WHERE DID FUZZY LOGIC COME FROM?

The concept of Fuzzy Logic (FL) was conceived by Lotfi Zadeh, a professor at the University of California at Berkley

Presented as a way of processing data by allowing partial set membership rather than crisp set membership or non-membership.

This approach to set theory was not applied to control systems until the 70's due to insufficient small-computer capability prior to that time.

Professor Zadeh reasoned that people do not require precise, numerical information input, and yet they are capable of highly adaptive control.

If feedback controllers could be programmed to accept noisy, imprecise input, they would be much more effective and perhaps easier to implement.

U.S. manufacturers have not been so quick to embrace this technology while the Europeans and Japanese have been aggressively building real products around it.

WHAT IS FUZZY LOGIC?

FL is a problem-solving control system methodology

It lends itself to implementation in systems ranging from simple, small, embedded micro-controllers to large, networked, multi-channel PC or workstation-based data acquisition and control systems.

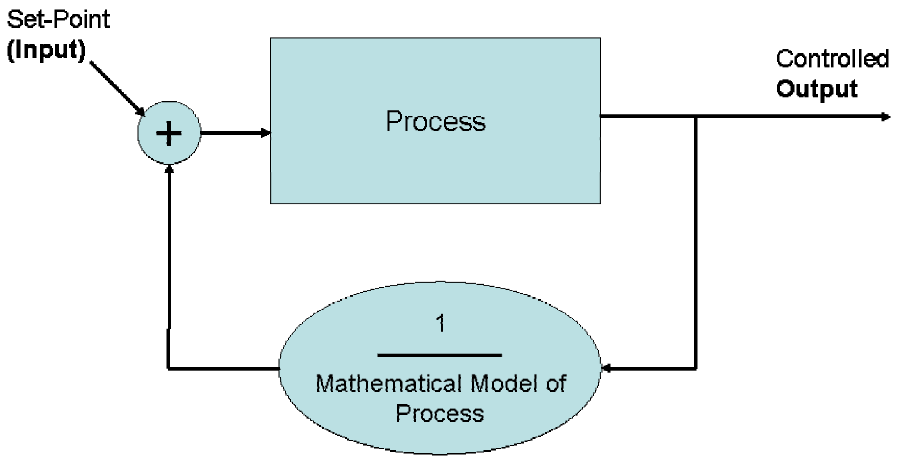
It can be implemented in hardware, software, or a combination of both.

FL provides a simple way to arrive at a definite conclusion based upon vague, ambiguous, imprecise, noisy, or missing input information.

FL's approach to control problems mimics how a person would make decisions, only much faster.

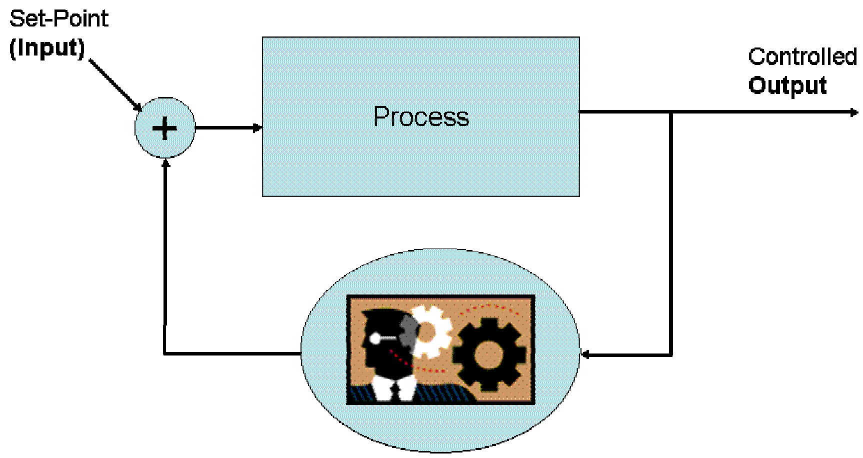
A Bit on Classic Control Theory:

Classic control is made by using a Mathematical model of the process to be controlled to create a feedback mechanism that is intended to keep the error from a set-point to a minimum (as near to cero as possible). In theory it works very nice, since you have the process and the inverse of it, mathematically you have a **1 to 1** relationship between the **Input** (the set-point) and the **Output**, thus the system **will have the controlled variable exactly to the set-point you want.** … That is in theory…



But the problem is that usually it is very difficult to determine the **exact** mathematical model of the process. Then, if the mathematical model is NOT exact, the control is not exact either, and depending on the process and other external factors, the control may become very difficult or erratic.

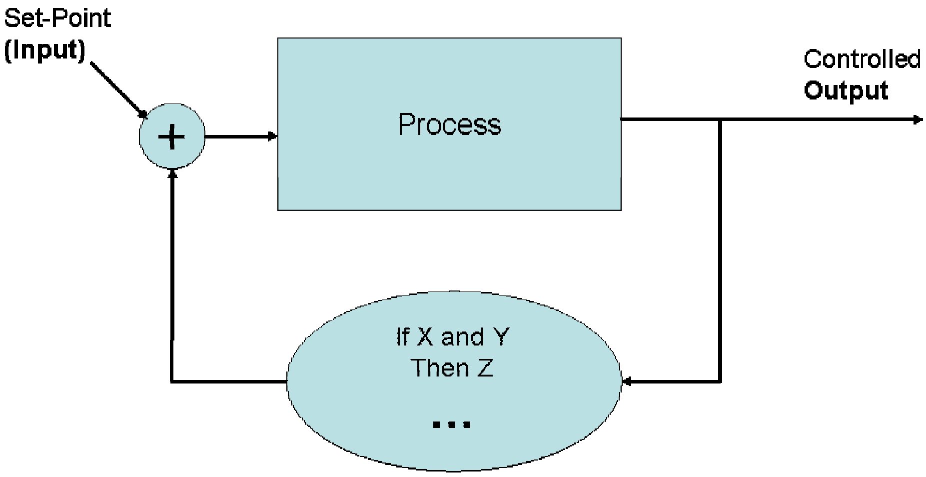
An alternative is to control the process by experience, like a person would do. When you control something (say a car in the road) you don not have all the equations that model the physical behaviour of the vehicle and the environment, you just "know" how to control the car by experience. And this works very well. You could then replace the model:



Then, what we want to do know is replace our experience with a computer program that represents that knowledge and is able to control the process for us, in a similar way that we would do. This is what a **Fuzzy Logic** controller is meant to do!

HOW IS FL DIFFERENT FROM CONVENTIONAL CONTROL METHODS?

FL incorporates a simple, rule-based IF X AND Y THEN Z approach to a solving control problem rather than attempting to model a system mathematically.



The FL model is empirically-based, relying on an operator's experience rather than their technical understanding of the system.

For example, rather than dealing with temperature control in terms such as:

> "SP =500F", "T <1000F", or "210C <TEMP <220C",

> Terms like "IF (process is too cool) AND (process is getting colder) THEN (add heat to the process)" are used.

These terms are imprecise and yet very descriptive of what must actually happen:

> Consider what you do in the shower if the temperature is too cold: you will make the water comfortable very quickly with little trouble.

FL is capable of mimicking this type of behaviour but at very high rate.

HOW DOES FL WORK?

FL requires some numerical parameters in order to operate such as what is considered significant error and significant rate-of-change-of-error, but exact values of these numbers are usually not critical unless very responsive performance is required in which case empirical tuning would determine them.

For example, a simple temperature control system could use a single temperature feedback sensor whose data is subtracted from the command signal to compute "error" and then time-differentiated to yield the error slope or rate-of-change-of-error, hereafter called "error-dot".

Error might have units of degs F and a small error considered to be 2F while a large error is 5F.

The "error-dot" might then have units of degs/min with a small error-dot being 5F/min and a large one being 15F/min.

These values don't have to be symmetrical and can be "tweaked" once the system is operating in order to optimize performance.

Generally, FL is so forgiving that the system will probably work the first time without any tweaking.

What is a Fuzzy Set?

The very basic notion of fuzzy systems is a *Fuzzy (sub)set.*

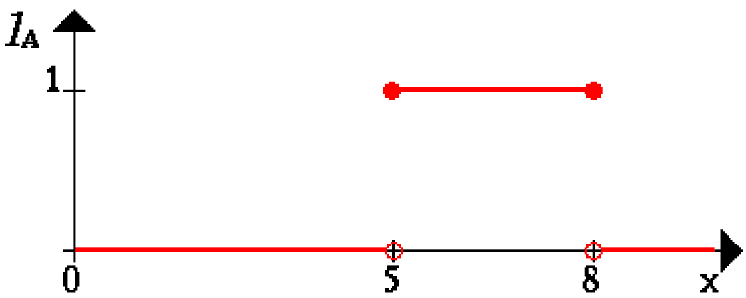
In classical mathematics we are familiar with what we call *crisp sets.*

Here is an example:

First we consider a set *X* of all real numbers between 0 and 10 which we call the universe of discourse. Now, let's define a subset *A* of *X* of all real-numbers in the range between 5 and 8.

`A = [5,8]`

We now show the set *A* by its characteristic function, i.e. this function assigns a number 1 or 0 to each element in *X*, depending on whether the element is in the subset *A* or not. This results in the following figure:



We can interpret the elements which have assigned the number 1 as *The elements are in the set A* and the elements which have assigned the number 0 as *The elements are not in the set A.*

This concept is sufficient for many areas of applications. But we can easily find situations where it lacks in flexibility. In order to show this, consider the following example on the next page:

In this example we want to describe the set of young people. More formally we can denote

B = {set of young people}

Since - in general - age starts at 0 the lower range of this set ought to be clear. The upper range, on the other hand, is rather hard to define. As a first attempt we set the upper range to, say, 20 years. Therefore we get *B* as a crisp interval, namely:

B = [0,20]

Now the question arises: why is somebody on his 20th birthday *young* and right on the next day *not young*? Obviously, this is a structural problem, for if we move the upper bound of the range from 20 to an arbitrary point we can pose the same question.

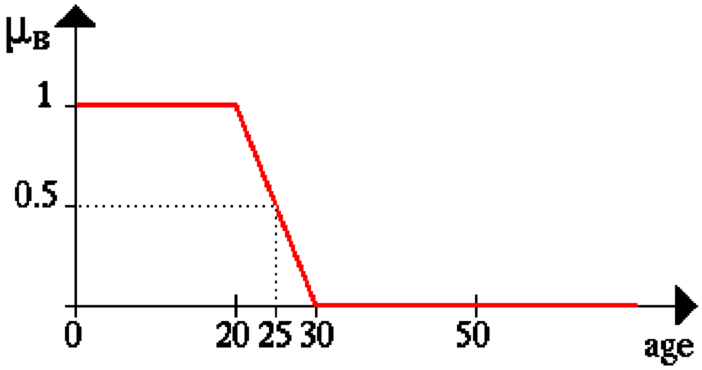
A more natural way to construct the set *B* would be to relax the strict separation between *young* and *not young*. We will do this by allowing not only the (crisp) decision *YES he/she is in the set of young people* or *NO he/she is not in the set of young people* but more flexible phrases like *Well, he/she belongs a little bit more to the set of young people* or *NO, he/she belongs nearly not to the set of young people.*

The next page shows how a fuzzy set allows us to define such a notion as *s/he is a little young*.

As stated in the introduction we want to use fuzzy sets to make computers smarter, we now have to code the above idea more formally. In our first example we coded all the elements of the Universe of Discourse with 0 or 1. A straight way to generalize this concept is to allow more values between 0 and 1. In fact, we even allow infinite many alternatives between 0 and 1, namely the unit interval $I = [0, 1]$.

The interpretation of the numbers now assigned to all elements of the Universe of Discourse is much more difficult. Of course, again the number 1 assigned to an element means that the element is in the set *B* and 0 means that the element is definitely not in the set *B*. All other values mean a gradual membership to the set *B*.

To be more concrete we now show the set of young people similar to our first example graphically by its characteristic function.



This way a 25 years old person would still be *young* to a **degree of 50 percent**.

Now you know what a **fuzzy set** is. But what can you do with it? Continue to the next lecture!