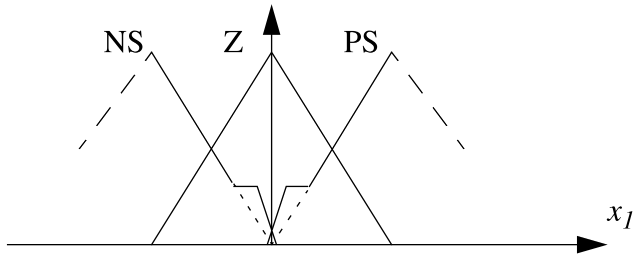

Figure 3.13: Modified membership functions of x_1

U

x_2

x_1

Figure 3.14: Change of control surface due to ZMMF by Figure 3.13

In both Chapter 4 and Chapter 5 ZMMF is applied to improve the behaviour of fuzzy controllers in simulation and real-time experiments.

Usually, however, one will use fine-tuning to have a smaller oscillatory behaviour of x_1, in other words, to get the value of x_1 closer to zero. Instead of a more flat strip around x_1 in the control surface, a more steeper strip around x_1 is desired, steeper then in the original of Figure 3.8. A steeper strip means that a small error results in a relative large control signal to correct this small error. The most logical modification to obtain this change might seem to manipulate the membership function in the opposite way from the modifications in Figure 3.9, namely widening the membership functions of NS and PS instead of narrowing them. (see Figure 3.11)

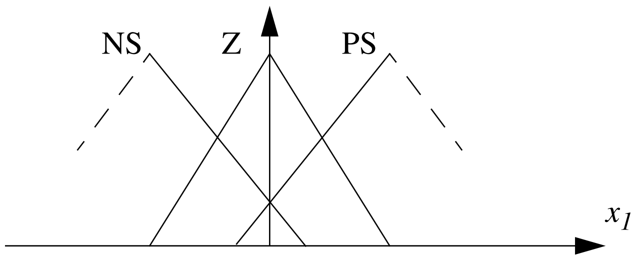

Figure 3.11: Modified membership functions of x_1

U

x_2

x_1

Figure 3.12: Change of control surface due to ZMMF by figure 3.11

Due to the modifications, described by Figure 3.11, the strip around x_1 is not steeper at all (see Figure 3.12) if related to the control surface of Figure 3.8. One can notice that control surface in the direction of x_1 is smoothed and the sharp folds have disappeared. This can be explained. By overlapping of the membership functions NS and PS, the FLC will be characterized by an even more gradual transition from NS to PS, thus a smoother control surface. The modifications carrying out a more steep strip around x_1 are described in Figure 3.13. Figure 3.14 shows this. The control surface is left in tact so the overall behaviour is still like that of the original one (Figure 3.8), only a small strip is modified. (Note that the control surface is just partly shown.)

Figure 3.8)

U

x_2 x_1

Figure 3.8: Control surface as ZMMF is not applied

Figure 3.8 shows a ‘linear’ control surface, for as far as that is possible with a FLC. Next, the membership functions of x_1 are changed in the following way (see Figure 3.9):

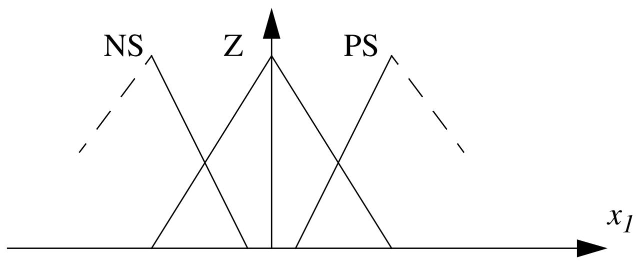

Figure 3.9: Modification of the membership functions of x_1

This means that small values of x_1 are accepted by the controller. In practice this will result in a less accurate control of x_1. In the control surface (see Figure 3.10) this is expressed by a horizontal strip around x_1,what is exactly what one may expect. This modification might be useful to reduce limit cycling [Chapter 7, Jamshidi *et al.*, 1993] as lower precision in x_1 is allowed.

U

x_2 x_1

Figure 3.10: Change of control surface due to ZMMF by figure 3.9

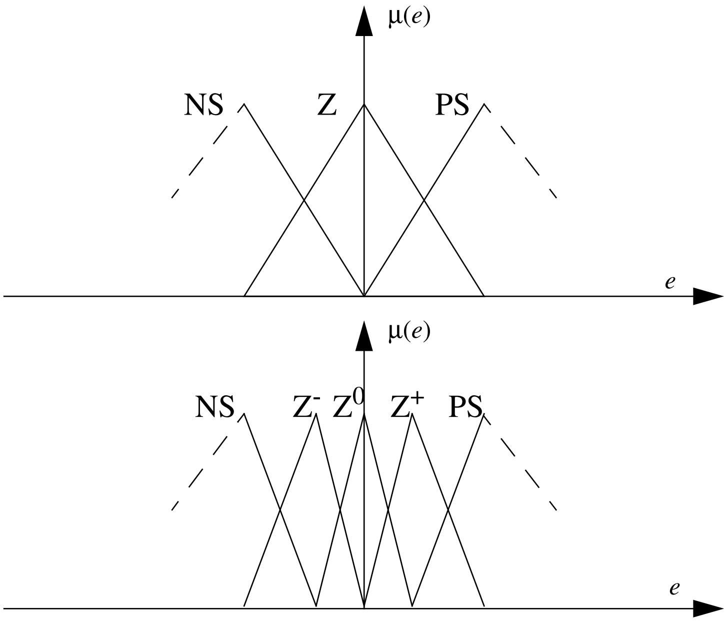

Figure 3.6: Membership functions of error *e* without and with zooming

In this way, the goal of defining the control behaviour in steady state more accurate can be reached, however, more rules have to be added to the original rule-base. Instead of $r = m_e m_{\Delta e}$= 5*5 = 25 rules for the unzoomed FLC, the number of rules increases to r = 7*5 = 35.

In the next part of this Section the above mentioned ZMMF approach will be introduced. Because ZMMF is based on modifying the already existing membership functions, the number of membership functions for each variable stays the same and with that the number of rules. In the following examples consider a FLC with two inputs x_1 and x_2 and one output U as described in Figure 3.17 and a McVicar-Whelan rule-base. The variables x_1 and x_2 have a normalized universe of discourse [-1,1] and the domain for U is defined as [-10,10]. For every example the effect on the control behaviour is illustrated by the variations of the control surface. Notice that in every figure the middle of the control surface is zoomed around the middle (x_1, x_2, U = 0), and thus just partly shown. The middle represents the desired steady state.

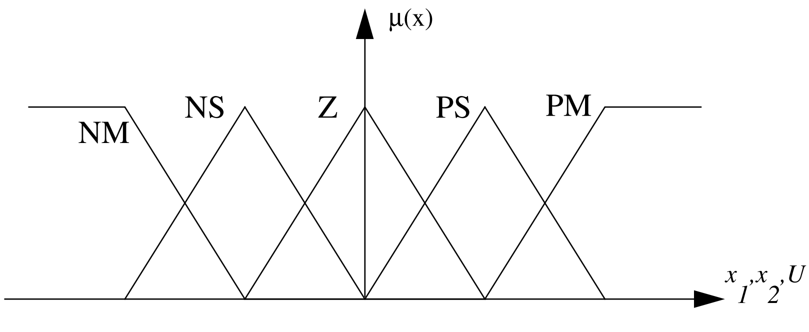

Figure 3.7: Membership functions of x_1, x_2 and U

First, examine the control surface around the steady state when ZMMF is not applied. (see

B).

-Click ‘Make’ to start the rule generation process. A name can be chosen for every input variable. For each input one has to select the number of membership functions, from 2 up to 5. The more membership functions are chosen, the higher will be the accuracy of the approximation, but the higher the number of rules. (The number of rules is the product of the number of membership functions of each input variable). Most times, the calculated singletons of the output variable will exceed the boundaries of the universe of discourse. A prompt will appear asking it is ‘OK’ to rescale. Answer ‘YES’.

-Click ‘Demo’ to compare the data from the data file to the generated fuzzy system. The status bar will report the R-squared correlation coefficient, a measure how well the rules emulate the observed data. According to Dreier [1994] the correlation is acceptable, if R^2 is greater than 0.99. The graphs of the observed data and rule generated data will bear witness to that.

The rules can be found in the left window of the ‘Autorule’ screen. For detailed information about the membership functions of the input variables and the values of output singletons, one has to return to the main menu and click ‘Memberships’ from the ‘Edit’ drop-down menu. In the left-upper corner of the membership function editor all the names of the membership functions and singletons of the generated FLC are presented. By dubble-clicking one of the names, both numerical and graphical information is shown on the screen.

Step 4:

At last, the obtained FLC has to be implemented. Since it is a Takagi-Sugeno-type fuzzy controller, the FLC can be easily be implemented even without a fuzzy logic design tool (for an example of a C-language implementation of a Takagi-Sugeno-type FLC see Appendix E.?).

3.5 Zooming by modifying the membership functions

This section can be divided in two parts. First will be explained when zooming or fine-tuning is wished and how it is applied [Jamshidi, 1995, Titli, 1995]. Afterwards will be shown that zooming can be accomplished by modifying the membership functions. This last approach is abbreviated as ZMMF (Zooming by Modifying the Membership Functions).

Suppose that the performance of a FLC is satisfactory over the complete universe of discourse, except for the behaviour about the setpoint. Suppose that the oscillations in steady state are larger than the desired limits. If the control behaviour around the setpoint could be specified more precise, the problems might be solved.

Consider an fuzzy controller with 2 input variables, being the error e and the change of error Δe both represented by 5 linguistic variables as mentioned in Section 3.2. In order to more precisely control a system around $e = 0$, divide the linguistic term Z of input e in Z^-, Z^0 and Z^+, as illustrated by Figure 3.6.

step.

Step 2:

Now, the FLC has to be characterized by a finite number of input/output combinations, so the ‘Autorule’ function can generate the desired FLC with the reduced rule-base. The question arises how to collect the input/output combination and how many to collect. A FLC can be graphically characterized by the control surface. The most logical and easy way to collect the input/output combinations is to take equidistant samples of this control surface. The number of input/output combinations is in this way depending on the sample resolution. If the samples built up a smooth control surface, then samples have been taken at a high enough resolution, else the samples have to be taken closer to each other, leading to more observed data. The ‘Autorule’ function will approximate the observed data and thus the control surface of the original controller. This rule-base reduction method is therefore called the Control Surface Approximation (CSA) approach. These sample points have to be written to a file with a specific format and file name, so that the ‘Autorule’ function can deal with the data points. The file must have the form: *filename.DAT*

The first line of the data file consists of three numbers seperated by commas. The first number is the number of inputs, the second is the number of outputs, and the third is the number of input/output combinations

In the following lines all the input/output combinations are written, also seperated by commas. This results in a file as described below:

number of inputs, number of outputs, number of data

input value 1, input value 2, ... , output value

....

....

As mentioned before, the algorithm is only capable to calculate the fuzzy inference rules for one output at a time and a maximum of 4 input variables can be given.

In Chapter 5 the CSA approach is applied with success to a 25 rules rule-base and led to 9 rules. In Appendix D.1 a Matlab file can be found that calculates a matrix of input/output combinations given a normalized FLC represented as a function. After calculating the data points the control surface is shown. In this way one can check the smoothness and if necessary, the resolution can be adapted. If a satisfactory set of data points is obtained, the matrix with these can be saved to the desired .DAT file using a similar file as the Matlab file that can be found in Appendix D.2.

Step 3:

Next, the FLC with the reduced rule-base can be calculated by the ‘Autorule’ function.

-First, start FULDEK (version 3.2 or higher, student or professional edition).

-Choose ‘Autorule’ from the ‘Edit’ drop-down menu.

-Click ‘Clear’ from the ‘File’ drop-down menu if necessary. Else click ‘Input file’ to select the in Step 2 obtained data file.

-Click the ‘Options’ menu to select Minimum or Multiplicative composition (see Appendix

Figure 3.5: Rule reduction percentage using the hierarchical sensory fusion approach

3.4 Fuzzy systems as approximators for rule-reduction

It is known (Kosko [1992] and Wang [1992]) that fuzzy systems are in fact universal approximators. Therefore, a fuzzy system can approximate any linear and non-linear function or system. And also, a fuzzy system can approximate *another* fuzzy system, e.g. a fuzzy controller. Remaining within the framework of rule-reduction, it would be interesting to approximate, for example, a 25 or 49 rules FLC by a 4 or 9 rules fuzzy system. In the remainder of this section one specific method will be discussed, achieving rule-base reduction according to the above mentioned idea. This method uses the approximation algorithm, that is explained in Appendix B. The algorithm generates a Takagi-Sugeno-type fuzzy controller by examining a finite number of input/output combinations of a multiple input, single output system. The algorithm is implemented and as the **'Autorule'** function available in the fuzzy logic development environment FULDEK (see Section 2.2.2). In four steps is explained how the 'Autorule' function can be used for rule reduction.

Step 1:

First, a FLC of any conceivable form has to developed. The only limitations are that the maximum number of input variables is 4 and that there may be just one output variable. When the fuzzy controller satisfies in simulation or real-time experiments, one may proceed to the next

Figure 3.4: Graphical comparison of rule reduction methods

Figure 3.3: Rule-base reduction using hierarchical and sensory fusion

Figure 3.2: The hierarchical fuzzy controller

In fact, a hierarchical fuzzy controller solves the control problem serial instead of parallel. The number of rules r as a function of the number of input variables n is given by equation (3.4) (m is the number of linguistic terms per input):

$$r = (n - 1)\, m^2 \tag{3.4}$$

To compare this method with sensory fusion, consider again the situation of 4 sensory signals. The number of fuzzy rules reduce from $5^4 = 625$ to $(3)*5^2 = 75$. Clearly, for $n = 4$ sensory fusion is the favourable method. However, for $n = 10$ the number of rules is $(9)*5^2 = 225$. Although this is an unmanageable number of rules, the profit in reduction compared to sensory fusion is considerable.

Combining both methods discussed above leads to the so-called '*Hierarchical and sensory fusion approach*'. Here, the variables are simply combined first, as in Figure 3.1, and are then organized into a hierarchical structure similar to that of Figure 3.2. Figure 3.3 shows the results of this approach for 3, 5, 8 and 10 input variables. The reduction in the size of the rule base is quite remarkable. For the four cases shown in the size of the rule reduces from 80% for $n = 3$, 98.4% for $n = 5$, 99.98% for $n = 8$ to nearly 100% (99.998976%) for $n = 10$. In figure 3.4 the different approaches for rule-reduction are expressed graphically. Ideal sensory fusion (see Figure 3.4b), which could be obtained by fusing all variables, is impossible in practice. In Figure 3.5 one finds the percentage of rule-base reduction using the hierarchical sensory fusion reduction method versus no rule reduction at all as function of the number of sensory variables.

Deciding which variables can be fused and where they are put into the hierarchy is an extremely important but difficult process, mostly based on knowledge of the system. General design rules are hard to draw up. One that can be mentioned (see Section 4.3) is that the fusion of an error signal e and its derivative Δe is usually exceedingly successful.

Figure 3.1: Sensory fusion for 2, 3 and 4 sensory signals

3.3 Hierarchical structures

The *hierarchical fuzzy controller* was first introduced by Raju *et al.* (1991). The clearest way to explain his rule-reduction method is to examine the graphical representation of his method. (see Figure 3.2)

Firstly, consider a controller with $n = 3$ input signals (y_1, y_2, y_3). Traditionally, this leads to an $r = m^n = 5^3 = 125$ rules rule-base, described as follows:

$$R_1\text{: } IF\ y_1\ is\ A_1\ and\ y_2\ is\ B_1\ and\ y_3\ is\ C_1\ THEN\ u\ is\ D_1$$

$$R_2\text{: } IF\ y_1\ is\ A_2\ and\ y_2\ is\ B_2\ and\ y_3\ is\ C_2\ THEN\ u\ is\ D_2$$

$$\ldots\ldots \tag{3.2}$$

$$R_r\text{: } IF\ y_1\ is\ A_r\ and\ y_2\ is\ B_r\ and\ y_3\ is\ C_r\ THEN\ u\ is\ D_r$$

where $A_i, B_i, C_i \in \{\mathrm{NM}, \mathrm{NS}, \mathrm{Z}, \mathrm{PS}, \mathrm{PM}\}$. However, one could combine variables y_1, y_2 and y_3 linearly, in this example, in one of the following possible four ways: (*a*, *b* and *c* are positive parameters dictated by physical considerations and designers knowledge and experience)

- Variables 1 and 2 fused:

  $Y_1 = ay_1 + by_2$

  $Y_2 = y_3$

- Variables 1 and 3 fused:

  $Y_1 = ay_1 + by_3$

  $Y_2 = y_2$

- Variables 2 and 3 fused:

  $Y_1 = ay_2 + by_3$

  $Y_2 = y_2$

- All three variables fused:

  $Y = ay_1 + by_2 + cy_3$

In this manner number of fuzzy rules would be reduced from 125 to 25 when fusion 2 of the variables is applied. The combination of all three variables results in one fused variable and therefore 5 fuzzy rules, shown in equation 3.2.

$$R_1\text{: } IF\ Y\ is\ A_1\ THEN\ u\ is\ D_1$$

$$\ldots\ldots \tag{3.3}$$

$$R_5\text{: } IF\ Y\ is\ A_5\ THEN\ u\ is\ D_5$$

Using sensory fusion, the reduction has a lower bound if all variables could, somehow, be fused. It may be clear that all variables cannot be combined trivially. Every combination has to be reasoned and explained. In practice, usually just 2 variables are fused. Would every 2 variables be combined, then for even number of variables, the reduction is more pronounced. For example, if $n = 4$, then the number of rules would reduce $5^4 = 625$ to $5^2 = 25$, a 96% reduction versus a 80% for $n = 3$. Figure 3.1 illustrates the sensory fusion approach as discussed above.

In spite of the remarkable reduction for lower values of n, if n is large, say $n = 10$, there are $5^{10} = 9{,}765{,}625$ rules in the original base could be reduced to $5^5 = 3{,}125$ rules if every two sensory variables can be fused. In the next Section is shown that using hierarchical structures, in combination with sensory fusion, one can even handle the situation with $n = 10$.

3 Rule-base reduction methods

3.1 Introduction

One of the major problems of fuzzy control is that the number of rules grows exponentially with respect to the number system's sensory feedback variables [Olafsson *et al.*, 1995]. Furthermore, as a Fuzzy Logic Controller (FLC) is applied to systems with 3 states or more, the rule-base design is obstructed by lack of tractability. In fact, for a 3-state system the designer has to construct a 4 dimensional control surface, which is rather difficult. Solutions for these problems have been found in rule-base reduction methods.

In the last couple of years several rule-base reduction methods have been proposed [Raju *et al.*, 1991, Yamakawa, 1991, Jamshidi, 1995, 1996, Olafsson *et al.*, 1995]. In the next sections (Section 3.2 and 3.3) the use of *sensory fusion* and *hierarchical structures* for rule-base reduction will be discussed. It will be shown that a combination of these both methods gives the most interesting results. In that case, the FLC consists of seperated controllers that have just 2 inputs and 1 output, which leads to an easy rule base design.

In Section 3.4 a method is proposed that can be applied as a general final reduction after finishing the design of a FLC. This method is based on *Control Surface Approximation* (CSA).

In some cases fine-tuning of the controller about the setpoint is needed. Jamshidi describes in [Jamshidi, 1995] a method which causes an increasement of the number of rules. In the last section of this chapter a method is discussed that leaves the rule-base intact. The method is based on modification of the shape of the membership functions.

3.2 Sensory fusion

The sensory fusion approach is based on fusion or combining of sensor signals before being given as input for the FLC. In this section this method will be explained by examining several cases. In every case it is assumed that the input signals of the FLC are represented by $m = 5$ linguistic variables: NM (Negative Medium), NS (Negative Small), Z (Zero), PS (Positive Small), and PM (Positive Medium).