that for the linear nonfuzzy PI controller given in Eq. (2) above.

## 2. Definition of a linear fuzzy controller

We begin by defining, for a simple linear fuzzy proportional-integral controller with one input and one output, the fuzzification membership functions; the control rules; and the defuzzification algorithm. We will assume $N$ members of the fuzzy sets ERROR and error rate of change, or RATE, and $2N - 1$ members of the fuzzy set for derivative of controller output, or DERIV.

### *Fuzzification membership functions*

We define the principal values of the fuzzy sets for ERROR and RATE to be $x[i]$, $i = 1, \ldots, N$, ordered from most negative to most positive. We require these values to be equally spaced on the real line, and centered on zero. Without loss of generality, with scaling performed externally, we require the extreme values $x[1]$ and $x[N]$ to be $-1$ and $+1$ respectively. We require the membership functions for the fuzzy set members ERROR$[j]$ and RATE$[j]$ to be piecewise linear from zero at $x[j-1]$ to one at $x[j]$ and from one at $x[j]$ to zero at $x[j+1]$, if $x[j-1]$ and $x[j+1]$ exist, and zero elsewhere.

Although no fuzzification is needed for the fuzzy set DERIV, its members also have principal values which will be needed in defuzzification; these principal values are also equally spaced, but at half the interval of the members of ERROR and RATE.

For convenience, using the notation employed in our expert system shell FLOPS [5], we will denote the grades of membership of the members of fuzzy set SET by:

$$\text{SET.member}[k] := \text{gm}(\text{SET}, \text{member}[k])$$

where gm(SET, member[$k$]) is the grade of membership of the $k$-th member of fuzzy set SET.

As a simple example, if we have three members of fuzzy sets ERROR and RATE, denoted by n (negative), z (zero) and p (positive), their grades of membership are ERROR.n, ERROR.z, ERROR.p and RATE.n, RATE.z and RATE.p. Principal values for these would be −1 for member n, zero for member z and +1 for member p. The fuzzy set DERIV would have five members nl (negative large), ns (negative small), z (zero), ps (positive small) and pl (positive large) with principal values −1, −0.5, 0, +0.5 and +1 respectively.

## *Control rules*

We define our linear control rules to be of the form

```
(ERROR.error[m] AND RATE.rate[n])          OR
(ERROR.error[m + 1] AND RATE.rate[n − 1])  OR
...                                        OR
(ERROR.error[n] AND RATE.rate[m])
→ DERIV.deriv[2N − (m + n) + 1],
  m = 1, n = 1 to N; m = 2 to N, n = N (yields 2N − 1 rules).
```

These rules state, in effect, that the grade of membership of the $k$-th member of fuzzy set DERIV is the logical OR of the logical ANDS of the grades of membership of fuzzy set member pairs ERROR[$m$], RATE[$n$] whose principal values are symmetric about the principal value of DERIV[$k$] with sign reversed.

Consider the simple example given above with $N = 3$, with three fuzzy set members of ERROR and RATE $\{n, z, p\}$ with grades of membership ERROR.n, ERROR.z, ERROR.p and RATE.n, RATE.z, RATE.p. We have then $2N - 1$ or 5 members of fuzzy set DERIV; the members are DERIV.nl, DERIV.ns, DERIV.z, DERIV.ps, and DERIV.pl, for members negative large, negative small, zero, positive small and positive large. The control rules are:

1. (ERROR.n AND RATE.n)→DERIV.pl;
2. ((ERROR.n AND RATE.z) OR (ERROR.z AND RATE.n))→DERIV.ps;
3. ((ERROR.n AND RATE.p) OR
   (ERROR.z AND RATE.z) OR
   (ERROR.p AND RATE.n))→DERIV.z;
4. ((ERROR.z AND RATE.p) OR (ERROR.p AND RATE.z))→DERIV.ns;
5. (ERROR.p AND RATE.p)→DERIV.nl.

*Defuzzification algorithm*

The principal values of the $N$ members of the fuzzy sets ERROR and RATE, from the previous definition, span the range from $-1$ to $+1$, and are spaced $2/(N-1)$ apart on the real line. Without loss of generality we also restrict the range of the output derivative to $[-1, +1]$, subject to external scaling. There are $2N-1$ members of the fuzzy set DERIV. The principal value of the $i$-th member of fuzzy set DERIV is then $-1+(i-1)/(N-1)$. We defuzzify by summing the principal values of the members of fuzzy set DERIV weighted by their grades of memberships as derived from the control rules, or:

$$\text{derivative} = \sum_{i=1}^{i=2N-1} (\text{DERIV.deriv}[i] * (-1 + (i-1)/(N-1))).$$

The derivatives so obtained are integrated and scaled to obtain the actual controller output.

We note that if a member of DERIV has principal value zero, it need not be included in the control rules since its grade of membership is multiplied by zero in the defuzzification procedure.

## 3. Comparison: Linear fuzzy PI controller and non-fuzzy controller

So that the characteristics of a controller over its entire range may be conveniently viewed, we will plot isocontours of constant derivative on the phase plane of error on the abscissa and rate on the ordinate. With error, rate and derivative normalized to the range $[-1, 1]$ and scaling done externally, the simple non-fuzzy PI controller may be written as

$$d(\text{output})/dt = -(\text{error} + \text{rate})/2.$$

Figure 1 shows the output derivative on the phase plane plot. The isocontours are

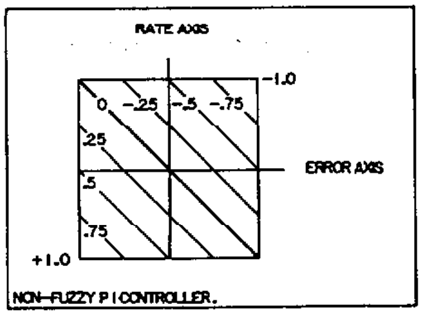

Fig. 1. Characteristics of linear non-fuzzy proportional-integral (PI) controller. Error on abscissa, rate-of-change-of-error on ordinate; isolines drawn of constant controller output derivative. Output derivative is integrated within the controller to yield controller output. Error, rate and output derivative are normalized to range $[-1, 1]$.

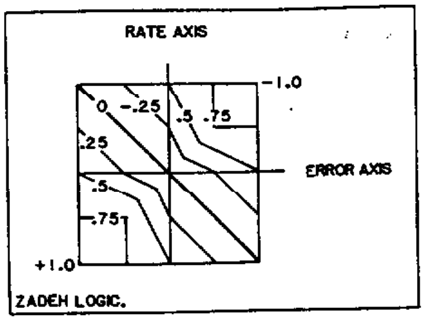

Fig. 2. Performance of linear fuzzy controller with rules evaluated using Zadeh logic only. Control is linear in second and fourth quadrants; hyper-linear in first and third quadrants.

straight lines with a slope of $-1$, the zero isocontour line running through the origin.

Since our fuzzy PI controller has been carefully designed to be linear, we would expect its isocontour plots to be identical to that of the non-fuzzy controller.

### *Rule evaluation using conventional fuzzy logics*

We first evaluate the control rules using the Zadeh fuzzy logic, with

$$a \text{ AND } b = \min(a, b), \qquad a \text{ OR } b = \max(a, b),$$

where $a$ and $b$ are grades of membership. Details of the calculations are shown in the Appendix. Characteristics of the controller with Zadeh logic are shown in Figure 2. The control is linear in the second and fourth quadrants, but is hyperlinear in the first and third quadrants; that is, the Zadeh fuzzy controller gives greater output than the linear non-fuzzy PI controller.

Now we evaluate the control rules using probability logic, with

$$a \text{ AND } b = a * b, \qquad a \text{ OR } b = a + b - ab.$$

Again, details of the calculations are given in the Appendix. Characteristics of the controller with probability logic are shown in Figure 3. Again, the control is linear in the second and fourth quadrants, but is slightly hypolinear in the first and third quadrants; that is, the probability fuzzy controller gives slightly less output than the linear non-fuzzy PI controller.

Now we evaluate the controller rules using Lukasiewicz logic, with

$$a \text{ AND } b = \max(0, 1 - (a + b)), \qquad a \text{ OR } b = \min(1, a + b).$$

Again, details of the calculations are given in the Appendix. Characteristics of the controller with Lukasiewicz logic are shown in Figure 4. Again the control is linear in the second and fourth quadrants, but is strongly hypolinear in the first and third quadrants; that is, the Lukasiewicz fuzzy controller gives less output

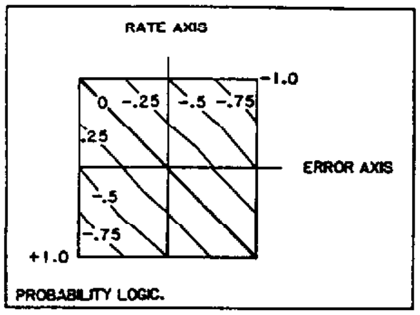

Fig. 3. Performance of linear fuzzy controller with rules evaluated using probability logic only. Control is linear in second and fourth quadrants; slightly hypolinear in first and third quadrants.

than the linear non-fuzzy PI controller. An additional problem now appears; the controller output is zero not only on the phase plane line from $(-1, +1)$ to $(+1, -1)$ but also on the line from $(-0.5, -0.5)$ to $(-0.5, +0.5)$, a region where control is expected but none is exerted.

Something is obviously wrong; our supposedly linear fuzzy controller is in fact nonlinear. However, our trouble is simple. It has been previously pointed out [1, 4] that the correct choice of fuzzy logics depends on any prior associations between the logic operands; that strongly positive associations imply the applicability of the Zadeh logic; zero associations, or independence, imply the applicability of probability logic; and negative associations imply the applicability of Lukasiewicz logic. Our rules inherently imply such associations.

Consider the rule

((ERROR.n AND RATE.z) OR (ERROR.z AND RATE.n)) $\rightarrow$ DERIV.ps.

We have two clauses which are ORd to yield the grade of membership DERIV.ps. These clauses are antithetical. If the grade of membership ERROR.n is nearly one, then ERROR.z is nearly zero; conversely, if ERROR.z is nearly one, then ERROR.n is

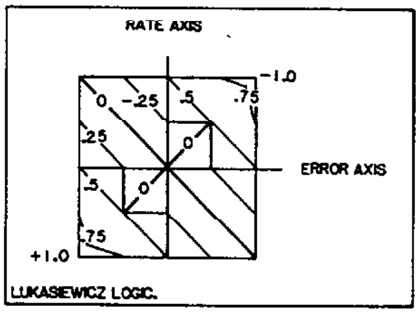

Fig. 4. Performance of linear fuzzy controller with rules evaluated using Lukasiewicz logic only. Control is linear in second and fourth quadrants; hypolinear in first and third quadrants. No control on line from $(-0.5, -0.5)$ to $(0.5, 0.5)$.

nearly zero. Similarly, if RATE.z is nearly one then RATE.n is nearly zero, and if RATE.z is nearly zero then RATE.n is nearly one. The associations are strongly negative, and the Lukasiewicz logic is appropriate.

Now consider the pair of rules

(ERROR.n AND RATE.n) $\rightarrow$ DERIV.pl;

((ERROR.n AND RATE.z) OR (ERROR.z AND RATE.n)) $\rightarrow$ DERIV.ps.

If the association between ERROR.n and RATE.n is strongly positive, then the association between ERROR.n and RATE.z, and between ERROR.z and RATE.n must be strongly negative; if the Zadeh logic is used in evaluating the first rule, then the Lukasiewicz logic should be used in evaluating the two clauses of the second rule. If the association between ERROR.n and RATE.n is zero, then we expect the association between ERROR.n and RATE.z, and between ERROR.z and RATE.n, also to be zero, and probability logic may be used in evaluating the first rule and the clauses of the second rule. Finally, if the association between ERROR.n and RATE.n is strongly negative, then the association between ERROR.n and RATE.z, and between ERROR.z and RATE.n, will be strongly positive; in this case the Lukasiewicz logic should be employed in evaluating the first rule and the Zadeh logic in evaluating the clauses of the second rule.

## *Rule evaluation using appropriate mixed fuzzy logics*

We now evaluate the rules using appropriate mixed fuzzy logics. First, we use the Zadeh logic for rules 1 and 5; the Lukasiewicz logic for the AND clauses of rules 2 and 4, and the Lukasiewicz logic for the OR of rules 2 and 4. Next, we use probability logic for rules 1 and 5; probability logic for the clauses of rules 2 and 4, and Lukasiewicz logic for the OR of rules 2 and 4. Finally, we use Lukasiewicz logic for rules 1 and 5; Zadeh logic for the AND clauses of rules 2 and 4; and Lukasiewicz logic for the OR of rules 2 and 4. Results are shown in Figure 5.

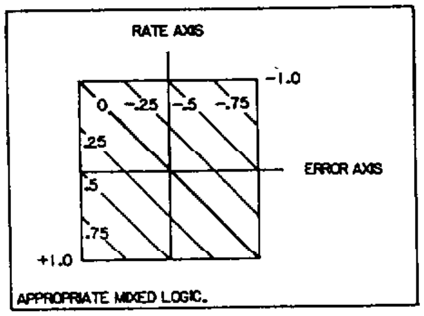

Fig. 5. Performance of linear fuzzy controller with rules evaluated using appropriate mixed logic. Appropriate logics include Zadeh logic for rules 1 and 5, and Lukasiewicz logic for rules 2 and 4; probability logic for rules 1 and 5, probability AND and Lukasiewicz OR for rules 2 and 4; Lukasiewicz logic for rules 1 and 5, Zadeh AND and Lukasiewicz OR for rules 2 and 4. Rule 3 need not be evaluated, since its consequent DERIV.z is multiplied by zero in defuzzification.

Details of the calculations are shown in the Appendix. In all cases the controller characteristics are precisely and theoretically identical to the non-fuzzy PI controller.

## 4. Discussion

In a fuzzy expert system, if there is no prior knowledge of associations, the Zadeh logic is a desirable default; but if prior association exists or is implied by the rules or operands, the correct fuzzy logic should be used. For example, in testing whether one fuzzy number is equal to OR greater than another, a negative association is implied, and the Lukasiewicz logic is correct and should be employed [6]. There are, in a fuzzy controller, four sources of nonlinearity. These include nonlinear fuzzification; nonlinear control rules, in the sense of departure from the linear control rules here defined; nonlinear defuzzification; and inappropriate choice of fuzzy logic for rule evaluation. Unfortunately, the designer might not be aware of nonlinearity due to inappropriate fuzzy logic; that is certainly to be avoided, if rational design of fuzzy controllers in a critical environment is to be achieved.

In the sample fuzzy controller given above, the hyperlinear control furnished in the first and third quadrants might well be a desirable characteristic; but in higher order fuzzy controllers, with more than three fuzzy set members for ERROR and RATE, the peicewise nonlinearity introduced by the Zadeh logic is probably undesirable, and needed nonlinearities should probably be otherwise introduced.

In the example of fuzzy logic application given in this paper we know a priori what the answer should be from non-fuzzy control theory. It is easy to spot erroneous answers and hence an inappropriate choice of fuzzy logic, and to evaluate the consequences of an inapppropriate choice, However, in many applications of fuzzy logic, such as in fuzzy expert systems, errors in the choice of fuzzy logic are much harder to spot, and a careful review of implicit associations and their consequences for a choice of fuzzy logic are even more important. In the example given, the inappropriate choice of probability logic has the least undesirable consequences, but it would be quite premature to conclude that this would always be the case. Our experience with our fuzzy expert system shell FLOPS over the past three years leads us to feel relatively comfortable with the Zadeh logic as a default for fuzzy expert systems.

## Appendix: Derivation of fuzzy controller equations

The input data are 'error' and 'rate'; the output is 'derivative'. Error, rate and derivative are all scaled to lie in the interval $[-1, +1]$.

The fuzzy sets ERROR, RATE and DERIV calculated have these members:

$$\text{ERROR} = \{n, z, p\},$$
$$\text{RATE} = \{n, z, p\},$$
$$\text{DERIV} = \{nl, ns, ps, pl\},$$

The fuzzy set DERIV does not have to have the zero member included, since its grade of membership is multiplied by zero in the defuzzification procedure. The fuzzy set members have these grades of membership:

| | |
|---|---|
| ERROR: | E.n, E.z, E.p |
| RATE: | R.n, E.z, R.p |
| DERIV: | D.nl, D.ns, D.ps, D.pl |

Membership functions for ERROR and RATE members are given by:

$$\text{E.n} = -e,\ -1 \leqslant e\ (\text{error}) \leqslant 0;$$
$$\text{R.n} = -r,\ -1 \leqslant r\ (\text{rate}) \leqslant 0;$$
$$\text{E.n} = 0,\ 0 \leqslant e\ (\text{error}) \leqslant +1;$$
$$\text{R.n} = 0,\ 0 \leqslant r\ (\text{rate}) \leqslant +1;$$
$$\text{E.z} = 1 + e,\ -1 \leqslant e\ (\text{error}) \leqslant 0;$$
$$\text{R.z} = 1 + r,\ -1 \leqslant r\ (\text{rate}) \leqslant 0;$$
$$\text{E.z} = 1 - e,\ 0 \leqslant e\ (\text{error}) \leqslant +1;$$
$$\text{R.z} = 1 - r,\ 0 \leqslant r\ (\text{rate}) \leqslant +1;$$
$$\text{E.p} = 0,\ -1 \leqslant e\ (\text{error}) \leqslant 0;$$
$$\text{R.p} = 0,\ -1 \leqslant r\ (\text{rate}) \leqslant 0;$$
$$\text{E.p} = e,\ 0 \leqslant e\ (\text{error}) \leqslant +1;$$
$$\text{R.p} = r,\ 0 \leqslant r\ (\text{rate}) \leqslant +1.$$

Control rules that must be evaluated are:

A. (ERROR.n, AND DERIV.n)→DERIV.pl;
B. ((ERROR.n AND DERIV.z) OR (ERROR.z AND DERIV.n))→DERIV.ps;
C. ((ERROR.z AND RATE.p) OR (ERROR.p AND RATE.z))→DERIV.ns;
D. (ERROR.p AND RATE.p)→DERIV.nl;

Defuzzification is obtained by

$$\text{derivative} = -\text{D.nl} - \text{D.ns}/2 + \text{D.ps}/2 + \text{D.pl}.$$

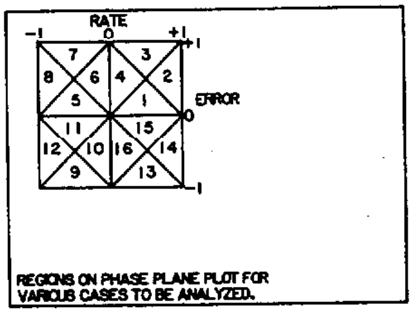
RATE
-1
0
+1
+1
ERROR
0
-1
7
3
8
6
4
2
5
1
11
15
12
10
16
14
9
13
REGIONS ON PHASE PLANE PLOT FOR VARIOUS CASES TO BE ANALYZED.

Fig. A1. Regions on phase plane plot of error versus error-rate-of-change which correspond to the case numbers in the analysis Tables A1–A6.

Table A1. Zadeh logic fuzzy controller
$x$ AND $y = \min(x, y)$, $x$ OR $y = \max(x, y)$

| Case | E.p | E.n | R.p | R.n | D.pl | D.nl |
|---|---|---|---|---|---|---|
| 1–2 | $e$ | 0 | $r$ | 0 | 0 | $r$ |
| 3–4 | $e$ | 0 | $r$ | 0 | 0 | $e$ |
| 5–8 | 0 | $-e$ | $r$ | 0 | 0 | 0 |
| 9–10 | 0 | $-e$ | 0 | $-r$ | $-e$ | 0 |
| 11–13 | 0 | $-e$ | 0 | $-r$ | $-r$ | 0 |
| 13–16 | $e$ | 0 | 0 | $-r$ | 0 | 0 |

| Case | E.p | E.z | R.p | R.z | E.p ∧ R.z | E.z ∧ R.p | D.ns = E.p ∧ R.z ∨ E.z ∧ R.p |
|---|---|---|---|---|---|---|---|
| 1 | $e$ | $1-e$ | $r$ | $1-r$ | $e$ | $r$ | $e$ |
| 2 | $e$ | $1-e$ | $r$ | $1-r$ | $1-r$ | $1-e$ | $1-r$ |
| 3 | $e$ | $1-e$ | $r$ | $1-r$ | $1-r$ | $1-e$ | $1-e$ |
| 4 | $e$ | $1-e$ | $r$ | $1-r$ | $e$ | $r$ | $r$ |
| 5–6 | 0 | $1+e$ | $r$ | $1-r$ | 0 | $r$ | $r$ |
| 7–8 | 0 | $1+e$ | $r$ | $1-r$ | 0 | $1+e$ | $1+e$ |
| 9–12 | 0 | $1+e$ | 0 | $1+r$ | 0 | 0 | 0 |
| 13–14 | $e$ | $1-e$ | 0 | $1+r$ | $1+r$ | 0 | $1+r$ |
| 15–16 | $e$ | $1-e$ | 0 | $1+r$ | $e$ | 0 | $e$ |

| Case | E.z | E.n | R.z | R.n | E.n ∧ R.z | E.z ∧ R.n | D.ps = E.n ∧ R.z ∨ E.z ∧ R.n |
|---|---|---|---|---|---|---|---|
| 1–4 | $1-e$ | 0 | $1-r$ | 0 | 0 | 0 | 0 |
| 5–6 | $1+e$ | $-e$ | $1-r$ | 0 | $-e$ | 0 | $-e$ |
| 7–8 | $1+e$ | $-e$ | $1-r$ | 0 | $1-r$ | 0 | $1-r$ |
| 9 | $1+e$ | $-e$ | $1+r$ | $-r$ | $1+r$ | $1+e$ | $1+e$ |
| 10 | $1+e$ | $-e$ | $1+r$ | $-r$ | $-e$ | $-r$ | $-r$ |
| 11 | $1+e$ | $-e$ | $1+r$ | $-r$ | $-e$ | $-r$ | $-e$ |
| 12 | $1+e$ | $-e$ | $1+r$ | $-r$ | $1+r$ | $1+e$ | $1+r$ |
| 13–14 | $1-e$ | 0 | $1+r$ | $-r$ | 0 | $1-e$ | $1-e$ |
| 15–16 | $1-e$ | 0 | $1+r$ | $-r$ | 0 | $-r$ | $-r$ |

| Case | D.nl | D.ns | D.ps | D.pl | derivative = D.pl + D.ps/2 − D.ns/2 − D.nl |
|---|---|---|---|---|---|
| 1 | $r$ | $e$ | 0 | 0 | $-(e+r)/2-r/2$ |
| 2 | $r$ | $1-r$ | 0 | 0 | $-(e+r)/2-(1-e)/2$ |
| 3 | $e$ | $1-r$ | 0 | 0 | $-(e+r)/2-(1-r)/2$ |
| 4 | $e$ | $r$ | 0 | 0 | $-(e+r)/2-e/2$ |
| 5–6 | 0 | $r$ | $-e$ | 0 | $-(e+r)/2$ |
| 7–8 | 0 | $1+e$ | $1-r$ | 0 | $-(e+r)/2$ |
| 9 | 0 | 0 | $1+e$ | $-e$ | $-(e+r)/2+(1+r)/2$ |
| 10 | 0 | 0 | $-r$ | $-e$ | $-(e+r)/2-e/2$ |
| 11 | 0 | 0 | $-e$ | $-r$ | $-(e+r)/2-r/2$ |
| 12 | 0 | 0 | $1+r$ | $-r$ | $-(e+r)/2+(1+e)/2$ |
| 13–14 | 0 | $1+r$ | $1-e$ | 0 | $-(e+r)/2$ |
| 15–16 | 0 | $e$ | $-r$ | 0 | $-(e+r)/2$ |

The piecewise nature of the fuzzification rules and of the Zadeh and Lukasiewicz logics requires that 16 cases be considered, as shown in Figure A1. Since the value for derivative as given by the non-fuzzy lineary PI controller is $-(e+r)/2$, where $e$ is error and $r$ is rate, values shown in the calculation tables for derivative are given as $-(e+r)/2$ plus or minus any additional terms.

Tables A1 through A3 show calculation details for evaluating the rules above with a single logic type. Table A1 gives calculation details for evaluating for the sample case using Zadeh logic alone; Table A2, for probability logic; and Table A3, for Lukasiewicz logic.

Table A2. Probability logic fuzzy controller

$x$ AND $y = xy$, $x$ OR $y = x + y - xy$

| Case | E.p | E.n | R.p | R.n | D.pl | D.nl |
|---|---|---|---|---|---|---|
| 1–4 | $e$ | 0 | $r$ | 0 | 0 | $er$ |
| 5–8 | 0 | $-e$ | $r$ | 0 | 0 | 0 |
| 9–12 | 0 | $-e$ | 0 | $-r$ | $er$ | 0 |
| 13–16 | $e$ | 0 | 0 | $-r$ | 0 | 0 |

| Case | E.p | E.z | R.p | R.z | E.p ∧ R.z | E.z ∧ R.p | D.ns = E.p ∧ R.z ∨ E.z ∧ R.p |
|---|---|---|---|---|---|---|---|
| 1–4 | $e$ | $1-e$ | $r$ | $1-r$ | $e(1-r)$ | $r(1-e)$ | (1) |
| 5–8 | 0 | $1+e$ | $r$ | $1-r$ | 0 | $r(1+e)$ | $r(1+e)$ |
| 9–10 | 0 | $1+e$ | 0 | $1+r$ | 0 | 0 | 0 |
| 13–16 | $e$ | $1-e$ | 0 | $1+r$ | $e(1+r)$ | 0 | $e(1+r)$ |

(1) $= e(1-r) + r(1-e) - er(1-e)(1-r)$

| Case | E.z | E.n | R.z | R.n | E.n ∧ R.z | E.z ∧ R.n | D.ps = E.n ∧ R.z ∨ E.z ∧ R.n |
|---|---|---|---|---|---|---|---|
| 1–4 | $1-e$ | 0 | $1-r$ | 0 | 0 | 0 | 0 |
| 5–8 | $1+e$ | $-e$ | $1-r$ | 0 | $-e(1-r)$ | 0 | $-e(1-r)$ |
| 9–12 | $1-e$ | $-e$ | $1+r$ | $-r$ | $-e(1+r)$ | $-r(1+e)$ | (2) |
| 13–16 | $1-e$ | 0 | $1+r$ | $-r$ | 0 | $-r(1-e)$ | $-r(1-e)$ |

(2) $= -e(1+r) - r(1+e) - er(1+e)(1+r)$

| Case | D.nl | D.ns | D.ps | D.pl | derivative = D.pl + D.ps/2 − D.ns/2 − D.nl |
|---|---|---|---|---|---|
| 1–4 | $er$ | (1) | 0 | 0 | $-er - (e(1-r) + r(1-)$ $- er(1-e)(1-r))/2$ |
| 5–8 | 0 | $r(1+e)$ | $-e(1-r)$ | 0 | $(-e(1-r) - r(1+e))/2$ |
| 9–12 | 0 | 0 | (2) | $er$ | $er - (e(1+r) + r(1+e)$ $+ er(1+e)(1+r))/2$ |
| 13–16 | 0 | $e(1+r)$ | $-r(1-e)$ | 0 | $(-r(1-e) - e(1+r))/2$ |

(1) $= e(1-r) + r(1-e) - er(1-e)(1-r)$
(2) $= -e(1+r) - r(1+e) - er(1+e)(1+r)$

Table A3. Lukasiewicz logic fuzzy controller
$x$ AND $y = \max(0, a + b - 1)$, $x$ OR $y = \min(1, a + b)$

| Case | E.p | E.n | R.p | R.n | D.pl | D.nl |
|---|---|---|---|---|---|---|
| 1 | $e$ | 0 | $r$ | 0 | 0 | 0 |
| 2–3 | $e$ | 0 | $r$ | 0 | 0 | $e+r-1$ |
| 3 | $e$ | 0 | $r$ | 0 | 0 | $e+r-1$ |
| 5–8 | 0 | $-e$ | $r$ | 0 | 0 | 0 |
| 9 | 0 | $-e$ | 0 | $-r$ | $-e-r-1$ | 0 |
| 10–11 | 0 | $-e$ | 0 | $-r$ | 0 | 0 |
| 12 | 0 | $-e$ | 0 | $-r$ | $-e-r-1$ | |
| 13–16 | $e$ | 0 | 0 | $-r$ | 0 | 0 |

| Case | E.p | E.z | R.p | R.z | E.p ∧ R.z | E.z ∧ R.p | D.ns = E.p ∧ R.z ∨ E.z ∧ R.p |
|---|---|---|---|---|---|---|---|
| 1–2 | $e$ | $1-e$ | $r$ | $1-r$ | $e-r$ | 0 | $e-r$ |
| 3–4 | $e$ | $1-e$ | $r$ | $1-r$ | 0 | $-e+r$ | $-e+r$ |
| 5 | 0 | $1+e$ | $r$ | $1-r$ | 0 | 0 | 0 |
| 6–7 | 0 | $1+e$ | $r$ | $1-r$ | 0 | $e+r$ | $e+r$ |
| 8 | 0 | $1+e$ | $r$ | $1-r$ | 0 | 0 | 0 |
| 9–12 | 0 | $1+e$ | 0 | $1+r$ | 0 | 0 | 0 |
| 13 | $e$ | $1-e$ | 0 | $1+r$ | 0 | 0 | 0 |
| 14–15 | $e$ | $1-e$ | 0 | $1+r$ | $e+r$ | 0 | $e+r$ |
| 16 | $e$ | $1-e$ | 0 | $1+r$ | 0 | 0 | 0 |

| Case | E.z | E.n | R.z | R.n | E.n ∧ R.z | E.z ∧ R.n | D.ps = E.n ∧ R.z ∨ E.z ∧ R.n |
|---|---|---|---|---|---|---|---|
| 1–4 | $1-e$ | 0 | $1-r$ | 0 | 0 | 0 | 0 |
| 5 | $1+e$ | $-e$ | $1-r$ | 0 | $-(e+r)$ | 0 | $-(e+r)$ |
| 6–7 | $1+e$ | $-e$ | $1-r$ | 0 | 0 | 0 | 0 |
| 8 | $1+e$ | $-e$ | $1-r$ | 0 | $-(e+r)$ | 0 | $-(e+r)$ |
| 9–10 | $1+e$ | $-e$ | $1+r$ | $-r$ | 0 | $-e+r$ | $e-r$ |
| 11–12 | $1+e$ | $-e$ | $1+r$ | $-r$ | $-e+r$ | 0 | $-e+r$ |
| 13 | $1-e$ | 0 | $1+r$ | $-r$ | 0 | $-(e+r)$ | $-(e+r)$ |
| 14–15 | $1-e$ | 0 | $1+r$ | $-r$ | 0 | 0 | 0 |
| 16 | $1-e$ | 0 | $1+r$ | $-r$ | 0 | $-(e+r)$ | $-(e+r)$ |

| Case | D.nl | D.ns | D.ps | D.pl | derivative = D.pl + D.ps/2 − D.ns/2 − D.nl |
|---|---|---|---|---|---|
| 1 | 0 | $e-r$ | 0 | 0 | $-(e+r)/2+r$ |
| 2 | $e+r-1$ | $e-r$ | 0 | 0 | $-(e+r)/2+(1-e)$ |
| 3 | $e+r-1$ | $-e+r$ | 0 | 0 | $-(e+r)/2+(1-r)$ |
| 4 | 0 | $-e+r$ | 0 | 0 | $-(e+r)/2+e$ |
| 5 | 0 | 0 | $-(e+r)$ | 0 | $-(e+r)/2$ |
| 6–7 | 0 | $e+r$ | 0 | 0 | $-(e+r)/2$ |
| 8 | 0 | 0 | $-(e+r)$ | 0 | $-(e+r)/2$ |
| 9 | 0 | 0 | $e-r$ | $-e-r-1$ | $-(e+r)/2-(1+r)$ |
| 10 | 0 | 0 | $e-r$ | 0 | $-(e+r)/2+e$ |
| 11 | 0 | 0 | $-e+r$ | 0 | $-(e+r)/2+r$ |
| 12 | 0 | 0 | $-e+r$ | $-e-r-1$ | $-(e+r)/2-(1+e)$ |
| 13 | 0 | 0 | $-(e+r)$ | 0 | $-(e+r)/2$ |
| 14–15 | 0 | $e+r$ | 0 | 0 | $-(e+r)/2$ |
| 16 | 0 | 0 | $-(e+r)$ | 0 | $-(e+r)/2$ |

Table A4. Mixed logic fuzzy controller I

E.p ∧ R.p, E.n ∧ R.n: Zadeh
E.p ∧ R.z ∨ E.z ∧ R.p, E.n ∧ R.z ∨ E.z ∧ R.n: Lukasiewicz

| Case | E.p | E.n | R.p | R.n | D.pl | D.nl |
|---|---|---|---|---|---|---|
| 1–2 | $e$ | 0 | $r$ | 0 | 0 | $r$ |
| 3–4 | $e$ | 0 | $r$ | 0 | 0 | $e$ |
| 5–8 | 0 | $-e$ | $r$ | 0 | 0 | 0 |
| 9–10 | 0 | $-e$ | 0 | $-r$ | $-e$ | 0 |
| 11–12 | 0 | $-e$ | 0 | $-r$ | $-r$ | 0 |
| 13–16 | $e$ | 0 | 0 | $-r$ | 0 | 0 |

| Case | E.p | E.z | R.p | R.z | E.p ∧ R.z | E.z ∧ R.p | D.ns = E.p ∧ R.z ∨ E.z ∧ R.p |
|---|---|---|---|---|---|---|---|
| 1–2 | $e$ | $1-e$ | $r$ | $1-r$ | $e-r$ | 0 | $e-r$ |
| 3–4 | $e$ | $1-e$ | $r$ | $1-r$ | 0 | $-e+r$ | $-e+r$ |
| 5 | 0 | $1+e$ | $r$ | $1-r$ | 0 | 0 | 0 |
| 6–7 | 0 | $1+e$ | $r$ | $1-r$ | 0 | $e+r$ | $e+r$ |
| 8 | 0 | $1+e$ | $r$ | $1-r$ | 0 | 0 | 0 |
| 9–12 | 0 | $1+e$ | 0 | $1+r$ | 0 | 0 | 0 |
| 13 | $e$ | $1-e$ | 0 | $1+r$ | 0 | 0 | 0 |
| 14–15 | $e$ | $1-e$ | 0 | $1+r$ | $e+r$ | 0 | $e+r$ |
| 16 | $e$ | $1-e$ | 0 | $1+r$ | 0 | 0 | 0 |

| Case | E.z | E.n | R.z | R.n | E.n ∧ R.z | E.z ∧ R.n | D.ps = E.n ∧ R.z ∨ E.z ∧ R.n |
|---|---|---|---|---|---|---|---|
| 1–4 | $1-e$ | 0 | $1-r$ | 0 | 0 | 0 | 0 |
| 5 | $1+e$ | $-e$ | $1-r$ | 0 | $-(e+r)$ | 0 | $-(e+r)$ |
| 6–7 | $1+e$ | $-e$ | $1-r$ | 0 | 0 | 0 | 0 |
| 8 | $1+e$ | $-e$ | $1-r$ | 0 | $-(e+r)$ | 0 | $-(e+r)$ |
| 9–10 | $1+e$ | $-e$ | $1+r$ | $-r$ | 0 | $e-r$ | $e-r$ |
| 11–12 | $1+e$ | $-e$ | $1+r$ | $-r$ | $-e+r$ | 0 | $-e+r$ |
| 13 | $1-e$ | 0 | $1+r$ | $-r$ | 0 | $-(e+r)$ | $-(e+r)$ |
| 14–15 | $1-e$ | 0 | $1+r$ | $-r$ | 0 | 0 | 0 |
| 16 | $1-e$ | 0 | $1+r$ | $-r$ | 0 | $-(e+r)$ | $-(e+r)$ |

| Case | D.nl | D.ns | D.ps | D.pl | derivative = D.pl + D.ps/2 − D.ns/2 − D.nl |
|---|---|---|---|---|---|
| 1–2 | $r$ | $e-r$ | 0 | 0 | $-(e+r)/2$ |
| 3–4 | $e$ | $-e+r$ | 0 | 0 | $-(e+r)/2$ |
| 5 | 0 | 0 | $-(e+r)$ | 0 | $-(e+r)/2$ |
| 6–7 | 0 | $e+r$ | 0 | 0 | $-(e+r)/2$ |
| 8 | 0 | 0 | $-(e+r)$ | 0 | $-(e+r)/2$ |
| 9–10 | 0 | 0 | $e-r$ | $-e$ | $-(e+r)/2$ |
| 11–12 | 0 | 0 | $-e+r$ | $-r$ | $-(e+r)/2$ |
| 13 | 0 | 0 | $-(e+r)$ | 0 | $-(e+r)/2$ |
| 14–15 | 0 | $e+r$ | 0 | 0 | $-(e+r)/2$ |
| 16 | 0 | 0 | $-(e+r)$ | 0 | $-(e+r)/2$ |

Tables A4 through A6 show calculation details for evaluating the rules with appropriate mixed logics. Table A4 gives details for Zadeh logic for the rules A and D above, and Lukasiewicz logic for rule B and C. Table A5 is for probability logic for the rules A and D, and the probabilistic AND and Lukasiewicz OR for rules B and C. Finally, Table A6 is for Lukasiewicz logic for the rules A and C, and the Zadeh AND and Lukasiewicz OR for rule B and C. In all three cases, the Lukasiewicz OR is used in rules B and C.

Table A5. Mixed logic fuzzy controller II

E.p ∧ R.p, E.n ∧ R.n: Probability
E.p ∧ R.z, E.z ∧ R.p, E.n ∧ R.z, E.z ∧ R.n: Probability
(E.p ∧ R.z) ∨ (E.z ∧ R.p), (E.n ∧ R.z) ∨ (E.z ∧ R.n): Lukasiewicz

| Case | E.p | E.n | R.p | R.n | D.pl | D.nl |
|---|---|---|---|---|---|---|
| 1–4 | $e$ | 0 | $r$ | 0 | 0 | $er$ |
| 5–8 | 0 | $-e$ | $r$ | 0 | 0 | 0 |
| 9–12 | 0 | $-e$ | 0 | $-r$ | $er$ | 0 |
| 13–16 | $e$ | 0 | 0 | $-r$ | 0 | 0 |

| Case | E.p | E.z | R.p | R.z | E.p ∧ R.z | E.z ∧ R.p | D.ns = E.p ∧ R.z ∨ E.z ∧ R.p |
|---|---|---|---|---|---|---|---|
| 1–4 | $e$ | $1-e$ | $r$ | $1-r$ | $e(1-r)$ | $r(1-e)$ | $(e+r)-2r$ |
| 5–8 | 0 | $1+e$ | $r$ | $1-r$ | 0 | $r(1+e)$ | $r(1+e)$ |
| 9–10 | 0 | $1+e$ | 0 | $1+r$ | 0 | 0 | 0 |
| 13–16 | $e$ | $1-e$ | 0 | $1+r$ | $e(1+r)$ | 0 | $e(1+r)$ |

| Case | E.z | E.n | R.z | R.n | E.n ∧ R.z | E.z ∧ R.n | D.ps = E.n ∧ R.z ∨ E.z ∧ R.n |
|---|---|---|---|---|---|---|---|
| 1-4 | $1-e$ | 0 | $1-r$ | 0 | 0 | 0 | 0 |
| 5–8 | $1+e$ | $-e$ | $1-r$ | 0 | $-e(1-r)$ | 0 | $-e(1-r)$ |
| 9–12 | $1+e$ | $-e$ | $1+r$ | $-r$ | $-e(1+r)$ | $-r(1+e)$ | $-(e+r)-2er$ |
| 13–16 | $1-e$ | 0 | $1+r$ | $-r$ | 0 | $-r(1-e)$ | $-r(1-e)$ |

| Case | D.nl | D.ns | D.ps | D.pl | derivative = D.pl + D.ps/2 − D.ns/2 − D.nl |
|---|---|---|---|---|---|
| 1–4 | $er$ | $(e+r)-2er$ | 0 | 0 | $-(e+r)/2$ |
| 5–8 | 0 | $r(1+e)$ | $-e(1-r)$ | 0 | $-(e+r)/2$ |
| 9–12 | 0 | 0 | $-(e+r)-2er$ | $er$ | $-(e+r)/2$ |
| 13–16 | 0 | $e(1+r)$ | $-r(1-e)$ | 0 | $-(e+r)/2$ |

Table A6. Mixed logic fuzzy controller III

$E.p \wedge R.p$, $E.n \wedge R.n$: Lukasiewicz
$E.p \wedge R.z$, $E.z \wedge R.n$, $E.n \wedge R.z$, $E.z \wedge R.n$: Zadeh
$(E.p \wedge R.z) \vee (E.z \wedge R.n)$, $(E.n \wedge R.z) \vee (E.z \wedge R.n)$: Lukasiewicz

| Case | E.p | E.n | R.p | R.n | D.pl | D.nl |
|---|---|---|---|---|---|---|
| 1 | $e$ | 0 | $r$ | 0 | 0 | 0 |
| 2–3 | $e$ | 0 | $r$ | 0 | 0 | $e+r-1$ |
| 3 | $e$ | 0 | $r$ | 0 | 0 | 0 |
| 5–8 | 0 | $-e$ | $r$ | 0 | 0 | 0 |
| 9 | 0 | $-e$ | 0 | $-r$ | $-e-r-1$ | 0 |
| 10–11 | 0 | $-e$ | 0 | $-r$ | 0 | 0 |
| 12 | 0 | $-e$ | 0 | $-r$ | $-e-r-1$ | |
| 13–16 | $e$ | 0 | 0 | $-r$ | 0 | 0 |

| Case | E.p | E.z | R.p | R.z | $E.p \wedge R.z$ | $E.z \wedge R.p$ | D.ns $= E.p \wedge R.z \vee E.z \wedge R.p$ |
|---|---|---|---|---|---|---|---|
| 1 | $e$ | $1-e$ | $r$ | $1-r$ | $e$ | $r$ | $e+r$ |
| 2–3 | $e$ | $1-e$ | $r$ | $1-r$ | $1-r$ | $1-e$ | $2-(e+r)$ |
| 4 | $e$ | $1-e$ | $r$ | $1-r$ | $e$ | $r$ | $e+r$ |
| 5–6 | 0 | $1+e$ | $r$ | $1-r$ | 0 | $r$ | $r$ |
| 7–8 | 0 | $1+e$ | $r$ | $1-r$ | 0 | $1+e$ | $1+e$ |
| 9–12 | 0 | $1+e$ | 0 | $1+r$ | 0 | 0 | 0 |
| 13–14 | $e$ | $1-e$ | 0 | $1+r$ | $1+r$ | 0 | $1+r$ |
| 15–16 | $e$ | $1-e$ | 0 | $1+r$ | $e$ | 0 | $e$ |

| Case | E.z | E.n | R.z | R.n | $E.n \wedge R.z$ | $E.z \wedge R.n$ | D.ps $= E.n \wedge R.z \vee E.z \wedge R.n$ |
|---|---|---|---|---|---|---|---|
| 1–4 | $1-e$ | 0 | $1-r$ | 0 | 0 | 0 | 0 |
| 5–6 | $1+e$ | $-e$ | $1-r$ | 0 | $-e$ | 0 | $-e$ |
| 7–8 | $1+e$ | $-e$ | $1-r$ | 0 | $1-r$ | 0 | $1-r$ |
| 9 | $1+e$ | $-e$ | $1+r$ | $-r$ | $1+r$ | $1+e$ | $2+(e+r)$ |
| 10–11 | $1+e$ | $-e$ | $1+r$ | $-r$ | $-e$ | $-r$ | $-(e+r)$ |
| 12 | $1+e$ | $-e$ | $1+r$ | $-r$ | $1+r$ | $1+e$ | $2+(e+r)$ |
| 13–14 | $1-e$ | 0 | $1+r$ | $-r$ | 0 | $1-e$ | $1-e$ |
| 15–16 | $1-e$ | 0 | $1+r$ | $-r$ | 0 | $-r$ | $-r$ |

| Case | D.nl | D.ns | D.ps | D.pl | derivative $=$ D.pl $+$ D.ps/2 $-$ D.ns/2 $-$ D.nl |
|---|---|---|---|---|---|
| 1 | 0 | $e+r$ | 0 | 0 | $-(e+r)/2$ |
| 2–3 | $e+r-1$ | $2-(e+r)$ | 0 | 0 | $-(e+r)/2$ |
| 4 | 0 | $e+r$ | 0 | 0 | $-(e+r)/2$ |
| 5–6 | 0 | $r$ | $-e$ | 0 | $-(e+r)/2$ |
| 7–8 | 0 | $1+e$ | $1-r$ | 0 | $-(e+r)/2$ |
| 9 | 0 | 0 | $2+(e+r)$ | $-e-r-1$ | $-(e+r)/2$ |
| 10–11 | 0 | 0 | $-(e+r)$ | 0 | $-(e+r)/2$ |
| 12 | 0 | 0 | $2+(e+r)$ | $-e-r-1$ | $-(e+r)/2$ |
| 13–14 | 0 | $1+r$ | $1-e$ | 0 | $-(e+r)/2$ |
| 15–16 | 0 | $e$ | $-r$ | 0 | $-(e+r)/2$ |

## References

[1] J.J. Buckley and W. Siler, Fuzzy operators for possibility interval sets, *Fuzzy Sets and systems* 22 (1987) 215–227.

[2] K. Ozata, *Modern Control Engineering* (Prentice-Hall, New York, 1970).

[3] T.J. Procyk and F.Z. Mamdani, A linguistic self-organizing process controller, *Automatica* 15 (1979) 15–30.

[4] E.H. Ruspini, Possibility theory approaches for advanced information systems, *Computer* 15 (1982) 83–91.

[5] W. Siler, FLOPS: a fuzzy expert system shell, in: *Preprints of Second IFSA Congress* (International Fuzzy Systems Association, 1987) 848–850.

[6] W. Siler and J.J. Buckley, Fuzzy numbers for expert systems, in: *Proceedings of International Symposium of Fuzzy Systems and Knowledge Engineering* (Guangdong Higher Education Publishing House, Guangdong, China, 1987) 103–109.

[7] M. Sugeno, *Industrial Applications of Fuzzy Control* (Elsevier Science Publishers, New York, 1985).