Fifth IFSA World Congress (1993), 1266-1269

FUZZY PROCESSING BASED ON ALPHA-CUT MAPPING

Adrian Stoica

Dept of Electrical & Electronic Engineering
Victoria University of Technology
PO Box 14428, MMC, Melbourne, Vic 3000, Australia
E-mail: adrian=stoica@vut.edu.au

Abstract

The paper introduces a new method for fuzzy processing. The method allows handling a piece of information lost in the classic fuzzification process, and thus neglected by other methods. Processing the result after fuzzification is sustained by the interpretation that the input-output set mapping, specified by the IF-THEN rules, can be regarded as a direct mapping of their corresponding alpha-cuts. Processing involves just singletons as intermediary results, the final result being a combination of singletons obtained from fired rules.

1. INTRODUCTION

A fuzzy set A on the universe of discourse X, is the set of ordered pairs [1]

$$A = \{(x, \mu_A(x)) \mid x \epsilon X\}. \tag{1}$$

We consider in the following $\mu_A(x) \epsilon [0,1]$.
In classic "fuzzification" operation, values from the universe of discourse are assigned membership degrees to fuzzy sets covering the universe. For convex fuzzy sets, which feature non-sharp boundaries on both sides, there are always two input values (one at each side of the prototype) which have the same degree of membership to the set. Consider for example the fuzzy sets Young (Y), Middle Aged (MA) and Old (O), with MA centred around 45 years old. Let us consider that a 50 years person belongs to the set MA in a degree of 0.8. There is another age, assume 40, for which the degree of membership is also 0.8. Thus, the value of 0.8 might be a sufficient characterisation of the degree of membership of a value to a set, but is not embedding the complete information provided by the input value, since we are not able to uniquely specify the age from {0.8, MA}.

Some information seems to be lost. If a complete overlap of transitory regions for two neighbouring sets exists (which is not the case in all reported applications, e.g. [2], [3]) then the complete initial information is better found in the conjunction "0.8 MA AND 0.2 O".
Let us observe that we are still unable to deduce the initial value using the combination of sets and defuzzification (which is not the exact reverse of fuzzification). It is the case that for an identical fuzzy system, i.e. mapping identically fuzzy sets on themselves (e.g. Low->Low, Medium->Medium, High->High), the input-output characteristic is generally nonlinear, with the nonlinearity being ***arbitrarily*** introduced by the processing method (e.g. MIN-MAX-COG).

We suggest here that, instead of using {0.8, MA}, we should use the form {0.8, MA, O} corresponding to the input value equal to 50, reading for example as 0.8 Middle Aged towards Old. Thus, ***through fuzzification we'll specify not only the degree of membership to the set, but also the position relative to the prototype, in the set of ordered pairs.***

2. ALPHA-CUT SET BASED INFERENCE

2.1. Mapping α-cuts

The alpha cut (α-cut) A_α and the strong alpha cut A'_α are defined by (2), (3) [1].

$$A_\alpha = \{x \epsilon X \mid \mu_A(x) \geq \alpha\} \tag{2}$$

$$A'_\alpha = \{x \epsilon X \mid \mu_A(x) > \alpha\} \tag{3}$$

The fuzzy set A could be represented as a union of its α-cuts (representation theorem [4]):

$$A = \bigcup_{\alpha \epsilon [0,1]} \alpha A_\alpha \tag{4}$$

Consider a fuzzy system, with two input and one output variables, and the rules specified expressed by:

$$\text{IF A and B THEN C} \tag{5}$$

specified in the following by: r[A,B] = C, with A, B, C fuzzy sets (graphical correspondence in fig. 1)

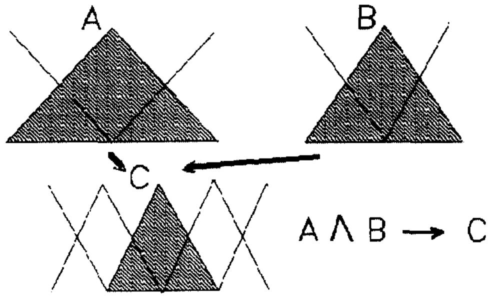

Fig. 1 Mapping fuzzy sets

Using the representation theorem this could be interpreted as:

$$\text{IF} \quad \Big(\bigcup_{\alpha_a \in [0,1]} \alpha_a A_{\alpha a} \Big) \wedge \Big(\bigcup_{\alpha_b \in [0,1]} \alpha_b B_{\alpha b} \Big)$$

$$\text{THEN} \quad \bigcup_{\alpha_c \in [0,1]} \alpha_c C_{\alpha c} \qquad (6)$$

Classic inference is related with fig. 2 and in terms of α-cuts this can be stated as:

$$\bigcup_{\alpha_a \in [0,\alpha a]} \alpha_a A_{\alpha a} \wedge \bigcup_{\alpha_b \in [0,\alpha b]} \alpha_b B_{\alpha b} \rightarrow \bigcup_{\alpha_c \in [0,Min(\alpha a,\alpha b)]} \alpha_c C_{\alpha c} \qquad (7)$$

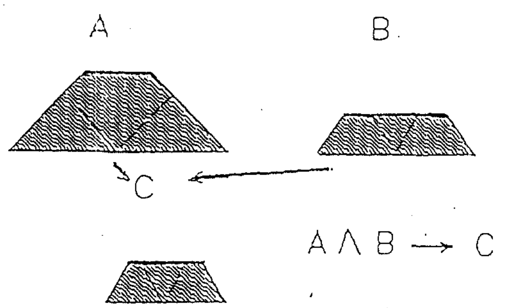

Fig. 2 Classical interpretation for set mapping

We consider the basic mapping to be the mapping of α-cuts:

$$A_{\alpha a} \wedge B_{\alpha b} \rightarrow C_{Min(\alpha a,\alpha b)} \qquad (8)$$

as suggested in fig.3.

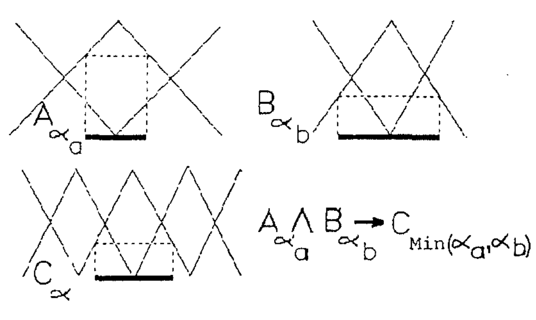

Fig. 3 Interpretation in terms of α-cut mapping

This can be interpreted as a mapping of intervals of confidence, with the intervals of confidence being ordinary subsets of R representing a type of uncertainty [5].
Although Min is used in the exemples, in general, any t-norm could be acceptable.

2.2 Mapping α-cut borders

Consider the difference between the two sets A_α and A'_α, the alpha-cut and the strong alpha-cut.

$$\text{Bd } A_\alpha = \{a, a^*\} = A_\alpha - A'_\alpha \qquad (9)$$

This delimits the borders of the interval of confidence α.
An input value x corresponds to one of the two borders of an α-cut, the fuzzification process specifying now the α-cut ($\alpha=\mu_A(x)$) and which of the two borders of the α-cut is addressed, e.g. by indicating the neighbouring fuzzy set towards the prototype of which the border is oriented. Thus the notation $\{\alpha 1$, set1, set2$\}$ will indicate the border of the $\alpha 1$-cut of set1 on the side of the prototype of set2.
From alpha-cut mapping we will continue with the mapping of alpha-cut borders.

Consider the simple case of sets in fig.4 and the rules given by (10):

r[A,B] = C
r[A',B] = C' (10)
r[A,B'] = C'
r[A',B'] = C"

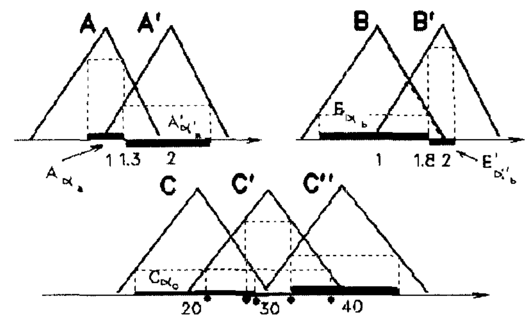

Fig. 4 Mapping borders of α-cuts

Consider the input values be a and b.
a-> $\{\alpha A$, A, A'$\}$, $\{\alpha A'$, A', A$\}$
b-> $\{\alpha B$, B, B'$\}$, $\{\alpha B'$, B', B$\}$

$$\begin{aligned} A_{\alpha a} &\wedge B_{\alpha b} \rightarrow C_{Min(\alpha a,\alpha b)} \\ A_{\alpha a} &\wedge B'_{\alpha b} \rightarrow C'_{Min(\alpha a,\alpha' b)} \\ A'_{\alpha a} &\wedge B_{\alpha b} \rightarrow C'_{Min(\alpha' a,\alpha b)} \\ A'_{\alpha a} &\wedge B'_{\alpha b} \rightarrow C''_{Min(\alpha' a,\alpha' b)} \end{aligned} \qquad (11)$$

Let us explain in more detail the inference related to the first rule in (11).
The rule refers to the mapping of α_a- cut of A and α_b-cut of B, to the α_c- cut of C. The input value a is the border of α_a- cut set towards the prototype of A'.
a-> $\{\alpha A$, A, A'$\}$, b is the border of α_b- cut set of B towards the prototype of B',
b-> $\{\alpha B$, B, B'$\}$.

The result infered through the rule is the border of alpha-cut of C (r[A,B]=C) towards the prototype of the set specified by rules as being associated to A' ∧ B' that is C" (r[A',B']=C").
A_1, A'_1, B_1, B'_1, C_1, C'_1, C''_1, (1-cuts) are prototypes.

{αA, A, A'} ∧ {αB, B, B'} ->
c^1 = {Min[αA,αB],r[A,B],r[A',B']}, border of α-cut in C (r[A,B]=C) towards the prototype C''_1 (which corresponds to A'_1 ∧ B'_1, r[A',B']=C")

{αA, A, A'} ∧ {αB, B', B} ->
c^2 = {Min[αA,αB],r[A,B'],r[A',B]}

{αA, A', A} ∧ {αB, B, B'} ->
c^3 = {Min[αA,αB],r[A',B],r[A,B']}

{αA, A', A} ∧ {αB, B', B} ->
c^4 = {Min[αA,αB],r[A',B'],r[A,B]}

The final output value can be calculated as a weighted (with corresponding α^i) average of c^i, or as a simple average as prefered in the following,

$$c = \sum_{i=1,n} c^i / n \qquad (12)$$

A numerical example:
a=1.3, b=1.8 (fig.4).
α_a = 0.7, α'_a = 0.3, α_b = 0.2, α'_b = 0.8.
c^1 select in {12,28} towards 40, c^1 = 28
c^2 select in {27,33} towards 30, (both) c^{21} = 27, c^{22} = 33
c^3 select in {22,38} towards 30, (both) c^{31} = 22, c^{32} = 38
c^4 select in {33,47} towards 20, c^1= 28. c^3 = 33, using (12) with c^{21} and c^{22}, c^{31} and c^{32}, contributing independently, we obtain c = 30.16.

3. A PARALLEL PRESENTATION OF CLASSIC AND ALPHA-CUT BASED FUZZY PROCESSING

Classic method:

fuzzify[x_,set$_i$_]:={μ,set$_i$};

inference[{μ1_,set1_},
{μ2_,set2_}]:=
{Min[μ1,μ2], r[set1,set2]}; (13)

Alpha-cut based method:

fuzzify[x_,set$_i$_]:={α,set$_i$_,set$_{i\pm1}$};

inference[{α1_,set1_,set1'_},
{α2_,set2_,set2'_}]:=
{Min[α1,α2],r[set1,set2],r[set1',set2']}; (14)

Apart of the similarity of the above formalism, the values in the two approaches have different meanings (as differentiated by fig. 3 and fig.4, and equations (7) and (8). After (13), and the combination of results from all fired rules, we need to adopt a method for obtaining a crisp value, method which is not using the same convention as used in fuzzification, while after (14) we can use the same convention as we the one we used in fuzzification.

4. SIMULATIONS

The viability of the proposed method was tested on the vertical stabilisation of the inverted pendulum. The parameters for error, change in error and current are the ones used in TILShell+ demo [8]. The fuzzy system is pictured in fig. 5. The corresponding membership functions are presented in fig. 6. The rules are presented in table 1. System response using the α-cut based method is compared with the one obtained using Min-Max-COG in fig 7. It is observed that the α-cut based method gives a better performance in the first part of the response. This is easy to predict if we observe that for large values of Error as it is the case in the first moments, COG defuzzification outputs a value corresponding to the COG of PM, which is less than the the value of the prototype obtained using α-cut processing. It was difficult to outperform another method for which the membership functions have been tuned to obtain a control suiting the system to be controlled. However even in this conditions the proposed method performed satisfactory.

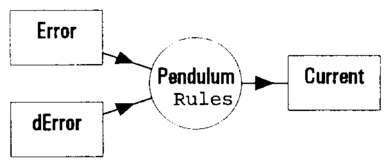

Fig. 5 Fuzzy controller for inverted pendulum

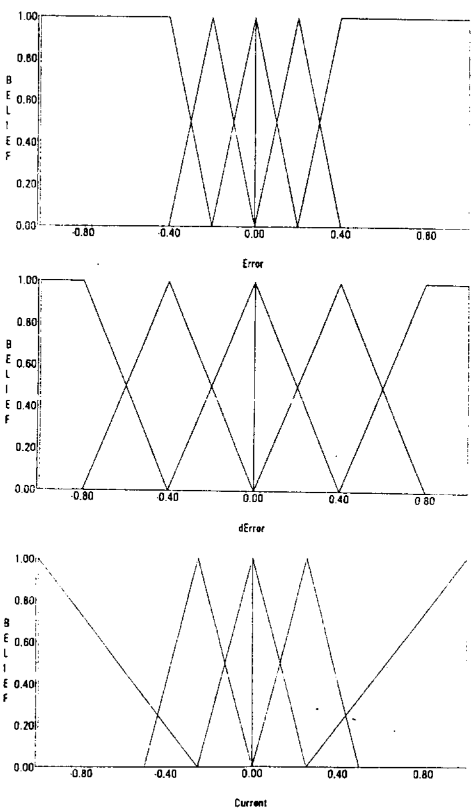

Fig. 6 Membership functions for error, change in error and current

Tabel 1. Rules for controlling the pendulum

dE\E	NM	NS	ZR	PS	PM
NM	PM	PM	PM	PS	ZR
NS	PM	PS	PS	ZR	NS
ZR	PM	PS	ZR	NS	NM
PS	PS	ZR	NS	NS	NM
PM	ZR	NS	NM	NM	NM

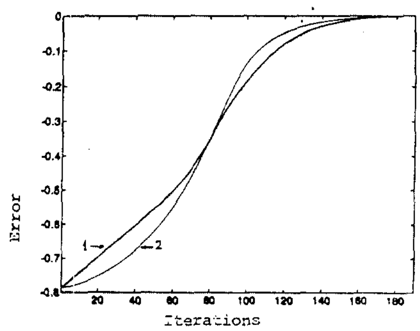

Fig. 7 System response: 1 - α-cut method, 2 - Min-Max-COG

5. DISCUSSION

There are many degrees of freedom we can handle in optimising the performance of a fuzzy system. For a particular inference method, we can modify the membership functions or/and the rules. Tuning could be manual, or automatic using different learning techniques. Zimmermann [1] enumerates eight important criteria for selecting appropriate aggregation operators for fuzzy sets. The literature offers studies on the effect of different inference or defuzzification methods on system's output [6],[7]. However, it is difficult to compare two fuzzy processing methods, since with proper membership functions adjustment and rule selection, the corresponding control surface can be shaped to almost any form. If we "freeze" the membership functions and the rules one method can perform better, but how can we be sure we had frozen the most appropriate selection? An other selection for membership functions and rules might lead to another "better" suited inference method.

A good criteria could be a time-based one, considering both time for tunning the system and time for processing the information.
The "good" fuzzy processing method should be the one which best matches two conditions:
1. Leads to good results for the initial description acquired from an expert, without extensive tunning.
2. It is fast.

In relation with the above conditions, in respect to the first one, we have the argument that the here proposed method is making better use of the information provided on the system, including in processing a piece of information neglected by other methods from the moment of fuzzification.
For the second requirement, in software simulations the method is faster than methods which require defuzzification of an output set, having approximately the speed of methods that defuzzify using prototypes (singletons). A hardware perspective will be the subject of another paper.

6. CONCLUSION

A new method for fuzzy processing is introduced, having the following characteristics: 1) Allows the handling of a piece of information lost in classic fuzzification stage, 2) Offers an interpretation in the sense that input membership functions in terms of their α-cuts are mapped to output membership functions in terms of their α-cuts, according to the if-then rules, i.e. a mapping of intervals of confidence, 3) It is simple and faster in software implementations comparing with methods that use defuzzification of an output set for deriving a final crisp value, offering comparative results in the tests performed.

REFERENCES

[1] Zimmermann, H. J. (1985) 'Fuzzy Set Theory and its Applications', Kluwer-Nijhoff Publishing

[2] Ollero, A. and Garcia-Cerezo, A. J. (1989) 'Direct Digital Control, Auto-Tuning and Supervision Using Fuzzy Logic'. Fuzzy Sets and Systems 30, 135-153

[3] Okada, H., et al (1992) 'Knowledge Implementation Multilayer Neural Networks with Fuzzy Logic'. Proc of the 2nd Int Conf on Fuzzy Logic & Neural Networks (Iizuka, Japan, July 17-22, 1992) pp. 99-102

[4] Negoita, C. V. and Ralescu, D. A. (1975) 'Applications of Fuzzy Sets to System Analysis'. Birkhauser Verlag, Basel.

[5] Kaufmann, A. and Gupta, M. M. (1988) 'Fuzzy Mathematical Models in Engineering and Management Science'. Elsevier, Amsterdam.

[6] Mizumoto, M.(1991) 'Min-Max-Gravity Method versus Product-Sum- Gravity Method for Fuzzy Controllers'. In Proc of the 4th IFSA Congress, (Engineering) Brussels, 1991 pp. 127-130.

[7] Lembessis, E. and Tanscheit, R. 'The Influence of Implication Operators and Defuzzification Methods on the Deterministic Output of a Fuzzy Rule-Based Controller' In Proc of the 4th IFSA Congress, (Engineering) Brussels, 1991 pp. 109-114.

[8] TogaiInfraLogic (1992) TIL Shell+