# Estimation of Pump Motor Speed and Operating Time for Irrigation Based on Real-Time Air Temperature and Soil Moisture Data Using Fuzzy Logic

**Baybars SERENER**
Mechatronics Engineering
Faculty of Technology / Afyon Kocatepe
Universitybaybars.serener@gmail.com

## Introduction

This study aims to estimate the speed and operating time of an irrigation pump motor for agricultural purposes by using real-time air temperature and soil moisture data. This is achieved through fuzzy logic control and measurements, implemented using Arduino and MATLAB.

## 1.Model Hardware

An Arduino is used to control the model, and the connections and hardware components are shown in Figure 1. The model utilizes a single motor. Soil temperature and moisture values obtained from potentiometers connected to the Arduino are transmitted to MATLAB via serial communication. Fuzzy logic outputs for the motor's speed and operating time are then sent back to the Arduino. This process involves processing soil temperature and moisture data using fuzzy logic control algorithms to determine motor operating parameters, ensuring real-time control on the Arduino.

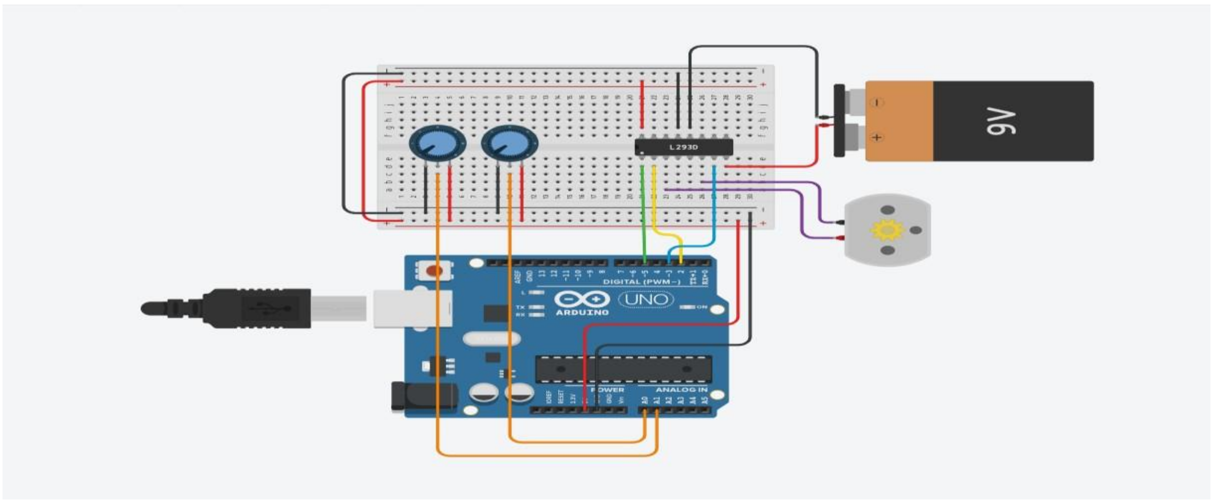
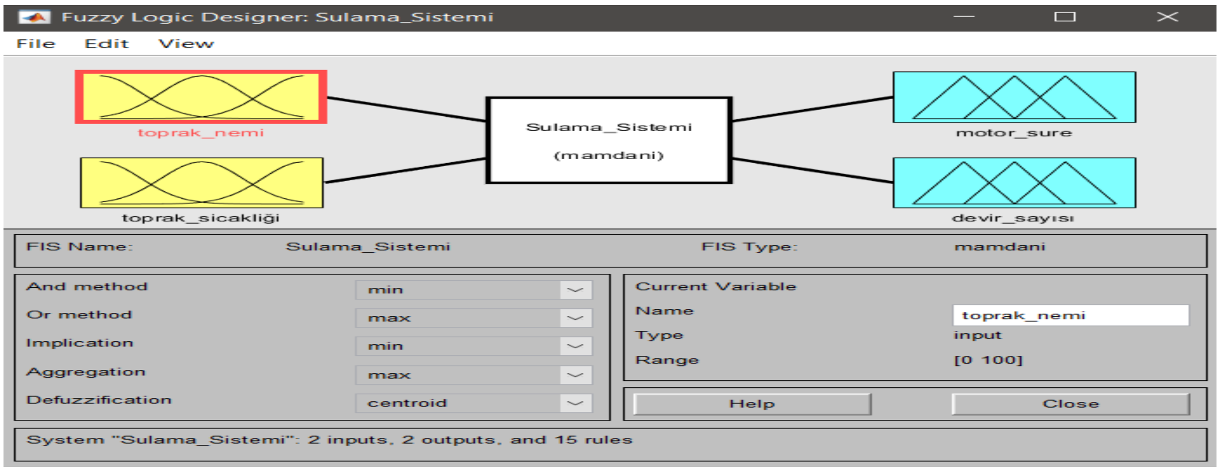


Figure 1 - Circuit Model

## 2. Design and Implementation of Fuzzy Logic Control System for Irrigation

The developed fuzzy logic control system consists of two inputs and two outputs. The first input represents soil temperature, while the second represents soil moisture. These two values are obtained from potentiometers. The outputs include the speed of the irrigation motor and the motor's operating time. The fuzzy logic control is designed as shown in Figure 2, using triangular membership functions for inputs and outputs. The details of these variables are provided below. In defuzzification, the AND (min) compositional inference rule and Mamdani centroid method were used.



FIS Name: Sulama_Sistemi
FIS Type: mamdani

And method: min
Or method: max
Implication: min
Aggregation: max
Defuzzification: centroid

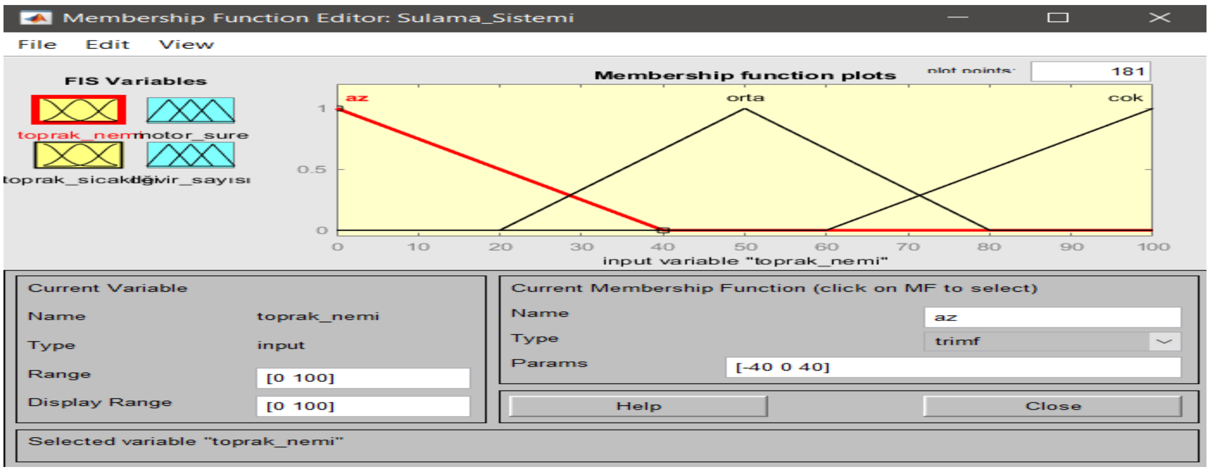
Current Variable
Name: toprak_nemi
Type: input
Range: [0 100]

System "Sulama_Sistemi": 2 inputs, 2 outputs, and 15 rules

Figure 2 - Fuzzy Logic Mamdani Control Unit

2.1) **The input variable, soil moisture, ranges from 0 to 100. The values are:**

- **Low**: [-40, 0, 40] (trimf)
- **Medium**: [20, 50, 80] (trimf)
- **High**: [60, 100, 140] (trimf)



Current Variable
Name: toprak_nemi
Type: input
Range: [0 100]
Display Range: [0 100]

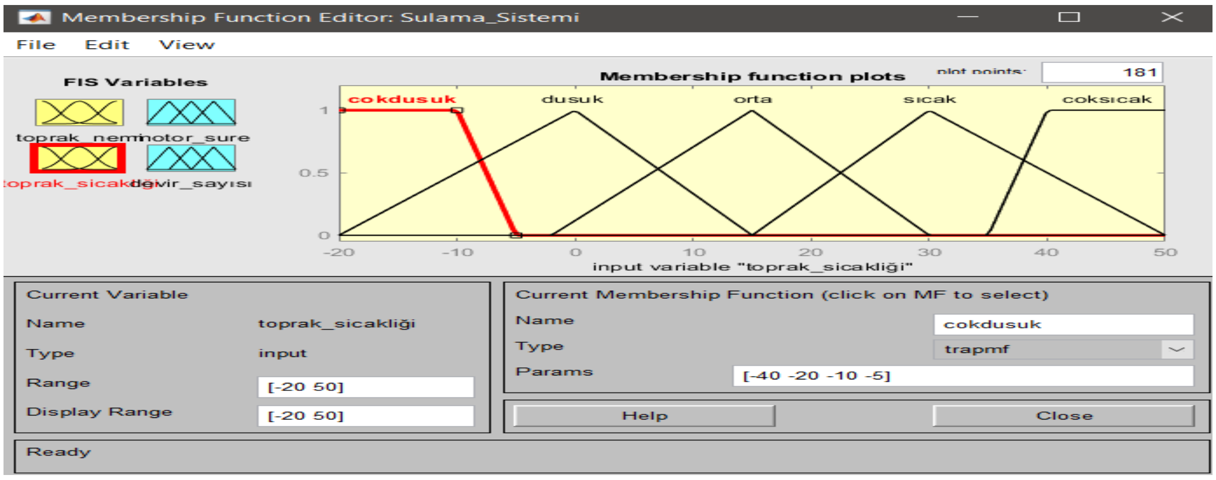
Current Membership Function (click on MF to select)
Name: az
Type: trimf
Params: [-40 0 40]

Selected variable "toprak_nemi"

Figure 3 - Fuzzy Logic Membership Functions for Soil Moisture Input Variable

**2.2) The input variable, soil temperature, ranges from -20 to 50. The values are:**

- **Very Low**: [-40, -20, -10, -5] (trapmf)
- **Low**: [-20, 0, 15] (trimf)
- **Medium**: [-2, 15, 30] (trimf)
- **High**: [-40, 0, 40] (trimf)
- **Very High**: [15, 30, 50] (trapmf)



plot points: 181

Current Variable

| | |
|---|---|
| Name | toprak_sicakliği |
| Type | input |
| Range | [-20 50] |
| Display Range | [-20 50] |

Current Membership Function (click on MF to select)

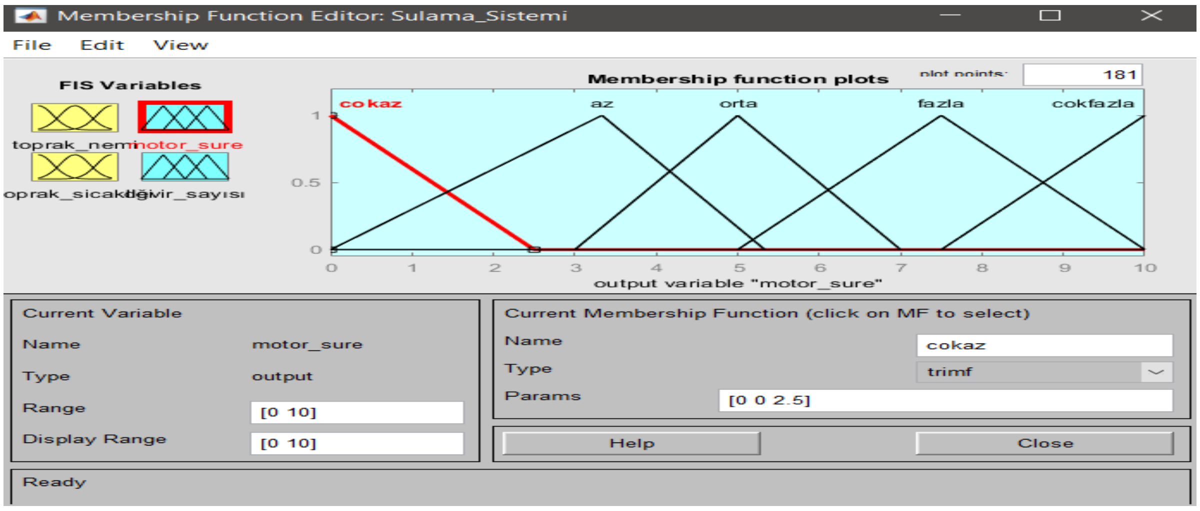
| | |
|---|---|
| Name | cokdusuk |
| Type | trapmf |
| Params | [-40 -20 -10 -5] |

Ready

Figure 4 - Fuzzy Logic Membership Functions for Soil Temperature Input Variable

2.3) **The output variable, motor time, ranges from 0 to 10. The values are:**

- **Very Short**: [0, 0, 2.5] (trimf)
- **Short**: [0, 3.333, 5.333] (trimf)
- **Medium**: [3, 5, 7] (trimf)
- **Long**: [5, 7.5, 10] (trimf)
- **Very Long**: [7.5, 10, 16.67] (trimf)



plot points: 181

Current Variable

| | |
|---|---|
| Name | motor_sure |
| Type | output |
| Range | [0 10] |
| Display Range | [0 10] |

Current Membership Function (click on MF to select)

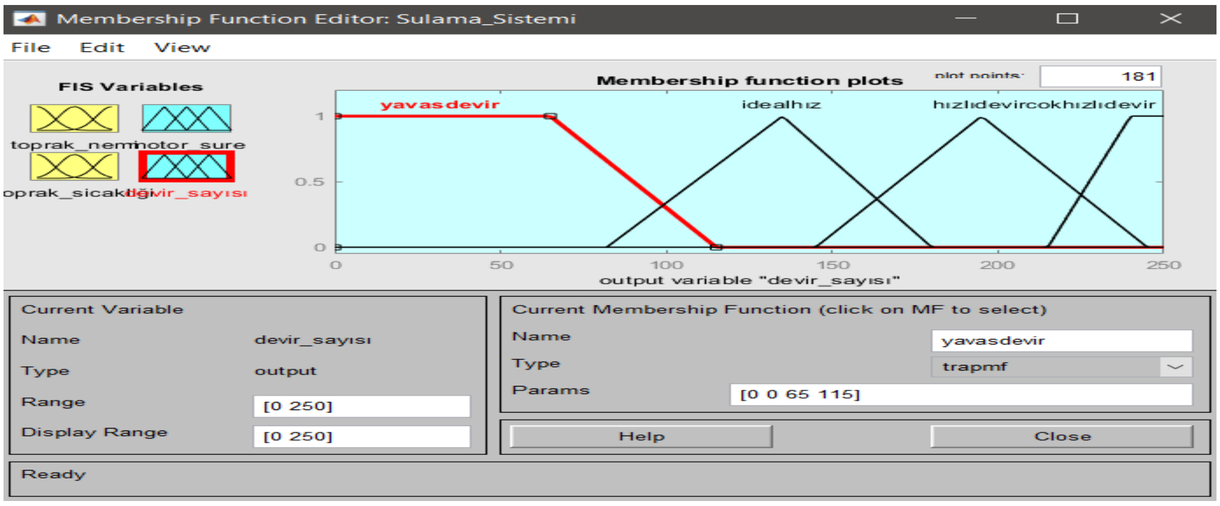
| | |
|---|---|
| Name | cokaz |
| Type | trimf |
| Params | [0 0 2.5] |

Ready

Figure 5 - Fuzzy Logic Membership Functions for Motor Time Output Variable

**2.4) The output variable, motor speed, ranges from 0 to 250. The values are:**

- **Slow Speed**: [0, 0, 65, 115] (trapmf)
- **Medium Speed**: [82, 135, 180] (trimf)
- **Fast Speed**: [145, 195, 245] (trimf)
- **Very Fast Speed**: [215, 240, 250, 280] (trapmf)



Current Variable

Name devir_sayısı

Type output

Range [0 250]

Display Range [0 250]

Current Membership Function (click on MF to select)

Name yavasdevir

Type trapmf

Params [0 0 65 115]

Ready

Figure 6 - Fuzzy Logic Membership Functions for Motor Speed Output Variable

2.5) Rules Created for Obtaining Optimal Values

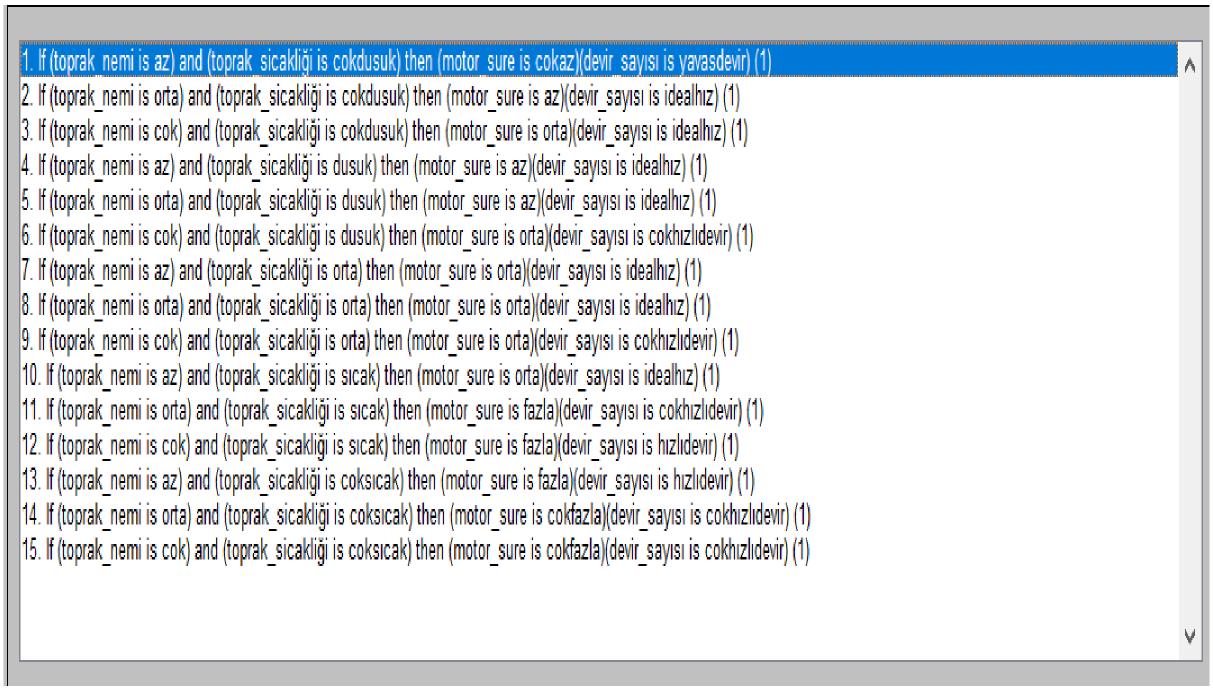
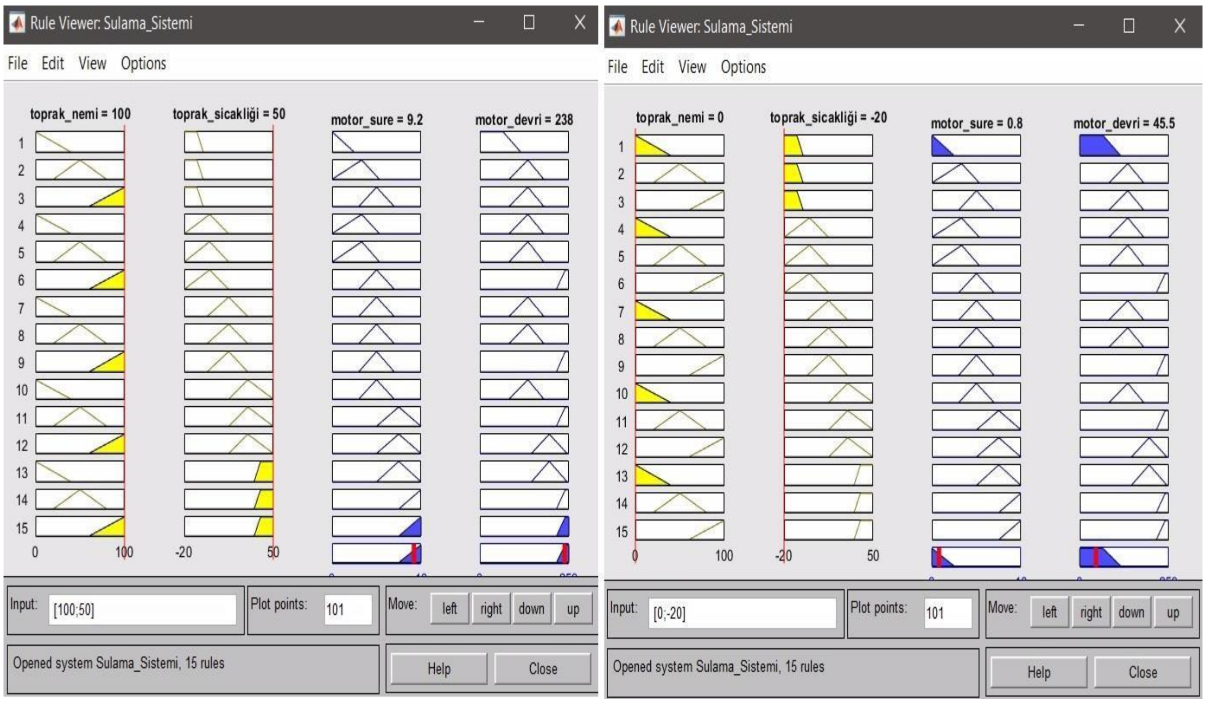

1. If (toprak_nemi is az) and (toprak_sicakliği is cokdusuk) then (motor_sure is cokaz)(devir_sayısı is yavasdevir) (1)
2. If (toprak_nemi is orta) and (toprak_sicakliği is cokdusuk) then (motor_sure is az)(devir_sayısı is idealhız) (1)
3. If (toprak_nemi is cok) and (toprak_sicakliği is cokdusuk) then (motor_sure is orta)(devir_sayısı is idealhız) (1)
4. If (toprak_nemi is az) and (toprak_sicakliği is dusuk) then (motor_sure is az)(devir_sayısı is idealhız) (1)
5. If (toprak_nemi is orta) and (toprak_sicakliği is dusuk) then (motor_sure is az)(devir_sayısı is idealhız) (1)
6. If (toprak_nemi is cok) and (toprak_sicakliği is dusuk) then (motor_sure is orta)(devir_sayısı is cokhızlıdevir) (1)
7. If (toprak_nemi is az) and (toprak_sicakliği is orta) then (motor_sure is orta)(devir_sayısı is idealhız) (1)
8. If (toprak_nemi is orta) and (toprak_sicakliği is orta) then (motor_sure is orta)(devir_sayısı is idealhız) (1)
9. If (toprak_nemi is cok) and (toprak_sicakliği is orta) then (motor_sure is orta)(devir_sayısı is cokhızlıdevir) (1)
10. If (toprak_nemi is az) and (toprak_sicakliği is sıcak) then (motor_sure is orta)(devir_sayısı is idealhız) (1)
11. If (toprak_nemi is orta) and (toprak_sicakliği is sıcak) then (motor_sure is fazla)(devir_sayısı is cokhızlıdevir) (1)
12. If (toprak_nemi is cok) and (toprak_sicakliği is sıcak) then (motor_sure is fazla)(devir_sayısı is hızlıdevir) (1)
13. If (toprak_nemi is az) and (toprak_sicakliği is coksıcak) then (motor_sure is fazla)(devir_sayısı is hızlıdevir) (1)
14. If (toprak_nemi is orta) and (toprak_sicakliği is coksıcak) then (motor_sure is cokfazla)(devir_sayısı is cokhızlıdevir) (1)
15. If (toprak_nemi is cok) and (toprak_sicakliği is coksıcak) then (motor_sure is cokfazla)(devir_sayısı is cokhızlıdevir) (1)

Figure 7 - Fuzzy Logic Rule List


Rule Viewer: Sulama_Sistemi

toprak_nemi = 100 | toprak_sicakliği = 50 | motor_sure = 9.2 | motor_devri = 238

Input: [100;50] Plot points: 101

Opened system Sulama_Sistemi, 15 rules

Rule Viewer: Sulama_Sistemi

toprak_nemi = 0 | toprak_sicakliği = -20 | motor_sure = 0.8 | motor_devri = 45.5

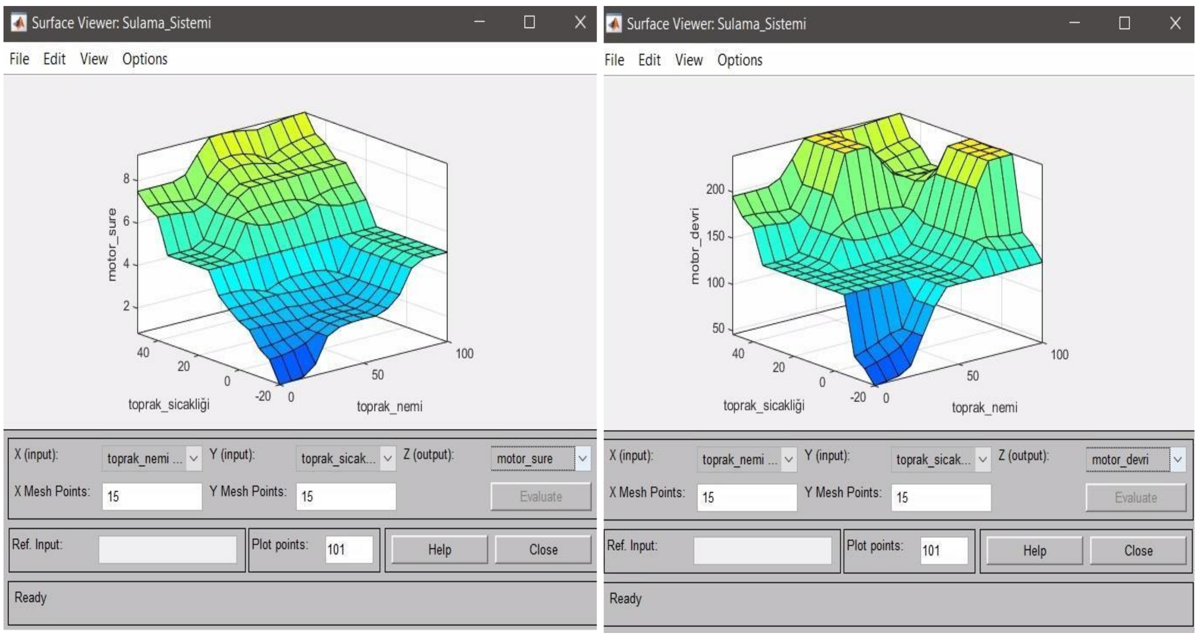
Input: [0;-20] Plot points: 101

Opened system Sulama_Sistemi, 15 rules

Figure 8 - Fuzzy Logic Rule Table

### 2.6) Surfaces Created with Optimal Rules

In fuzzy logic systems, a surface represents the combination of input variables and is used to determine how the system will respond. In fuzzy logic, an output variable value is determined for each surface. Surfaces are generally used to define the rules of fuzzy logic systems and determine how the system will react.


Surface Viewer: Sulama_Sistemi

X (input): toprak_nemi ... Y (input): toprak_sicak... Z (output): motor_sure

X Mesh Points: 15 Y Mesh Points: 15

Plot points: 101

Ready

Surface Viewer: Sulama_Sistemi

X (input): toprak_nemi ... Y (input): toprak_sicak... Z (output): motor_devri

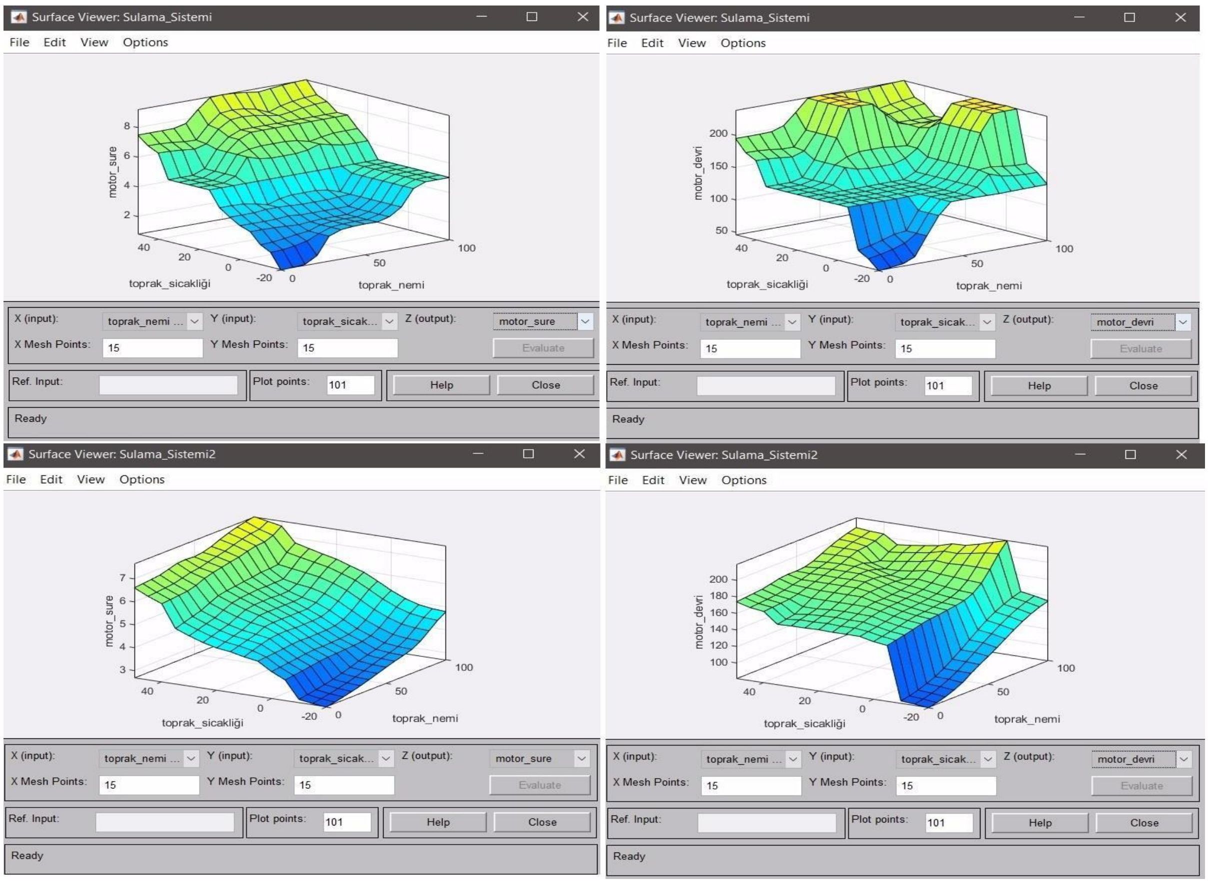
X Mesh Points: 15 Y Mesh Points: 15

Plot points: 101

Ready

Figure 9 - Three-Dimensional Surface Graphs for Motor Time and Motor Speed

## 3) Comparison of Triangular MF and Gaussian MF



X (input): toprak_nemi ... Y (input): toprak_sicak... Z (output): motor_sure
X Mesh Points: 15 Y Mesh Points: 15
Ref. Input: Plot points: 101

X (input): toprak_nemi ... Y (input): toprak_sicak... Z (output): motor_devri
X Mesh Points: 15 Y Mesh Points: 15
Ref. Input: Plot points: 101

X (input): toprak_nemi ... Y (input): toprak_sicak... Z (output): motor_sure
X Mesh Points: 15 Y Mesh Points: 15
Ref. Input: Plot points: 101

X (input): toprak_nemi ... Y (input): toprak_sicak... Z (output): motor_devri
X Mesh Points: 15 Y Mesh Points: 15
Ref. Input: Plot points: 101

Figure 10 - Comparison of Triangular MF (Top) and Gaussian MF (Bottom)

### Effects of Using Gaussmf

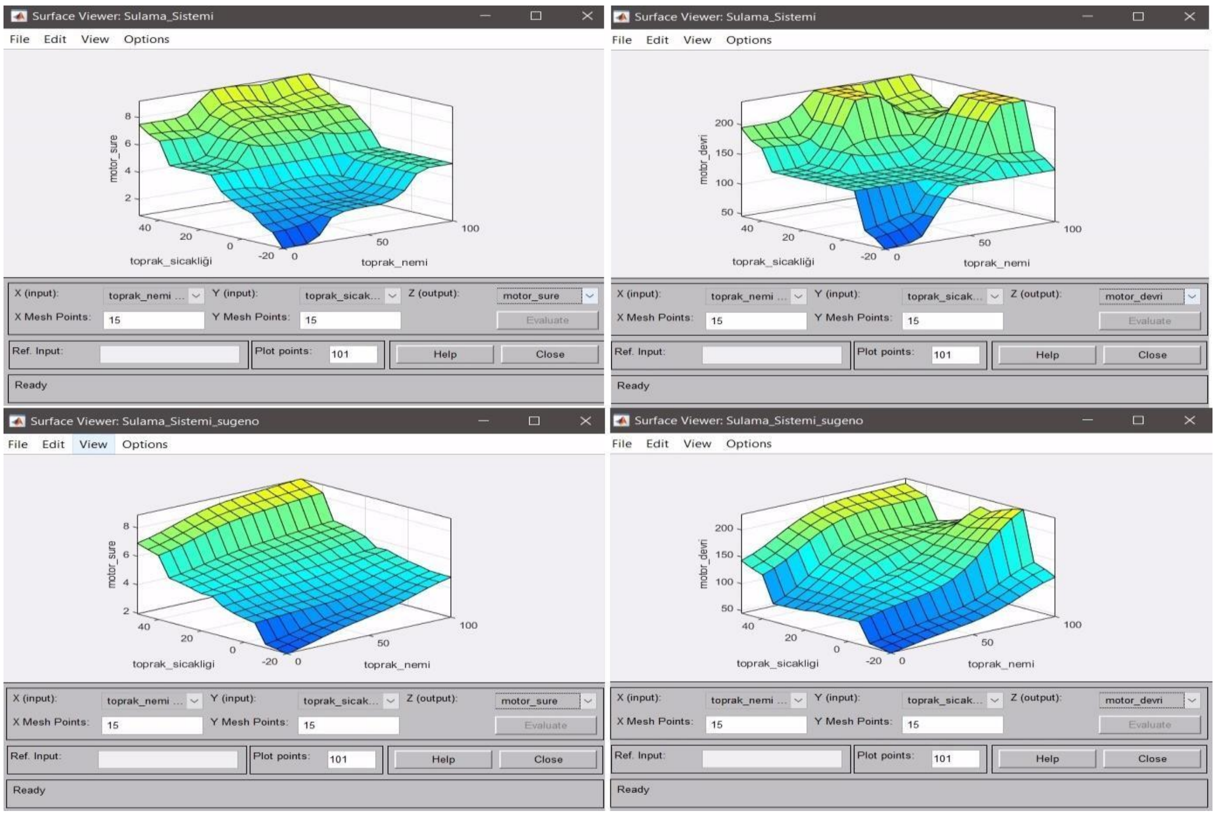
Using Gaussmf resulted in smoother output values. When trimf and trapmf were used, the output values were sharper, while Gaussmf provided a smoother output curve.

With Gaussmf, the effect of input values on the output had a more gradual transition. Trimf and trapmf produced more distinct transitions.

Gaussmf offered a higher level of precision, particularly in cases without sharp transitions between input values. However, due to its more complex mathematical structure, Gaussmf may increase computational cost compared to the simpler calculations of trimf and trapmf.

## 4) Comparison of Mamdani and Sugeno

In fuzzy logic, Mamdani and Sugeno are two different inference methods. Mamdani inference produces a fuzzy output for each rule, and these outputs are combined to form the final fuzzy output. Sugeno inference, on the other hand, uses weighted rule results with a specific formula to produce a crisp output.



X (input): toprak_nemi ... Y (input): toprak_sicak... Z (output): motor_sure
X Mesh Points: 15 Y Mesh Points: 15
Ref. Input: Plot points: 101
Ready

X (input): toprak_nemi ... Y (input): toprak_sicak... Z (output): motor_devri
X Mesh Points: 15 Y Mesh Points: 15
Ref. Input: Plot points: 101
Ready

X (input): toprak_nemi ... Y (input): toprak_sicak... Z (output): motor_sure
X Mesh Points: 15 Y Mesh Points: 15
Ref. Input: Plot points: 101
Ready

X (input): toprak_nemi ... Y (input): toprak_sicak... Z (output): motor_devri
X Mesh Points: 15 Y Mesh Points: 15
Ref. Input: Plot points: 101
Ready

Şekil 10_ Mamdani (üst) - Sugeno (alt) Karşılaştırması

If this code had been developed using Sugeno, we would have obtained a model where the outputs are crisp values. Specifically, the outputs would be directly expressed using a mathematical formula, making system behavior more straightforward and generally easier to interpret. In contrast, Mamdani logic expresses outputs as fuzzy sets, and combining these sets results in a more flexible but more complex model to interpret.

## 5) Arduino Code

```
#define ADC0 A0 // ADC0, analog 0 pinini temsil ediyor.
#define ADC1 A1 // ADC1, analog 1 pinini temsil ediyor.

int MOTOR_PIN_1 = 2; // L239D sürücüsü 1. pin (yön)
int MOTOR_PIN_2 = 3; // L239D sürücüsü 2. pin (yön)
int MOTOR_SPEED_PIN = 5; // L239D sürücüsü enable pin
float devir;
float sure;

void setup() {

  Serial.begin(9600);

  pinMode(MOTOR_PIN_1, OUTPUT);
  pinMode(MOTOR_PIN_2, OUTPUT);
  pinMode(MOTOR_SPEED_PIN, OUTPUT);
}
void loop() {

  Serial.print(analogRead(ADC0));
  Serial.print(analogRead(ADC1));
  delay(100);

  if (Serial.available() > 0) {

    Serial.readBytes((char*)&devir, sizeof(devir));

    Serial.readBytes((char*)&sure, sizeof(sure));

    analogWrite(MOTOR_SPEED_PIN, map(devir, 0, 100, 0, 255));
    digitalWrite(MOTOR_PIN_1, HIGH);
    digitalWrite(MOTOR_PIN_2, LOW);
    analogWrite(MOTOR_SPEED_PIN, devir);

    unsigned long baslangicZamani = millis();
    while (millis() - baslangicZamani < sure * 1000) {

    }

    digitalWrite(MOTOR_PIN_1, LOW);
    digitalWrite(MOTOR_PIN_2, LOW);

    delay(2000);
  }
}
```

## 6) MATLAB Fuzzy Logic Code

```
clear;clc;

b = newfis ('Sulama_Sistemi');
b = addvar(b, 'input', 'toprak_nemi', [0 100]);
b = addmf(b, 'input', 1, 'az', 'trimf', [-40 0 40]);
b = addmf(b, 'input', 1, 'orta', 'trimf', [20 50 80]);
b = addmf(b, 'input', 1, 'cok', 'trimf', [60 100 140]);

b = addvar(b, 'input', 'toprak_sicakliği', [-20 50]);
b = addmf(b, 'input', 2, 'cokdusuk', 'trapmf', [-40 -20 -10 -5]);
b = addmf(b, 'input', 2, 'dusuk', 'trimf', [-20 0 15]);
b = addmf(b, 'input', 2, 'orta', 'trimf', [-2 15 30]);
b = addmf(b, 'input', 2, 'sıcak', 'trimf', [15 30 50]);
b = addmf(b, 'input', 2, 'coksıcak', 'trapmf', [35 40 50 50]);

b = addvar(b, 'output', 'motor_sure', [0 10]);
b = addmf(b, 'output', 1, 'cokaz', 'trimf', [0 0 2.5]);
b = addmf(b, 'output', 1, 'az', 'trimf', [0 3.333 5.333]);
b = addmf(b, 'output', 1, 'orta', 'trimf', [3 5 7]);
b = addmf(b, 'output', 1, 'fazla', 'trimf', [5 7.5 10]);
b = addmf(b, 'output', 1, 'cokfazla', 'trimf', [7.5 10 16.67]);

b = addvar(b, 'output', 'motor_devri', [0 250]);
b = addmf(b, 'output', 2, 'yavasdevir', 'trapmf',   [0 0 65 115]);
b = addmf(b, 'output', 2, 'ortadevir', 'trimf',   [82 135 180]);
b = addmf(b, 'output', 2, 'hızlıdevir', 'trimf',   [145 195 245]);
b = addmf(b, 'output', 2, 'cokhızlıdevir', 'trapmf',  [215 240 250 280]);

rulkey = [
    1 1 1 1 1 1;
    2 1 2 2 1 1;
    3 1 3 2 1 1;
    1 2 2 2 1 1;
    2 2 2 2 1 1;
    3 2 3 4 1 1;
    1 3 3 2 1 1;
    2 3 3 2 1 1;
    3 3 3 4 1 1;
    1 4 3 2 1 1;
    2 4 4 4 1 1;
    3 4 4 3 1 1;
    1 5 4 3 1 1;
    2 5 5 4 1 1;
    3 5 5 4 1 1
];
b = addrule(b, rulkey); writefis(b, 'Sulama_Sistemi.fis');
```

## 7) MATLAB Code for Communication with Arduino

```
clc;
clear;
close all;
delete(instrfindall);
g = serial('COM3', 'BaudRate', 9600);
fopen(g);
b = readfis('D:\Allahin Adami Program File\MathLab\bin\sulama_sistemi.fis'); %
Oluşturulan bulanık mantık kontrol sistemi cagırıldı.
x = 1;
while x
    sicaklik=fscanf(g,'%s');
    s=strsplit(sicaklik,'.');
    sicaklik=str2double(s(2));
    sicaklik=sicaklik*5/1024; %gelen değer 0-1024 aralıgından 0-5 aralığına
çekildi.
    nem=str2double(s(1));
    nem=nem*5/1024;%gelen değer 0-1024 aralıgından 0-5 aralıgına çekildi.
    sicaklik(x)=14*sicaklik-20;%pot'tan gelen değer (-20,50) arasına uyarlanı.
    nem(x)=20*nem;%sensörden gelen değer (0,100) arasına uyarlandı.
    y = evalfis(b, [nem(x) sicaklik(x)]); % belirtilen girişlere (nem(x) ve
sicaklik(x)) dayalı olarak bulanık mantık modelinin çıkışları hesaplandı.
    sure(x)=y(2);
    devir(x)=y(1);
    fprintf('Sıcaklık=%f Nem=%f Süre= %f Devir=%f\n',sicaklik(x),nem(x),y(1),
y(2));
    fwrite(g, [sure(x)], 'float');
    fwrite(g, [devir(x)], 'float');
    x=x+1;
end
```

## 8) Conclusion

In this study, we aimed to estimate the speed and operating time of an irrigation pump motor for agricultural purposes using real-time air temperature and soil moisture data, achieved through fuzzy logic control and measurements using Arduino and MATLAB.

As a result, this study presents a model for designing a fuzzy logic-controlled irrigation system. This model can help make agricultural irrigation systems more efficient in water usage and ensure that the water requirements of plants are determined more accurately.