CSE 4110 : Artificial Intelligence Laboratory

Name : Tanzir Mannan Turzo
Roll : 2007048

Fuzzy Logic–Based Water Tank Monitoring System

Problem Statement

Design a fuzzy logic–based control system for an automated water tank that controls two main functions:

- **Motor:** Starts when the water level is low and stops when it's high.
- **Heater:** Turns on when the water is cold and gradually turns off as the temperature increases.

The system fuzzifies two key inputs:

1. Water Level (0–100%)
2. Temperature (°C)

Using fuzzy membership functions, the system infers appropriate motor and heater intensities through fuzzy rules and produces crisp output values via defuzzification.

Algorithm Steps

Step 1: Fuzzification

Convert crisp inputs (water level and temperature) into fuzzy membership values.

- Water Level — {Low, Medium, High}
- Temperature — {Cold, Warm, Hot}

Step 2: Rule Evaluation

Apply fuzzy rules of the form:

- If (Water Level is Low) → Motor = ON

- If (Water Level is Medium) → Motor = MEDIUM
- If (Water Level is High) → Motor = OFF
- If (Temperature is Cold) → Heater = ON
- If (Temperature is Warm) → Heater = MEDIUM
- If (Temperature is Hot) → Heater = OFF

Step 3: Aggregation

Combine all rule outputs to get final fuzzy values for Motor and Heater.

Step 4: Defuzzification

Convert fuzzy outputs to crisp control percentages (0–100%) using the centroid method.

Step 5: Visualization

Plot:

1. Membership functions for water level and temperature.
2. Final control percentages for Motor and Heater.

Code Implementation

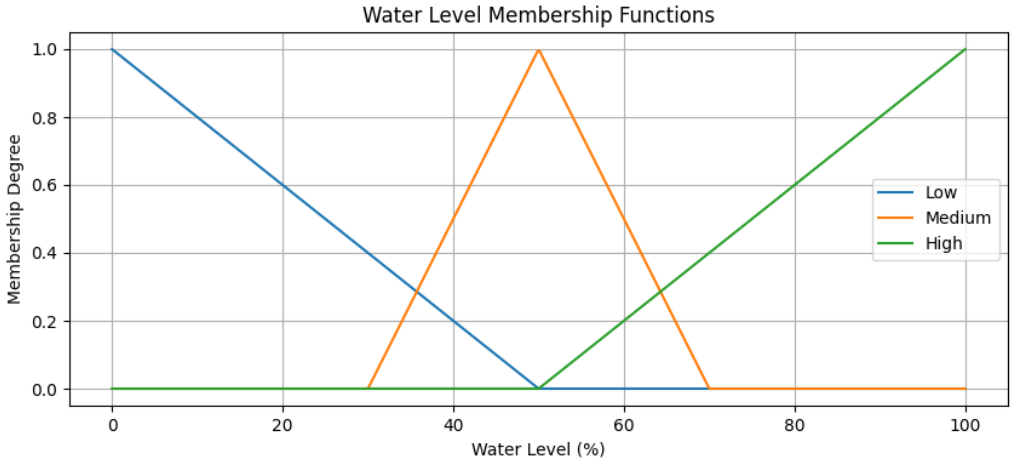
Listing 1: Fuzzy Logic-Based Water Tank System

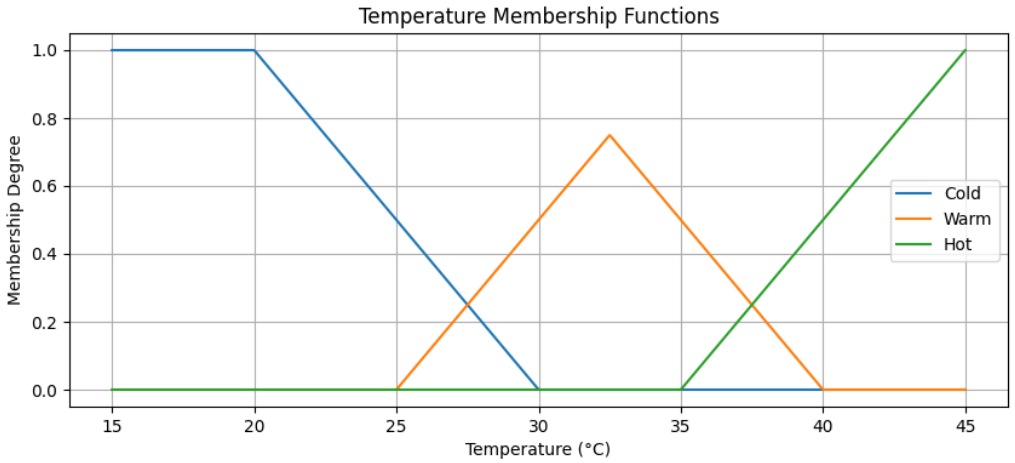
```
import numpy as np

def fuzzify_water_level(x):
    low = max(0, min(1, (50 - x) / 50))
    medium = max(0, min((x - 30) / 20, (70 - x) / 20))
    high = max(0, min(1, (x - 50) / 50))
    return {"low": round(low, 2), "medium": round(medium, 2), "high":
        round(high, 2)}

def fuzzify_temperature(x):
    cold = max(0, min(1, (30 - x) / 10))
    warm = max(0, min((x - 25) / 10, (40 - x) / 10))
    hot = max(0, min(1, (x - 35) / 10))
    return {"cold": round(cold, 2), "warm": round(warm, 2), "hot":
        round(hot, 2)}

def apply_rules(water_level, temperature):
    motor = {"off": water_level["high"],
             "medium": water_level["medium"],
             "on": water_level["low"]}
    heater = {"off": temperature["hot"],
              "medium": temperature["warm"],
              "on": temperature["cold"]}
    return motor, heater

def defuzzify(rules):
    values = {"off": 0, "medium": 50, "on": 100}
    numerator = sum(rules[k] * values[k] for k in rules)
    denominator = sum(rules.values())
    return numerator / denominator if denominator != 0 else 0

def water_tank_system(water_level_input, temperature_input):
    wl_mf = fuzzify_water_level(water_level_input)
    temp_mf = fuzzify_temperature(temperature_input)
    motor_rules, heater_rules = apply_rules(wl_mf, temp_mf)
    motor_val = defuzzify(motor_rules)
    heater_val = defuzzify(heater_rules)
    print(f"Water Level={water_level_input}%, Temperature={
        temperature_input} C ")
    print(f"Motor Control = {motor_val:.2f}% | Heater Control = {
        heater_val:.2f}%")
    return motor_val, heater_val

# Example run
water_tank_system(30, 28)
```

Sample Output

Water Level=30%, Temperature=28°C
Motor Control = 83.33% | Heater Control = 75.00%

Water Level MFs: {'low': 0.4, 'medium': 0.5, 'high': 0.0}
Temperature MFs: {'cold': 0.2, 'warm': 0.8, 'hot': 0.0}
Motor Rule Strengths: {'off': 0.0, 'medium': 0.5, 'on': 0.4}
Heater Rule Strengths: {'off': 0.0, 'medium': 0.8, 'on': 0.2}
Membership function





Conclusion

The fuzzy water tank system effectively controls motor and heater intensity by interpreting continuous sensor data. It mimics human decision-making, ensuring the motor starts only when the tank is low and the heater maintains the water at an optimal temperature range.