ACADEMIC TASK- 2

SYNOPSIS

ON

DESIGN A FUZZY LOGIC BASED EXPERT FOR AIR CONDITINOR

COURSE CODE: INT 246

SOFT COMPUTING TECHNIQUES

By

ASHOK KUMAR MEGHVANSHI- (11718652)

ASHISH PAL-(11717426)
SAI PRASAD VARMA-(11715439)

Under the Guidance of

Assi. Prof. USHA MITTAL MAM

SCHOOL OF COMPUTER SCIENCE AND ENGINEERING

LOVELY PROFESSIONAL UNIVERSITY

PHAGWARA, PUNJAB, (INDIA) -144402

2019-20

TABLE OF CONTENTS

1. Abstract……………………………………………………………………..2
2. Introduction……………………………………………………………….2
3. Air System Organization……………………………………………..3
4. Fuzzy Membership Function……………………………………….5
5. Fuzzy Rule Base…………………………………………………….….8
6. Fuzzy logic Algorithm……………………………………………….8
7. Program Code……………………………………………………….…9
8. Output…………………………………………………………………….27
9. Conclusion……………………………………………………………….29

ABSTRACT

Fuzzy logic control was developed to control the compressor motor speed, fan speed, fin direction and operation mode to maintain the room temperature at or closed to the set point temperature and save energy and keep devices from damage. This paper describes the development of Fuzzy logic algorithm for Air Condition control system. This system consists of four sensors for feedback control: first for input electric volt which used to save devices from damage due to alternated voltages, second for temperature and third for humidity and fourth for dew point. Simulation of the Fuzzy logic algorithm for Air Condition controlling system is carried out.

INTRODUCTION

The human brain can reason with imprecise environments or decision based on uncertain or value judgment like the air is fine or the speed is fast or facts that are partially true. The fuzzy logic is a branch of machine intelligence that help computers to process and utilize vague data of humanistic systems. Fuzzy inference systems (FIS) are information processing systems using fuzzy logic mechanism to represent the human reasoning process and to make decisions based on uncertain, imprecise environments. FIS consists of four parts fuzzifier, rules, inference engine, and DE fuzzifier as shown in the following. To design FIS, needed the perioral experiences of human experts about field of research or knowledge base that observed and collected from operations of systems. Fuzzy logic manipulates such imprecise and vague data as fine or fast help engineering to controls and describes systems using commonsense rules that refer to indefinite quantities. So that it is possible to transition from one rule to another as the input is varied smoothly. These rules are linguistically natural representation of human's (or expert's) Knowledge base, that provides easy understanding knowledge representing scheme for explain information that has been learnt by a computer. For air condition system to manipulate temperature and the humidity close to an aimed value, and to save the electrical energy that taken by Air Condition compressor / Fan while utilizing all available resources in the most efficient manner. Fuzzy logic system structure consists of database or prior knowledge that have to be crisp value to allow fuzzification using membership function, fuzzy control that manage fuzzification, rules evaluation, and defuzzification the output that also is crisp values as shown in figure (1).

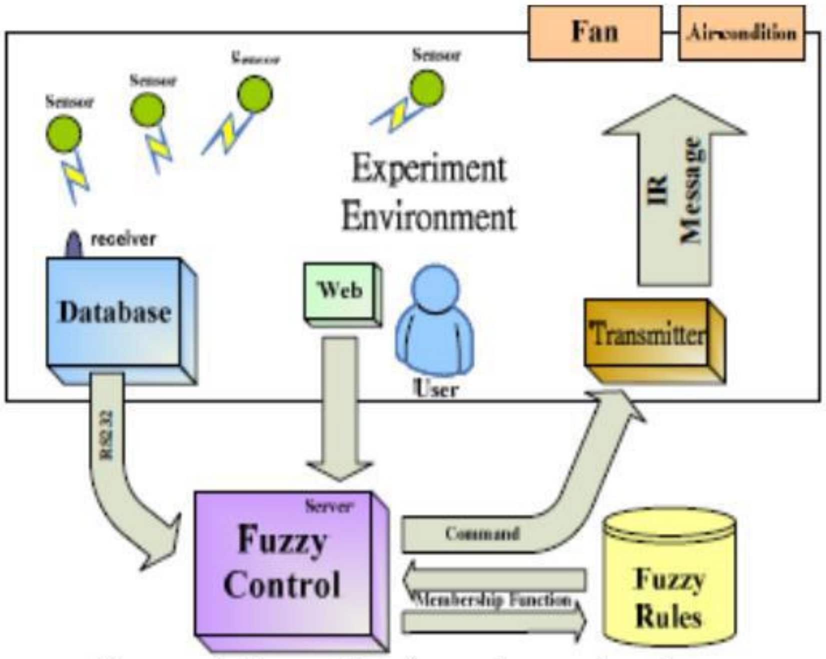
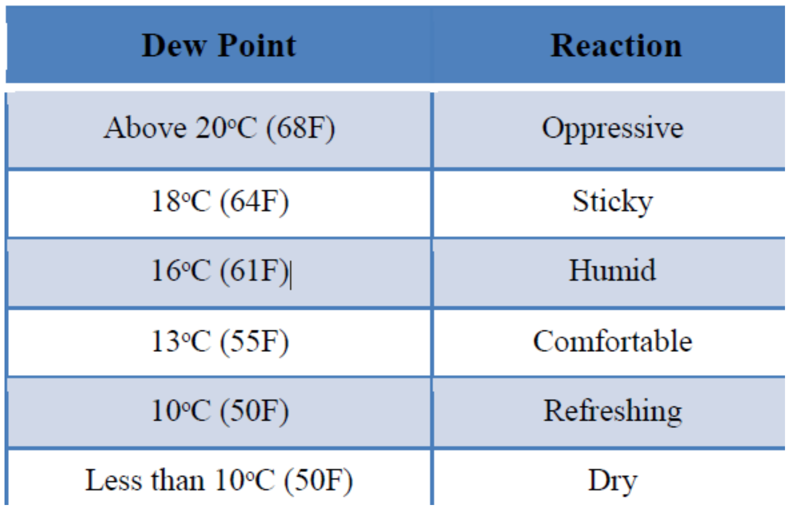


Figure 1.Fuzzy Logic system structure

To develop Air Condition system that control in humidity in own way without giving users any scope for changing the set point for the target humidity unlike the scope, it offers to change the set point for the target temperature through thermostat. That's leads to huge reducing energy using and providing necessary conditions for comfort living inside building. This system consists of four Sensors that captures temperature, Electric Volt, user temperature and humidity these reading are fuzzified figure 1. These are used to decide the fuzzy qualifier, which is decoded into a crisp value that

2. AIR SYSTEM ORGANIZATION

The dew point temperature determines what combinations of temperature and relative humidity (RH) will be possible in the storage environment. At a constant dew point, when the temperature goes up, the RH goes down and when the temperature goes down, the RH goes up. Controlling the dew point is key to managing the risk of material decay. Used Dew Point Calculator that designed to calculate and visually present the relationship between temperature, relative humidity and dew point. These evaluations are used to evaluate the preservation quality of the environment. Dew point temperature is used to measure humidity instead of relative humidity H). A standard Dew Point Human Reaction table are evaluated as in table (1).



Dew Point	Reaction
Above 20°C (68F)	Oppressive
18°C (64F)	Sticky
16°C (61F)	Humid
13°C (55F)	Comfortable
10°C (50F)	Refreshing
Less than 10°C (50F)	Dry

Table (1) : Human reaction to different levels of dew point

Air Condition Fuzzy logic control System takes four variables into consideration showing in the following block:

(1) User temperature (16°C→30°C continuous control).

(2) Actual temperature.

(3) dew point temperature.

(4) Electricity Volt

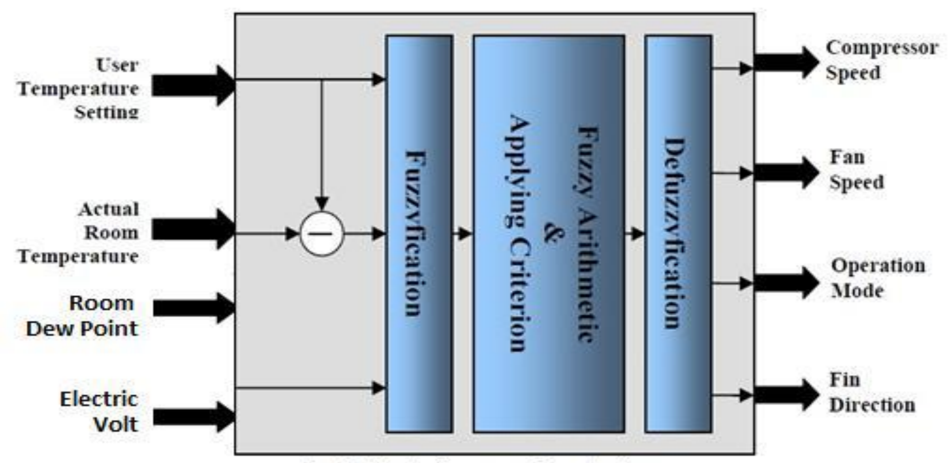
User temperature subtracted from actual temperature before sending data for fuzzification step. Fuzzy arithmetic and criterion step is applied on these variables and final result is DE fuzzified step to get following crisp results as showing in the following figure (2):

(1) Compressor Speed.

(2) Fan Speed.

(3) Mode of operation.

(4) Fin Direction.



Basic block diagram of controller

3. FUZZY MEMBERSHIP FUNCTION

The membership function editor in Fuzzy tool box is used to define the shapes of all membership functions associated with each membership variable [9,10]. In that system for each of the input and output variables the membership function is defined as follows

3.1 Input Variables

3.1.1 User temperature (UT)

User temperature that received by electronic, thermostat, are represented by three membership functions low, optimal, high as in table (2).

Table (2) : classification of user temperature (UT)

Input field	Range	Fuzzy set
User temperature (UT)	16 – 25	Low
	22 – 28	Optimal
	25 – 30	High

3.1.2 Temperature difference (Tdif)

The different between room temperature actually and user temperature, are represented by four membership functions negative, zero, positive, large as in table (3).

Table (3) : Classification of Temperature Difference (Tdif)

Input field	Range	Fuzzy set
Temperature Difference (Tdif)	-1 – 0	Negative
	-0.5 –0.5	Zero
	0 – 2	Positive
	1 - 3	Large

3.1.3 Dew Point (Td)

Dew point describes information about dew point temperature inside the room. Using two membership functions optimal, humid as in table (4).

Table (4) : Classification of Dew Point (Td)

Input field	Range	Fuzzy set
Dew Point (Td)	10 – 14	Optimal
	12 – 18	Humid

3.1.4 Electric Volt (EV)

Electric volt describes the information about alternating volt that difference and then according to that difference by controlling in compressor and fan to stand by or work as rule listed. Using two membership functions low, regular as in table (5)

Table (5) : Classification of Electric Volt (EV)

Input field	Range	Fuzzy set
Electric Volt (EV)	130 – 180	Low
	170 – 220	Regular

3.2 Output membership function

3.2.1 Compressor Speed (SC)

Compressor speed is characterized as low, medium and high different from 0 to 100%. To evaluate the membership function (MF) by scaling the ranges from 0 to 100 based on speed. calculated MF and the range are given as in table (6).

Table (6) : Classification of Compressor Speed (SC)

Input field	Range	Fuzzy set
Compressor Speed (SC)	0 – 50	Low
	40 – 80	Medium
	70 – 100	Fast

3.2.2 Fan Speed (Fc)

Fan speed is characterized as low, medium and high varied from 0 to 100%. To calculate membership function (MF) by scaling the ranges from 0 to 100 based on speed. Calculated MF and the ranges are given as in table (7).

Table (7) : Classification of Fan Speed (FC)

Input field	Range	Fuzzy set
Fan Speed (FC)	0 – 50	Low
	40 – 80	Medium
	70 – 100	Fast

3.2.3 Mode of operation (Mo)

Mode of operation decides whether air condition work like a dehumidifier only or normal.

3.2.4 Fin direction (Fn)

Fin directions directs air from air condition towards or away from occupants. Assuming top mounted air condition, $\theta = 0^o$ can be considered as towards and $\theta = 90^o$ as away from occupant.

4. FUZZY RULE BASE

Fuzzy rules referred to as the IF-THEN rule base form and deductive form. And rules are based on natural language representation and models which are themselves based on fuzzy sets and fuzzy logic. It typically expressed inference like if the fact (premise, hypothesis, antecedent), then infer, or derive another fact called a conclusion. It designs manual by a user, or automatic. The rules are defined by selecting the right sequence in the If-then sequence**.** It represents human empirical and heuristic knowledge in our language of communication that can be represented by fuzzy sets and logical connectivity of these sets. user temperature having three fuzzy ranges (low, optimal and high), temperature difference with four fuzzy ranges (negative, zero, positive and large), dew point with two fuzzy ranges (optimal and humid) and electric volt with two fuzzy ranges (low and regular) give a rule base matrix with size 3*4*2*2 = 48 cells. Every cell has four outputs, each for compressor speed, fan speed, mode of operation and fin direction.

5. FUZZY LOGIC ALGORITHM

Fuzzy logic algorithm is an algorithm that solves the problems expressed in the basic IF-THEN rule format. It consists of four steps as following:

Step 1 : linguistic Variables are the input variables of the system whose values are words or sentences from a natural language, instead of numerical values. A linguistic variable is generally decomposed into a set of linguistic terms.

Membership Functions Membership functions are used in the fuzzification and defuzzification steps of a FLS, to map the non-fuzzy input values to fuzzy linguistic terms and vice versa. A membership function is used to quantify a linguistic term.

Fuzzy Rules In a FLS, a rule base is constructed to control the output variable. A fuzzy rule is a simple IF-THEN rule with a condition and a conclusion.

Step 2: Fuzzification means adding uncertainty by design to crisp sets or to sets that are already fuzzy and spreading the information provided by a crisp number or symbol to its vicinity so that the close neighborhood of the crisp number can be recognize by the computational tools.

Step 3: Inference For each rule which represented in fuzzy level as set of restriction on the output based on certain conditions of the input. That restriction modeled by fuzzy set and relations and connected by linguistic connections like and, or, not and else, Obtaining the output which is a new fuzzy set which is the conclusion of rule since an implication operator is applied to the value of the antecedent obtained.

Step 4: defuzzification is the process of converting the result in fuzzy set form to a crisp result. It is important process for hardware application which process based on crisp data exchange. There is not theory to justify behavior of exchange other than commonsense reasoning such that the defuzzied output must represent a weight, voted, or must suitable solution. There are two main mechanism centroid method which based on finding a balance point of a property and maxima method which based in search for the highest pack whereas

6. PROGRAM CODE

```
import numpy as np

import skfuzzy as fuzz

from skfuzzy import control as ctrl


#input from user

UserTemp=ctrl.Antecedent(np.arange(16,30,1),'UserTemp')

TempDiff=ctrl.Antecedent(np.arange(-1,3,1),'TempDiff')

DewPoint=ctrl.Antecedent(np.arange(10,18,1),'DewPoint')

ElectricVolt=ctrl.Antecedent(np.arange(130,220,1),'ElectricVolt')


#output to user

CompSpeed=ctrl.Consequent(np.arange(0,100,1),'CompSpeed')

FanSpeed=ctrl.Consequent(np.arange(0,100,1),'FanSpeed')
```

```
ModeOfOperation=ctrl.Consequent(np.arange(0,10,1),'ModeOfOperation')
FinDir=ctrl.Consequent(np.arange(0,90,1),'FinDir')

#auto-membership function population is possible with .automf(3,5, or 7)
UserTemp.automf(3)
TempDiff.automf(3)
DewPoint.automf(3)
ElectricVolt.automf(3)

#LOW/NEGATIVE/ZERO=poor
#OPTIM/MEDIUM/POSITIVE=average
#HIGH/LARGE=good
#FAST=good
#OPTIMAL=poor
#HUMID=good
#AC=LOW
#DE=HIGH

#it is for compressor speed
CompSpeed['LOW'] =fuzz.trimf(CompSpeed.universe,[0,30,50])
CompSpeed['MEDIUM'] =fuzz.trimf(CompSpeed.universe,[40,60,80])
CompSpeed['FAST'] =fuzz.trimf(CompSpeed.universe,[70,90,100])
CompSpeed.view()

rule1=ctrl.Rule(UserTemp['poor'] | TempDiff['poor'] | DewPoint['poor'] | ElectricVolt['poor'] ,
CompSpeed['LOW'])
```

```
rule2=ctrl.Rule(UserTemp['average'] | TempDiff['poor'] | DewPoint['poor'] | ElectricVolt['poor']
, CompSpeed['LOW'])
```

```
rule3=ctrl.Rule(UserTemp['good'] | TempDiff['poor'] | DewPoint['poor'] | ElectricVolt['poor'] ,
CompSpeed['LOW'])

rule4=ctrl.Rule(UserTemp['poor'] | TempDiff['poor'] | DewPoint['poor'] | ElectricVolt['poor'] ,
CompSpeed['LOW'])

rule5=ctrl.Rule(UserTemp['average'] | TempDiff['poor'] | DewPoint['poor'] | ElectricVolt['poor']
, CompSpeed['LOW'])

rule6=ctrl.Rule(UserTemp['good'] | TempDiff['poor'] | DewPoint['poor'] | ElectricVolt['poor'] ,
CompSpeed['LOW'])

rule7=ctrl.Rule(UserTemp['poor'] | TempDiff['average'] | DewPoint['poor'] | ElectricVolt['poor']
, CompSpeed['LOW'])

rule8=ctrl.Rule(UserTemp['average'] | TempDiff['average'] | DewPoint['poor'] |
ElectricVolt['poor'] , CompSpeed['LOW'])

rule9=ctrl.Rule(UserTemp['good'] | TempDiff['average'] | DewPoint['poor'] | ElectricVolt['poor']
, CompSpeed['LOW'])

rule10=ctrl.Rule(UserTemp['poor'] | TempDiff['good'] | DewPoint['poor'] | ElectricVolt['poor'] ,
CompSpeed['LOW'])

rule11=ctrl.Rule(UserTemp['average'] | TempDiff['good'] | DewPoint['poor'] |
ElectricVolt['poor'] , CompSpeed['LOW'])

rule12=ctrl.Rule(UserTemp['good'] | TempDiff['good'] | DewPoint['poor'] | ElectricVolt['poor'] ,
CompSpeed['LOW'])

rule13=ctrl.Rule(UserTemp['poor'] | TempDiff['poor'] | DewPoint['poor'] | ElectricVolt['good'] ,
CompSpeed['LOW'])

rule14=ctrl.Rule(UserTemp['average'] | TempDiff['poor'] | DewPoint['poor'] |
ElectricVolt['good'] , CompSpeed['LOW'])

rule15=ctrl.Rule(UserTemp['good'] | TempDiff['poor'] | DewPoint['poor'] | ElectricVolt['good'] ,
CompSpeed['LOW'])

rule16=ctrl.Rule(UserTemp['poor'] | TempDiff['poor'] | DewPoint['poor'] | ElectricVolt['good'] ,
CompSpeed['LOW'])

rule17=ctrl.Rule(UserTemp['average'] | TempDiff['poor'] | DewPoint['poor'] |
ElectricVolt['good'] , CompSpeed['LOW'])

rule18=ctrl.Rule(UserTemp['good'] | TempDiff['poor'] | DewPoint['poor'] | ElectricVolt['good'] ,
CompSpeed['LOW'])
```

```
rule19=ctrl.Rule(UserTemp['poor'] | TempDiff['average'] | DewPoint['poor'] | ElectricVolt['good'], CompSpeed['FAST'])

rule20=ctrl.Rule(UserTemp['average'] | TempDiff['average'] | DewPoint['poor'] | ElectricVolt['good'] , CompSpeed['MEDIUM'])

rule21=ctrl.Rule(UserTemp['good'] | TempDiff['average'] | DewPoint['poor'] | ElectricVolt['good'] , CompSpeed['MEDIUM'])

rule22=ctrl.Rule(UserTemp['poor'] | TempDiff['good'] | DewPoint['poor'] | ElectricVolt['good'] , CompSpeed['LOW'])

rule23=ctrl.Rule(UserTemp['average'] | TempDiff['good'] | DewPoint['poor'] | ElectricVolt['good']  , CompSpeed['LOW'])

rule24=ctrl.Rule(UserTemp['good'] | TempDiff['good'] | DewPoint['poor'] | ElectricVolt['good']  , CompSpeed['LOW'])

rule25=ctrl.Rule(UserTemp['poor'] | TempDiff['poor'] | DewPoint['good'] | ElectricVolt['poor'] , CompSpeed['LOW'])

rule26=ctrl.Rule(UserTemp['average'] | TempDiff['poor'] | DewPoint['good'] | ElectricVolt['poor'] , CompSpeed['LOW'])

rule27=ctrl.Rule(UserTemp['good'] | TempDiff['poor'] | DewPoint['good'] | ElectricVolt['poor'] , CompSpeed['LOW'])

rule28=ctrl.Rule(UserTemp['poor'] | TempDiff['poor'] | DewPoint['good'] | ElectricVolt['poor'] , CompSpeed['LOW'])

rule29=ctrl.Rule(UserTemp['average'] | TempDiff['poor'] | DewPoint['good'] | ElectricVolt['poor'] , CompSpeed['LOW'])

rule30=ctrl.Rule(UserTemp['good'] | TempDiff['poor'] | DewPoint['good'] | ElectricVolt['poor'] , CompSpeed['LOW'])

rule31=ctrl.Rule(UserTemp['poor'] | TempDiff['average'] | DewPoint['good'] | ElectricVolt['poor'] , CompSpeed['LOW'])

rule32=ctrl.Rule(UserTemp['average'] | TempDiff['average'] | DewPoint['good'] | ElectricVolt['poor'] , CompSpeed['LOW'])

rule33=ctrl.Rule(UserTemp['good'] | TempDiff['average'] | DewPoint['good'] | ElectricVolt['poor'] , CompSpeed['LOW'])

rule34=ctrl.Rule(UserTemp['poor'] | TempDiff['good'] | DewPoint['good'] | ElectricVolt['poor'] , CompSpeed['LOW'])
```

rule35=ctrl.Rule(UserTemp['average'] | TempDiff['good'] | DewPoint['good'] | ElectricVolt['poor'] , CompSpeed['LOW'])

rule36=ctrl.Rule(UserTemp['good'] | TempDiff['good'] | DewPoint['good'] | ElectricVolt['poor'] , CompSpeed['LOW'])

rule37=ctrl.Rule(UserTemp['poor'] | TempDiff['poor'] | DewPoint['good'] | ElectricVolt['good'] , CompSpeed['FAST'])

rule38=ctrl.Rule(UserTemp['average'] | TempDiff['poor'] | DewPoint['good'] | ElectricVolt['good'], CompSpeed['LOW'])

rule39=ctrl.Rule(UserTemp['good'] | TempDiff['poor'] | DewPoint['good'] | ElectricVolt['good'] , CompSpeed['LOW'])

rule40=ctrl.Rule(UserTemp['poor'] | TempDiff['poor'] | DewPoint['good'] | ElectricVolt['good'] , CompSpeed['FAST'])

rule41=ctrl.Rule(UserTemp['average'] | TempDiff['poor'] | DewPoint['good'] | ElectricVolt['good'] , CompSpeed['MEDIUM'])

rule42=ctrl.Rule(UserTemp['good'] | TempDiff['poor'] | DewPoint['good'] | ElectricVolt['good'] , CompSpeed['MEDIUM'])

rule43=ctrl.Rule(UserTemp['poor'] | TempDiff['average'] | DewPoint['good'] | ElectricVolt['good'] , CompSpeed['FAST'])

rule44=ctrl.Rule(UserTemp['average'] | TempDiff['average'] | DewPoint['good'] | ElectricVolt['good'] , CompSpeed['FAST'])

rule45=ctrl.Rule(UserTemp['good'] | TempDiff['average'] | DewPoint['good'] | ElectricVolt['good'] , CompSpeed['MEDIUM'])

rule46=ctrl.Rule(UserTemp['poor'] | TempDiff['good'] | DewPoint['good'] | ElectricVolt['good'] , CompSpeed['FAST'])

rule47=ctrl.Rule(UserTemp['average'] | TempDiff['good'] | DewPoint['good'] | ElectricVolt['good'] , CompSpeed['FAST'])

rule48=ctrl.Rule(UserTemp['good'] | TempDiff['good'] | DewPoint['good'] | ElectricVolt['good'] , CompSpeed['FAST'])

CompSpeed_ctrl=ctrl.ControlSystem([rule1,rule2,rule3,rule4,rule5,rule6,rule7,rule8,rule9,rule10,rule11,rule12,rule13,rule14,rule15,rule16,rule17,rule18,rule19,rule20,rule21,rule22,rule23,ru

```
le24,rule25,rule26,rule27,rule28,rule29,rule30,rule31,rule32,rule33,rule34,rule35,rule36,rule37
,rule38,rule39,rule40,rule41,rule42,rule43,rule44,rule45,rule46,rule47,rule48])

CompSpeed1=ctrl.ControlSystemSimulation(CompSpeed_ctrl)


#UserTemp range(16,30,1)

#TempDiff range(-1,3,1)

#DewPoint range(10,18,1)

#ElectricVolt range(130,220,1)


CompSpeed1.input['UserTemp']=30

CompSpeed1.input['TempDiff']=3

CompSpeed1.input['DewPoint']=18

CompSpeed1.input['ElectricVolt']=220


CompSpeed1.compute()

print('')

print('*********************************************************')

print('*','    Compressor Speed :=', CompSpeed1.output['CompSpeed'],'              *')

CompSpeed.view(sim=CompSpeed1)


#it is for fan speed

FanSpeed['LOW']=fuzz.trimf(FanSpeed.universe,[0,30,50])

FanSpeed['MEDIUM']=fuzz.trimf(FanSpeed.universe,[40,60,80])

FanSpeed['FAST']=fuzz.trimf(FanSpeed.universe,[70,90,100])

FanSpeed.view()
```

```
rule1=ctrl.Rule(UserTemp['poor'] | TempDiff['poor'] | DewPoint['poor'] | ElectricVolt['poor'] ,
FanSpeed['LOW'] )
```

```
rule2=ctrl.Rule(UserTemp['average'] | TempDiff['poor'] | DewPoint['poor'] | ElectricVolt['poor']
, FanSpeed['LOW'] )

rule3=ctrl.Rule(UserTemp['good'] | TempDiff['poor'] | DewPoint['poor'] | ElectricVolt['poor'] ,
FanSpeed['LOW'] )

rule4=ctrl.Rule(UserTemp['poor'] | TempDiff['poor'] | DewPoint['poor'] | ElectricVolt['poor'] ,
FanSpeed['LOW'] )

rule5=ctrl.Rule(UserTemp['average'] | TempDiff['poor'] | DewPoint['poor'] | ElectricVolt['poor']
, FanSpeed['LOW'] )

rule6=ctrl.Rule(UserTemp['good'] | TempDiff['poor'] | DewPoint['poor'] | ElectricVolt['poor'] ,
FanSpeed['LOW'] )

rule7=ctrl.Rule(UserTemp['poor'] | TempDiff['average'] | DewPoint['poor'] | ElectricVolt['poor']
, FanSpeed['LOW'] )

rule8=ctrl.Rule(UserTemp['average'] | TempDiff['average'] | DewPoint['poor'] |
ElectricVolt['poor'] , FanSpeed['LOW'] )

rule9=ctrl.Rule(UserTemp['good'] | TempDiff['average'] | DewPoint['poor'] | ElectricVolt['poor']
, FanSpeed['LOW'] )

rule10=ctrl.Rule(UserTemp['poor'] | TempDiff['good'] | DewPoint['poor'] | ElectricVolt['poor'] ,
FanSpeed['LOW'] )

rule11=ctrl.Rule(UserTemp['average'] | TempDiff['good'] | DewPoint['poor'] |
ElectricVolt['poor'] , FanSpeed['LOW'] )

rule12=ctrl.Rule(UserTemp['good'] | TempDiff['good'] | DewPoint['poor'] | ElectricVolt['poor'] ,
FanSpeed['LOW'] )

rule13=ctrl.Rule(UserTemp['poor'] | TempDiff['poor'] | DewPoint['poor'] | ElectricVolt['good'] ,
FanSpeed['LOW'] )

rule14=ctrl.Rule(UserTemp['average'] | TempDiff['poor'] | DewPoint['poor'] |
ElectricVolt['good'] , FanSpeed['LOW'] )

rule15=ctrl.Rule(UserTemp['good'] | TempDiff['poor'] | DewPoint['poor'] | ElectricVolt['good'] ,
FanSpeed['LOW'] )

rule16=ctrl.Rule(UserTemp['poor'] | TempDiff['poor'] | DewPoint['poor'] | ElectricVolt['good'] ,
FanSpeed['FAST'] )

rule17=ctrl.Rule(UserTemp['average'] | TempDiff['poor'] | DewPoint['poor'] |
ElectricVolt['good'] , FanSpeed['MEDIUM'] )
```

```
rule18=ctrl.Rule(UserTemp['good'] | TempDiff['poor'] | DewPoint['poor'] | ElectricVolt['good'] , FanSpeed['LOW'] )

rule19=ctrl.Rule(UserTemp['poor'] | TempDiff['average'] | DewPoint['poor'] | ElectricVolt['good'], FanSpeed['FAST'] )

rule20=ctrl.Rule(UserTemp['average'] | TempDiff['average'] | DewPoint['poor'] | ElectricVolt['good'] , FanSpeed['MEDIUM'] )

rule21=ctrl.Rule(UserTemp['good'] | TempDiff['average'] | DewPoint['poor'] | ElectricVolt['good'] , FanSpeed['MEDIUM'] )

rule22=ctrl.Rule(UserTemp['poor'] | TempDiff['good'] | DewPoint['poor'] | ElectricVolt['good'] , FanSpeed['FAST'] )

rule23=ctrl.Rule(UserTemp['average'] | TempDiff['good'] | DewPoint['poor'] | ElectricVolt['good']  , FanSpeed['FAST'] )

rule24=ctrl.Rule(UserTemp['good'] | TempDiff['good'] | DewPoint['poor'] | ElectricVolt['good']  , FanSpeed['FAST'] )

rule25=ctrl.Rule(UserTemp['poor'] | TempDiff['poor'] | DewPoint['good'] | ElectricVolt['poor'] , FanSpeed['LOW'] )

rule26=ctrl.Rule(UserTemp['average'] | TempDiff['poor'] | DewPoint['good'] | ElectricVolt['poor'] , FanSpeed['LOW'] )

rule27=ctrl.Rule(UserTemp['good'] | TempDiff['poor'] | DewPoint['good'] | ElectricVolt['poor'] , FanSpeed['LOW'] )

rule28=ctrl.Rule(UserTemp['poor'] | TempDiff['poor'] | DewPoint['good'] | ElectricVolt['poor'] , FanSpeed['LOW'] )

rule29=ctrl.Rule(UserTemp['average'] | TempDiff['poor'] | DewPoint['good'] | ElectricVolt['poor'] , FanSpeed['LOW'] )

rule30=ctrl.Rule(UserTemp['good'] | TempDiff['poor'] | DewPoint['good'] | ElectricVolt['poor'] , FanSpeed['LOW'] )

rule31=ctrl.Rule(UserTemp['poor'] | TempDiff['average'] | DewPoint['good'] | ElectricVolt['poor'] , FanSpeed['LOW'] )

rule32=ctrl.Rule(UserTemp['average'] | TempDiff['average'] | DewPoint['good'] | ElectricVolt['poor'] , FanSpeed['LOW'] )
```

```
rule33=ctrl.Rule(UserTemp['good'] | TempDiff['average'] | DewPoint['good'] | ElectricVolt['poor'] , FanSpeed['LOW'] )
```

```
rule34=ctrl.Rule(UserTemp['poor'] | TempDiff['good'] | DewPoint['good'] | ElectricVolt['poor'] ,
FanSpeed['LOW'] )

rule35=ctrl.Rule(UserTemp['average'] | TempDiff['good'] | DewPoint['good'] |
ElectricVolt['poor'] , FanSpeed['LOW'] )

rule36=ctrl.Rule(UserTemp['good'] | TempDiff['good'] | DewPoint['good'] | ElectricVolt['poor'] ,
FanSpeed['LOW'] )

rule37=ctrl.Rule(UserTemp['poor'] | TempDiff['poor'] | DewPoint['good'] | ElectricVolt['good']  ,
FanSpeed['FAST'] )

rule38=ctrl.Rule(UserTemp['average'] | TempDiff['poor'] | DewPoint['good'] |
ElectricVolt['good'], FanSpeed['LOW'] )

rule39=ctrl.Rule(UserTemp['good'] | TempDiff['poor'] | DewPoint['good'] | ElectricVolt['good'] ,
FanSpeed['LOW'] )

rule40=ctrl.Rule(UserTemp['poor'] | TempDiff['poor'] | DewPoint['good'] | ElectricVolt['good'] ,
FanSpeed['FAST'] )

rule41=ctrl.Rule(UserTemp['average'] | TempDiff['poor'] | DewPoint['good'] |
ElectricVolt['good'] , FanSpeed['FAST'] )

rule42=ctrl.Rule(UserTemp['good'] | TempDiff['poor'] | DewPoint['good'] | ElectricVolt['good'] ,
FanSpeed['MEDIUM'] )

rule43=ctrl.Rule(UserTemp['poor'] | TempDiff['average'] | DewPoint['good'] |
ElectricVolt['good']  , FanSpeed['FAST'] )

rule44=ctrl.Rule(UserTemp['average'] | TempDiff['average'] | DewPoint['good'] |
ElectricVolt['good'] , FanSpeed['FAST'] )

rule45=ctrl.Rule(UserTemp['good'] | TempDiff['average'] | DewPoint['good'] |
ElectricVolt['good'] , FanSpeed['FAST'] )

rule46=ctrl.Rule(UserTemp['poor'] | TempDiff['good'] | DewPoint['good'] | ElectricVolt['good'] ,
FanSpeed['FAST'] )

rule47=ctrl.Rule(UserTemp['average'] | TempDiff['good'] | DewPoint['good'] |
ElectricVolt['good'] , FanSpeed['FAST'] )

rule48=ctrl.Rule(UserTemp['good'] | TempDiff['good'] | DewPoint['good'] | ElectricVolt['good']  ,
FanSpeed['FAST'] )
```

```
FanSpeed_ctrl=ctrl.ControlSystem([rule1,rule2,rule3,rule4,rule5,rule6,rule7,rule8,rule9,rule10,rule11,rule12,rule13,rule14,rule15,rule16,rule17,rule18,rule19,rule20,rule21,rule22,rule23,rule24,rule25,rule26,rule27,rule28,rule29,rule30,rule31,rule32,rule33,rule34,rule35,rule36,rule37,rule38,rule39,rule40,rule41,rule42,rule43,rule44,rule45,rule46,rule47,rule48])

FanSpeed1=ctrl.ControlSystemSimulation(FanSpeed_ctrl)


#UserTemp range(16,30,1)

#TempDiff range(-1,3,1)

#DewPoint range(10,18,1)

#ElectricVolt range(130,220,1)


FanSpeed1.input['UserTemp']=33

FanSpeed1.input['TempDiff']=3

FanSpeed1.input['DewPoint']=18

FanSpeed1.input['ElectricVolt']=220


FanSpeed1.compute()

print('*','    Fan Speed :=', FanSpeed1.output['FanSpeed'],'                    *')

FanSpeed.view(sim=FanSpeed1)


#it is for mode of operation

ModeOfOperation['AC']=fuzz.trimf(ModeOfOperation.universe,[0,9,9])

ModeOfOperation['DE']=fuzz.trimf(ModeOfOperation.universe,[0,0,9])

ModeOfOperation.view()


rule1=ctrl.Rule(UserTemp['poor']  |  TempDiff['poor']  |  DewPoint['poor']  |  ElectricVolt['poor']
,ModeOfOperation['AC'])
```

```
rule2=ctrl.Rule(UserTemp['average'] | TempDiff['poor'] | DewPoint['poor'] | ElectricVolt['poor']
,ModeOfOperation['AC'])
```

```
rule3=ctrl.Rule(UserTemp['good'] | TempDiff['poor'] | DewPoint['poor'] | ElectricVolt['poor']
,ModeOfOperation['AC'])

rule4=ctrl.Rule(UserTemp['poor'] | TempDiff['poor'] | DewPoint['poor'] | ElectricVolt['poor']
,ModeOfOperation['AC'])

rule5=ctrl.Rule(UserTemp['average'] | TempDiff['poor'] | DewPoint['poor'] | ElectricVolt['poor']
,ModeOfOperation['AC'])

rule6=ctrl.Rule(UserTemp['good'] | TempDiff['poor'] | DewPoint['poor'] | ElectricVolt['poor']
,ModeOfOperation['AC'])

rule7=ctrl.Rule(UserTemp['poor'] | TempDiff['average'] | DewPoint['poor'] | ElectricVolt['poor']
,ModeOfOperation['AC'])

rule8=ctrl.Rule(UserTemp['average'] | TempDiff['average'] | DewPoint['poor'] |
ElectricVolt['poor'] ,ModeOfOperation['AC'])

rule9=ctrl.Rule(UserTemp['good'] | TempDiff['average'] | DewPoint['poor'] | ElectricVolt['poor']
,ModeOfOperation['AC'])

rule10=ctrl.Rule(UserTemp['poor'] | TempDiff['good'] | DewPoint['poor'] | ElectricVolt['poor']
,ModeOfOperation['AC'])

rule11=ctrl.Rule(UserTemp['average'] | TempDiff['good'] | DewPoint['poor'] |
ElectricVolt['poor'] ,ModeOfOperation['AC'])

rule12=ctrl.Rule(UserTemp['good'] | TempDiff['good'] | DewPoint['poor'] | ElectricVolt['poor']
,ModeOfOperation['AC'])

rule13=ctrl.Rule(UserTemp['poor'] | TempDiff['poor'] | DewPoint['poor'] | ElectricVolt['good']
,ModeOfOperation['AC'])

rule14=ctrl.Rule(UserTemp['average'] | TempDiff['poor'] | DewPoint['poor'] |
ElectricVolt['good'] ,ModeOfOperation['AC'])

rule15=ctrl.Rule(UserTemp['good'] | TempDiff['poor'] | DewPoint['poor'] | ElectricVolt['good']
,ModeOfOperation['AC'])

rule16=ctrl.Rule(UserTemp['poor'] | TempDiff['poor'] | DewPoint['poor'] | ElectricVolt['good']
,ModeOfOperation['AC'])

rule17=ctrl.Rule(UserTemp['average'] | TempDiff['poor'] | DewPoint['poor'] |
ElectricVolt['good']  ,ModeOfOperation['AC'])

rule18=ctrl.Rule(UserTemp['good'] | TempDiff['poor'] | DewPoint['poor'] | ElectricVolt['good']
,ModeOfOperation['AC'])
```

```
rule19=ctrl.Rule(UserTemp['poor'] | TempDiff['average'] | DewPoint['poor'] |
ElectricVolt['good'] ,ModeOfOperation['AC'])

rule20=ctrl.Rule(UserTemp['average'] | TempDiff['average'] | DewPoint['poor'] |
ElectricVolt['good']  ,ModeOfOperation['AC'])

rule21=ctrl.Rule(UserTemp['good'] | TempDiff['average'] | DewPoint['poor'] |
ElectricVolt['good']  ,ModeOfOperation['AC'])

rule22=ctrl.Rule(UserTemp['poor'] | TempDiff['good'] | DewPoint['poor'] | ElectricVolt['good']
,ModeOfOperation['AC'])

rule23=ctrl.Rule(UserTemp['average'] | TempDiff['good'] | DewPoint['poor'] |
ElectricVolt['good']   ,ModeOfOperation['AC'])

rule24=ctrl.Rule(UserTemp['good'] | TempDiff['good'] | DewPoint['poor'] | ElectricVolt['good']
,ModeOfOperation['AC'])

rule25=ctrl.Rule(UserTemp['poor'] | TempDiff['poor'] | DewPoint['good'] | ElectricVolt['poor']
,ModeOfOperation['AC'])

rule26=ctrl.Rule(UserTemp['average'] | TempDiff['poor'] | DewPoint['good'] |
ElectricVolt['poor'] ,ModeOfOperation['AC'])

rule27=ctrl.Rule(UserTemp['good'] | TempDiff['poor'] | DewPoint['good'] | ElectricVolt['poor']
,ModeOfOperation['AC'])

rule28=ctrl.Rule(UserTemp['poor'] | TempDiff['poor'] | DewPoint['good'] | ElectricVolt['poor']
,ModeOfOperation['AC'])

rule29=ctrl.Rule(UserTemp['average'] | TempDiff['poor'] | DewPoint['good'] |
ElectricVolt['poor'] ,ModeOfOperation['AC'])

rule30=ctrl.Rule(UserTemp['good'] | TempDiff['poor'] | DewPoint['good'] | ElectricVolt['poor']
,ModeOfOperation['AC'])

rule31=ctrl.Rule(UserTemp['poor'] | TempDiff['average'] | DewPoint['good'] |
ElectricVolt['poor'] ,ModeOfOperation['AC'])

rule32=ctrl.Rule(UserTemp['average'] | TempDiff['average'] | DewPoint['good'] |
ElectricVolt['poor'] ,ModeOfOperation['AC'])

rule33=ctrl.Rule(UserTemp['good'] | TempDiff['average'] | DewPoint['good'] |
ElectricVolt['poor'] ,ModeOfOperation['AC'])

rule34=ctrl.Rule(UserTemp['poor'] | TempDiff['good'] | DewPoint['good'] | ElectricVolt['poor']
,ModeOfOperation['AC'])
```

```
rule35=ctrl.Rule(UserTemp['average'] | TempDiff['good'] | DewPoint['good'] |
ElectricVolt['poor'] ,ModeOfOperation['AC'])

rule36=ctrl.Rule(UserTemp['good'] | TempDiff['good'] | DewPoint['good'] | ElectricVolt['poor']
,ModeOfOperation['AC'])

rule37=ctrl.Rule(UserTemp['poor'] | TempDiff['poor'] | DewPoint['good'] | ElectricVolt['good']
,ModeOfOperation['DE'])

rule38=ctrl.Rule(UserTemp['average'] | TempDiff['poor'] | DewPoint['good'] |
ElectricVolt['good'],ModeOfOperation['DE'])

rule39=ctrl.Rule(UserTemp['good'] | TempDiff['poor'] | DewPoint['good'] | ElectricVolt['good']
,ModeOfOperation['DE'])

rule40=ctrl.Rule(UserTemp['poor'] | TempDiff['poor'] | DewPoint['good'] | ElectricVolt['good']
,ModeOfOperation['DE'])

rule41=ctrl.Rule(UserTemp['average'] | TempDiff['poor'] | DewPoint['good'] |
ElectricVolt['good']  ,ModeOfOperation['DE'])

rule42=ctrl.Rule(UserTemp['good'] | TempDiff['poor'] | DewPoint['good'] | ElectricVolt['good']
,ModeOfOperation['DE'])

rule43=ctrl.Rule(UserTemp['poor'] | TempDiff['average'] | DewPoint['good'] |
ElectricVolt['good']   ,ModeOfOperation['AC'])

rule44=ctrl.Rule(UserTemp['average'] | TempDiff['average'] | DewPoint['good'] |
ElectricVolt['good']  ,ModeOfOperation['AC'])

rule45=ctrl.Rule(UserTemp['good'] | TempDiff['average'] | DewPoint['good'] |
ElectricVolt['good']  ,ModeOfOperation['AC'])

rule46=ctrl.Rule(UserTemp['poor'] | TempDiff['good'] | DewPoint['good'] | ElectricVolt['good']
,ModeOfOperation['AC'])

rule47=ctrl.Rule(UserTemp['average'] | TempDiff['good'] | DewPoint['good'] |
ElectricVolt['good']  ,ModeOfOperation['AC'])

rule48=ctrl.Rule(UserTemp['good'] | TempDiff['good'] | DewPoint['good'] | ElectricVolt['good']
,ModeOfOperation['AC'])
```

```
ModeOfOperation_ctrl=ctrl.ControlSystem([rule1,rule2,rule3,rule4,rule5,rule6,rule7,rule8,rule9
,rule10,rule11,rule12,rule13,rule14,rule15,rule16,rule17,rule18,rule19,rule20,rule21,rule22,rul
e23,rule24,rule25,rule26,rule27,rule28,rule29,rule30,rule31,rule32,rule33,rule34,rule35,rule36,
rule37,rule38,rule39,rule40,rule41,rule42,rule43,rule44,rule45,rule46,rule47,rule48])

ModeOfOperation1=ctrl.ControlSystemSimulation(ModeOfOperation_ctrl)


#UserTemp range(16,30,1)

#TempDiff range(-1,3,1)

#DewPoint range(10,18,1)

#ElectricVolt range(130,220,1)


ModeOfOperation1.input['UserTemp']=30

ModeOfOperation1.input['TempDiff']=3

ModeOfOperation1.input['DewPoint']=18

ModeOfOperation1.input['ElectricVolt']=220


ModeOfOperation1.compute()

print('*','    Mode Of Operation :=', ModeOfOperation1.output['ModeOfOperation'],'              *')

ModeOfOperation.view(sim=ModeOfOperation1)


#it is for fin direction

FinDir['AWAY']=fuzz.trimf(FinDir.universe,[30,70,90])

FinDir['TOWARD']=fuzz.trimf(FinDir.universe,[0,30,70])

FinDir.view()


rule1=ctrl.Rule(UserTemp['poor'] | TempDiff['poor'] | DewPoint['poor'] | ElectricVolt['poor']
,FinDir['AWAY'])
```

```
rule2=ctrl.Rule(UserTemp['average'] | TempDiff['poor'] | DewPoint['poor'] | ElectricVolt['poor']
,FinDir['AWAY'])
```

rule3=ctrl.Rule(UserTemp['good'] | TempDiff['poor'] | DewPoint['poor'] | ElectricVolt['poor'] ,FinDir['AWAY'])

rule4=ctrl.Rule(UserTemp['poor'] | TempDiff['poor'] | DewPoint['poor'] | ElectricVolt['poor'] ,FinDir['AWAY'])

rule5=ctrl.Rule(UserTemp['average'] | TempDiff['poor'] | DewPoint['poor'] | ElectricVolt['poor'] ,FinDir['AWAY'])

rule6=ctrl.Rule(UserTemp['good'] | TempDiff['poor'] | DewPoint['poor'] | ElectricVolt['poor'] ,FinDir['AWAY'])

rule7=ctrl.Rule(UserTemp['poor'] | TempDiff['average'] | DewPoint['poor'] | ElectricVolt['poor'] ,FinDir['AWAY'])

rule8=ctrl.Rule(UserTemp['average'] | TempDiff['average'] | DewPoint['poor'] | ElectricVolt['poor'] ,FinDir['AWAY'])

rule9=ctrl.Rule(UserTemp['good'] | TempDiff['average'] | DewPoint['poor'] | ElectricVolt['poor'] ,FinDir['AWAY'])

rule10=ctrl.Rule(UserTemp['poor'] | TempDiff['good'] | DewPoint['poor'] | ElectricVolt['poor'] ,FinDir['AWAY'])

rule11=ctrl.Rule(UserTemp['average'] | TempDiff['good'] | DewPoint['poor'] | ElectricVolt['poor'] ,FinDir['AWAY'])

rule12=ctrl.Rule(UserTemp['good'] | TempDiff['good'] | DewPoint['poor'] | ElectricVolt['poor'] ,FinDir['AWAY'])

rule13=ctrl.Rule(UserTemp['poor'] | TempDiff['poor'] | DewPoint['poor'] | ElectricVolt['good'] ,FinDir['AWAY'])

rule14=ctrl.Rule(UserTemp['average'] | TempDiff['poor'] | DewPoint['poor'] | ElectricVolt['good'] ,FinDir['AWAY'])

rule15=ctrl.Rule(UserTemp['good'] | TempDiff['poor'] | DewPoint['poor'] | ElectricVolt['good'] ,FinDir['AWAY'])

rule16=ctrl.Rule(UserTemp['poor'] | TempDiff['poor'] | DewPoint['poor'] | ElectricVolt['good'] ,FinDir['TOWARD'])

rule17=ctrl.Rule(UserTemp['average'] | TempDiff['poor'] | DewPoint['poor'] | ElectricVolt['good'], FinDir['TOWARD'])

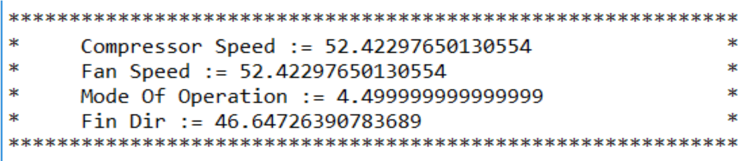
---

rule18=ctrl.Rule(UserTemp['good'] | TempDiff['poor'] | DewPoint['poor'] | ElectricVolt['good'] ,FinDir['AWAY'])

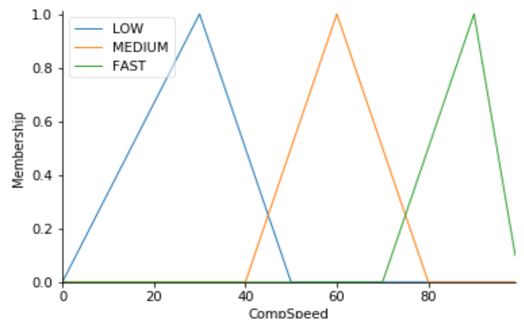
```
rule19=ctrl.Rule(UserTemp['poor'] | TempDiff['average'] | DewPoint['poor'] | ElectricVolt['good'] ,FinDir['TOWARD'])
rule20=ctrl.Rule(UserTemp['average'] | TempDiff['average'] | DewPoint['poor'] | ElectricVolt['good']  ,FinDir['TOWARD'])
rule21=ctrl.Rule(UserTemp['good'] | TempDiff['average'] | DewPoint['poor'] | ElectricVolt['good']  ,FinDir['TOWARD'])
rule22=ctrl.Rule(UserTemp['poor'] | TempDiff['good'] | DewPoint['poor'] | ElectricVolt['good'] ,FinDir['TOWARD'])
rule23=ctrl.Rule(UserTemp['average'] | TempDiff['good'] | DewPoint['poor'] | ElectricVolt['good']   ,FinDir['TOWARD'])
rule24=ctrl.Rule(UserTemp['good'] | TempDiff['good'] | DewPoint['poor'] | ElectricVolt['good'] ,FinDir['TOWARD'])
rule25=ctrl.Rule(UserTemp['poor'] | TempDiff['poor'] | DewPoint['good'] | ElectricVolt['poor'] ,FinDir['AWAY'])
rule26=ctrl.Rule(UserTemp['average'] | TempDiff['poor'] | DewPoint['good'] | ElectricVolt['poor'] ,FinDir['AWAY'])
rule27=ctrl.Rule(UserTemp['good'] | TempDiff['poor'] | DewPoint['good'] | ElectricVolt['poor'] ,FinDir['AWAY'])
rule28=ctrl.Rule(UserTemp['poor'] | TempDiff['poor'] | DewPoint['good'] | ElectricVolt['poor'] ,FinDir['AWAY'])
rule29=ctrl.Rule(UserTemp['average'] | TempDiff['poor'] | DewPoint['good'] | ElectricVolt['poor'] ,FinDir['AWAY'])
rule30=ctrl.Rule(UserTemp['good'] | TempDiff['poor'] | DewPoint['good'] | ElectricVolt['poor'] ,FinDir['AWAY'])
rule31=ctrl.Rule(UserTemp['poor'] | TempDiff['average'] | DewPoint['good'] | ElectricVolt['poor'] ,FinDir['AWAY'])
rule32=ctrl.Rule(UserTemp['average'] | TempDiff['average'] | DewPoint['good'] | ElectricVolt['poor'] ,FinDir['AWAY'])
rule33=ctrl.Rule(UserTemp['good'] | TempDiff['average'] | DewPoint['good'] | ElectricVolt['poor'] ,FinDir['AWAY'])
```

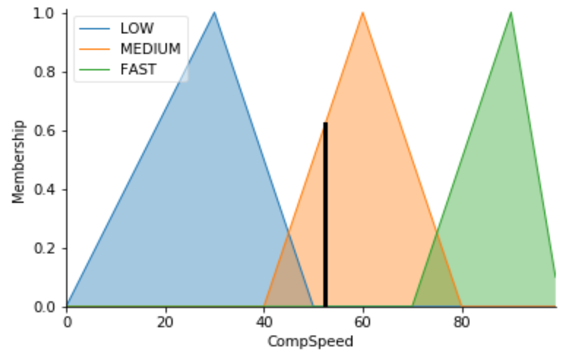
```
rule34=ctrl.Rule(UserTemp['poor'] | TempDiff['good'] | DewPoint['good'] | ElectricVolt['poor'] ,FinDir['AWAY'])

rule35=ctrl.Rule(UserTemp['average'] | TempDiff['good'] | DewPoint['good'] | ElectricVolt['poor'] ,FinDir['AWAY'])

rule36=ctrl.Rule(UserTemp['good'] | TempDiff['good'] | DewPoint['good'] | ElectricVolt['poor'] ,FinDir['AWAY'])

rule37=ctrl.Rule(UserTemp['poor'] | TempDiff['poor'] | DewPoint['good'] | ElectricVolt['good'] ,FinDir['TOWARD'])

rule38=ctrl.Rule(UserTemp['average'] | TempDiff['poor'] | DewPoint['good'] | ElectricVolt['good'],FinDir['AWAY'])

rule39=ctrl.Rule(UserTemp['good'] | TempDiff['poor'] | DewPoint['good'] | ElectricVolt['good'] ,FinDir['AWAY'])

rule40=ctrl.Rule(UserTemp['poor'] | TempDiff['poor'] | DewPoint['good'] | ElectricVolt['good'] ,FinDir['TOWARD'])

rule41=ctrl.Rule(UserTemp['average'] | TempDiff['poor'] | DewPoint['good'] | ElectricVolt['good']  ,FinDir['TOWARD'])

rule42=ctrl.Rule(UserTemp['good'] | TempDiff['poor'] | DewPoint['good'] | ElectricVolt['good'] ,FinDir['TOWARD'])

rule43=ctrl.Rule(UserTemp['poor'] | TempDiff['average'] | DewPoint['good'] | ElectricVolt['good']   ,FinDir['TOWARD'])

rule44=ctrl.Rule(UserTemp['average'] | TempDiff['average'] | DewPoint['good'] | ElectricVolt['good']  ,FinDir['TOWARD'])

rule45=ctrl.Rule(UserTemp['good'] | TempDiff['average'] | DewPoint['good'] | ElectricVolt['good']  ,FinDir['TOWARD'])

rule46=ctrl.Rule(UserTemp['poor'] | TempDiff['good'] | DewPoint['good'] | ElectricVolt['good'] ,FinDir['TOWARD'])

rule47=ctrl.Rule(UserTemp['average'] | TempDiff['good'] | DewPoint['good'] | ElectricVolt['good']  ,FinDir['TOWARD'])

rule48=ctrl.Rule(UserTemp['good'] | TempDiff['good'] | DewPoint['good'] | ElectricVolt['good'] ,FinDir['TOWARD'])

FinDir_ctrl=ctrl.ControlSystem([rule1,rule2,rule3,rule4,rule5,rule6,rule7,rule8,rule9,rule10,rule11,rule12,rule13,rule14,rule15,rule16,rule17,rule18,rule19,rule20,rule21,rule22,rule23,rule24,r
ule25,rule26,rule27,rule28,rule29,rule30,rule31,rule32,rule33,rule34,rule35,rule36,rule37,rule3
8,rule39,rule40,rule41,rule42,rule43,rule44,rule45,rule46,rule47,rule48])

FinDir1=ctrl.ControlSystemSimulation(FinDir_ctrl)



#UserTemp range(16,30,1)

#TempDiff range(-1,3,1)

#DewPoint range(10,18,1)

#ElectricVolt range(130,220,1)


FinDir1.input['UserTemp']=33

FinDir1.input['TempDiff']=3

FinDir1.input['DewPoint']=18

FinDir1.input['ElectricVolt']=220


FinDir1.compute()

print('*','    Fin Dir :=', FinDir1.output['FinDir'],'                    *')

print('************************************************************')

FinDir.view(sim=FinDir1)
```

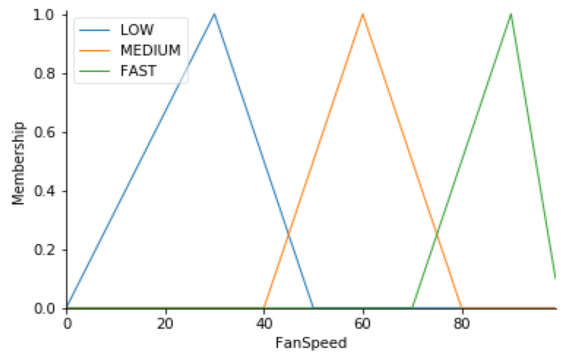
OUTPUT OF FUZZY LOGIC BASED EXPERT FOR AIR CONDITINOR

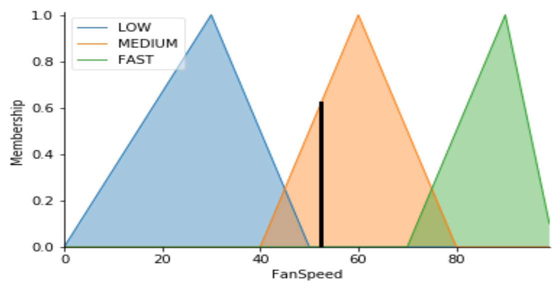


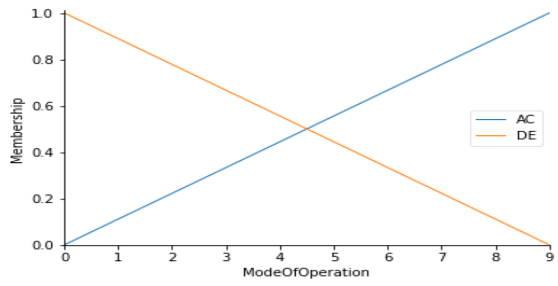
```
************************************************************
*      Compressor Speed := 52.42297650130554                *
*      Fan Speed := 52.42297650130554                       *
*      Mode Of Operation := 4.499999999999999               *
*      Fin Dir := 46.64726390783689                         *
************************************************************
```

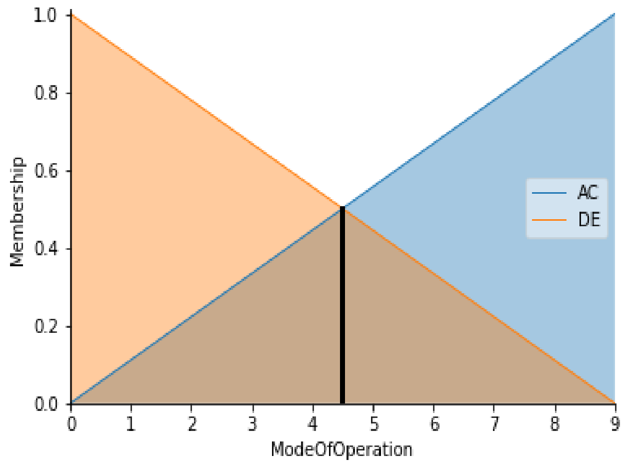


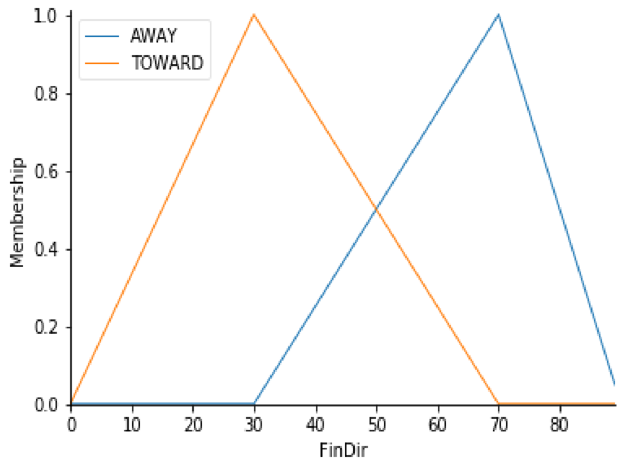


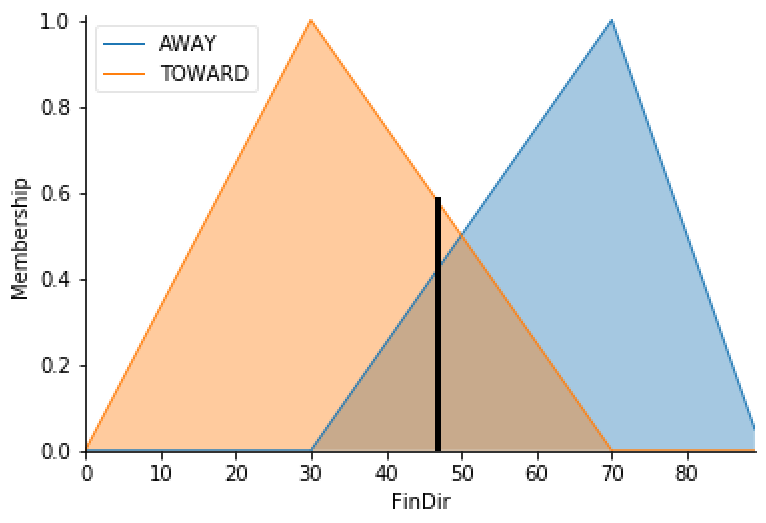



LOW
MEDIUM
FAST
Membership
FanSpeed
0
20
40
60
80
0.0
0.2
0.4
0.6
0.8
1.0


AC
DE
Membership
ModeOfOperation
0
1
2
3
4
5
6
7
8
9


AC
DE
Membership
ModeOfOperation
0
1
2
3
4
5
6
7
8
9


AWAY
TOWARD
Membership
FinDir
0
10
20
30
40
50
60
70
80



7. CONCLUSION

Previous Air Conditioning systems that are used to cool the rooms now can perform variety of functions. By adding intelligence to the system like fuzzy logic which is dealing with problems that are difficult and complex to study analytical that is now easy to solve in terms of linguistic variables. With most of the problems encountered in day to day life falling in this category, like washing machines, vacuum cleaners, etc., fuzzy logic is sure to make a great impact in human life. The neural net acts like computer because it maps input to output the neurons and synapses may be silicon component or equations in software that simulate their behavior. Supervised networks tune the rules of fuzzy system as if they were synapses. We will use neural network that can help fuzzy systems learn rules which can accepts pairs of input and output data and cluster them in a small number of classes.

DIVISION OF WORK AMONG GROUP MEMBERS:

1. **ASHOK KUMAR MEGHVANSHI: -** Create Rules of Fuzzy Logic, Compute Result of Data with the Help of Fuzzy Rules and Collecting Data and Information About Fuzzy.
2. **SAI PRASAD VARMA: -** Training of Model.
3. **ASHISH PAL: -** Input and Output Handling.