Military Technical College
Kobry El-Kobbah
Cairo, Egypt

10th International Conference
On Aerospace Sciences&
Aviation Technology

DESIGN OF A FUZZY LOGIC CONTROL ENGINE USED IN CONTROLLING ANESTHESIA DELIVERY

S. M. ALIAN *, M. A. ELKFAFI *, and M. A. HUSSEIN *

ABSTRACT

This paper aims to design a fuzzy logic control engine. The inputs of this proposed engine are the heart rate, blood pressure and the primary depth of anesthesia obtained from mid-latency auditory evoked response (MLAER). The output is a parameter applied to the syringe pump to control the amount of the delivered drug to the patient during surgical operation. The operation of the proposed system is based on the experience of anesthetists. Good results have been obtained for controlling the amount of drug.

KEYWORDS

Fuzzy Logic, Fuzzy Logic Control System, Fuzzy Rule, Fuzzy Engine, Anesthesia, Depth of Anesthesia (DOA), PDOA, Heart Rate (HR), Blood Pressure (BP), Auditory Evoked Response, Mid-Latency Auditory Evoked Response (MLAER)

* Egyptian Armed Forces.

INTRODUCTION

Both sleep and general anesthesia are states of unresponsiveness which vary in depth. Sleep is natural and healthy [1], whereas anesthesia is an artificial state maintained by the continuing presence of chemical agents in brain. Anesthesia is the art or science of removing sensations of and reactions to a surgical procedure. Anesthesia means loss of all modalities of sensation whether it be the sense of pain, touch, temperature, or position sense.

Ideally, anesthesia should rapidly achieve a depth appropriate for surgery. Anesthesia which is too light is associated with movement, muscle rigidity, cardio-vascular instability, and worse, patient awareness which may cause serious distress at the time and in the long-term lead to chronic physiological disturbances. Likewise, excessive anesthesia can be harmful producing depression of the cardio-vascular and the respiratory systems, and prolonging the recovery period. Thus the signs of anesthetic depth are very important. So, depth of anesthesia can be viewed as the position on a scale of a pointer which reflects the balance between the depressant effects of anesthetics on the central nervous system and the stimulation by sensory events such as surgery [2]. Depth of anesthesia is much harder to define and not readily measurable. A desired feature of any anesthetic depth index is that it can be used to control the anesthesia automatically via a fuzzy control system.

Control itself is defined as the manipulation of the inputs of a system, to produce a desired output. There are two types of control systems: open loop control systems and closed lop control system. An open loop control system implies that the response of the system to its inputs is not incorporated into future inputs, i.e. there is no feedback. Closed loop control system uses feedback from the output of the system being controlled to provide improved accuracy [3]

Zadeh proposed fuzzy sets [4] because of his dissatisfaction with increasing precision in control theory. However, fuzzy logic control (FLC) research was started by Mamdani's pioneering work.

Fuzzy logic provides a practical and inexpensive solution for controlling complex or ill-defined systems. It can handle the concept of partial truth. It resembles human decision making with its ability to work from approximate data and find precise solutions [5].

Here we build up a real fuzzy logic control engine that is to be used in any application. We then apply the engine in anesthesia delivery

FUZZY ENGINE REQUIREMENTS AND TOP LEVEL DESIGN

The fuzzy logic control engine implemented here is designed to be flexible and extendable. This way any number of fuzzy inputs as well as fuzzy rules can be supported easily. The fuzzy logic control parameters in the engine should also be modified easily. The following considerations were taken in the design of the fuzzy logic control system engine:

- Open number of fuzzy input variables.
- Open number of fuzzy sets per fuzzy input variable.
- Open number of fuzzy sets per fuzzy output variable.
- Easy way for representing the membership function for input / output fuzzy variables.
- Open number of fuzzy rules.
- Easy way for representing the set of fuzzy rules.

The representation of the membership function of a fuzzy set in a fuzzy input or output variable is achieved using a simple method. This method assigns a set of ordered pairs to build up this membership function. Each pair contains a value from the universe of discourse of the fuzzy variable and the degree of membership of this value to the set. Each ordered pair represents a change in the way of the membership function that we can call simply a *knee*. The engine is responsible for determining the degree of membership of any other value that could be introduced to the engine between the values in the set of ordered pairs. This is achieved by linear interpolation in each period between two adjacent knees. This way, any arbitrary membership function can be easily obtained by adding or modifying the set of ordered pairs for this fuzzy set as shown in Fig. 1.

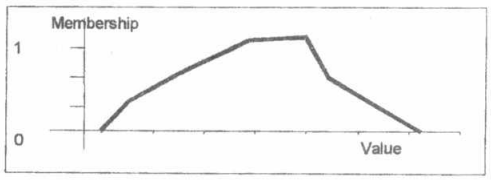


Fig. 1. Arbitrary member function representation

For example: to buildup the fuzzy set "Mid Heart Rate" of the fuzzy variable "Heart Rate" three ordered pairs are used: (50, 0), (70, 1), and (90, 0) as shown in Fig. 2.

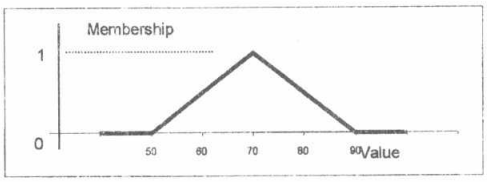


Fig. 2. Triangular member function representation

The first ordered pair specifies that at the value of 50 in the universe of discourse of the crisp input the degree of membership to this set will be 0 i.e. no membership at all. This also means that any value less than 50 will be 0. The second ordered pair gives the meaning that the value 70 of the heart rate crisp input gives full membership to the set "Mid Heart Rate". To determine the degree of membership to the set in the period between 50 and 70 of the crisp input the engine will interpolate. So it is obvious that the degree of membership of the value 60 to the set will be 0.5 and the degree of membership of the value 65 will be 0.75 and so on. The third ordered pair of the set specifies that at the value 90 again there will be no membership to the set again. And this will be the case for any input higher than this value.

We can find out that this set of ordered pairs construct a triangle membership function. If we wanted to change the shape of this membership function to be trapezoidal for example we can add another ordered pair (or a knee). The set of ordered pairs will be (50, 0), (65, 1), (75, 1), (90, 0) as shown in Fig. 3.

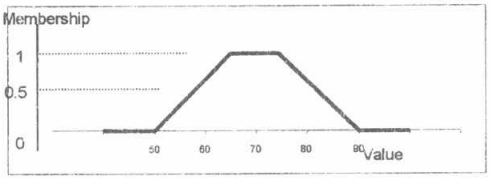


Fig. 3. Trapezoidal member function representation

The fuzzy rules are represented in the system as a set of vectors. A fuzzy rule is a vector of values that are the premises for every fuzzy input variable. The last value in the vector is the conclusion or the result. Each premise, which is a fuzzy set label or a part from the if statement of the rule takes a value starting from zero. This value specifies the strength of the fuzzy variable being evaluated. The output, or the else part of the fuzzy rule, goes the same.

An example for a 3 input rule is:

0 1 2 0

This means that if

It is the first set label of the first fuzzy input variable
AND
It is the second set label of the second fuzzy input variable
AND
It is the third set label of the third fuzzy input variable
THEN
It will be the first set label of the fuzzy output variable

If we know that the first fuzzy input variable is the "Heart Rate", the second is the "Blood Pressure", the third is the "Primary Depth of Anesthesia", and the fuzzy output variable is the "Pump-On Period" we can read the rule as:

If it's "Low Heart Rate" AND "Mid Blood Pressure" AND "Low-Mid Depth of Anesthesia" THEN make it "Short Pump-On Period"

This is after looking up the values in the vector and substituting the right fuzzy set label in its place.

The design also recommends using flat text files for simplifying the process of editing and modifying the structure of the system (specifying inputs and outputs), the number of sets per fuzzy variable, the membership functions for each set, and the set of fuzzy rules.

FUZZY ENGINE IMPLEMENTATION

As the design suggests, the system should be open and should be reusable on any other control system. The fuzzy logic control engine is implemented in the system using C++ programming language. This gives the power for the system to be ported to many platforms and machines smoothly. In fact using C++ for implementing the engine has many advantages:

- C++ is a language found in early 70's and will last virtually forever
- C++ supports object-oriented programming that simplifies implementation of the system
- C++ compilers are found on every platform / operating system so it's easy to find the engine run on a PC, on a SPARC machine, or on an embedded control in an aircraft
- C++ is considered a mid-level language useful for control applications

- The generated machine code for programs written in C++ is faster and smaller in size so it's appropriate to be used for embedded control systems that use smaller memory
- No dependency on other software modules like MATLAB

Microsoft Visual C++ was the compiler used for building up the engine. This gives the C++ requirements and provides many facilities regarding designing the graphical user interface. The environment where the engine was built was an IBM compatible PC with Intel Celeron 333 MHz processor and 128 MB of RAM. The operating system running on this machine was the Windows2000Professional.

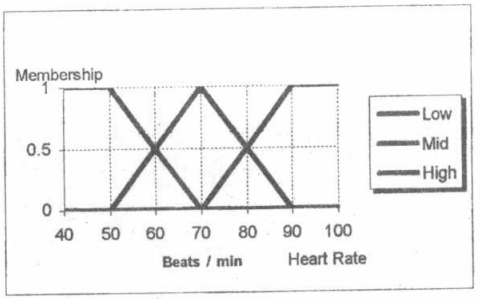
It's simple to run the engine, simply instantiate an object from the class representing the engine, load the engine object from a text file, apply input values to all input variables, then call the run() function of the class and get the output value from the output variable.

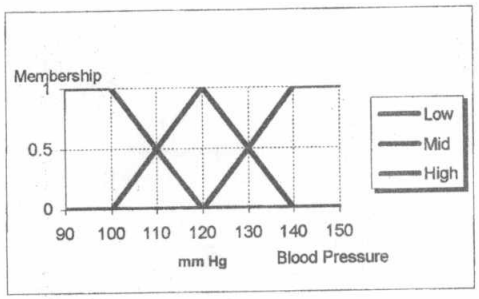
The loading process should be done once and the run function should be called whenever you want the engine to produce output or whenever there is a new system input ready at the fuzzy input variables. So it is normal to put this function in the main control loop. Then the engine output is then accessible through a variable in the output object (which is an object of a *fuzzy variable* class.)

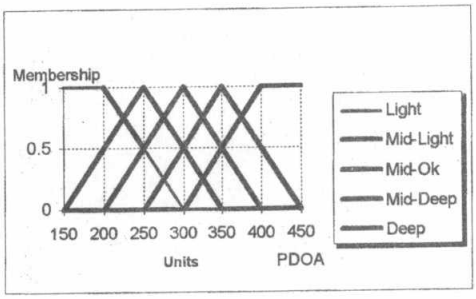
APPLYING THE ENGINE TO THE ANESTHESIA DELIVERY PROCESS

Here we apply the fuzzy logic engine to control the rate of delivery of an anesthetic agent through an electronic syringe to the patient undergoing surgical operation. The inputs to this control system are three biological parameters: the heart rate, the mean blood pressure, and the primary depth of anesthesia. The first two parameters are of great significance regarding the patient's well being. The heart rate and the blood pressure signals are acquired directly from the patient and is then normalized and freed of the noise and artifacts. The third and the most dominant parameter; the primary depth of anesthesia is obtained in real time by a more complex process of digital signal processing in which the patient stimulated by auditory clicks [6]. The three input parameters are represented numerically as integers and are ready now to be introduced to the controller. The controller is based on fuzzy logic. The data of fuzzy membership functions and fuzzy rules are static and was previously acquired from anesthetists. Then they are normalized and modeled to be suitable for the fuzzy logic inference engine, upon which the fuzzy logic controller is based. The fuzzy logic inference engine is software made up of number of C++ classes.

The output from the engine is a number representing the pulse width that represent the time in which the syringe will inject the anesthetic agent to the patient. The fuzzy membership functions of the three inputs to the system and the output are shown in Fig. 4.







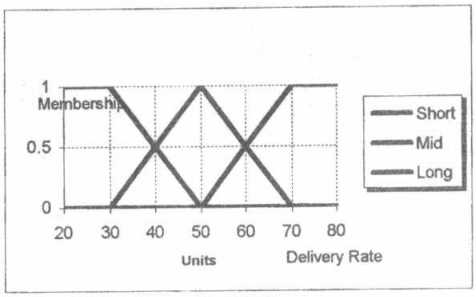


Fig. 4. The membership functions of the system inputs and the system output

The set of rules used in the system are listed in Fig. 5.

#	HR	BP	PDOA	Output
1	Low	Low	Light	Short
2	Low	Low	Mid-Light	Short
3	Low	Low	Mid-OK	Short
4	Low	Low	Mid-Deep	Short
5	Low	Low	Deep	Short
6	Low	Mid	Light	Short
7	Low	Mid	Mid-Light	Short
8	Low	Mid	Mid-OK	Short
9	Low	Mid	Mid-Deep	Short
10	Low	Mid	Deep	Short
11	Low	High	Light	Short
12	Low	High	Mid-Light	Short
13	Low	High	Mid-OK	Short
14	Low	High	Mid-Deep	Short
15	Low	High	Deep	Short
#	HR	BP	PDOA	Output
16	Mid	Low	Light	Short
17	Mid	Low	Mid-Light	Short
18	Mid	Low	Mid-OK	Short
19	Mid	Low	Mid-Deep	Short
20	Mid	Low	Deep	Short
21	Mid	Mid	Light	Long
22	Mid	Mid	Mid-Light	Long
23	Mid	Mid	Mid-OK	Mid
24	Mid	Mid	Mid-Deep	Mid
25	Mid	Mid	Deep	Short
26	Mid	High	Light	Long
27	Mid	High	Mid-Light	Long
28	Mid	High	Mid-OK	Long
29	Mid	High	Mid-Deep	Mid
30	Mid	High	Deep	Short
#	HR	BP	PDOA	Output
31	High	Low	Light	Short
32	High	Low	Mid-Light	Short
33	High	Low	Mid-OK	Short
34	High	Low	Mid-Deep	Short
35	High	Low	Deep	Short
36	High	Mid	Light	Long
37	High	Mid	Mid-Light	Long
38	High	Mid	Mid-OK	Long
39	High	Mid	Mid-Deep	Mid
40	High	Mid	Deep	Short
41	High	High	Light	Long
42	High	High	Mid-Light	Long
43	High	High	Mid-OK	Long
44	High	High	Mid-Deep	Mid
45	High	High	Deep	Short

Fig. 5. The set of rules used in the system

After building up the engine, defining the inputs and the output, building the membership function for each, and gathering the rules from experts, we made an application that put all together simulating the automated anesthesia process.

The application simulates the status of the patient. It generates a stream of random values of the heart rate, blood pressure, and primary depth of anesthesia. The resulted curve of values are guaranteed to be continuous since the random numbers used here don't represent the new value, instead, they represent the change of the value.

Fig. 6 represents a snapshot from the application. The first three curves are the system inputs and the last one is the system output.

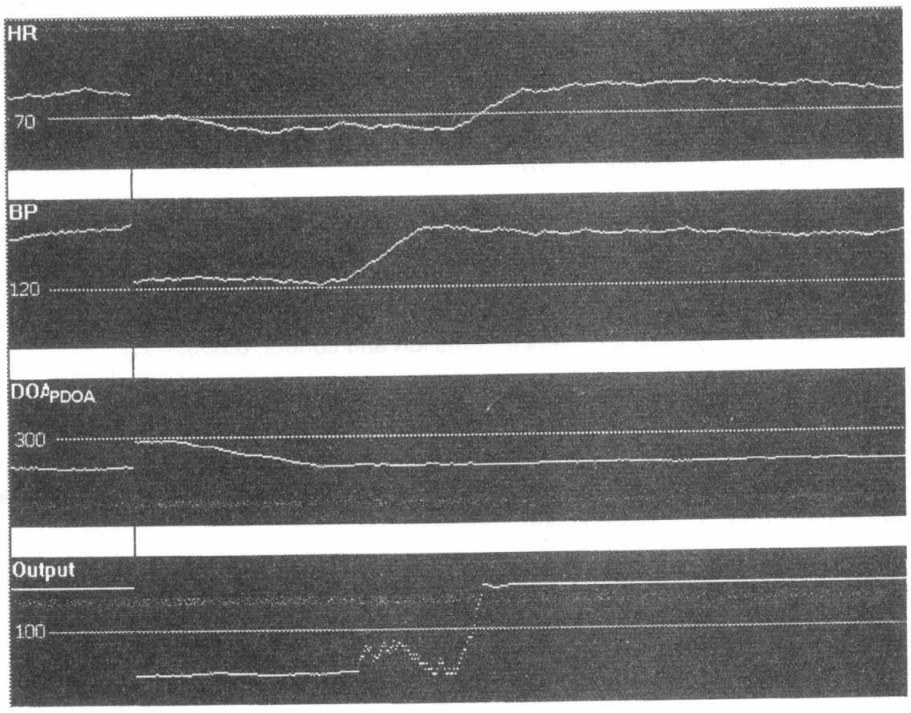


Fig. 6. A snapshot of the application

CONCLUSION AND FUTURE WORK

The fuzzy logic control engine implemented here is designed to be very flexible and ready to be used in any system easily. When used in the proposed system of anesthesia, it gave trusted results conforming to the set of rules system. Future work should consider hardware implementation to both the engine and the system

REFERENCES:

[1] Oswald, 1989, The physiology and pharmacy of sleep, Anesthesia, vol. 1, Eds. Nimmo, W.S and Smith, G Blackwell Scientific Publications, UK, pp 3-9

[2] Thoronton, C., 1991, Evoked Potentials in anesthesia, European journal of anesthesiology, 1991, 8:89-107

[3] Rampil I.J. and Smith, N.T., 1984, Computer in anesthesiology, Monitoring in Anesthesia, Eds. Saidmary L.J. and Smith, N.T., Butterworth Pubs., USA, pp 441-496

[4] Zadeh, L. A., 1965, Fuzzy Sets Information and control, 8:338: 535.

[5] Motorola, The fuzzy logic educating program

[6] Elkfafi, 1995, Intelligent signal processing of evoked potentials for control of depth of anesthesia, PhD thesis, department of automatic control and system engineering, univ. of Sheffield.