First steps in using of FuzzyTech5.5 Online Edition at College of Nyíregyháza

András Botos, Ágnes B. Simon

College of Nyiregyháza, Rákóczi út 69., Nyíregyháza

Abstract:9 In this paper you can read about the possibilities of FuzzyTECH 5.52 g program in our research work on application of fuzzy logic

Keywords:9 fuzzyTECH

1 Introduction

The *fuzzy*TECH was developed as an internal productivity tool in 1987 by INFORM GmbH founded by Professor Zimmermann.,The product the *fuzzy*TECH was introduced to the European and Asian markets. In 1990 it is one of the world leading families of software development tools for fuzzy logic and neural-fuzzy solutions. The *fuzzy*TECH Editions are specialized for technical applications while the *fuzzy*TECH for Business products are specialized for applications in business and finance. We ordered an educational licence of fuzzyTech5.52 Online Edition (includes NeuroFuzzy & FuzzyCluster module) recently. We have installed it on PentiumII PC under WIN98 OS. A hardwarekey allows us to use all functionalities of the purchased licence version without any restrictions. Implentiny Fuzzy-Runtime System parts of a Runtime System. There are 3 modules: 1. Library of basic fuzzy algorithms 2.

The *fuzzy*TECH Professional and Online Editions generate portable C code that can be implemented on any hardware that is supported by a C compiler. In addition, both let you export your fuzzy logic system as a plug-in runtime module for PC-based applications. Both Editions also support the special "plug-and-play" type runtime modules provided for specific process control software such as InTouch™, FactoryLink™, TheFIX™, Genesis™, and WinCC™. In addition to the M source code generation provided by every *fuzzy*TECH Edition, the *fuzzy*TECH Professional and Online Editions also contain a MEX runtime module that provides high-performance computation of fuzzy logic systems in the Matlab/Simulink™ environment

2.1.Technical Specifications

Variables					Rules			
Total	*Input*	*Out-put*	*Terms per Variable Linguistic/Categorical*	*Total Terms*	*Rule Blocks (RB)*	*Inputs per RB*	*Outputs per RB*	*Total Rules*
255	255	32	32/255	65535	32	11 [1)]	11 [1)]	-

1) The total number of Inputs and Outputs per RB is limited to 12.

Table 1. Limitations of variables and rules

Table 1 shows the maximum number of interfaces, variables, terms, rule blocks and rules. A "-" sign indicates that no practical limit exists. The total number of Variables (Total) represent the number of input, output and intermediate variables of the entire fuzzy logic system. The columns *Input and Output* show the maximum amount of input and output variables. *Terms per Variable* relates to the total number of terms for each variable. The column *Total Terms* shows the maximum number of terms that may occur in one fuzzy logic project. The Rules section shows the maximum total number of *Rule Blocks* and Rules that a fuzzy logic project may contain, as well as the maximum total number of input variables (*Inputs per RB*) and output variables (*Outputs per RB*) that can be assigned to a rule block.

2.2. Membership Functions

The following table lists methods of fuzzification and Types of membership functions (MBF). Standard MBFs are sometimes called "4-point definitions". *Arbitrary MBFs* can be defined withup to 16 points of definition. *Inverse MBFs* (inverse terms) are useful for filling a rule part with the negated form of an already existing term. The column MBF Shape shows the available approximation functions for membership functions. The column Fuzzification Method lists the supported algorithms for the fuzzification step of the fuzzy logic inference. *Fuzzy Input* indicates that variables are inputted as fuzzy values (i.e. as vectors of membership degrees) instead of crisp values. For *Look-up-MBF* fuzzification, MBFs are computed as look-up-tables by code generators. This speeds up the computation on some microcontrollers and fuzzy processors, but consumes considerable memory space. The standard fuzzification method is computation at run time (*MBF Computation*). For most target hardware implementations, this is the most efficient approach. On some microcontroller families, the slope fuzzification method (*Fast Computation of MBF*) is faster. Slope fuzzification is a variant of *MBF Computation*.

Type			Shape		Fuzzification Methods				
Standard MBF	*Arbitrary MBF*	*Inverse MBF*	*linear*	*S-shape*	*Fuzzy Input*	*Look-up MBF[1)]*	*MBF Computation*	*Fast MBF Computation [2)]*	*Categorical*
x	x	x	x	x	x	-	x	-	x

[1)] MBFs stored as look up table, [2)] Slope fuzzification

Table 2: Memberships types and fuzzification methodes

2.3. Inference and Defuzzification

The table 3. summarizes supported methods of fuzzy logic inference and defuzzification. The fuzzy inference consists of three computational steps: Aggregation, Composition, and Result Aggregation. Different operators can be chosen for aggregation (*Input aggregation*) and result aggregation. Fuzzy operators used for aggregation (Minimum or Maximum) combine the preconditions of each fuzzy rule. Beside this standard operators, support compensatory operators (Gamma, Min-Avg, Min-Max), that help to compute relations between rules formulated with the logic standard operators AND (Minimum) and NOR (Maximum). The second step of the fuzzy rule inference, the composition, works generally with the PROD-Operator as fixed operator. Standard Rules are rules with a fixed rule weight (Degree of Support = 1.0) that cannot be changed. FAM Rules stands for "Fuzzy Associative Maps" and refers to individually weighted rules (Degree of Support = DoS). The last step of fuzzy inference is the so-called result aggregation. Its MAX operator selects the maximum firing degree of all rules matching to the term. The BSUM operator uses a bounded sum. Thus, all firing degrees are summed by using a bound of one. Note that BSUM result aggregation is different from BSUM MoM and BSUM CoA. The bounds are zero and one.

Aggregation Operators Standard / Compensatory				Composition		Result Aggregation		Defuzzification				
Minimum Maximum	*Min-Max*	*Min-Avg*	*Gamma*	*Standard Rules*	*FAM Rules (DoS)*	*Max*	*BSUM*	*CoM*	*CoA [1)]*	*MoM*	*Fuzzy Output*	*Hyper CoM [2)]*
x	x	x	x	x	x	x	x	x	x	x	x	x

1) Fast CoA, neglects overlaps, 2) Only available as add-on module

Table 3: Inference and defuzzification

The result of the fuzzy inference is a fuzzy value that has to be re-transformed into a crisp value. This transformation is called Defuzzification. Different computation methods can be applied for defuzzification. The standard defuzzification is CoM (Center-of-Maximum), delivering the "best compromise" for the inference result. It is equivalent to most implementations of Center-of-Area (*CoA*) and Center-of-Gravity (CoG) methods. The MoM (*Mean-of-Maximum*) method delivers the "most plausible" result. Hence, it is mostly used in applications such as pattern recognition, data analysis, and decision support. The defuzzification method *HyperCoM* (see last column) is also available in this edition. HyperCoM is a defuzzification method that takes both positive and negative experience into consideration (e.g. in the form of recommendations and warnings). A hyperdefuzzification strategy weights these recommendations and warnings against each other and computes a membership function, from which HyperCoM then computes the optimum based output value.

2.4. System Optimization and Analysis, Add-on Modules

All fuzzyTECH Editions come with a complete set of debugging features (Debug Modes) and analysis tools Analyzers (see Table below). Remote debugging, where fuzzyTECH running on the PC debugs a fuzzy logic system running on a different computer/microcontroller, is facilitated by Online and RTRCD debug modes. The RTRCD debug mode (Real Time Remote Control Debugging) lets you analyze the running system and modify rules and membership functions. The RTRCD mode is best for embedded systems design and featured as an ad-on to most fuzzyTECH MCU Editions. The Online debug mode in addition allows for all types of modifications on-the-fly. To expand the capabilities of fuzzyTECH, several Add-on Modules are available. The NeuroFuzzy Module uses neural network technology to automatically generate fuzzy logic rules and membership functions. It contains the FuzzyCluster Module that is suited for preparing training data for the NeuroFuzzy Module. The HyperInference Module expands traditional fuzzy logic rule inference with the ability of "prohibitive" rules.

Debug Modes (internal)						**Communication Channels**				**Analyzer**	**Add-on Modules**	
Interactive	*File / Batch*	*Serial Link*	*RCU, DDE*	*RTRCD*	*Online*	*TCP/IP, IPX/SPX, DDE*	*Serial Interface (RS232)*	*SFS*	*User-defined (FTOCC)*	*TransferPlot, 3D Plot, Time Plot, Trace*	*Neuro-Fuzzy*	*Hyper Inference*
x	x	x	x	-	x	x	x	x	x	x	x	x

Table 4: Tools of debugging and optimizing

2.5. Code Generation

The table 5. lists supported code generator options. The File Code option generates a complete C source code for an executable program which accepts file data as input and writes outputs to a file as well. Since fuzzyTECH generates the complete fuzzy logic system as a single function, input and output values can be transferred as function parameters (Function Call) over the system stack or as global variables (Public I/O). Some fuzzyTECH MCU Editions support only Public I/O, since passing variables over the stack can be inefficient. All fuzzyTECH MCU Editions utilizing an assembly library as a kernel support calling fuzzy logic functions from Assembly code and from C code (C interface). The C Code column lists which C compiler standard is supported. Besides, Code Interface Resolution indicates the data types of the interfaces, too.

Code-Options		**Hardware specific Code**		**C-Code**		**Java-Code**	**ActiveX Java FTRUN**	**COBOL**
Funct. Call[1)]	*Public I/O*	*C*	*Assembly*	*ANSI*	*K & R*	*Java*	*FTR*	*COBOL* [2)]
		Code Interface Resolution (8 Bit, 16 Bit, double)						
x	x	-	-	8/16/d	8/16/d	16/d	16/d	16

[1)] *I/O passing as function parameter* [2)] *Only available as add on module*

Table 5: Code Generation Options

3. Demonstration of the online technology

The animated Steam Generator Drum simulation demonstrates the possibility of remote controlling. This simulated application uses fuzzy logic control for the start up of a steam generator in a power plant. The development system was our FuzzyTech5.52g online educational version. In our experiment the target system was the fuzzy runtime system simulating the process of putting a steam generator drum into operation (it is a sample system closed to the FuzzyTech5.52g), the online communication channel was the MS-Windows DDE, because the runtime system and developing sytem were the same PC.

3.1 Starting the simulating program

The *Start\fuzzyTech5.5\Examples\Simulations\SteamGenerationDrum* menu started the visual application program. The steam generator window however does'nt show the internal particulars of the fuzzy logic systems. You need to start the remote access.

3.2 Configuring Online Connection

Clicking the the treeview entry „Online Connections" by the right mouse key we can choose the option Online Wizard.

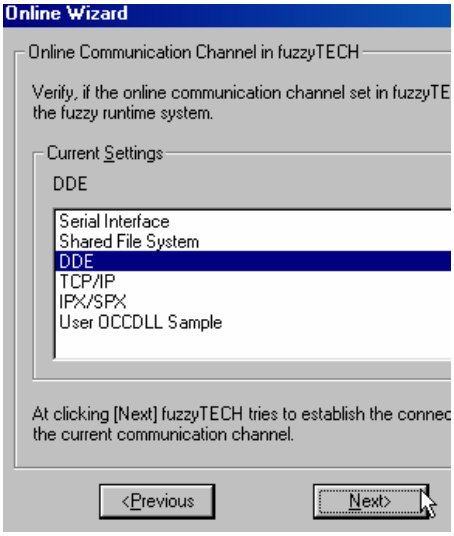

Figure1. Online channels in our version

After the successful configuration we can see the information by the treeview entry as in Figure 2.

Figure 2.

To study the effect of modifications on the sample runtime system we used the „Online- Monitor and Modify" Debug tool.

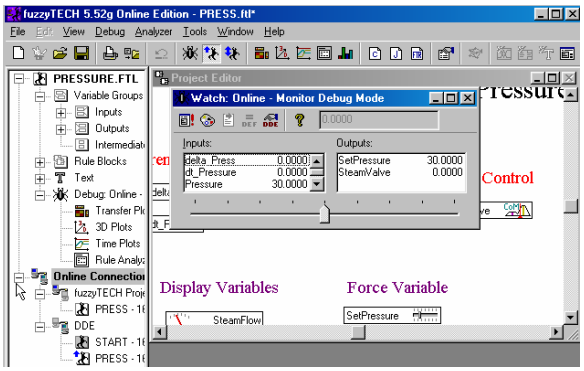

Figure 3.

4 References

1. IEC 1131-Programmable Controllers
   Part 7 – Fuzzy Control Programming
   Commitee Draft CD 1.0 (rel. 19 Jan 97)
2. User's Manual for all *fuzzy*TECH 5.5 Edition
3. *fuzzy*TECH 5.52 g Edition program package on MS-WIN98 PC