FUZZY LOGIC: AN INTRODUCTION

Fuzzy logic is a kind of statistical reasoning, whose foundations can be said to have been laid in the 18th century by the British philosopher Thomas Bayes. With this technique, large amounts of data can be condensed into a much smaller set of variable rules than with rigid logic. The result is an expert system that can process information faster, and provide a more flexible, more human-like response than conventional logic.

THE great German polymath, Gottfried Leibniz (1646–1716) dreamed about devising a way whereby a couple of philosophers could discuss and settle any human argument once and for all by pure logic. But he and many other thinkers after him have discovered that there are many problems that cannot be solved by just logic. This realization gave rise to another way of attempting to solve problems: the use of statistics. In statistical reasoning, probabilities express the idea of 'perhaps'. One method of statistical reasoning, whose foundations can be said to have been laid in the 18th century by the British philosopher Thomas Bayes, is called fuzzy logic. In fuzzy logic, there is not just 'true' and 'false', 1s and 0s, 'black' and 'white', but also all the various grades of grey in between. Fuzzy logic can condense large amounts of data into a much smaller set of variable rules than rigid logic. The result is an expert system that can process information faster, and provide a more flexible, more human-like response than conventional logic. For example, a washing machine controlled by fuzzy logic will wash very dirty clothes very hard; not-so-dirty clothes get a milder wash, and so on. Fuzzy logic is already being used in many domestic appliances, cameras and passenger trains.

Traditional control technology is based on a mathematical model that describes the control process. Although this is perfectly satisfactory for simple processes, it gets more difficult as the process becomes more complex. In such cases, the solution is derived from a simplified model or from a set of values that was determined empirically.

An example from everyday life would be when you are driving along in your car and you want to turn left (or right): without conscious calculation you determine the moment when you have to start turning the steering wheel. Without precise information of the width of the roads, the position of your car, the way the front wheels of your car react to the turning of the steering wheel, the wheel base of your car, and so on, you will normally act so that you do not get on the wrong side of the road or on the pavement. In the same easy manner, you can steer your car or that of your neighbours through completely different bends. What you are doing is reacting in a 'fuzzy-logical' way to the effect of an action. You turn the steering wheel a little, your eyes register the effect of this and your brain corrects if necessary and without complex analyses and calculations the action. If we were to have this, to us simple, operation carried out by a digital control system, we would have to design a surprisingly complex system that would, moreover, require a fairly large computer power. However, for a control system based on fuzzy logic, the rules would be based on human practical experience. For instance, at home:

if the room temperature is much *too low*, turn up the thermostat *to maximum*.

if the room temperature drops *slowly*, turn up the thermostat *a little*.

But how do we define *too low*, *slowly*, and *a little*? Fortunately, fuzzy logic can cope with these terms, as we will see later on.

Collecting data

An important principle of fuzzy logic is *set theory* (in mathematics, a set is a collection of elements chosen for membership of the set because it possesses some required property). This may be illustrated by, say, our desire to go out and buy fragrant red roses. We may go to a market and find a stall that sells flowers. We make our wishes known to the stall-holder, who subconsciously may reason: '*if* the flower is a rose, *and* if it is red, and *if* it is fragrant, *then* the customer will buy a bunch'. In other words, if the flower is an *element* of all three collections (rose, red, fragrant), it is the desired one. This is illus-

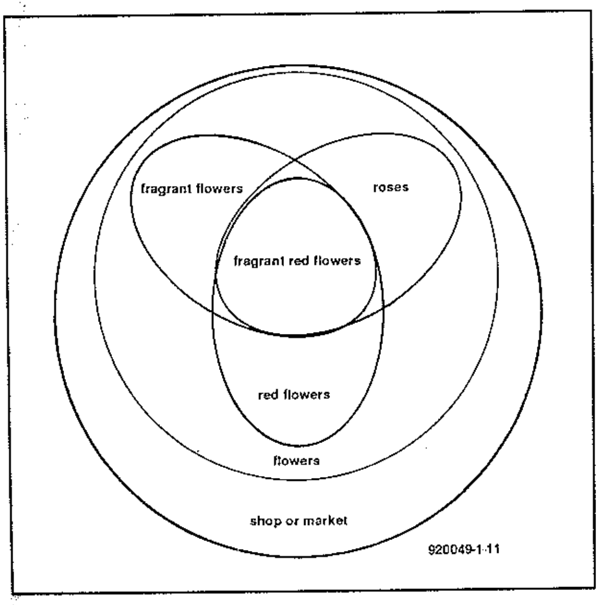

Fig. 1. Venn diagram of choices in a flower shop.

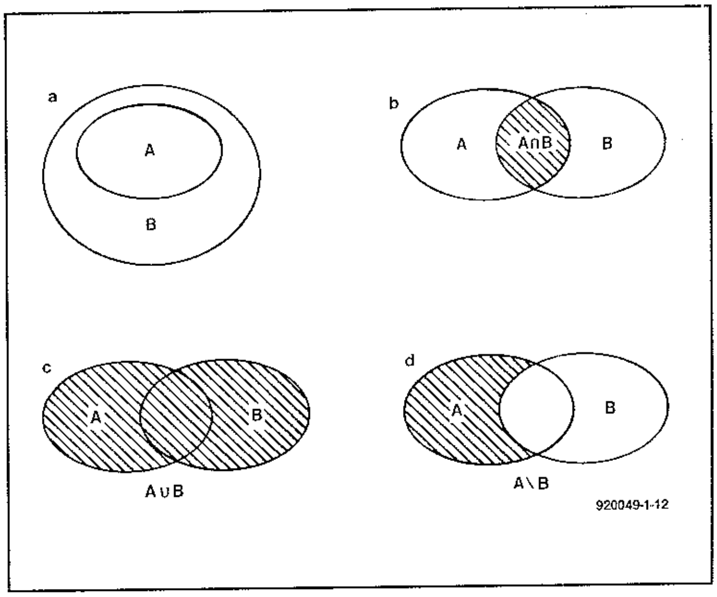

Fig. 2. Various basic operations in Set Theory.

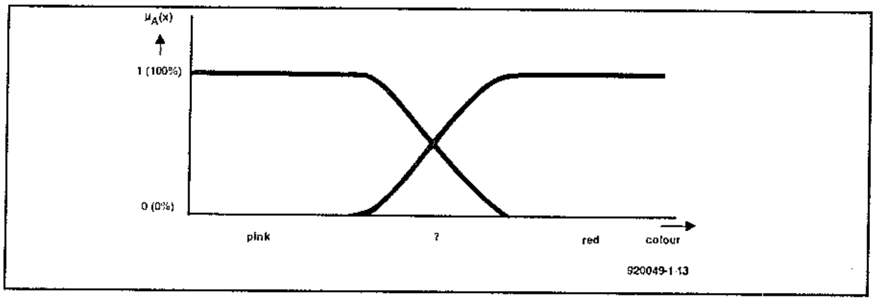

Fig. 3. Typical 'degree of association' cruve.

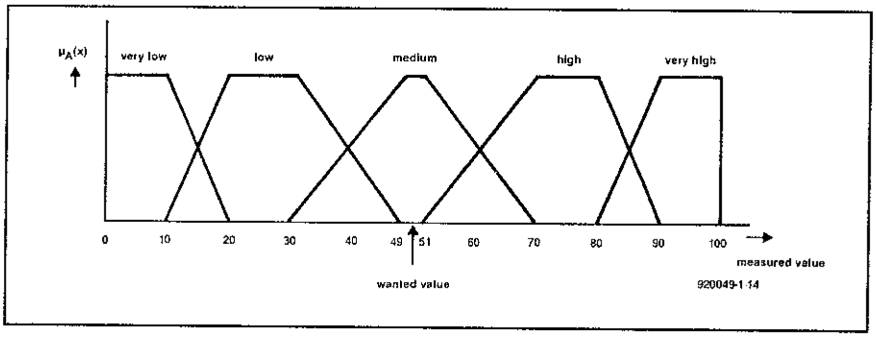

Fig. 4. Example of a fuzzy division of a measured value.

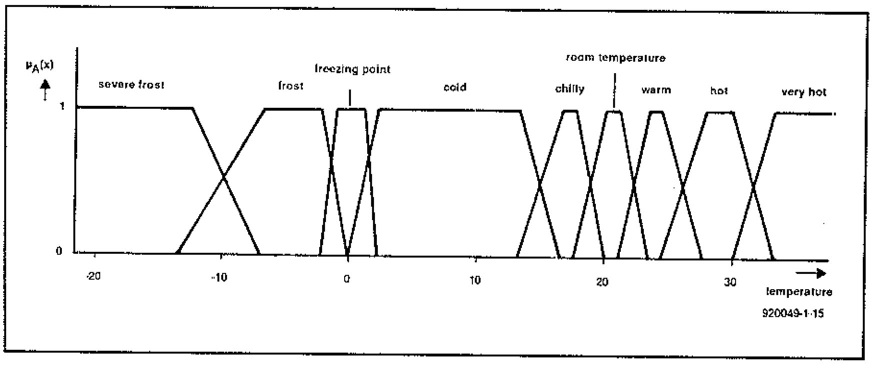

Fig. 5. Allocating likely values of temperature to fuzzy sets.

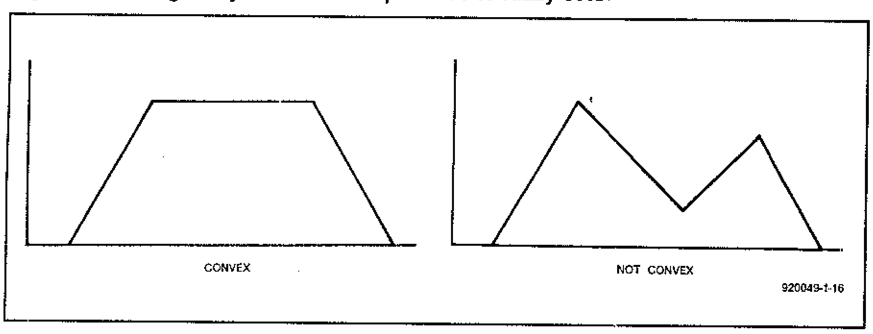

Fig. 6. The 'degree of association' curve should be as at the left, not as on the right.

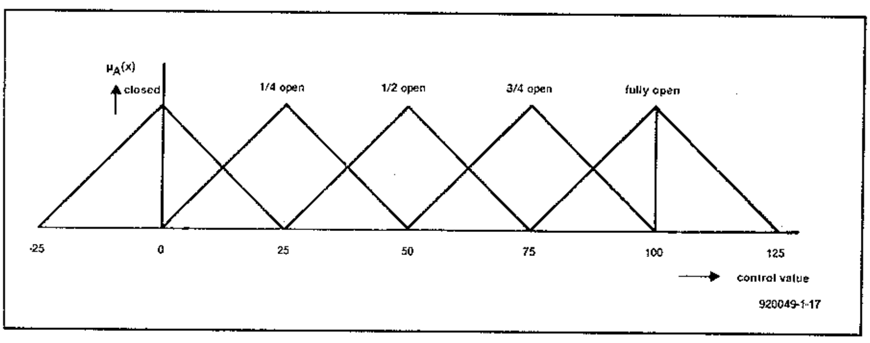

Fig. 7. Fuzzy sub-division of output signals.

trated in the Venn diagram in Fig. 1

In the logic rule we use to arrive at a final conclusion, we make use of a number of basic operations that are illustrated in Fig. 2, another Venn diagram. Figure 2a shows the simplest situation that can occur: from a set B, a new set A is formed, such that all elements of A are also elements of B. Such a set A is called a *subset* of B. Mathematically, this is expressed as $B \supset A$, read as 'A is a subset of B'. A more general situation arises when two sets A and B are involved, each of which possesses elements that are not common to to the other, so that neither $A \supset B$ nor $B \supset A$ is true. The set of elements C that is common to the two sets is called the *intersection* of sets A and B and is written $C = A \cap B$; this is illustrated in Fig. 2b. Another important set related to sets A and B is the set C containing all the elements belonging to A, to B or to both A and B. This is called the *union* of sets A and B and is written $C = A \cup B$, read as 'A cup B'; it is illustrated in Fig. 2c. Finally, in connection with sets A and B, there is the *complement* of B relative to A, which is written as $A \backslash B$ and read as 'A minus B'; this is illustrated in Fig. 2d.

All this is still clearly defined, but in the earlier instance of the red roses, we could ask: 'What is red; where does pink begin?'. In general, the colours red and pink will be recognized as such by most people, but in between them there is a range of hues that is not clearly red or pink. That sort of problem is solved by the use of fuzzy sets, in which a clearly red flower is entirely common to the red set and not at all to the pink set. A flower with a colour in between red and pink is common to both sets, for instance 70% red and 30% pink. This is called the *degree of association*, μ. For example, the degree of association of an element x to a set A is written as $\mu_A(x)$. The degree of association is shown by the curves in Fig. 3.

The type of characteristic shown in Fig. 3 is an important aid in the application of fuzzy logic in control engineering, because it enables measured values to be arranged in sets. The measured values (input signals) are entered on the x-axis, while the degree of association curves for a number of sets are plotted on the y-axis—see Fig. 4. In the design of control systems, it is usual to take an odd number of sets and to place the centre one in a position where it coincides with the desired value: here, 50.

Another instance of allocating likely values of ambient temperatures to fuzzy sets is shown in Fig. 5. The boundaries between areas are not always clearly defined; in fact, in this way it may be determined how 'fuzzy' the boundary between two sets is. It is customary, but not obligatory, to allow the boundaries to overlap to such an extent that the combined border areas have a degree of association of 100%, that is, $\mu=1$.

It will have been noticed in these examples that the curves of μ are trapezoidal. This is the most customary shape, since it allows straightforward arithmetic. Other shapes are possible, as long as they are convex, that is,

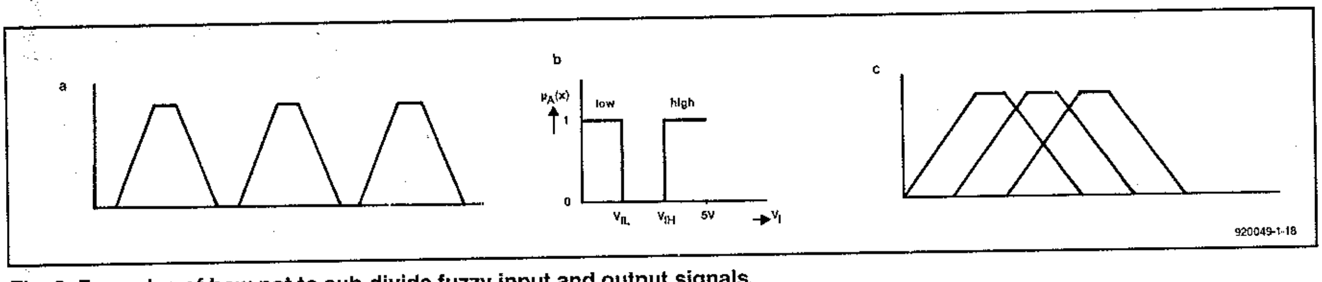

Fig. 8. Examples of how not to sub-divide fuzzy input and output signals.

their edges should not have transitions as shown in Fig 6 It is, however, possible to omit the horizontal top of the trapezium so that the curves attain a triangular shape This is done, for instance, in the case of a set that represents the desired value of a control system to obtain a very accurate setting. It is always done when the subdivision of output signals is fuzzy—see Fig. 7, which shows the positions of a boiler valve in a heating system It may look strange that the μ-characteristic for 'closed' extends to –25% and that for 'open' to +125%, but that is how these sets are weighted to the same scale as the other three positions. Once the output signals have been brought back to concrete values, the valve can be set exactly between 0% and 100%, no more, no less.

Examples of how not to subdivide input and output signals are shown in Fig 8 The curves in Fig 8a do not overlap, which means that there is no defined μ for a number of values A well-known example of this is the inputs of TTL-gates—see Fig 8b In these, a certain range of values belongs to the set 'low' and another range to the set 'high'. Values between these ranges will lead to unpredictable behaviour This is, by the way, a special fuzzy set: a so-called crisp-set. In Fig. 8c, the edges of the various curves spill over into various other sets: this will lead to instability

Logic combining of fuzzy sets

We have seen how input and output signals can be divided into fuzzy sets To use these to make a practical control system, certain rules are required to indicate the logic connections between input and output sets These rules, which describe and determine the behaviour of the system, can be arrived at through practical experience of the system or by trial and error

As an example of how to go about setting the rules, we will use a system that has a switch-on behaviour as shown in Fig. 9b. This is quite a common behaviour: for a little while after switch-on, the measured value will swing around the wanted value It is the task of the control system to bring the measured value to the wanted value quickly and to keep it there in spite of possible interference To design the system, we set out the error, that is, the difference between measured and wanted values (Fig 9c) and also the variations in the measured value (Fig. 9d). The x-axes of these figures show no concrete values, only a 0. In practice, the allocating of concrete values together with the selecting of sensible sets (and the rules mentioned earlier) will be the key to a successful design

The fuzzy division of the control signal is approached in two ways First, we divide the magnitude of the signal into seven sets—see Fig 9e. These sets enable us to give the system a proportional-control behaviour. To ensure that the system responds timely to the reactions of the process (integrating and differentiating), we need a number of sets that indicate by how much the control signal must be corrected to prevent overshoot, or to limit it to a wanted minimum value—see Fig. 9f

Next, we have to formulate the rules necessary to keep the measured value equal to the wanted value. They are summarized in Table 1. Rules 1–7 must ensure that through proportional control the measured and wanted values are kept equal, or nearly so Rules 8–14 ensure through integrating control that the value of the control signal is corrected as relevant to obtain the wanted value Overshoot of the control signal after the wanted value has been reached is prevented by rules 15–21, which, when the error becomes smaller, slow down the altering of the measured value. All the rules together ensure that the system reacts rapidly without overshoot

In fuzzy logic, all these considerations can be evaluated by simple arithmetic and a computer or analogue electronics. There are processors and controllers designed specifically to carry out fuzzy-logic computations It is interesting to note that an algorithm to carry out these computations would need a 32-bit microprocessor, whereas in fuzzy logic an 8-bit microcontroller is sufficient

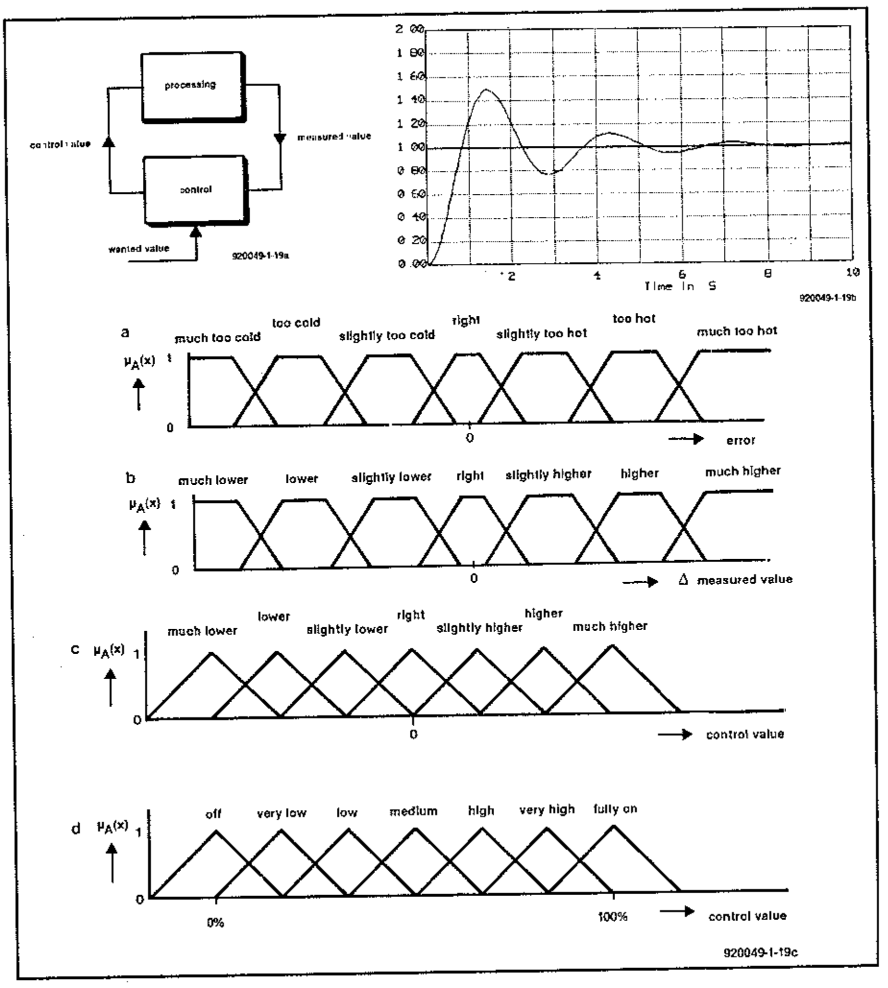

Fig. 9. Example of how to compute a control system.

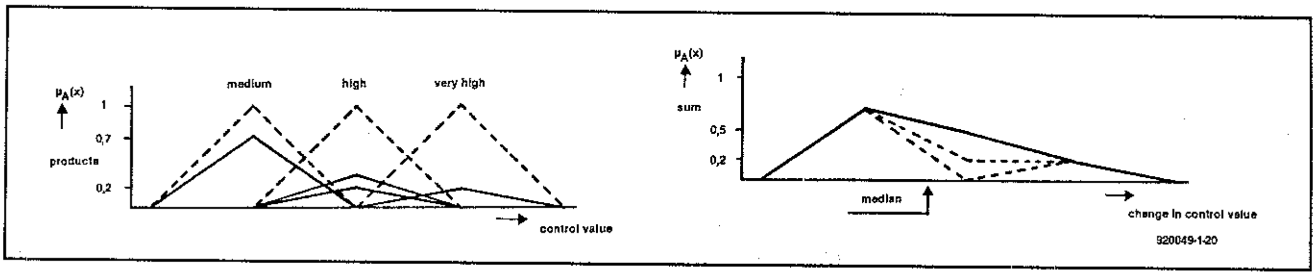

Fig. 10. Computation of the control signal for the system illustrated in Fig. 9.

Fuzzy logic arithmetic

The function $\mu_A(x)$ enables us to calculate to what degree an element x is common to set A. If we use a microcontroller and an analogue-to-digital (A-D) converter, the calculation becomes very simple, because the converter provides concrete values. For each of these values, the associated μ can be found in relevant tables.

Evaluating the logic rules is normally simplicity itself. There are three basic operations: AND, OR and NOT. In an AND-operation, the smallest μ-value of the relevant sets is allocated to the output set. If, for instance, two inputs, x and y, have values of $\mu_A(x)=0.8$ and $\mu_B(y)=0.3$, it follows from the logic rule 'A AND B gives C' ($A.B=C$) that the smallest μ, that is, 0.3, must be allocated to C. This is done by adding an element of value 0.3 to set C. This operation is called minimum-operator (MIN{A,B}) and is one of the special instructions in fuzzy logic.

The OR-operation allocates the largest μ-value to the output set. This operation is called maximum-operator (MAX{A,B}). With the values from the previous example the logic rule 'A OR B gives C' ($A+B=C$) gives an element with a μ-value of 0.8.

The NOT operation is just as simple: deduct the μ-value from 1: NOT $A=1-\mu_A(x)$.

Compensating operations yield results that lie somewhere between AND and OR; they add a sort of 'but' to the logic rule. For instance, you want to buy a house. It must be sound, in a good position, and not too expensive, *but*, if it is very nice and well situated, it may cost a little more. In pure logic terms, such a consideration is difficult to realize, but compensating operators make it possible. The most important of these is the gamma-operator. If the value of y lies between 0 and 1, this operator can be set between AND and OR. The simplified form of the gamma-operator (for three sets) is

$$\mu_r(x)=[\mu_A(x)\mu_B(x)\mu_C(x)]^{(1-y)}\times$$
$$\times\{1-[1-\mu_A(x)][1-\mu_B(x)][1-\mu_C(x)]\}^{y},$$

where $\mu_r=\mu_{result}$ and $0\leq y<1$

Fuzzy becomes distinct

After working through the logic rules, we have a number of indications (21 in Table 1) of what has to happen next. All the results in Table 1 refer to a signal that controls the relevant process. In complex processes, more control signals may be used. For the calculation of concrete values for these signals, a method is needed that somehow combines the results relevant to the signals. There are two usual methods: min-max-median and product-sum-median. The former is simple, but not suitable for the example in Table 1, because several logic rules apply to the same output set. In that case, the product-sum-median method must be used.

Assume that on evaluation of the logic rules the following results have been obtained: 0.7 median; 0.2 and 0.3 high; and 0.2 very high. How the product-sum-median method works is shown in Fig. 10. For each element, we multiply the height of the triangle indicating the μ of a set with the value of the elements and use the results to draw four new (smaller) triangles. We then add the areas of the triangles together and determine the median position of the resulting figure: the value on the x-axis underneath that position is the value we seek.

All this may sound pretty complicated, but the arithmetic is not too difficult. If we assume that the functions of μ for the control signals are isosceles triangles, the calculation becomes:

$$S_c=\frac{\sum_{x=1}^{q}[\alpha(x)\times S_a\times A]}{\sum_{x=1}^{q}[\alpha(x)\times A]}$$

where:

S_c is the value of the control signal;
q is the number of logic rules whose value is relevant to the magnitude of the control signal;
$\alpha[x]$ is the result of logic rule number x;
S_a is the value of the control signal immediately underneath the apex of the triangle (set) to which logic rule x refers;
A is the area of the relevant triangle (set) ($A=\frac{1}{2}\times$ base $\times$ height)

This formula is worked out in a computer in seconds: S_a and A are fixed data for all sets, which, therefore, can be stored in memory and need not be computed. If all triangles have identical areas, as in the example, the calculation becomes even simpler, because A can then be ignored both in the numerator and the denominator. ■

Table 1.

error	Δ measured value	control value
1. much too cold		full on
2. too cold		very high
3. slightly too cold		high
4. right		medium
5. slightly too hot		low
6. too hot		very low
7. much too hot		off
8. much too cold		much higher
9. too cold		higher
10. slightly too cold		slightly higher
11. right		right
12. slightly too hot		slightly lower
13. too hot		lower
14. much too hot		much lower
15. slightly too cold	much higher	off
16. slightly too cold	higher	very low
17. slightly too cold	slightly lower	high
18. slightly too cold	lower	high
19. slightly too hot	higher	off
20. slightly too hot	much higher	off
21. slightly too hot	much lower	high

Reference

Fuzzy sets, theory and its applications, by H.J Zimmermann, ISBN 0 7923 9075 X
Kluwer Academic Publishers