

Design of fuzzy systems

Andrea Bonarini



Artificial Intelligence and Robotics Lab
Department of Electronics, Information, and Bioengineering
Politecnico di Milano



E-mail: andrea.bonarini@polimi.it
URL: <http://www.deib.polimi.it/people/bonarini>

Steps for fuzzy system design

- Problem definition
- Parametrization of the model: concepts
- Mapping definition: rules
- Implementation
- Testing

Problem definition

Analogous to what done in classical model design

- Selection of the **input** variables
- Selection of the **output** variables
- Selection of the **goals** of the model

Selection of input variables

In principle, numerical or ordinal variables so that it is possible to define fuzzy sets

Variables that can be

- “perceived” from **sensors**, **data** or **users**
- computed from perceived variables (error, derivatives, composition of variables)

In general, there are no a priori preclusions to select variables: it depends on the problem and the designer sensibility

Selection of output variables

Output variables are the results of the model, so come directly from the modeler needs.

For instance:

- for a control application, we might either have the control variables, or the respective increments
- for a decision support system we might have the decision to be taken

Selection of the goals of the fuzzy model

The goals of the fuzzy models depend on the specifications

For instance, a control system may reach the set point as soon as possible, with the least overshooting, as robustly as possible, etc.

The goals should always be stated in advance, and guide the design.

System parametrization

- Selection of membership functions for all variables
- Selection of the inferential mechanism (which T- norms...)
- Selection of eventual fuzzyfication and defuzzyfication

Membership function selection

- Number of granules: not too many, nor too few (7?)
- Coverage: an output for any input value
- Boundary considerations
- How to reach equilibrium?
- Value density: more attention where needed
- Crosspoint: smoothness in the output
- Singletons or more general MFs for the output

How many MFs?

It depends on the application.

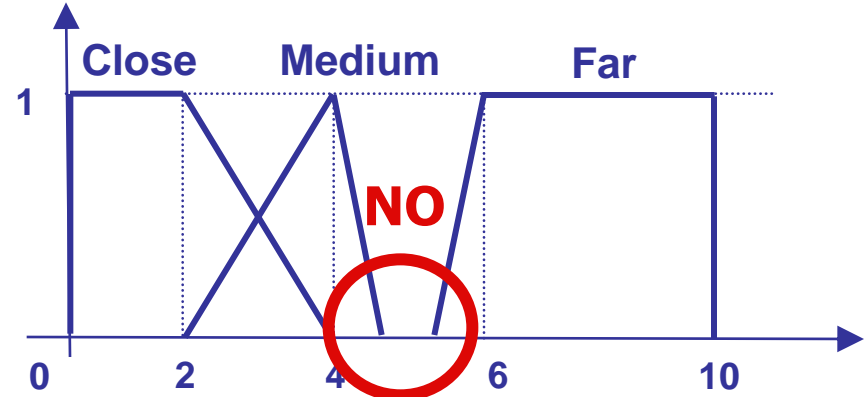
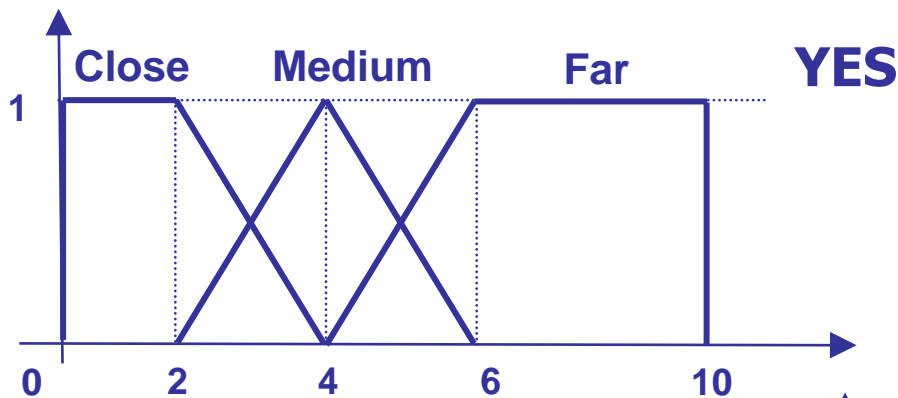
In general from 3 to 7.

Why?

The “magic” number 7: people usually cannot manage more than 7 ± 2 concepts at a time

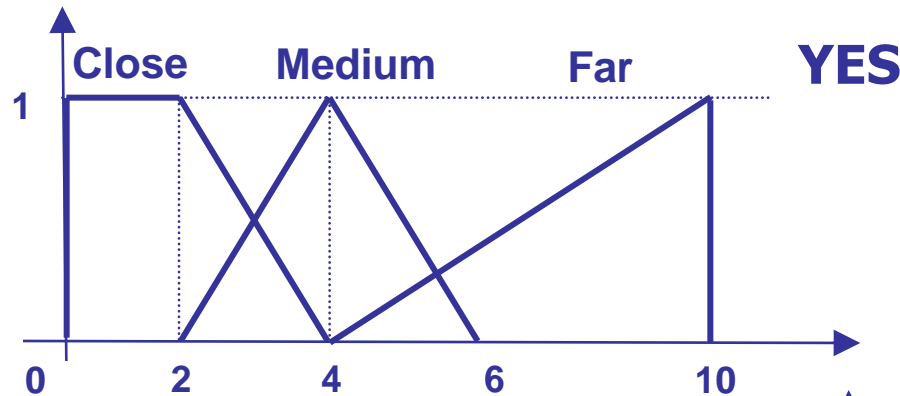
An output for each input

Any point in the range of input variables has to be covered by at least one fuzzy set participating to at least one rule

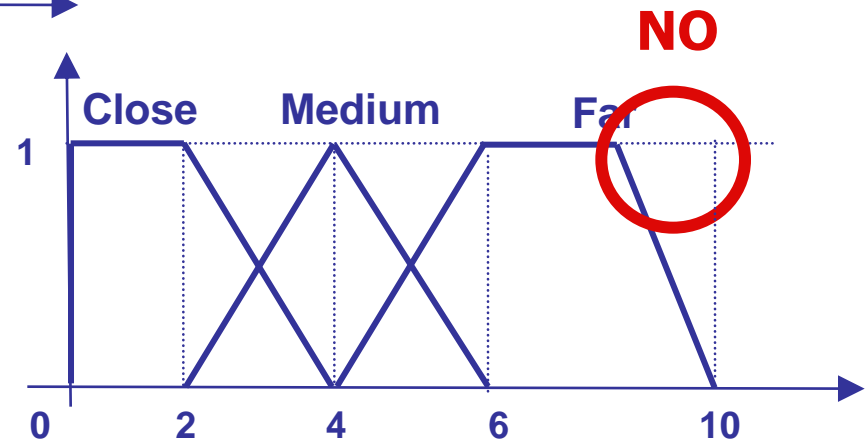


Boundary considerations

Boundary should be covered with maximum value

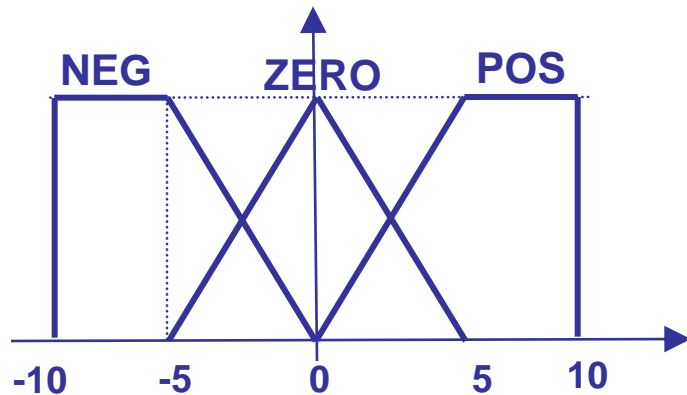


Saturation or not?



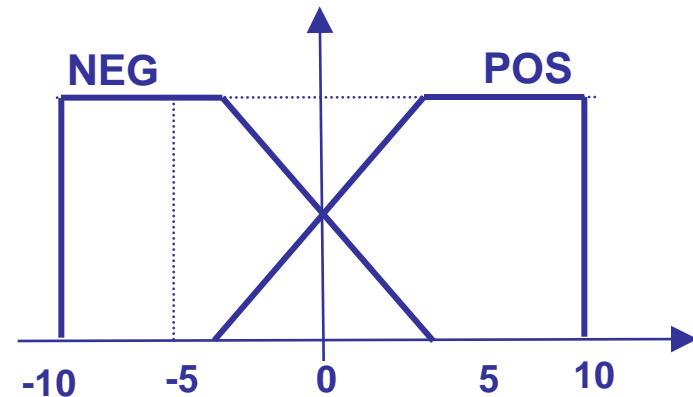
How to reach equilibrium

A mediating MF



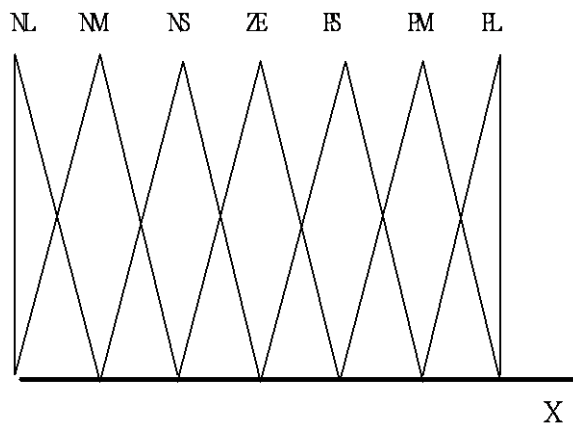
Just pushing...

IF (X is POS) THEN (U is NEG)
IF (X is NEG) THEN (U is POS)
IF (X is ZERO) THEN (U is ZERO)



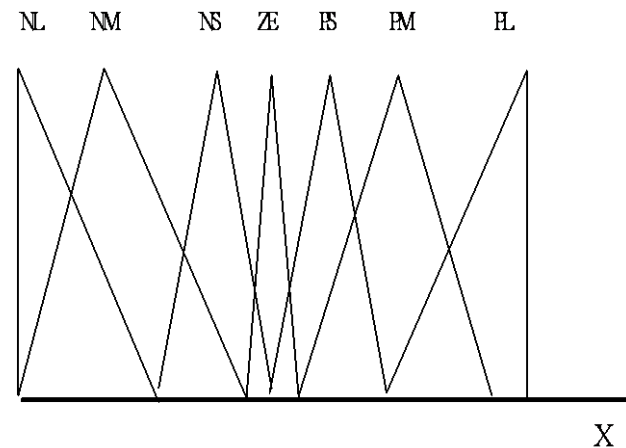
How to distribute the MFs

Evenly distributed



Max robustness to noise

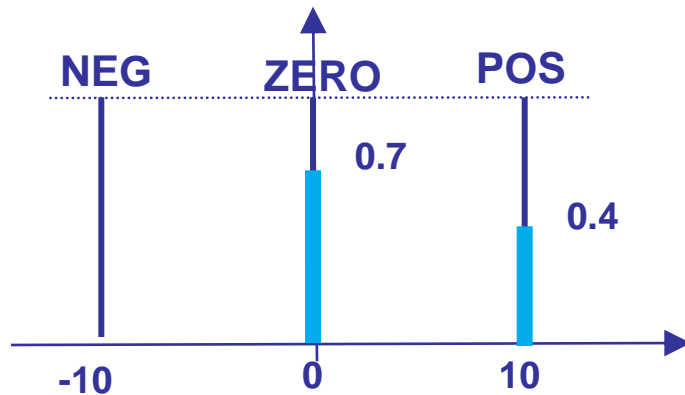
Unevenly distributed



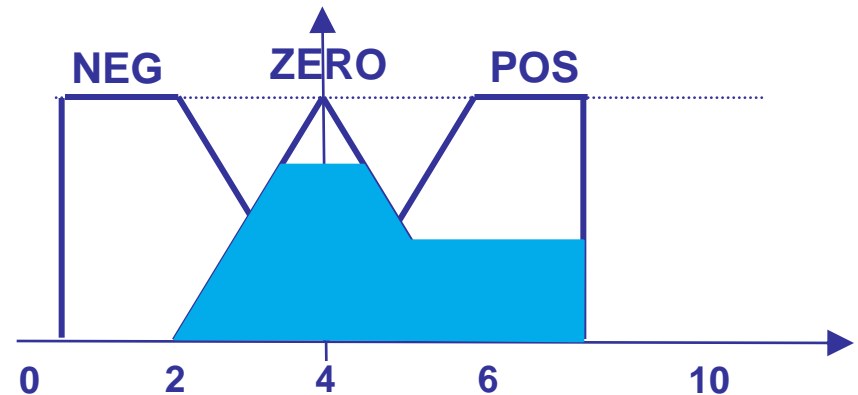
More precision where needed

Singletons or other for the output

Singltons: the weight of any cut is proportional to the cut level



Shapes: the relative weight is higher for lower cuts



How are MFs defined?

Single expert

- Objective evaluation (e.g.: error wrt a set point)
- Interview (e.g.: operator of a control room)

Multiple experts

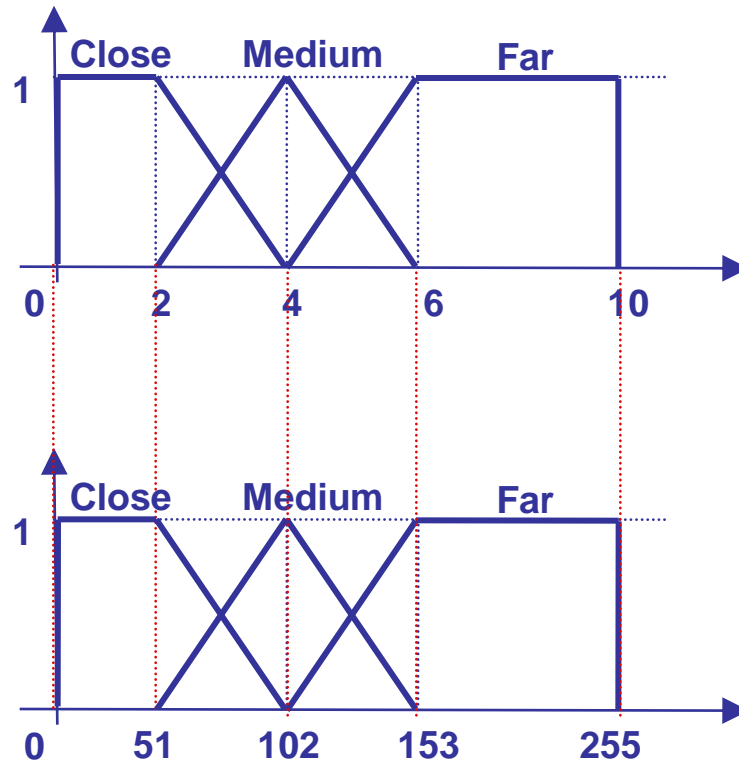
- Probabilistic elaboration, possibly weighted by the expert reliability

Automatic systems (NN, GA, ...) working on data

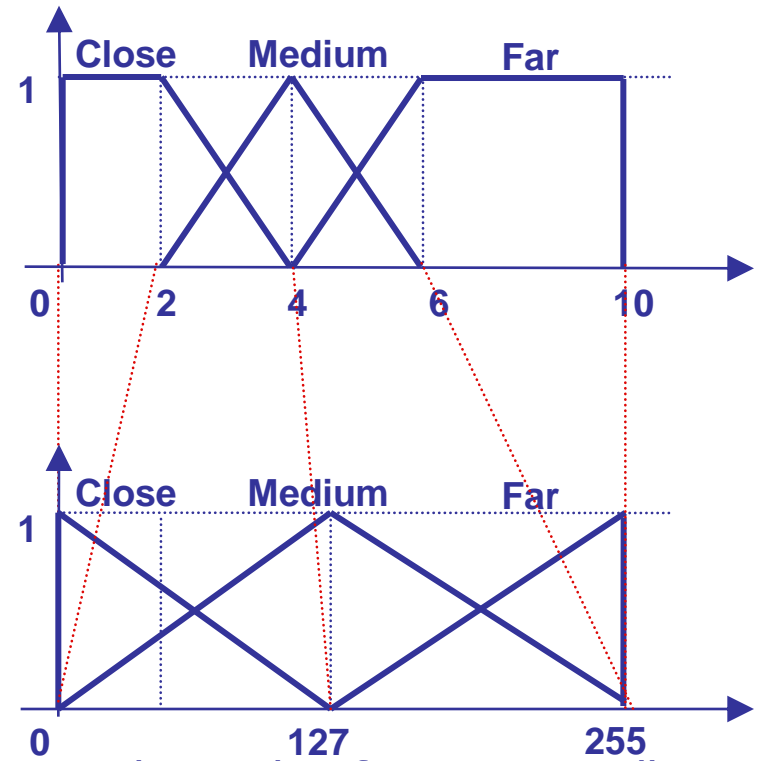
Normalization

It is common to normalize data on a standard range (e.g., 0 – 255)

Linear normalization



NON-linear normalizaton

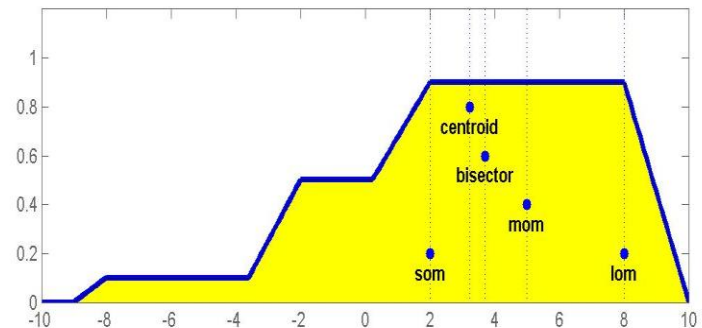
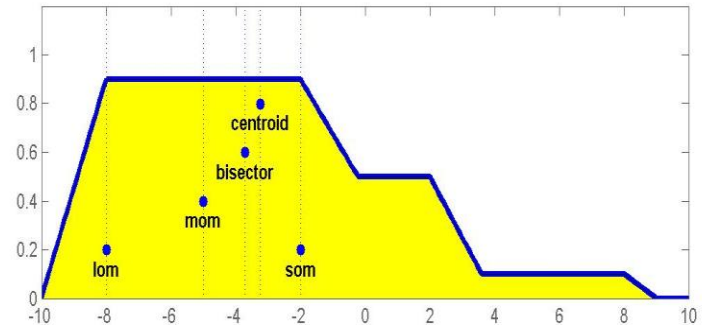


mainly used in fuzzy μ controllers
to exploit the available precision

Defuzzification

Many different possibilities:

- Centroid
- Bisector
- Average of maxima
- Lowest maximum
- Highest maximum
- Center of the highest area
- ...



Rule definition

From experience

- Introspective analysis
- Structured interview

From another model (e.g., a mathematical model)

By using machine learning, or self-tuning techniques (NN, GA, ...)

Selection of inferential engine

The inferential engine depends on the operators selected for:

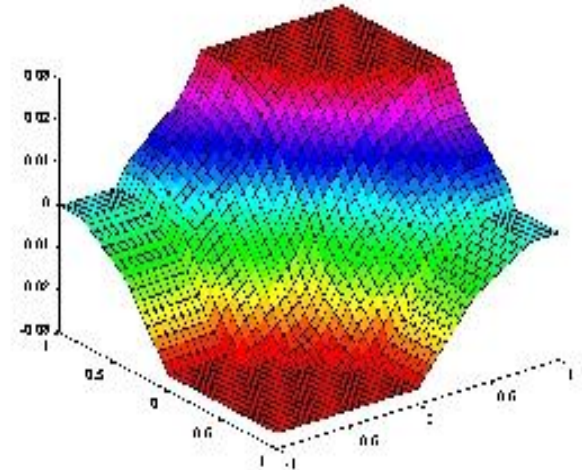
- **AND** of antecedent clauses
 - min: the worst degree of matching is the most relevant
 - product: all the degrees of matching are relevant
- **Detachment**: combination with the rule weight
 - min
 - product
- **Aggregation** of degrees of the same consequent
 - max: the best degree is the most relevant
 - probabilistic sum: all the collected knowledge is taken into account

Testing

Aim: verify the design goals

Possible activity

- Dynamic simulation, if the model of the subject is available
E.g., if the model of the controlled system is available
- "Static" simulation (single I/O check, output (control) surface study)
- Test on the process, possibly under safe conditions



Available development tools

Many tools, mainly for **fuzzy control** (including Matlab)

Common features:

- guided definition of rules and MF
- visualization of control surfaces
- suite of MFs, operators and defuzzification methods
- support to testing
- support to learning
- optimized code production, for many processors

Some tools for the development of **generic fuzzy rule systems**:

Xfuzzy, FOOL, Fuzzy Clips, FLIP, FCL, ...

FCL: an IEEE standard

IEEE-IEC document 61131-7

An example: model the aggressiveness of a character in a videogame

```
FUNCTION_BLOCK
```

```
\* VAR definition *\
```

```
VAR_INPUT
```

```
    Our_Health    REAL; (* RANGE(0 .. 100) *)
```

```
    Enemy_Health  REAL; (* RANGE(0 .. 100) *)
```

```
END_VAR
```

```
VAR_OUTPUT
```

```
    Aggressiveness REAL; (* RANGE(0 .. 4) *)
```

```
END_VAR
```

FCL Example (2): MF definition

* MF definition *\

FUZZIFY Our_Health

TERM Near_Death := (0, 0) (0, 1) (50, 0) ;

TERM Good := (14, 0) (50, 1) (83, 0) ;

TERM Excellent := (50, 0) (100, 1) (100, 0) ;

END_FUZZIFY

FUZZIFY Enemy_Health

TERM Near_Death := (0, 0) (0, 1) (50, 0) ;

TERM Good := (14, 0) (50, 1) (83, 0) ;

TERM Excellent := (50, 0) (100, 1) (100, 0) ;

END_FUZZIFY

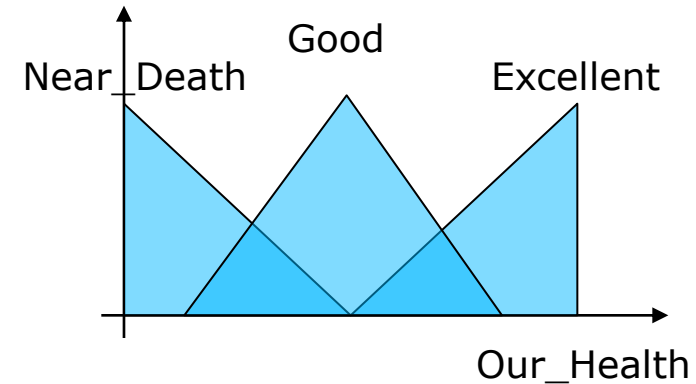
FUZZIFY Aggressiveness

TERM Run_Away := 1 ;

TERM Fight_Defensively := 2 ;

TERM All_Out_Attack := 3 ;

END_FUZZIFY



FCL Example (3): defuzzification

* Definition of the defuzzification method *\

DEFUZZIFY Aggressiveness

METHOD: MoM; \\Media of Maxima

END_DEFUZZIFY

FCL Example (4): rule definition

```
RULEBLOCK first
AND:MIN;
ACCU:MAX;
  RULE 0: IF (Our_Health IS Near_Death) AND (Enemy_Health IS Near_Death)
  THEN (Aggressiveness IS Fight_Defensively);
  RULE 1: IF (Our_Health IS Near_Death) AND (Enemy_Health IS Good)
  THEN (Aggressiveness IS Run_Away);
  RULE 2: IF (Our_Health IS Near_Death) AND (Enemy_Health IS Excellent)
  THEN (Aggressiveness IS Run_Away);
  RULE 3: IF (Our_Health IS Good) AND (Enemy_Health IS Near_Death)
  THEN (Aggressiveness IS All_Out_Attack);
  RULE 4: IF (Our_Health IS Good) AND (Enemy_Health IS Good)
  THEN (Aggressiveness IS Fight_Defensively);
  RULE 5: IF (Our_Health IS Good) AND (Enemy_Health IS Excellent)
  THEN (Aggressiveness IS Fight_Defensively);
  RULE 6: IF (Our_Health IS Excellent) AND (Enemy_Health IS Near_Death)
  THEN (Aggressiveness IS All_Out_Attack);
  RULE 7: IF (Our_Health IS Excellent) AND (Enemy_Health IS Good)
  THEN (Aggressiveness IS All_Out_Attack);
  RULE 8: IF (Our_Health IS Excellent) AND (Enemy_Health IS Excellent)
  THEN (Aggressiveness IS Fight_Defensively);
END_RULEBLOCK
END_FUNCTION_BLOCK
```

HW for fuzzy systems

PC and assembly, or C (C++, Java, ...) code

Standard microcontroller (compiled fuzzy code)

Fuzzy HW

- Fuzzy processor: almost only a RAM (everything is precomputed) with data acquisition (8 to 12 bit) and serial interface
- Standard microcontroller augmented with fuzzy instructions in the instruction set (e.g., fuzzyfy, defuzzyfy, ...)
- Fuzzy processor integrated with other devices (e.g., pwm generator, ...)

What to remember from these slides?

- **General design principles**
- **Why selecting a set of operators or another?**
- **How can we design a set of MFs (distribution shape)?**