Intelligent System Based on DSP to Monitoring and Maintain Optimum Environmental Conditions in a Fir Saplings Tree Greenhouse

MANUELA PANOIU, CAIUS PANOIU, LOREDANA GHIORGHIONI, RALUCA ROB
Electrical Engineering and Industrial Informatics Department
Polytechnic University of Timisoara, Engineering Faculty of Hunedoara
Revolutiei street, no 5 Hunedoara, cod 331128
ROMANIA
{m.panoiu, c.panoiu}@fih.upt.ro, anaderol_hd@yahoo.com, raluca.rob@fih.upt.ro

Abstract: -In this paper is presented an intelligent system for measure and maintain the optimum environmental conditions in a fir tree greenhouse. The proposed system used a DSP development board built TMS 320 F2812. In a fir tree greenhouse environment condition is more stringent than in a typical greenhouse in all respects. In such a greenhouse the most sensitive conditions refer to soil moisture, air humidity and air temperature. The proposed system allows acquisition of soil moisture, air humidity and air temperature. Depending on these the system sends three output commands. These commands are generated by using Fuzzy logic**.**

Key-Words: - Fuzzy logic, DSP, temperature control, humidity control, environmental conditions

1. Introduction

In the last years, a significant increase of number of various fuzzy applications has been notice [2], [5].
In the real situations, decision makers are often faced to a plurality of objectives and constraints in a world of imprecise data about the preferences of agents, the local constraints and the global environment [1]. Including of a microprocessor in a control and measure system gives the opportunity that the maneuvers of the operator to be taken by the microcontroller and additionally to obtain some important advantages [2]:

- The takings over of the operator commands such as zero adjust and scale adjustment, the scale selection assuring a plus of accuracy and safety in exploitation.
- Self-calibration and automatic equalization of the effect of the climate factors. Also are calculated the errors and the final result of the measurement is corrected.
- The improvement of the accuracy by abstraction the systematic errors (by self-calibration, self-correction).

In this paper is presented such a system based on fuzzy logic that allows maintaining optimal environmental conditions for a greenhouse for seedlings of trees. In such a hothouse environment condition is more stringent than in a typical greenhouse in all respects. The optimal values in a fir tree greenhouse are:

- Soil moisture must have the values between 50% and 80 %
- Environmental temperature must have the values between 10°C and 35°C
- Air humidity must have the values between 70% and 90 %.

2. Monitoring and measuring the environmental conditions

To obtain the optimum values for the environmental conditions it is requires an integrated system to maintain soil moisture, air humidity and ambient temperature within the limits mentioned above. This is proposed to be carried out by a DSP development board built TMS 320 F2812. This board is a stand-alone card allowing to the users to examine the TMS 320 F2812 digital signal processor (DSP) to determine if it meets their application requirements. Furthermore, the module is an excellent platform to develop and run software for the DSP [6]. In figure 1 it is presented the block diagram for the three inputs measure: air humidity, soil moisture and ambient temperature.

A fir saplings tree greenhouse effect works only in cold weather. In October are cut branches from mature trees. These branches are planted in an appropriate soil and up to March – April the braches development roots and can be planted in another outdoor greenhouse. In figure 2 it can be see such a fir saplings tree greenhouse.

Because the greenhouse effect works only in cold weather is not needed to air cooling. So, the three commands that are generated by Fuzzy logic are:

- Soil wetting command for maintaining appropriate soil moisture
- Air warming for maintaining an appropriate ambient temperature
- Steam spray for maintaining an appropriate ambient humidity

The proposed system allows generating these commands (to see figure 1).

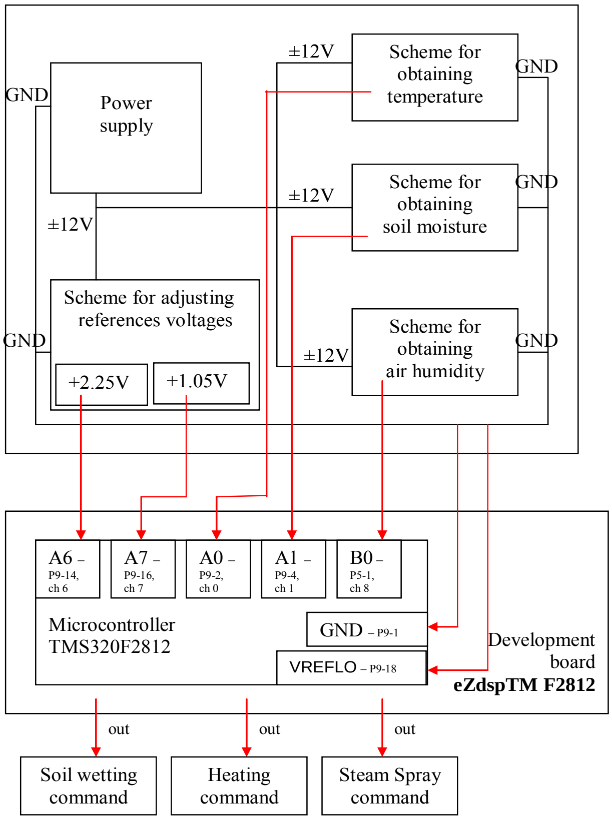


Fig. 1. The block diagram for the three inputs measure and for generate the three outputs commands

Fig. 2. A fir saplings tree greenhouse

2.1. Ambient temperature measurement

For ambient temperature measurement we used the scheme from figure 3.

Bridge consisting of R1, Rv1, R2, R3 and LM335 temperature sensor provides a proportional voltage to temperature witch is amplified by integrated circuit with two operational amplifiers LM4558C and is used for calibration through Rv1 and with the help of Rv2 is adjusted the amplification (the range of variation for ambient temperature: 0 ÷ 40 °C).

We used two operational amplifiers to have high input impedance to not affect the stage above.

O1 is a repeater circuit. Capacitor C4 is used for filtering the sensor (to eliminate noise) and capacitors C1, C2 and C3 are used for filtering the entire circuit. (3-Terminal 1A Positive Voltage Regulator is an integrated circuit which is a +5V voltage stabilizer.

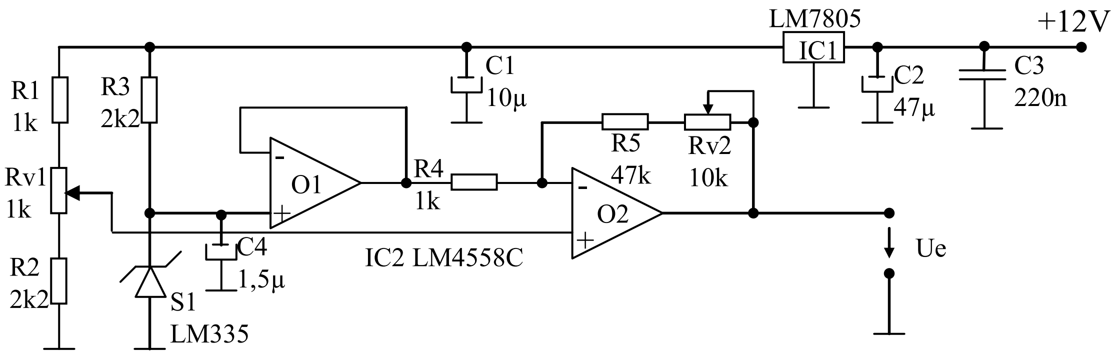


Fig.3 Electronic scheme for ambient temperature

2.2 Soil moisture measuring

For soil moisture measuring we used scheme from figure 4.

In this scheme we used the same elements as the temperature and air humidity measurements: integrated circuit LM4558C with two operational amplifiers, integrated circuit LM7805, witch were detailed at temperature measurement, resistors and capacitors but other values for them and of course the two stainless steel electrodes witch are placed in the ground at a constant distance to not change the resistance, because as we know even in the ground there is some resistance, witch is influenced by the distance between the electrodes.

$$R = (\rho * l) / s$$

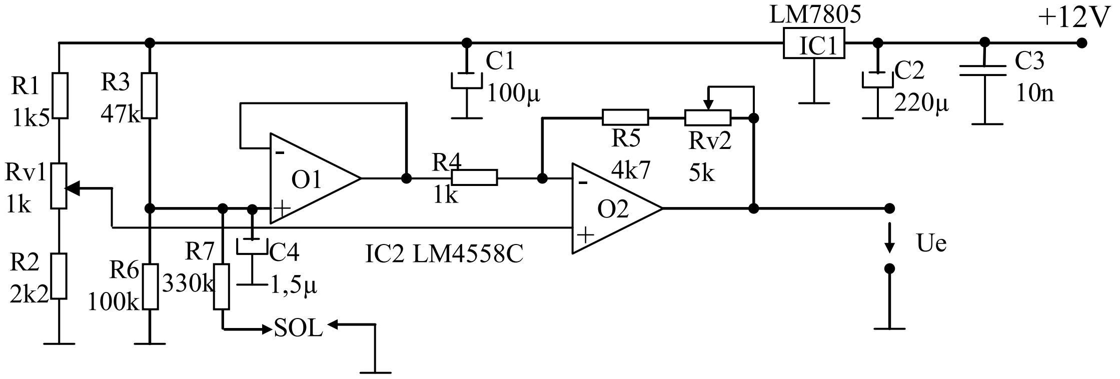


Fig.4 Electronic scheme for soil moisture

Bridge consisting of R1, Rv1, R2, R3 and the two stainless steel electrodes provides a proportional voltage to soil moisture witch is amplified by integrated circuit with two operational amplifiers LM4558C and is used for calibration through Rv1 and with the help of Rv2 is adjusted the amplification (the range of variation for soil moisture: 0÷100%).

Range of variation of soil moisture for saplings of pine is 70 ÷ 95 %, so it needs high humidity, but we took the range for soil moisture at 0 ÷ 100% because is was easier to calibrate.

2.3 Ambient humidity measure

For measuring air humidity we used scheme from figure 5. Was used air humidity sensor SY –HS – 230. This air humidity sensor is designed to measure relative humidity as an output voltage witch depends on humidity measured. It is suitable for wet and dry air.

In this scheme are mainly used the same elements as for the ambient temperature measurements: integrated circuit with two operational amplifiers LM4558C, integrated circuit LM7805, witch were detailed above, resistors and capacitors but other values for them and air humidity sensor SY – HS - 230.

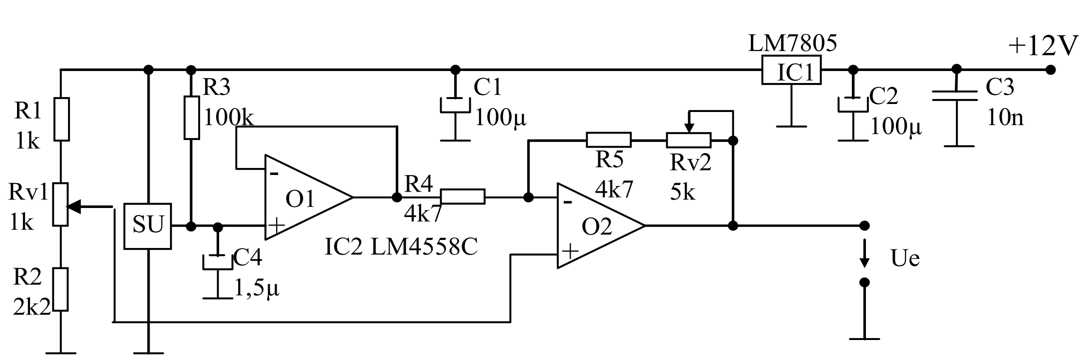


Fig.5 Electronic scheme for air humidity

Output signals obtained from the three measurement schemes are in the range 0 ÷ 3.3 V. This is the interval that data can be purchased by the digital to analog convertor that is placed on EzDsp TMF2812 development board.

According to data acquired for soil moisture and air humidity, respective for ambient temperature will be sent the three output commands throw GPIO pins (General Purpose Input Output pins). Scheme to generate output commands throw GPIO pins is presented in figure 6.

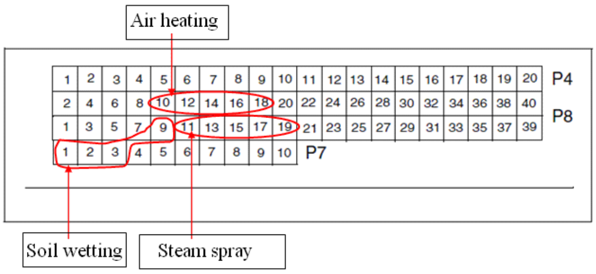


Fig. 6. Output commands generating through the GPIO pins

And for each of the three output commands are four levels that can be used:

- blocked – when the input values situate in normal limits
- level1 – when is needed a correction, but not very large
- level2 – is needed higher correction
- maxim – when the input values must be influenced to go in normal limits because otherwise saplings pine will not survive.

3. The proposed system based on Fuzzy logic

The implemented system uses three inputs and three outputs that will ensure optimal conditions of environment and soil in a greenhouse for seedlings of trees.

Input values are shown in paragraph above, and those are ambient temperature, air humidity, soil moisture. The outputs, also describe in paragraph above are: soil wetting, air warming and steam spray. The most common shape of membership functions is triangular, although trapezoids and bell curves are also used. In our case were used trapezoidal functions for all three sizes of entry. And for the output sizes were used singleton-type functions.

3.1 Definition of Fuzzy sets for the ambient temperature.

Considering that the optimum temperature for proper development of fir saplings tree is between Tmin and Tmax (10°C si 40°C), have defined the fuzzy sets from fig. 7. We choose for membership functions trapezoidal functions.

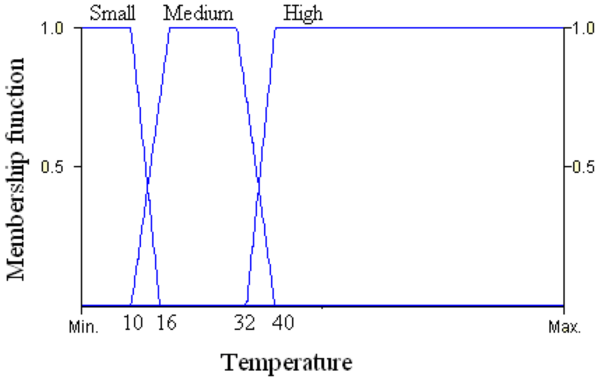


Fig. 7. Definition of Fuzzy sets for "temperature"

3.2. Definition of Fuzzy sets for the "air humidity"

Considering that the optimum air humidity for proper development of fir saplings tree is between AHmin and AHmax (50% and 70%), have defined the fuzzy sets from fig. 8. It was also choose trapezoidal functions for the membership functions.

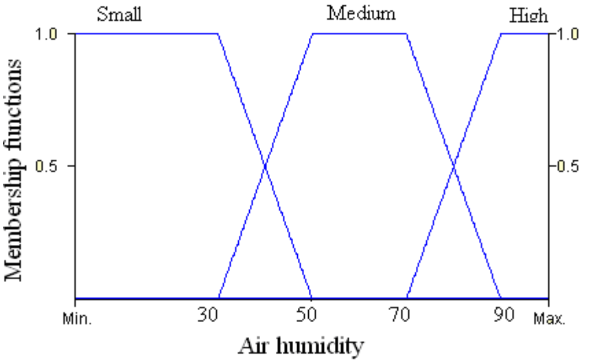


Fig. 8. Definition of Fuzzy sets for "air humidity"

3.3 Definition of Fuzzy sets for "soil moisture"

Considering that the optimum soil moisture for proper development of fir saplings tree is between SMmin and SMmax (75% and 95%), have defined the fuzzy sets from fig. 9. Like for the other two inputs, it was choose trapezoidal functions for the membership functions.

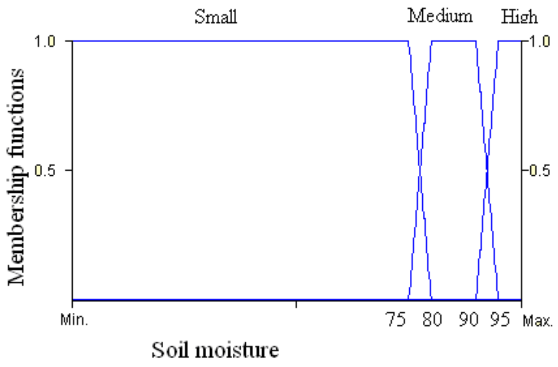


Fig. 9. Definition of Fuzzy sets for “soil moisture”

3.4. Definition of Fuzzy sets for the three outputs

In case of all three outputs were used two methods for definition of fuzzy sets:

- singleton-type functions thus can be seen in the figure 10, i.e. the functions proposed by the Sugeno defuzzification method and
- trapezoidal functions in case of use the Mamdani defuzzification method.

Using these fuzzy sets can be defined three sets of rules for the three outputs.

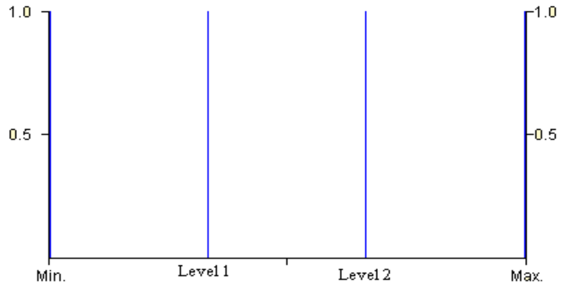
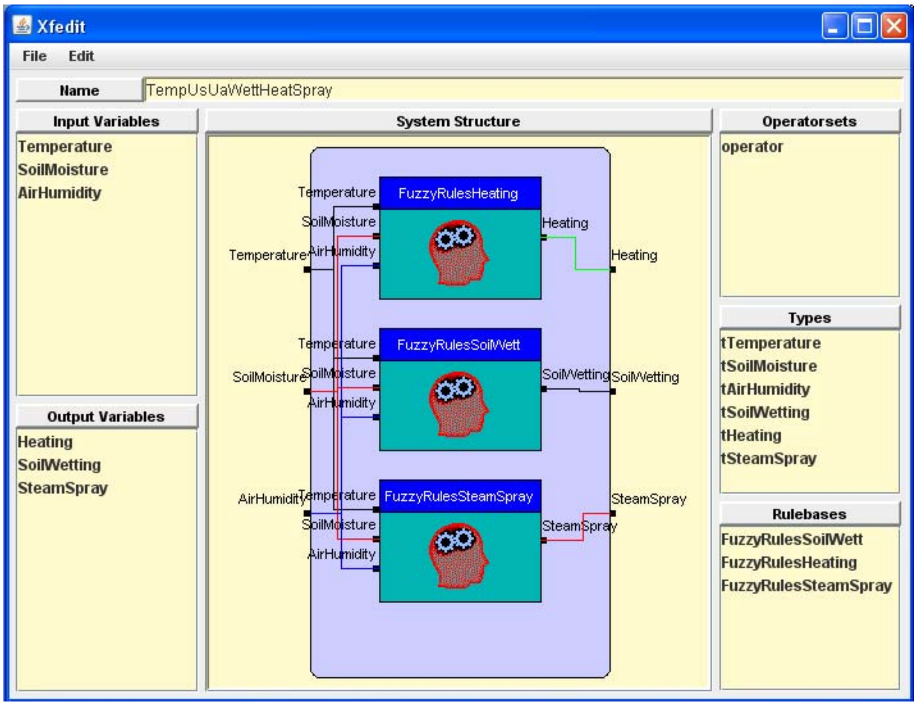


Fig. 10. Definition of the output commands

3.5. Simulation of the proposed system

To make the initial verification of the proposed system was developed a simulation system based on Fuzzy logic. The simulation was implemented in simulation software xFuzzy [3]. Xfuzzy 3.0 is a development environment for fuzzy-inference-based systems. It is composed of several tools that cover the different stages of the fuzzy system design process, from their initial description to the final implementation. Its main features are the capability for developing complex systems and the flexibility of allowing the user to extend the set of available functions. The environment has been completely programmed in Java, so it can be executed on any platform with JRE (Java Runtime Environment) installed. Thus the proposed system was implemented in xFuzzy as shown in Figure 11.



Name: TempUsUaWettHeatSpray

Input Variables
Temperature
SoilMoisture
AirHumidity

Output Variables
Heating
SoilWetting
SteamSpray

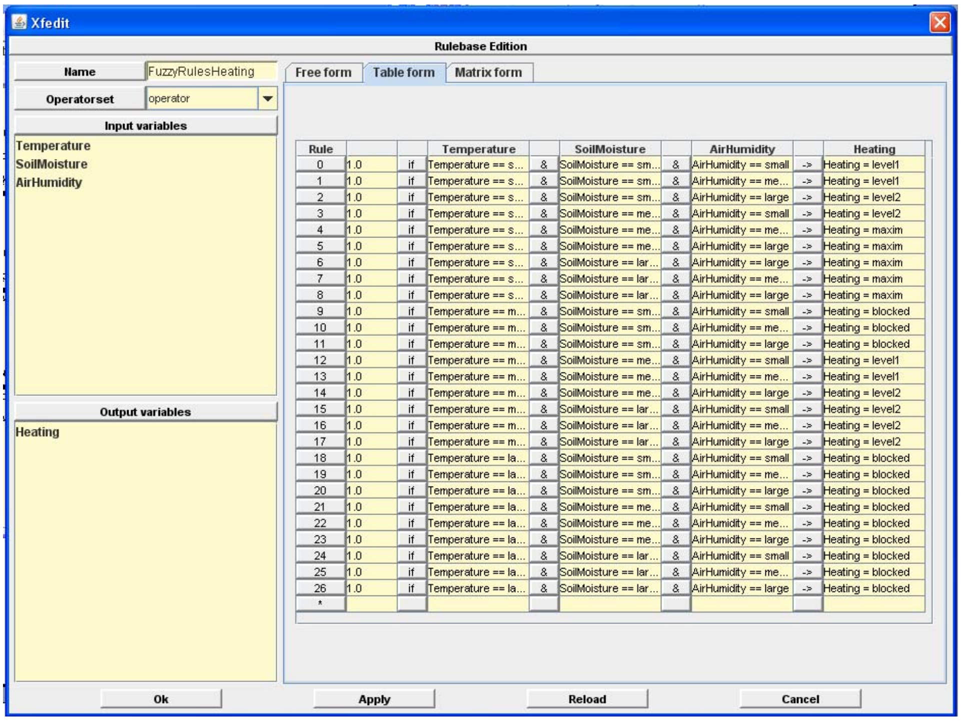
Operatorsets
operator

Types
tTemperature
tSoilMoisture
tAirHumidity
tSoilWetting
tHeating
tSteamSpray

Rulebases
FuzzyRulesSoilWett
FuzzyRulesHeating
FuzzyRulesSteamSpray

Fig. 11 Implementation of the proposed fuzzy system

XFuzzy simulation software allows a complete system monitoring functionality. For the output values, are defined the sets of rules that is shown in Figure 12, 13 and 14. The rules have been established for practical reasons after having consulted experts in maintain optimum environmental conditions in a Fir Tree Greenhouse. For the other two outputs the Fuzzy set of rules are defined in the same manner.



Name: FuzzyRulesHeating

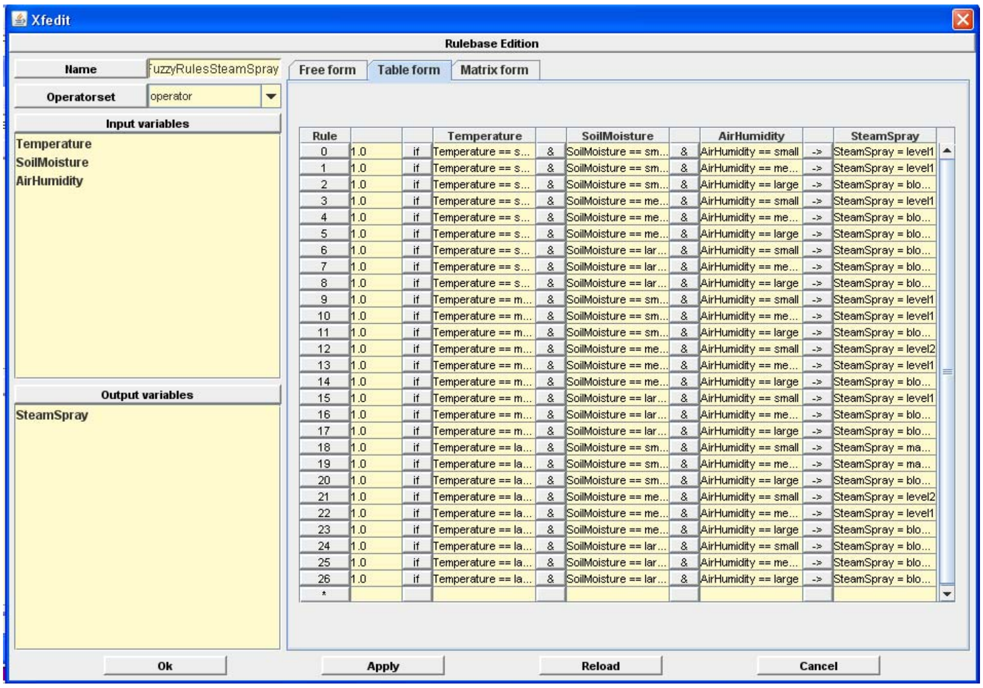
Operatorset: operator

Input variables:
Temperature
SoilMoisture
AirHumidity

Output variables:
Heating

Rule			Temperature		SoilMoisture		AirHumidity		Heating
0	1.0	if	Temperature == s...	&	SoilMoisture == sm...	&	AirHumidity == small	->	Heating = level1
1	1.0	if	Temperature == s...	&	SoilMoisture == sm...	&	AirHumidity == me...	->	Heating = level1
2	1.0	if	Temperature == s...	&	SoilMoisture == sm...	&	AirHumidity == large	->	Heating = level2
3	1.0	if	Temperature == s...	&	SoilMoisture == me...	&	AirHumidity == small	->	Heating = level2
4	1.0	if	Temperature == s...	&	SoilMoisture == me...	&	AirHumidity == me...	->	Heating = maxim
5	1.0	if	Temperature == s...	&	SoilMoisture == me...	&	AirHumidity == large	->	Heating = maxim
6	1.0	if	Temperature == s...	&	SoilMoisture == lar...	&	AirHumidity == large	->	Heating = maxim
7	1.0	if	Temperature == s...	&	SoilMoisture == lar...	&	AirHumidity == me...	->	Heating = maxim
8	1.0	if	Temperature == s...	&	SoilMoisture == lar...	&	AirHumidity == large	->	Heating = maxim
9	1.0	if	Temperature == m...	&	SoilMoisture == sm...	&	AirHumidity == small	->	Heating = blocked
10	1.0	if	Temperature == m...	&	SoilMoisture == sm...	&	AirHumidity == me...	->	Heating = blocked
11	1.0	if	Temperature == m...	&	SoilMoisture == sm...	&	AirHumidity == large	->	Heating = blocked
12	1.0	if	Temperature == m...	&	SoilMoisture == me...	&	AirHumidity == small	->	Heating = level1
13	1.0	if	Temperature == m...	&	SoilMoisture == me...	&	AirHumidity == me...	->	Heating = level1
14	1.0	if	Temperature == m...	&	SoilMoisture == me...	&	AirHumidity == large	->	Heating = level2
15	1.0	if	Temperature == m...	&	SoilMoisture == lar...	&	AirHumidity == small	->	Heating = level2
16	1.0	if	Temperature == m...	&	SoilMoisture == lar...	&	AirHumidity == me...	->	Heating = level2
17	1.0	if	Temperature == m...	&	SoilMoisture == lar...	&	AirHumidity == large	->	Heating = level2
18	1.0	if	Temperature == la...	&	SoilMoisture == sm...	&	AirHumidity == small	->	Heating = blocked
19	1.0	if	Temperature == la...	&	SoilMoisture == sm...	&	AirHumidity == me...	->	Heating = blocked
20	1.0	if	Temperature == la...	&	SoilMoisture == sm...	&	AirHumidity == large	->	Heating = blocked
21	1.0	if	Temperature == la...	&	SoilMoisture == me...	&	AirHumidity == small	->	Heating = blocked
22	1.0	if	Temperature == la...	&	SoilMoisture == me...	&	AirHumidity == me...	->	Heating = blocked
23	1.0	if	Temperature == la...	&	SoilMoisture == me...	&	AirHumidity == large	->	Heating = blocked
24	1.0	if	Temperature == la...	&	SoilMoisture == lar...	&	AirHumidity == small	->	Heating = blocked
25	1.0	if	Temperature == la...	&	SoilMoisture == lar...	&	AirHumidity == me...	->	Heating = blocked
26	1.0	if	Temperature == la...	&	SoilMoisture == lar...	&	AirHumidity == large	->	Heating = blocked
*									

Fig. 12. Definition of the set of rules for “air heating”.



Name: FuzzyRulesSteamSpray

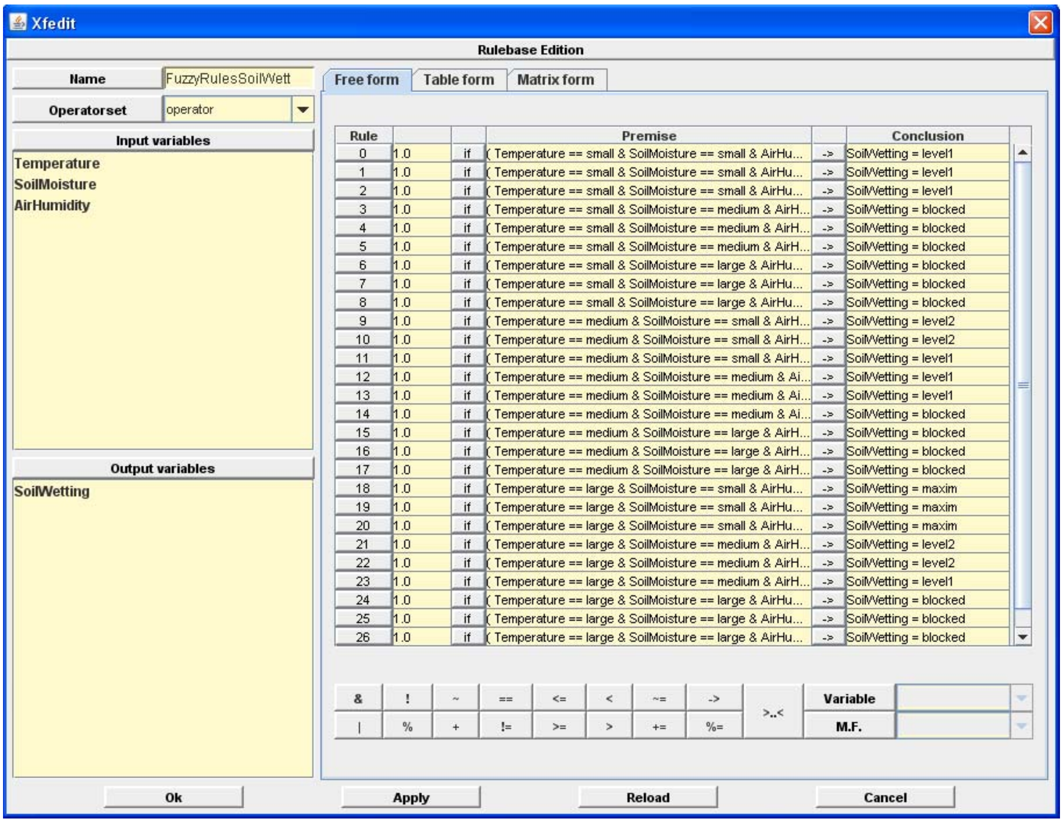
Operatorset: operator

Input variables:
Temperature
SoilMoisture
AirHumidity

Output variables:
SteamSpray

Rule			Temperature		SoilMoisture		AirHumidity		SteamSpray
0	1.0	if	Temperature == s...	&	SoilMoisture == sm...	&	AirHumidity == small	->	SteamSpray = level1
1	1.0	if	Temperature == s...	&	SoilMoisture == sm...	&	AirHumidity == me...	->	SteamSpray = level1
2	1.0	if	Temperature == s...	&	SoilMoisture == sm...	&	AirHumidity == large	->	SteamSpray = blo...
3	1.0	if	Temperature == s...	&	SoilMoisture == me...	&	AirHumidity == small	->	SteamSpray = level1
4	1.0	if	Temperature == s...	&	SoilMoisture == me...	&	AirHumidity == me...	->	SteamSpray = blo...
5	1.0	if	Temperature == s...	&	SoilMoisture == me...	&	AirHumidity == large	->	SteamSpray = blo...
6	1.0	if	Temperature == s...	&	SoilMoisture == lar...	&	AirHumidity == small	->	SteamSpray = blo...
7	1.0	if	Temperature == s...	&	SoilMoisture == lar...	&	AirHumidity == me...	->	SteamSpray = blo...
8	1.0	if	Temperature == s...	&	SoilMoisture == lar...	&	AirHumidity == large	->	SteamSpray = blo...
9	1.0	if	Temperature == m...	&	SoilMoisture == sm...	&	AirHumidity == small	->	SteamSpray = level1
10	1.0	if	Temperature == m...	&	SoilMoisture == sm...	&	AirHumidity == me...	->	SteamSpray = level1
11	1.0	if	Temperature == m...	&	SoilMoisture == sm...	&	AirHumidity == large	->	SteamSpray = blo...
12	1.0	if	Temperature == m...	&	SoilMoisture == me...	&	AirHumidity == small	->	SteamSpray = level2
13	1.0	if	Temperature == m...	&	SoilMoisture == me...	&	AirHumidity == me...	->	SteamSpray = level1
14	1.0	if	Temperature == m...	&	SoilMoisture == me...	&	AirHumidity == large	->	SteamSpray = blo...
15	1.0	if	Temperature == m...	&	SoilMoisture == lar...	&	AirHumidity == small	->	SteamSpray = level1
16	1.0	if	Temperature == m...	&	SoilMoisture == lar...	&	AirHumidity == me...	->	SteamSpray = blo...
17	1.0	if	Temperature == m...	&	SoilMoisture == lar...	&	AirHumidity == large	->	SteamSpray = blo...
18	1.0	if	Temperature == la...	&	SoilMoisture == sm...	&	AirHumidity == small	->	SteamSpray = ma...
19	1.0	if	Temperature == la...	&	SoilMoisture == sm...	&	AirHumidity == me...	->	SteamSpray = ma...
20	1.0	if	Temperature == la...	&	SoilMoisture == sm...	&	AirHumidity == large	->	SteamSpray = blo...
21	1.0	if	Temperature == la...	&	SoilMoisture == me...	&	AirHumidity == small	->	SteamSpray = level2
22	1.0	if	Temperature == la...	&	SoilMoisture == me...	&	AirHumidity == me...	->	SteamSpray = level1
23	1.0	if	Temperature == la...	&	SoilMoisture == me...	&	AirHumidity == large	->	SteamSpray = blo...
24	1.0	if	Temperature == la...	&	SoilMoisture == lar...	&	AirHumidity == small	->	SteamSpray = blo...
25	1.0	if	Temperature == la...	&	SoilMoisture == lar...	&	AirHumidity == me...	->	SteamSpray = blo...
26	1.0	if	Temperature == la...	&	SoilMoisture == lar...	&	AirHumidity == large	->	SteamSpray = blo...
*									

Fig. 13. Definition of the set of rules for “SteamSpray”.



Name: FuzzyRulesSoilWett

Operatorset: operator

Input variables:
Temperature
SoilMoisture
AirHumidity

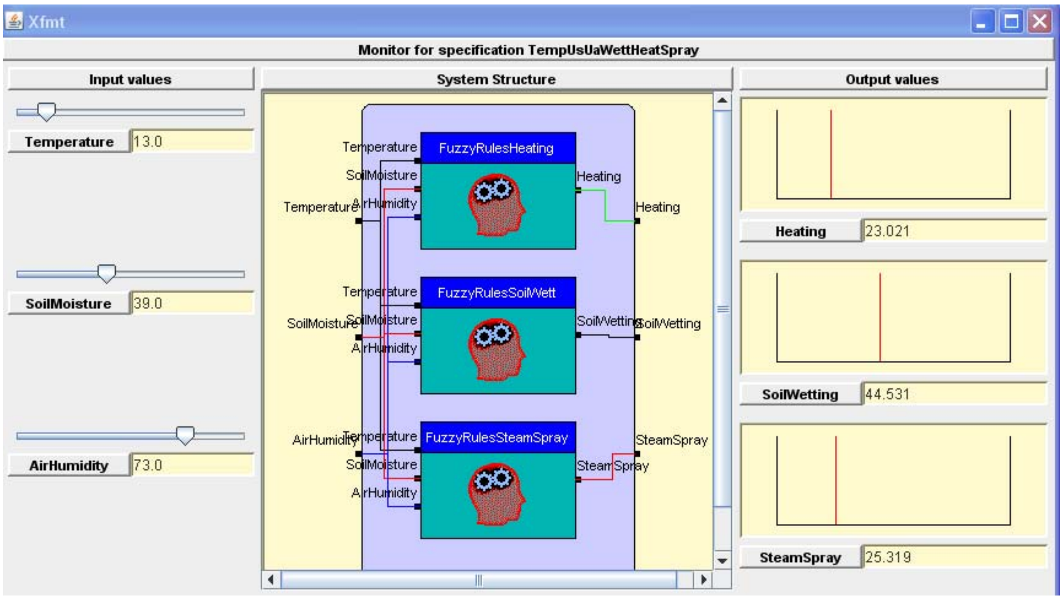
Output variables:
SoilWetting

Rule			Premise		Conclusion
0	1.0	if	(Temperature == small & SoilMoisture == small & AirHu...	->	SoilWetting = level1
1	1.0	if	(Temperature == small & SoilMoisture == small & AirHu...	->	SoilWetting = level1
2	1.0	if	(Temperature == small & SoilMoisture == small & AirHu...	->	SoilWetting = level1
3	1.0	if	(Temperature == small & SoilMoisture == medium & AirH...	->	SoilWetting = blocked
4	1.0	if	(Temperature == small & SoilMoisture == medium & AirH...	->	SoilWetting = blocked
5	1.0	if	(Temperature == small & SoilMoisture == medium & AirH...	->	SoilWetting = blocked
6	1.0	if	(Temperature == small & SoilMoisture == large & AirHu...	->	SoilWetting = blocked
7	1.0	if	(Temperature == small & SoilMoisture == large & AirHu...	->	SoilWetting = blocked
8	1.0	if	(Temperature == small & SoilMoisture == large & AirHu...	->	SoilWetting = blocked
9	1.0	if	(Temperature == medium & SoilMoisture == small & AirH...	->	SoilWetting = level2
10	1.0	if	(Temperature == medium & SoilMoisture == small & AirH...	->	SoilWetting = level2
11	1.0	if	(Temperature == medium & SoilMoisture == small & AirH...	->	SoilWetting = level1
12	1.0	if	(Temperature == medium & SoilMoisture == medium & Ai...	->	SoilWetting = level1
13	1.0	if	(Temperature == medium & SoilMoisture == medium & Ai...	->	SoilWetting = level1
14	1.0	if	(Temperature == medium & SoilMoisture == medium & Ai...	->	SoilWetting = blocked
15	1.0	if	(Temperature == medium & SoilMoisture == large & AirH...	->	SoilWetting = blocked
16	1.0	if	(Temperature == medium & SoilMoisture == large & AirH...	->	SoilWetting = blocked
17	1.0	if	(Temperature == medium & SoilMoisture == large & AirH...	->	SoilWetting = blocked
18	1.0	if	(Temperature == large & SoilMoisture == small & AirHu...	->	SoilWetting = maxim
19	1.0	if	(Temperature == large & SoilMoisture == small & AirHu...	->	SoilWetting = maxim
20	1.0	if	(Temperature == large & SoilMoisture == small & AirHu...	->	SoilWetting = maxim
21	1.0	if	(Temperature == large & SoilMoisture == medium & AirH...	->	SoilWetting = level2
22	1.0	if	(Temperature == large & SoilMoisture == medium & AirH...	->	SoilWetting = level2
23	1.0	if	(Temperature == large & SoilMoisture == medium & AirH...	->	SoilWetting = level1
24	1.0	if	(Temperature == large & SoilMoisture == large & AirHu...	->	SoilWetting = blocked
25	1.0	if	(Temperature == large & SoilMoisture == large & AirHu...	->	SoilWetting = blocked
26	1.0	if	(Temperature == large & SoilMoisture == large & AirHu...	->	SoilWetting = blocked

Fig. 14. Definition of the set of rules for “SoilWetting”.

Simulation allows the monitoring operation, as shown in Figure 15 and 16. During the simulation it can be modify the temperature, the soil moisture and the air humidity as can see in fig 15 and 16. In fig 16 it can be observed how the rules are activating during modifying the inputs.

In fig. 17. it can be analyze the surface plot for the system.



Temperature: 13.0

SoilMoisture: 39.0

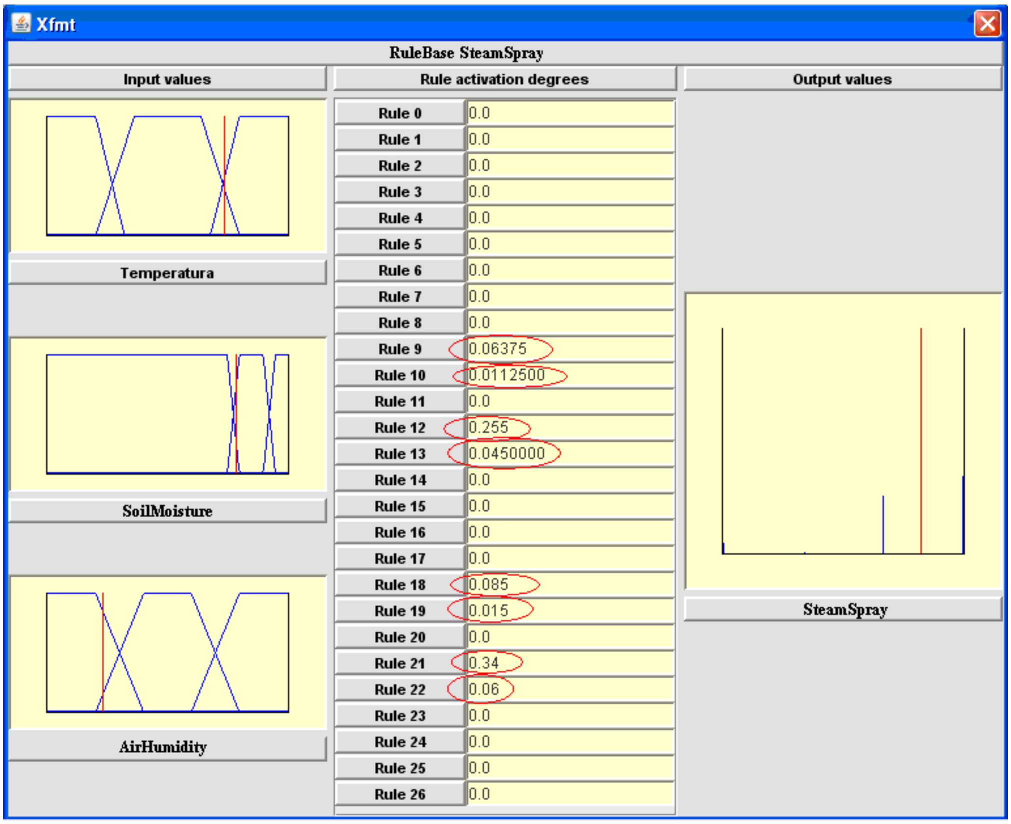
AirHumidity: 73.0

Heating: 23.021

SoilWetting: 44.531

SteamSpray: 25.319

Fig. 15. Monitoring the XFuzzy System

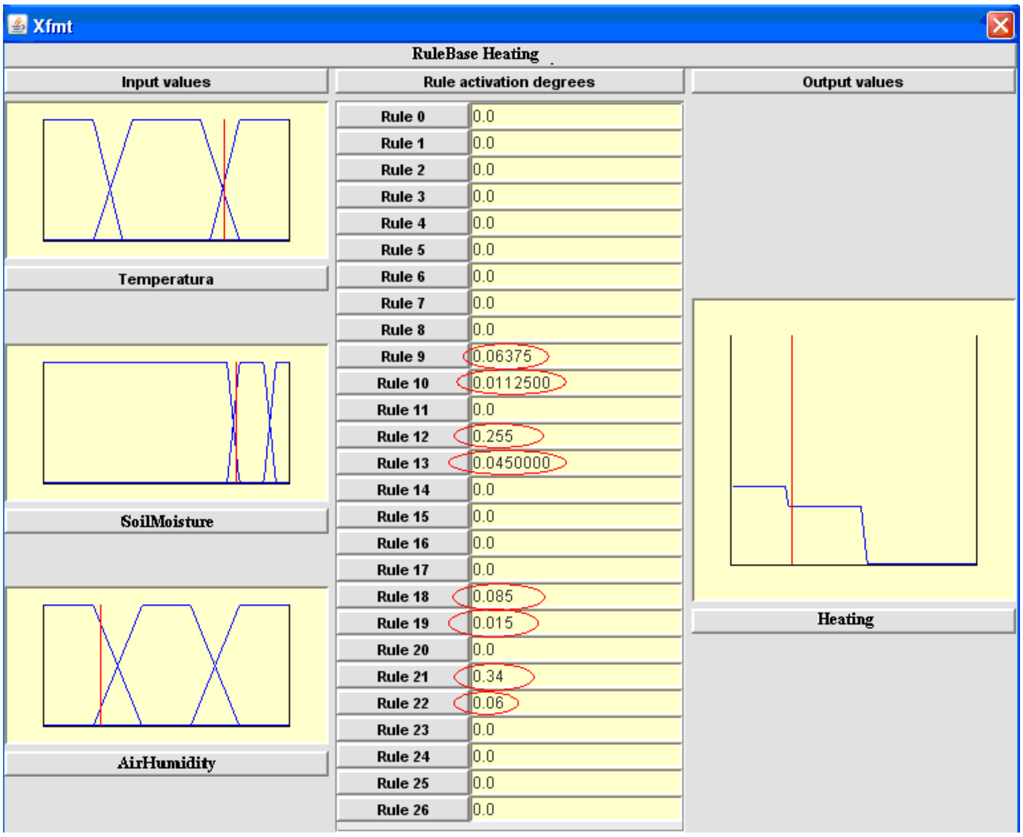


Xfmt

RuleBase SteamSpray

Rule activation degrees	
Rule 0	0.0
Rule 1	0.0
Rule 2	0.0
Rule 3	0.0
Rule 4	0.0
Rule 5	0.0
Rule 6	0.0
Rule 7	0.0
Rule 8	0.0
Rule 9	0.06375
Rule 10	0.0112500
Rule 11	0.0
Rule 12	0.255
Rule 13	0.0450000
Rule 14	0.0
Rule 15	0.0
Rule 16	0.0
Rule 17	0.0
Rule 18	0.085
Rule 19	0.015
Rule 20	0.0
Rule 21	0.34
Rule 22	0.06
Rule 23	0.0
Rule 24	0.0
Rule 25	0.0
Rule 26	0.0

a)



Xfmt

RuleBase Heating

Rule activation degrees	
Rule 0	0.0
Rule 1	0.0
Rule 2	0.0
Rule 3	0.0
Rule 4	0.0
Rule 5	0.0
Rule 6	0.0
Rule 7	0.0
Rule 8	0.0
Rule 9	0.06375
Rule 10	0.0112500
Rule 11	0.0
Rule 12	0.255
Rule 13	0.0450000
Rule 14	0.0
Rule 15	0.0
Rule 16	0.0
Rule 17	0.0
Rule 18	0.085
Rule 19	0.015
Rule 20	0.0
Rule 21	0.34
Rule 22	0.06
Rule 23	0.0
Rule 24	0.0
Rule 25	0.0
Rule 26	0.0

b)

Fig. 16. Rules monitoring
a) the proposed system for “SteamSpray” in case of Sugeno defuzzification method
b) the proposed system for “Heating” in case of Mamdani defuzzification method

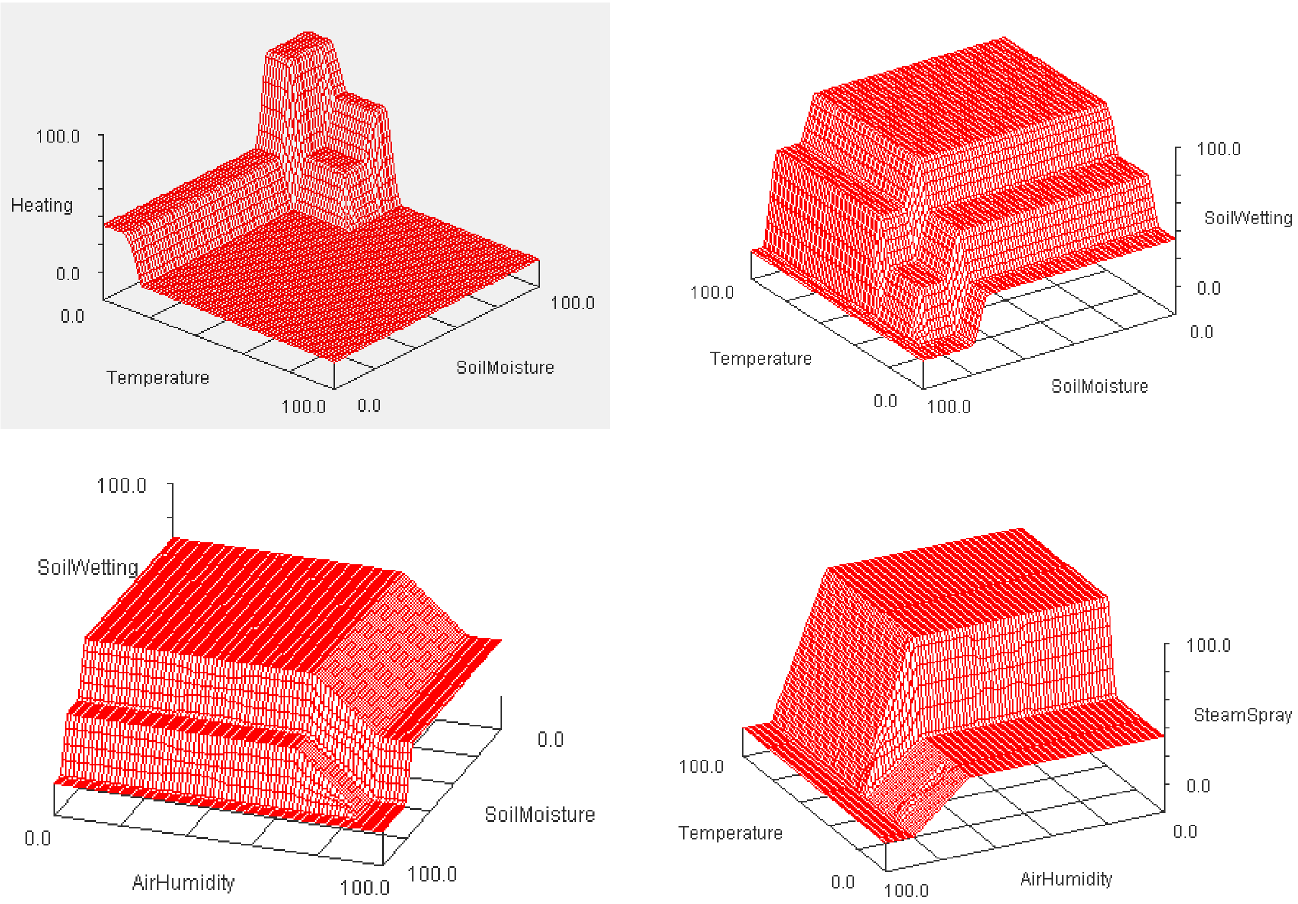


Fig. 17. The surfaces plot for the system

4. Implementation for the proposed system using the EZDSP TMS F2812.

To realize the proposed system using the development board EZDSP TMS F2812 were connected the inputs to three channels of the development board.

According to data acquired for soil moisture and air humidity, respective for ambient temperature will be sent the three output commands throw GPIO pins (General Purpose Input Output pins). For implementing the Fuzzy rules and to generate the output commands a program was implemented in C language using Code Composer Studio. Code Composer Studio v4 (CCS v4) is the integrated development environment for TI's DSPs, microcontrollers and application processors [6].

To verify the functioning of the proposed system it was connected to the output pins of the development board 9 LEDs for the three outputs taking into account that for each output there are three levels for the output command level 1, level 2 and maximum level). In figure 18 it can be see the proposed system realized with EzDSP TMS F2812.

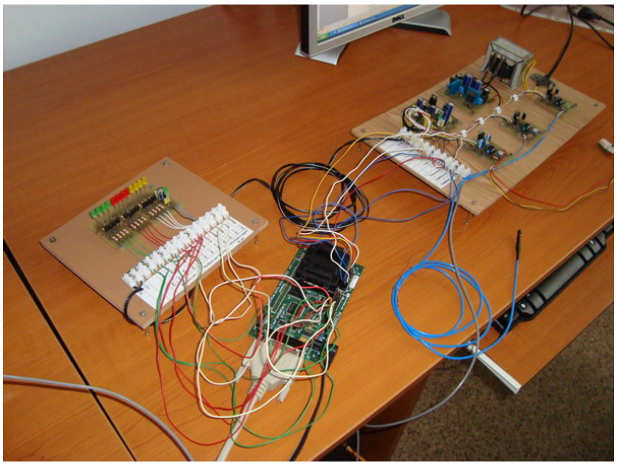


Fig. 18. The Fuzzy system implemented using EzDSP TMSF2812.

In fig. 19 it can be see the proposed system during functioning. In this case the SoiWetting is maxim, the Heating is blocked and the SteamSpray is also maxim.

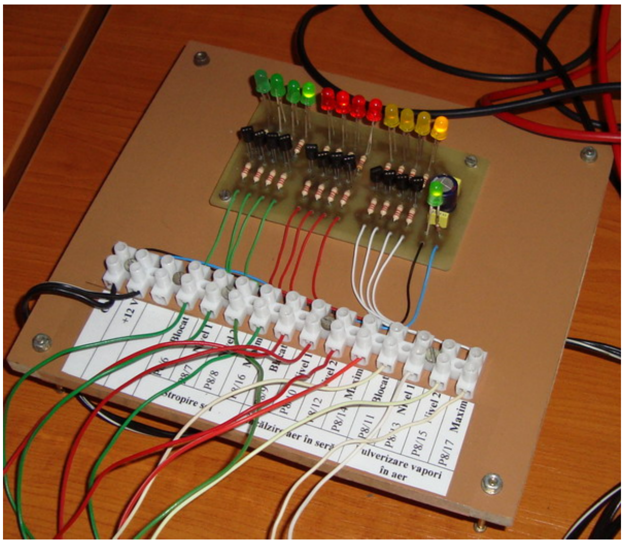


Fig. 19. Monitoring the outputs during system functioning

Below are some fragments from the program for the proposed system implemented in C by

```
//small temperature
#define     T_SMALL_a       0
#define     T_SMALL_b       0.0001
#define     T_SMALL_c       10.0
#define     T_SMALL_d       16.0
//medium temperature
#define     T_MEDIUM_a      10.0
#define     T_MEDIUM_b      18.0
#define     T_MEDIUM_c      32.0
#define     T_MEDIUM_d      40.0
// high emperature
#define     T_HIGH_a        34.0
#define     T_HIGH_b        40.0
#define     T_HIGH_c        99.999
#define     T_HIGH_d        100
//…
double  Temperature = 0, SoilMoisture
= 0, AirHumidity=0, outputS =0,
outputI =0,outputP =0 ;
double  *tSetPtr, *usSetPtr,*uaSetPtr;
set1          temperatureSet;
set2          SoilMoistureSet;
      set3          AirHumiditySet;

tSetPtr = (double *) & temperatureSet;
usSetPtr = (double *) &
SoilMoistureSet;
      uaSetPtr = (double *) &
AirHumiditySet;

while (TRUE)    {
AirHumidity = adc.ch8;
SoilMoisture = adc.ch1;
Temperature = adc.ch0;
ConvertData(&Temperature, &
SoilMoisture, & AirHumidity);
Fuzzifier(Temperature, SoilMoisture,
AirHumidity, tSetPtr, usSetPtr,
uaSetPtr);
Heating = RulesHeating((unsigned int
*) tSetPtr, (unsigned int *) usSetPtr,
(unsigned int *) uaSetPtr, SET1_SIZE,
SET2_SIZE, SET3_SIZE);
SoilWetting = RulesWetting((unsigned
int *) tSetPtr, (unsigned int *)
usSetPtr,
(unsigned int *) uaSetPtr, SET1_SIZE,
SET2_SIZE, SET3_SIZE);
SteamSpray =
RulesSteamSpraying((unsigned int *)
tSetPtr,           (unsigned int *)
usSetPtr,
(unsigned int *) uaSetPtr, SET1_SIZE,
SET2_SIZE, SET3_SIZE);
DINT;
IER = 0x0000;
IFR = 0x0000;

InitPieCtrl();
EALLOW;
GpioMuxRegs.GPAMUX.all=0X0000;
GpioMuxRegs.GPADIR.all=0XFFFF;
EALLOW;
GpioDataRegs.GPADAT.all= 0X0000;

//Fuzzy sets for SoilWetting
if
((SoilWetting>=0)&&(SoilWetting<=20))
GpioDataRegs.GPADAT.bit.GPIOA13 = 1;
      else
if((SoilWetting>20)&&(SoilWetting<=50)
) GpioDataRegs.GPADAT.bit.GPIOA14 =1;
       else
if((SoilWetting>50)&&(SoilWetting<=80)
) GpioDataRegs.GPADAT.bit.GPIOA15 =1;
            else
if((SoilWetting>80)&&(SoilWetting<=100
)) GpioDataRegs.GPADAT.bit.GPIOA0 =1;


//Fuzzy set for Heating
if ((Heating>=0)&&(Heating<=20))
GpioDataRegs.GPADAT.bit.GPIOA1 = 1;
      else
if((Heating>20)&&(Heating<=50))
GpioDataRegs.GPADAT.bit.GPIOA3 =1;
       else
if((Heating>50)&&(Heating<=80))
GpioDataRegs.GPADAT.bit.GPIOA5 =1;
            else
if((Heating>80)&&(Heating<=100))
GpioDataRegs.GPADAT.bit.GPIOA7 =1;

// Fuzzy set for SteamSpray
if ((SteamSpray>=0)&&(SteamSpray<=20))
GpioDataRegs.GPADAT.bit.GPIOA2 = 1;
```

```
      else
if((SteamSpray>20)&&(SteamSpray<=50))
GpioDataRegs.GPADAT.bit.GPIOA4 =1;
       else
if((SteamSpray>50)&&(SteamSpray<=80))
GpioDataRegs.GPADAT.bit.GPIOA6 =1;
            else
if((SteamSpray>80)&&(SteamSpray<=100))
GpioDataRegs.GPADAT.bit.GPIOA11 =1;

EDIS;
      }
}
```

The following code illustrate the rules definition in C language in case of using Sugeno defuzzification method.

```
unsigned int RulesHeating(memSet1,
memSet2, memSet3, memCount1,
memCount2, memCount3)
         double  *memSet1, *memSet2,
*memSet3, memCount1, memCount2,
memCount3;

{
 double suma = 0;
 int i,j,k,NrReguliActive = 0;
 static double reguli[SET1_SIZE]
[SET2_SIZE] [SET3_SIZE] =
                {
                    { {I_level2,
I_level2, I_level2},  {I_level2,
I_level2, I_level2 }, {I_maxim,
I_maxim, I_maxim} },
                    { {I_blocat,
I_blocat, I_blocat},  {I_blocat,
I_blocat, I_level1}, {I_level1,
I_level1, I_blocat} },
                    { {I_blocat,
I_blocat, I_blocat}, {I_blocat,
I_blocat, I_blocat}, {I_blocat,
I_blocat, I_blocat} },
                        };
   static double valori
[3][3][3]={{{0,0,0},{0,0,0},{0,0,0}},

{{0,0,0},{0,0,0},{0,0,0}},
{{0,0,0},{0,0,0},{0,0,0}}};

  double grade=0;
  for (i=0;i<3;i++)
    for (j=0;j<3;j++)
        for (k=0;k<3;k++){

valori[i][j][k]=(*(memSet1+i))*(*(memS
et2+j))*(*(memSet3+k));

grade=grade+valori[i][j][k];
                    }
for (i=0;i<3;i++)
    for (j=0;j<3;j++)
        for (k=0;k<3;k++)
        { if (valori[i][j][k]!=0)
NrReguliActive++;

valori[i][j][k]=valori[i][j][k]*reguli
[i][j][k];
  }
for (i=0;i<3;i++)
    for (j=0;j<3;j++)
        for (k=0;k<3;k++)
        { suma=suma+valori[i][j][k];

      }
      return(suma/grade);
}
```

The following code illustrate the rules definition in C language in case of using Mamdani defuzzification method.

```
unsigned int RulesHeating(memSet1,
memSet2, memSet3, memCount1,
memCount2, memCount3)
         double  *memSet1, *memSet2,
*memSet3;
         int memCount1, memCount2,
memCount3;

{
   double suma = 0;
   int i,j,k,NrReguliActive = 0;
   int Contor=0;

static double reguli[SET1_SIZE]
[SET2_SIZE] [SET3_SIZE] =
                {
                    { {I_Level2,
I_Level2, I_Level2},  {I_Level2,
I_Level2, I_Level2 }, {I_maxim,
I_maxim, I_maxim} },
                    { {I_blocat,
I_blocat, I_blocat},  {I_blocat,
I_blocat, I_Level1}, {I_Level1,
I_Level1, I_blocat} },
                    { {I_blocat,
I_blocat, I_blocat}, {I_blocat,
I_blocat, I_blocat}, {I_blocat,
I_blocat, I_blocat} },
                        };
   static double valori
[3][3][3]={{{0,0,0},{0,0,0},{0,0,0}},

{{0,0,0},{0,0,0},{0,0,0}},

{{0,0,0},{0,0,0},{0,0,0}}};

  double grade=0;
  for (i=0;i<3;i++)
    for (j=0;j<3;j++)
        for (k=0;k<3;k++){
```

```
valori[i][j][k]=(*(memSet1+i))*(*(memS
et2+j))*(*(memSet3+k));
                                }
for (i=0;i<3;i++)
    for (j=0;j<3;j++)
        for (k=0;k<3;k++)
        { if (valori[i][j][k]!=0)
NrReguliActive++;
          if
(reguli[i][j][k]==I_blocat)
             for
(Contor=I_BLOCAT_a;Contor<(I_BLOCAT_c+
I_BLOCAT_d)/2;Contor++) {

grade=grade+valori[i][j][k];

      suma=suma+valori[i][j][k]*reguli
[i][j][k];
                          }
  if (reguli[i][j][k]==I_Level1)
             for
(Contor=(I_BLOCAT_c+I_BLOCAT_d)/2;Cont
or<(I_Level1_c+I_Level1_d)/2;Contor++)
{

grade=grade+valori[i][j][k];

      suma=suma+valori[i][j][k]*reguli
[i][j][k];
                          }

        if (reguli[i][j][k]==I_Level2)
                for
(Contor=(I_Level1_c+I_Level1_d)/2;Cont
or<(I_Level2_c+I_Level2_d)/2;Contor++)
{

grade=grade+valori[i][j][k];

      suma=suma+valori[i][j][k]*reguli
[i][j][k];
                          }

          if
(reguli[i][j][k]==I_maxim)
              for
(Contor=(I_Level2_c+I_Level2_d)/2;Cont
or<=I_MAXIM_d;Contor++) {

grade=grade+valori[i][j][k];

      suma=suma+valori[i][j][k]*reguli
[i][j][k];
                          }
  }
      return(suma/grade);
}
```

5 . Conclusions

In the real situations, decision makers are often faced to a plurality of objectives and constraints in a world of imprecise data about the preferences of agents, the local constraints and the global environment [1]. For these reasons systems based on Fuzzy logic, like the proposed system are much simple and easier to implement. The simulation software, XFuzzy is also very useful for rule set definition and for testing the proposed system.

Texas Instruments TMS320F2812 DSP is ideally suited for fuzzy logic implementation. This chip may be used to produce low cost, high performance applications in a fraction of the time of conventional PID controllers [8], [9].

References

[1] Andre A. Keller, Fuzzy multiobjective optimization modeling with Mathematica, *WSEAS TRANSACTIONS on SYSTEMS* Issue 3, Volume 8, March 2009, pp 368-378

[2] Marius Constantin Popescu, Anca Petrisor, Adrian Drighiciu, Raducu Petrisor, Fuzzy Control Algorithm Implementation using LabWindows, *WSEAS TRANSACTIONS on SYSTEMS*, Issue 1, Volume 8, January 2009, pp 117-126

[3] http://www2.imse-cnm.csic.es/Xfuzzy

[4] Michael Negnevitsky *Artificial Intelligence A Guide to Intelligent Systems*, Addison-Wesley, 2005

[5] Gelu Ovidiu Tirian, Camelia Bretotean Pinca, Daniela Cristea, Marcel Topor, Applications of Fuzzy Logic in Continuous Casting, *WSEAS TRANSACTIONS on SYSTEMS and CONTROL*, Issue 3, Volume 5, March 2010, pp 133-142

[6] *TMS320 F2812, Implementation Tutorial*, Texas Instruments

[7] Byron Miller, *The Design and Development of Fuzzy Logic Controllers*, Impatient Publication, 1997

[8] *Fuzzy logic Meets DSPs*, Impatients Publication, 1998

[9] http://www.scribd.com/doc/28857722/F2812-DSP-Full-Tutorial