# Unit 5: Fuzzy Control Systems

## Learning Objectives

After studying this unit, students should be able to:

- Explain what a fuzzy control system is.
- Understand the control system design problem.
- Interpret a fuzzy control or decision surface.
- Identify assumptions used in fuzzy controller design.
- Design simple fuzzy logic controllers.
- Apply fuzzy control to real-world problems such as aircraft landing and robot navigation.
- Understand fuzzy optimization and fuzzy linear regression.
- Solve basic problems in fuzzy optimization and fuzzy regression.

---

# 5.1 Control System Design Problem

## 5.1.1 What is a Control System?

A **control system** is a system that manages, commands, directs, or regulates the behavior of another system.

Examples:

- Thermostat controlling room temperature
- Cruise control in a car
- Robot moving toward a target
- Washing machine choosing wash time
- Aircraft autopilot system

A control system usually has:

1. **Input**
2. **Controller**
3. **Plant/System**
4. **Output**
5. **Feedback**

---

## 5.1.2 Basic Control System Diagram

> Desired Output
> |
> v

```
+---------------+
|  Controller   |
+---------------+
       |
  Control Signal
       |
       v
+---------------+
|    Plant      |
+---------------+
       |
    Output
       |
       v
+---------------+
|  Feedback     |
+---------------+
       |
       +---------- back to controller
```

---

## 5.1.3 Classical Control System

In classical control, the controller is often designed using mathematical models.

For example, in a PID controller:

$$u(t) = K_p e(t) + K_i \int e(t)\,dt + K_d \frac{de(t)}{dt}$$

Where:

- $u(t)$ = control signal
- $e(t)$ = error
- $K_p$ = proportional gain
- $K_i$ = integral gain
- $K_d$ = derivative gain

The error is:

$$e(t) = r(t) - y(t)$$

Where:

- $r(t)$ = reference or desired output
- $y(t)$ = actual output

---

## 5.1.4 Problem with Classical Control

Classical control works well when:

- The system has an accurate mathematical model.

- The system is linear or nearly linear.
- The environment is predictable.

But many real systems are:

- Nonlinear
- Complex
- Uncertain
- Difficult to model mathematically

Examples:

- Human driving
- Aircraft landing
- Robot navigation
- Washing machine control
- Medical decision systems

This is where **fuzzy control** is useful.

---

## 5.1.5 What is a Fuzzy Control System?

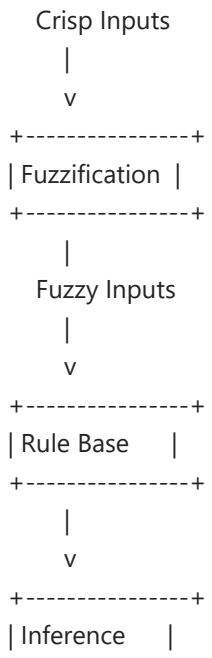
A **fuzzy control system** is a control system based on fuzzy logic instead of only precise mathematical equations.

It uses human-like rules such as:

> IF temperature is high THEN fan speed is fast.
> IF error is small THEN control action is small.
> IF obstacle is near THEN turn sharply.

---

## 5.1.6 General Structure of a Fuzzy Logic Controller



```
| Mechanism      |
+----------------+
        |
    Fuzzy Output
        |
        v
+----------------+
| Defuzzification|
+----------------+
        |
        v
    Crisp Output
```

---

## 5.1.7 Main Parts of a Fuzzy Controller

## 1. Fuzzification

Converts crisp numerical inputs into fuzzy values.

Example:

If temperature = 32°C, it may belong to:

- Warm with membership 0.4
- Hot with membership 0.7

---

## 2. Rule Base

Contains expert knowledge in the form of IF-THEN rules.

Example:

> IF temperature is hot THEN fan speed is high.

---

## 3. Inference Engine

Applies fuzzy rules and decides the fuzzy output.

Common inference methods:

- Mamdani inference
- Sugeno inference
- Tsukamoto inference

---

## 4. Defuzzification

Converts fuzzy output into a crisp control signal.

Common methods:

- Centroid method
- Mean of maxima
- Weighted average method

Centroid formula:

$$z^* = \frac{\int z\mu(z)\,dz}{\int \mu(z)\,dz}$$

For discrete values:

$$z^* = \frac{\sum z_i\mu(z_i)}{\sum \mu(z_i)}$$

---

## 5.1.8 Worked Example: Fan Speed Controller

## Problem

Design a simple fuzzy controller for fan speed based on temperature.

## Input

Temperature:

- Low
- Medium
- High

## Output

Fan speed:

- Slow
- Medium
- Fast

## Rule Base

| Rule | Temperature | Fan Speed |
|---|---|---|
| 1 | Low | Slow |
| 2 | Medium | Medium |
| 3 | High | Fast |

## Example

Suppose temperature = 30°C.

Membership values:

> Low = 0.0
> Medium = 0.4
> High = 0.6

Output membership:

> Slow = 0.0
> Medium = 0.4
> Fast = 0.6

Assume output values:

> Slow = 30 rpm
> Medium = 60 rpm
> Fast = 90 rpm

Using weighted average:

$$FanSpeed = \frac{0.4(60) + 0.6(90)}{0.4 + 0.6}$$

$$FanSpeed = \frac{24 + 54}{1} = 78 \text{ rpm}$$

So, the fan speed is **78 rpm**.

---

## 5.1.9 Python Code: Simple Fuzzy Fan Controller

```python
def fuzzy_fan_controller(temp):
    # Simple membership functions
    if temp <= 20:
        low = 1
    elif 20 < temp < 30:
        low = (30 - temp) / 10
    else:
        low = 0

    if 20 < temp < 30:
        medium = (temp - 20) / 10
    elif 30 <= temp < 40:
        medium = (40 - temp) / 10
    else:
        medium = 0

    if temp <= 30:
        high = 0
    elif 30 < temp < 40:
        high = (temp - 30) / 10
    else:
        high = 1

    # Output values
```

```
    slow = 30
    medium_speed = 60
    fast = 90

    # Weighted average defuzzification
    numerator = low * slow + medium * medium_speed + high * fast
    denominator = low + medium + high

    if denominator == 0:
        return 0

    return numerator / denominator


temperature = 35
speed = fuzzy_fan_controller(temperature)
print("Temperature:", temperature)
print("Fan Speed:", speed)
```

---

## Quick Revision Points

- A control system controls the behavior of another system.
- Classical controllers need accurate mathematical models.
- Fuzzy controllers use linguistic rules.
- A fuzzy controller has fuzzification, rule base, inference, and defuzzification.
- Fuzzy control is useful for nonlinear and uncertain systems.

---

## Questions and Answers

### Q1. What is a fuzzy control system?

A fuzzy control system is a control system that uses fuzzy logic and IF-THEN rules to make control decisions.

### Q2. Why are fuzzy controllers useful?

They are useful when the system is complex, nonlinear, uncertain, or difficult to model mathematically.

### Q3. What is defuzzification?

Defuzzification is the process of converting fuzzy output into a crisp numerical value.

### Q4. Give one example of a fuzzy control rule.

> IF temperature is high THEN fan speed is fast.

---

# 5.2 Control Decision Surface

## 5.2.1 Meaning of Control Surface

A **control surface**, also called a **decision surface**, shows the relationship between:

- Input variables
- Output control action

It is a graphical representation of how the fuzzy controller behaves.

For a controller with two inputs and one output:

> Input 1: Error
> Input 2: Change in Error
> Output: Control Signal

The control surface shows how output changes for different combinations of inputs.

---

## 5.2.2 Example

Suppose a fuzzy controller controls motor speed using:

- Error $e$
- Change in error $\Delta e$

The output is control signal $u$.

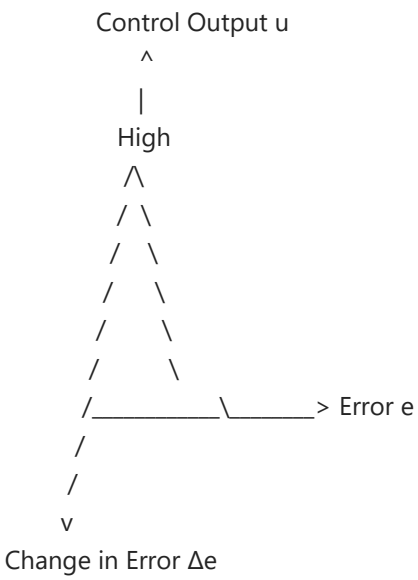
So:

$$u = f(e, \Delta e)$$

The control surface represents this function.

---

## 5.2.3 Control Surface Diagram




In actual software, this is plotted as a 3D surface.

---

## 5.2.4 Why Control Surface is Important

The control surface helps us understand:

- Whether the fuzzy controller behaves smoothly
- Whether rules conflict with each other
- Whether output changes gradually or suddenly
- Whether the system is stable
- Whether control action is reasonable

---

## 5.2.5 Smooth Control Surface

A good fuzzy controller usually has a smooth control surface.

Smooth surface means:

- Small input changes cause small output changes.
- The controller does not behave unpredictably.
- The system is easier to control.

---

## 5.2.6 Worked Example

Consider two inputs:

> Error: Negative, Zero, Positive
> Change in Error: Negative, Zero, Positive

Output:

> Control: Decrease, No Change, Increase

Rule table:

| Error / Change Error | Negative | Zero | Positive |
|---|---|---|---|
| Negative | Decrease | Decrease | No Change |
| Zero | Decrease | No Change | Increase |
| Positive | No Change | Increase | Increase |

If:

> Error = Positive
> Change in Error = Zero

Then:

> Control = Increase

---

## 5.2.7 Python Code to Plot a Simple Control Surface

```python
import numpy as np
import matplotlib.pyplot as plt

def control_output(error, derror):
    # Simple fuzzy-inspired rule approximation
    return 0.7 * error + 0.3 * derror

error_values = np.linspace(-10, 10, 50)
derror_values = np.linspace(-10, 10, 50)

E, DE = np.meshgrid(error_values, derror_values)
U = control_output(E, DE)

fig = plt.figure()
ax = fig.add_subplot(111, projection='3d')
ax.plot_surface(E, DE, U, cmap='viridis')

ax.set_xlabel('Error')
ax.set_ylabel('Change in Error')
ax.set_zlabel('Control Output')
ax.set_title('Simple Control Surface')

plt.show()
```

---

## Quick Revision Points

- A control surface shows input-output behavior of a fuzzy controller.
- For two inputs and one output, it is usually a 3D surface.
- A smooth surface is generally desirable.
- Control surfaces help analyze rule behavior.

---

## Questions and Answers

## Q1. What is a control surface?

A control surface is a graphical representation of the relationship between fuzzy controller inputs and output.

## Q2. Why should a control surface be smooth?

A smooth surface ensures that small changes in input do not produce sudden large changes in output.

## Q3. What are common inputs for a fuzzy control surface?

Common inputs are error and change in error.

---

# 5.3 Assumptions in a Fuzzy Control System Design

## 5.3.1 Why Assumptions Are Needed

When designing fuzzy controllers, we often make assumptions because:

- Exact mathematical model may not be available.
- Human expert knowledge may be incomplete.
- Sensors may be noisy.
- Environmental conditions may change.

Assumptions simplify the design process.

---

## 5.3.2 Common Assumptions

## Assumption 1: Expert Knowledge is Available

Fuzzy control often depends on human expertise.

Example:

An expert may say:

> IF speed is high AND obstacle is near THEN brake strongly.

---

## Assumption 2: Inputs and Outputs Can Be Described Linguistically

Variables are assumed to be expressible using words.

Example:

Temperature:

> Low, Medium, High

Speed:

> Slow, Normal, Fast

Distance:

> Near, Medium, Far

---

## Assumption 3: Membership Functions Can Represent Uncertainty

It is assumed that membership functions can describe vague concepts.

Example:

The term “hot” does not have one exact boundary.

> Temperature = 30°C may be partially warm and partially hot.

## Assumption 4: Rule Base Adequately Covers the Input Space

The rule base should cover possible input conditions.

Bad design:

> Only rules for low temperature are given.

Good design:

> Rules for low, medium, and high temperature are given.

---

## Assumption 5: Defuzzification Gives Useful Crisp Output

The final crisp output should be meaningful for the system.

Example:

A fan speed of 78 rpm should be physically possible.

---

## Assumption 6: Sensor Data is Reliable Enough

Fuzzy systems can handle uncertainty, but sensor data should not be completely wrong.

---

## Assumption 7: System Response is Acceptable

The system is assumed to respond properly to the control signal.

Example:

If the controller increases motor voltage, the motor speed should increase.

---

## 5.3.3 Diagram of Fuzzy Control Assumptions

> Expert Knowledge
>  |
>  v
> Linguistic Variables
>  |
>  v
> Membership Functions
>  |
>  v
> Rule Base Coverage
>  |
>  v
> Inference and Defuzzification
>  |

> v
> Practical Control Output

---

## Worked Example

## Problem

Design a fuzzy controller for room temperature.

## Assumptions

1. Temperature sensor gives reliable readings.
2. Temperature can be described as cold, comfortable, and hot.
3. Fan speed can be described as slow, medium, and fast.
4. Expert rules are available.
5. Fan speed output is physically possible.

## Example Rule

> IF temperature is hot THEN fan speed is fast.

---

## Quick Revision Points

- Assumptions simplify fuzzy controller design.
- Fuzzy controllers assume linguistic variables can represent real conditions.
- Membership functions are used to model vague ideas.
- Rule base coverage is important.
- Defuzzified output should be practical.

---

## Questions and Answers

## Q1. Why are assumptions important in fuzzy control design?

They simplify system design and make fuzzy control practical when exact models are unavailable.

## Q2. What happens if the rule base does not cover all input conditions?

The controller may produce weak, incorrect, or no output for some situations.

## Q3. Give one assumption in fuzzy controller design.

One assumption is that expert knowledge is available in the form of IF-THEN rules.

---

# 5.4 Simple Fuzzy Logic Controllers

## 5.4.1 What is a Simple Fuzzy Logic Controller?

A simple fuzzy logic controller uses a small number of inputs, outputs, membership functions, and rules.

Most common simple fuzzy controllers use:

- One input and one output
- Two inputs and one output

---

## 5.4.2 One-Input One-Output Controller

Example:

> Input: Temperature
> Output: Fan Speed

Rules:

> IF temperature is cold THEN fan speed is slow.
> IF temperature is warm THEN fan speed is medium.
> IF temperature is hot THEN fan speed is fast.

---

## 5.4.3 Two-Input One-Output Controller

Example:

> Input 1: Error
> Input 2: Change in Error
> Output: Control Signal

Rules:

> IF error is positive AND change in error is positive THEN control is high.
> IF error is zero AND change in error is zero THEN control is zero.
> IF error is negative AND change in error is negative THEN control is low.

---

## 5.4.4 Error and Change in Error

In many fuzzy controllers, the inputs are:

## Error

$$e(k) = r(k) - y(k)$$

## Change in Error

$$\Delta e(k) = e(k) - e(k-1)$$

Where:

- $r(k)$ = desired value
- $y(k)$ = actual value
- $e(k)$ = present error
- $e(k-1)$ = previous error

---

## 5.4.5 Common Linguistic Values

For error:

> NB = Negative Big
> NS = Negative Small
> ZE = Zero
> PS = Positive Small
> PB = Positive Big

For output:

> NL = Negative Large
> NM = Negative Medium
> ZE = Zero
> PM = Positive Medium
> PL = Positive Large

---

## 5.4.6 Rule Table Example

| Error / Change Error | NB | NS | ZE | PS | PB |
|---|---|---|---|---|---|
| NB | NL | NL | NM | NM | ZE |
| NS | NL | NM | NM | ZE | PM |
| ZE | NM | NM | ZE | PM | PM |
| PS | NM | ZE | PM | PM | PL |
| PB | ZE | PM | PM | PL | PL |

---

## 5.4.7 Steps to Design a Simple Fuzzy Logic Controller

## Step 1: Choose Inputs and Outputs

Example:

> Inputs: Error, Change in Error
> Output: Control Signal

## Step 2: Define Linguistic Variables

Example:

> Error: Negative, Zero, Positive
> Output: Decrease, No Change, Increase

## Step 3: Define Membership Functions

Common shapes:

- Triangular
- Trapezoidal
- Gaussian

Triangular membership function:

$$\mu_A(x) = \begin{cases} 0, & x < a \\ \frac{x-a}{b-a}, & a < x \leq b \\ \frac{c-x}{c-b}, & b < x < c \\ 0, & x > c \end{cases}$$

## Step 4: Create Rule Base

Example:

> IF error is positive THEN control is increase.

## Step 5: Choose Inference Method

Most common:

> Mamdani inference

## Step 6: Defuzzify Output

Common method:

> Centroid method

---

## Worked Example: Water Tank Level Control

## Problem

Maintain water level in a tank.

## Inputs

- Error = desired level - actual level
- Change in error

## Output

- Valve opening

## Linguistic Terms

Error:

> Low, Zero, High

Change in error:

> Decreasing, Stable, Increasing

Valve:

> Close, No Change, Open

---

## Rules

| Rule | Condition | Action |
|---|---|---|
| 1 | IF level is low | Open valve |
| 2 | IF level is correct | No change |
| 3 | IF level is high | Close valve |

## Example

Desired level = 100 cm
Actual level = 80 cm

$$e = 100 - 80 = 20$$

The tank level is low, so valve should open.

---

## Python Code: Simple Water Tank Controller

```python
def tank_controller(desired_level, actual_level):
    error = desired_level - actual_level

    if error > 10:
        action = "Open valve"
    elif error < -10:
        action = "Close valve"
    else:
        action = "No change"

    return action


desired = 100
actual = 80

print("Control Action:", tank_controller(desired, actual))
```

---

## Quick Revision Points

- Simple fuzzy controllers usually use one or two inputs.
- Error and change in error are common inputs.
- Rules are written in IF-THEN format.
- Membership functions map numerical values to fuzzy values.
- Defuzzification gives a crisp output.

---

## Questions and Answers

## Q1. What are the two common inputs in fuzzy controllers?

Error and change in error.

## Q2. What is a rule base?

A rule base is a collection of fuzzy IF-THEN rules.

## Q3. What is the formula for error?

$$e(k) = r(k) - y(k)$$

---

# 5.5 Examples of Fuzzy Control System Design

## 5.5.1 Example 1: Washing Machine Controller

## Problem

Control washing time based on:

- Dirtiness of clothes
- Load size

## Inputs

Dirtiness:

> Low, Medium, High

Load size:

> Small, Medium, Large

## Output

Wash time:

> Short, Normal, Long

---

## Rule Table

| Dirtiness / Load | Small | Medium | Large |
|---|---|---|---|
| Low | Short | Short | Normal |
| Medium | Short | Normal | Long |
| High | Normal | Long | Long |

## Example Rule

> IF dirtiness is high AND load is large THEN wash time is long.

---

## 5.5.2 Worked Example

Suppose:

> Dirtiness = High with membership 0.8
> Load = Large with membership 0.7

For AND operation, use minimum:

$$\alpha = \min(0.8, 0.7) = 0.7$$

So the rule fires with strength 0.7.

If long wash time = 60 minutes, then this rule strongly supports a long wash time.

---

## 5.5.3 Example 2: Air Conditioner Controller

## Inputs

- Room temperature
- Humidity

## Output

- Cooling power

## Rules

> IF temperature is high AND humidity is high THEN cooling is strong.
> IF temperature is medium AND humidity is medium THEN cooling is normal.
> IF temperature is low THEN cooling is weak.

---

## 5.5.4 Example 3: Car Braking System

## Inputs

- Distance to obstacle
- Speed of vehicle

## Output

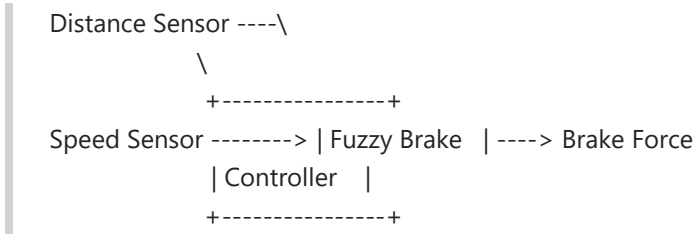
- Brake force

## Rules

> IF distance is near AND speed is high THEN brake force is strong.
> IF distance is far AND speed is low THEN brake force is weak.
> IF distance is medium AND speed is medium THEN brake force is medium.

---

## 5.5.5 Diagram: Fuzzy Braking System




---

## Python Code: Fuzzy Braking Controller

```python
def braking_controller(distance, speed):
    # distance in meters, speed in km/h

    if distance < 10 and speed > 60:
        brake = "Strong"
    elif distance < 20 and speed > 30:
        brake = "Medium"
    else:
        brake = "Weak"

    return brake


print(braking_controller(distance=8, speed=80))
print(braking_controller(distance=15, speed=50))
print(braking_controller(distance=40, speed=20))
```

---

## Quick Revision Points

- Fuzzy control can be used in washing machines, AC systems, cars, robots, and aircraft.
- Inputs and outputs are described using linguistic terms.
- Rule tables help organize fuzzy rules.
- AND operation often uses the minimum function.

## Questions and Answers

## Q1. Give two examples of fuzzy control applications.

Washing machine control and car braking control.

## Q2. In fuzzy logic, how is AND commonly calculated?

Using the minimum operator.

$$\mu_{A\cap B}(x) = \min(\mu_A(x), \mu_B(x))$$

## Q3. Write one fuzzy rule for a car braking system.

> IF distance is near AND speed is high THEN brake force is strong.

---

# 5.6 Aircraft Landing Control Problem

## 5.6.1 Introduction

Aircraft landing is a complex control problem.

During landing, the aircraft must control:

- Altitude
- Speed
- Descent rate
- Pitch angle
- Distance from runway
- Wind disturbances

A fuzzy controller can imitate the decision-making of an experienced pilot.

---

## 5.6.2 Why Fuzzy Logic is Useful for Aircraft Landing

Aircraft landing is difficult because:

- Exact mathematical model is complex.
- Wind and weather are uncertain.
- Pilot decisions are often linguistic.
- Smooth control is required.
- Safety is critical.

Example pilot rule:

> IF altitude is high AND descent rate is slow THEN decrease throttle slightly.

## 5.6.3 Inputs and Outputs

### Possible Inputs

| Input | Meaning |
|---|---|
| Altitude | Height of aircraft above runway |
| Vertical speed | Rate of descent |
| Horizontal distance | Distance from runway |
| Airspeed | Speed of aircraft |
| Pitch angle | Nose angle of aircraft |

### Possible Outputs

| Output | Meaning |
|---|---|
| Throttle command | Increase or decrease engine power |
| Elevator command | Change pitch angle |
| Flap adjustment | Adjust lift and drag |

## 5.6.4 Simplified Aircraft Landing Controller

Let us design a simple controller using two inputs:

> Altitude
> Descent Rate

Output:

> Throttle

## 5.6.5 Linguistic Variables

Altitude:

> Low, Medium, High

Descent Rate:

> Slow, Normal, Fast

Throttle:

> Decrease, Maintain, Increase

## 5.6.6 Rule Table

| Altitude / Descent Rate | Slow | Normal | Fast |
|---|---|---|---|
| Low | Decrease | Maintain | Increase |
| Medium | Decrease | Maintain | Increase |
| High | Decrease | Decrease | Maintain |

---

## 5.6.7 Interpretation of Rules

## Rule 1

> IF altitude is low AND descent rate is fast THEN increase throttle.

Meaning:

The aircraft is close to the ground and falling too quickly, so more engine power is needed.

## Rule 2

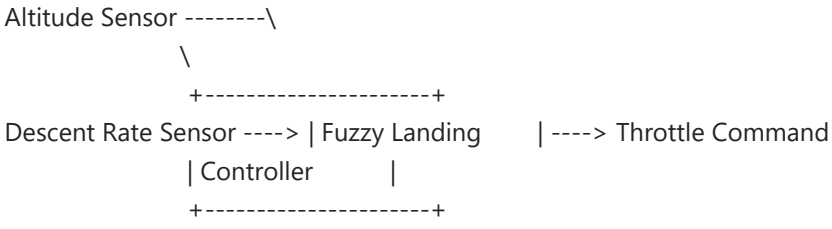
> IF altitude is high AND descent rate is slow THEN decrease throttle.

Meaning:

The aircraft is still high and descending slowly, so reduce power to descend faster.

---

## 5.6.8 Diagram




---

## 5.6.9 Worked Example

Suppose:

> Altitude = Low with membership 0.8
> Descent Rate = Fast with membership 0.6

Rule:

> IF altitude is low AND descent rate is fast THEN throttle is increase.

Firing strength:

$$\alpha = \min(0.8, 0.6) = 0.6$$

So the controller recommends increasing throttle with strength 0.6.

---

## 5.6.10 Python Code: Simplified Aircraft Landing Controller

```python
def landing_controller(altitude, descent_rate):
    # altitude in meters
    # descent_rate in meters/second

    if altitude < 100 and descent_rate > 5:
        throttle = "Increase throttle"
    elif altitude > 300 and descent_rate < 3:
        throttle = "Decrease throttle"
    else:
        throttle = "Maintain throttle"

    return throttle


print(landing_controller(altitude=80, descent_rate=7))
print(landing_controller(altitude=400, descent_rate=2))
print(landing_controller(altitude=200, descent_rate=4))
```

---

## 5.6.11 Safety Note

In real aircraft systems, fuzzy controllers must be:

- Carefully tested
- Verified mathematically
- Simulated under many conditions
- Used with backup systems
- Certified by aviation authorities

The example here is only for learning.

---

## Quick Revision Points

- Aircraft landing is a difficult nonlinear control problem.
- Fuzzy logic can model pilot-like reasoning.
- Inputs may include altitude, descent rate, speed, and distance.
- Outputs may include throttle and elevator commands.
- Safety verification is essential in real aircraft.

---

## Questions and Answers

## Q1. Why is fuzzy logic suitable for aircraft landing?

Because aircraft landing involves uncertainty, nonlinear behavior, and human-like decision-making.

## Q2. Write one fuzzy rule for aircraft landing.

> IF altitude is low AND descent rate is fast THEN increase throttle.

## Q3. What are two possible outputs of an aircraft landing controller?

Throttle command and elevator command.

---

# 5.7 Fuzzy Optimization

## 5.7.1 What is Optimization?

Optimization means finding the best solution from a set of possible solutions.

Examples:

- Minimize cost
- Maximize profit
- Minimize travel time
- Maximize efficiency
- Minimize energy consumption

A general optimization problem is:

$$\text{Minimize } f(x)$$

subject to:

$$g_i(x) < 0$$

---

## 5.7.2 What is Fuzzy Optimization?

**Fuzzy optimization** is optimization where:

- Goals may be vague.
- Constraints may be flexible.
- Parameters may be uncertain.

Instead of saying:

> Cost must be less than ₹10,000.

We may say:

> Cost should be approximately less than ₹10,000.

---

## 5.7.3 Difference Between Classical and Fuzzy Optimization

| Classical Optimization | Fuzzy Optimization |
|---|---|
| Exact constraints | Flexible constraints |
| Precise goals | Approximate goals |
| Sharp boundaries | Gradual membership |
| Feasible or infeasible | Degree of satisfaction |

---

## 5.7.4 Membership Function for Fuzzy Goal

Suppose the goal is:

> Cost should be low.

A membership function can be:

$$\mu_G(x) = \begin{cases} 1, & f(x) < f_0 \\ \frac{f_1 - f(x)}{f_1 - f_0}, & f_0 < f(x) < f_1 \\ 0, & f(x) > f_1 \end{cases}$$

Where:

- $f_0$ = fully satisfactory cost
- $f_1$ = unacceptable cost

---

## 5.7.5 Bellman-Zadeh Approach

In fuzzy optimization, the decision should satisfy both fuzzy goals and fuzzy constraints.

If:

- $\mu_G(x)$ = membership value of goal
- $\mu_C(x)$ = membership value of constraint

Then the fuzzy decision membership is:

$$\mu_D(x) = \min[\mu_G(x), \mu_C(x)]$$

The best solution is the one that maximizes:

$$\max_x \mu_D(x)$$

---

## 5.7.6 Worked Example

# Problem

Choose production quantity $x$.

Goal:

> Profit should be high.

Constraint:

> Labor hours should be reasonably low.

Suppose the following membership values are given:

| Quantity x | Goal Satisfaction | Constraint Satisfaction |
|---|---|---|
| 100 | 0.4 | 0.9 |
| 200 | 0.7 | 0.8 |
| 300 | 0.9 | 0.5 |
| 400 | 1.0 | 0.2 |

Using:

$$\mu_D(x) = \min[\mu_G(x), \mu_C(x)]$$

| Quantity x | Goal | Constraint | Decision |
|---|---|---|---|
| 100 | 0.4 | 0.9 | 0.4 |
| 200 | 0.7 | 0.8 | 0.7 |
| 300 | 0.9 | 0.5 | 0.5 |
| 400 | 1.0 | 0.2 | 0.2 |

Best decision:

> x = 200

because decision satisfaction is maximum at 0.7.

---

## Python Code: Fuzzy Optimization

```python
quantities = [100, 200, 300, 400]
goal = [0.4, 0.7, 0.9, 1.0]
constraint = [0.9, 0.8, 0.5, 0.2]

decision = []

for g, c in zip(goal, constraint):
    decision.append(min(g, c))

best_index = decision.index(max(decision))

print("Decision values:", decision)
print("Best quantity:", quantities[best_index])
print("Maximum satisfaction:", decision[best_index])
```

---

## Quick Revision Points

- Fuzzy optimization handles vague goals and flexible constraints.
- It uses degrees of satisfaction instead of true/false feasibility.
- Bellman-Zadeh decision uses the minimum of goal and constraint membership.
- Best solution maximizes the fuzzy decision membership.

---

## Questions and Answers

## Q1. What is fuzzy optimization?

Fuzzy optimization is optimization involving vague goals, flexible constraints, or uncertain parameters.

## Q2. What is the Bellman-Zadeh decision rule?

$$\mu_D(x) = \min[\mu_G(x), \mu_C(x)]$$

## Q3. How is the best fuzzy decision selected?

The best decision is the one with the maximum decision membership value.

---

# 5.8 Fuzzy Linear Regression

## 5.8.1 What is Linear Regression?

Linear regression is a method used to model the relationship between input and output variables.

For one input:

$$y = a + bx$$

Where:

- $y$ = output
- $x$ = input
- $a$ = intercept
- $b$ = slope

---

## 5.8.2 What is Fuzzy Linear Regression?

In **fuzzy linear regression**, some coefficients or outputs are fuzzy numbers.

This is useful when:

- Data is imprecise.
- Measurements are uncertain.

- Human judgment is involved.
- The relationship is approximate.

A fuzzy linear regression model can be written as:

$$\tilde{Y} = \tilde{A}_0 + \tilde{A}_1 x$$

Where:

- $\tilde{Y}$ = fuzzy output
- $\tilde{A}_0$ = fuzzy intercept
- $\tilde{A}_1$ = fuzzy slope

---

## 5.8.3 Fuzzy Number Representation

A triangular fuzzy number is represented as:

$$\tilde{A} = (a, l, r)$$

Where:

- $a$ = center value
- $l$ = left spread
- $r$ = right spread

For a symmetric triangular fuzzy number:

$$\tilde{A} = (a, s)$$

Where:

- $a$ = center
- $s$ = spread

---

## 5.8.4 Fuzzy Linear Model

A common simple fuzzy regression model is:

$$\tilde{Y}_i = A_0 + A_1 x_i$$

Where:

$$A_j = (a_j, c_j)$$

Then the predicted fuzzy output has:

## Center

$$\hat{y}_i = a_0 + a_1 x_i$$

## Spread

$$s_i = c_0 + c_1|x_i|$$

So the fuzzy prediction is:

$$\tilde{Y}_i = (\hat{y}_i, s_i)$$

---

## 5.8.5 Why Use Fuzzy Regression?

Classical regression assumes random error.

Fuzzy regression assumes uncertainty due to vagueness.

| Classical Regression | Fuzzy Regression |
|---|---|
| Random error | Fuzziness or vagueness |
| Crisp coefficients | Fuzzy coefficients |
| Gives one predicted value | Gives fuzzy interval |
| Statistical approach | Possibilistic approach |

---

## 5.8.6 Worked Example

Suppose the fuzzy linear model is:

$$\tilde{Y} = (2, 0.5) + (3, 0.2)x$$

Find fuzzy output when:

$$x = 4$$

## Center

$$\hat{y} = 2 + 3(4) = 14$$

## Spread

$$s = 0.5 + 0.2|4| = 0.5 + 0.8 = 1.3$$

So:

$$\tilde{Y} = (14, 1.3)$$

This means the predicted value is centered at 14 with uncertainty spread 1.3.

Approximate interval:

$$[14 - 1.3, 14 + 1.3] = [12.7, 15.3]$$

---

## Python Code: Fuzzy Linear Regression Prediction

```python
```

```
def fuzzy_linear_prediction(x, a0, c0, a1, c1):
    center = a0 + a1 * x
    spread = c0 + c1 * abs(x)

    lower = center - spread
    upper = center + spread

    return center, spread, lower, upper


x = 4
center, spread, lower, upper = fuzzy_linear_prediction(
    x=x,
    a0=2,
    c0=0.5,
    a1=3,
    c1=0.2
)

print("Center:", center)
print("Spread:", spread)
print("Interval:", lower, "to", upper)
```

---

## Quick Revision Points

- Fuzzy linear regression handles imprecise data.
- Coefficients can be fuzzy numbers.
- Output is a fuzzy value or interval.
- A fuzzy number may have a center and spread.
- Fuzzy regression is useful when uncertainty is vague, not random.

---

## Questions and Answers

## Q1. What is fuzzy linear regression?

Fuzzy linear regression is regression where coefficients or outputs are represented as fuzzy numbers.

## Q2. What is the standard linear regression equation?

$$y = a + bx$$

~

## Q3. If $Y = (2, 0.5) + (3, 0.2)x$ and $x = 4$, what is the fuzzy output?

The output is:

$$\tilde{Y} = (14, 1\ 3)$$

---

# 5.9 Problems on Fuzzy Optimization and Regression

## Problem 1: Fuzzy Optimization

A company wants to select a production level.

| Production Level | Profit Satisfaction | Resource Satisfaction |
|---|---|---|
| 50 | 0.3 | 1.0 |
| 100 | 0.6 | 0.8 |
| 150 | 0.9 | 0.5 |
| 200 | 1.0 | 0.3 |

Find the best production level using the Bellman-Zadeh method.

## Solution

Use:

$$\mu_D(x) = \min[\mu_G(x), \mu_C(x)]$$

| Production | Profit | Resource | Decision |
|---|---|---|---|
| 50 | 0.3 | 1.0 | 0.3 |
| 100 | 0.6 | 0.8 | 0.6 |
| 150 | 0.9 | 0.5 | 0.5 |
| 200 | 1.0 | 0.3 | 0.3 |

Maximum decision value = 0.6.

Therefore, best production level is:

> 100 units

---

## Problem 2: Fuzzy Optimization

Given:

| Option | Goal Membership | Constraint Membership |
|---|---|---|
| A | 0.5 | 0.9 |
| B | 0.7 | 0.6 |
| C | 0.8 | 0.8 |
| D | 0.9 | 0.4 |

Find the best option.

## Solution

| Option | Goal | Constraint | Decision |
|---|---|---|---|
| A | 0.5 | 0.9 | 0.5 |
| B | 0.7 | 0.6 | 0.6 |
| C | 0.8 | 0.8 | 0.8 |
| D | 0.9 | 0.4 | 0.4 |

Best option:

> C

because it has maximum decision satisfaction 0.8.

---

## Problem 3: Fuzzy Regression

Given fuzzy regression model:

$$\tilde{Y} = (5, 1) + (2, 0.5)x$$

Find fuzzy output for:

$$x = 6$$

## Solution

Center:

$$\hat{y} = 5 + 2(6) = 17$$

Spread:

$$s = 1 + 0.5|6| = 1 + 3 = 4$$

So:

$$\tilde{Y} = (17, 4)$$

Interval:

$$[17 - 4, 17 + 4] = [13, 21]$$

---

## Problem 4: Fuzzy Regression

Given:

$$\tilde{Y} = (10, 2) + (4, 1)x$$

Find fuzzy output when:

$$x = 3$$

## Solution

Center:

$$\hat{y} = 10 + 4(3) = 22$$

Spread:

$$s = 2 + 1|3| = 5$$

Fuzzy output:

$$\tilde{Y} = (22, 5)$$

Interval:

$$[17, 27]$$

---

## Problem 5: Combined Fuzzy Decision Problem

A student is selecting a project topic based on interest and difficulty.

| Topic | Interest Satisfaction | Difficulty Satisfaction |
|---|---|---|
| AI | 0.9 | 0.5 |
| IoT | 0.8 | 0.7 |
| Robotics | 1.0 | 0.4 |
| Data Science | 0.7 | 0.8 |

Find the best topic.

## Solution

Use minimum:

| Topic | Interest | Difficulty | Decision |
|---|---|---|---|
| AI | 0.9 | 0.5 | 0.5 |
| IoT | 0.8 | 0.7 | 0.7 |
| Robotics | 1.0 | 0.4 | 0.4 |
| Data Science | 0.7 | 0.8 | 0.7 |

Best topics:

> IoT and Data Science

Both have satisfaction 0.7.

---

## Quick Revision Points

- In fuzzy optimization, calculate the decision value using minimum membership.
- Choose the option with the highest decision membership.
- In fuzzy regression, calculate center and spread separately.

- A fuzzy output can be represented as an interval.
- Fuzzy regression is useful for uncertain or imprecise data.

---

## Questions and Answers

## Q1. What formula is used in Bellman-Zadeh fuzzy decision-making?

$$\mu_D(x) = \min[\mu_G(x), \mu_C(x)]$$

## Q2. If goal = 0.8 and constraint = 0.6, what is decision membership?

$$\mu_D = \min(0.8, 0.6) = 0.6$$

## Q3. If fuzzy output is $(20, 3)$, what is the approximate interval?

$$[20 - 3, 20 + 3] = [17, 23]$$

---

# 5.10 Case Study: Robot Navigation Using Fuzzy Logic

## 5.10.1 Introduction

Robot navigation means guiding a robot from one place to another while avoiding obstacles.

A robot may need to answer:

- Where is the target?
- Is there an obstacle nearby?
- Should I move forward?
- Should I turn left or right?
- How fast should I move?

Fuzzy logic is useful because robot navigation involves uncertainty.

For example:

> The obstacle is near.
> The target is slightly left.
> The robot is moving fast.

These are fuzzy concepts.

---

## 5.10.2 Robot Navigation Problem

## Goal

Move the robot toward a target while avoiding obstacles.

## Inputs

1. Distance to obstacle
2. Angle to target
3. Distance to target

## Outputs

1. Speed
2. Steering angle

---

## 5.10.3 Linguistic Variables

## Input 1: Obstacle Distance

> Near, Medium, Far

## Input 2: Target Angle

> Left, Straight, Right

## Input 3: Target Distance

> Near, Medium, Far

## Output 1: Speed

> Stop, Slow, Fast

## Output 2: Steering

> Turn Left, Go Straight, Turn Right

---

## 5.10.4 Fuzzy Rule Base

## Obstacle Avoidance Rules

> IF obstacle is near THEN speed is slow.
> IF obstacle is far THEN speed is fast.
> IF obstacle is near AND target is left THEN turn right.
> IF obstacle is near AND target is right THEN turn left.

## Target Following Rules

> IF target angle is left THEN steering is left.
> IF target angle is straight THEN steering is straight.
> IF target angle is right THEN steering is right.

## Stopping Rules

> IF target distance is near THEN speed is stop.

---

## 5.10.5 Rule Table for Steering

| Obstacle / Target Angle | Left | Straight | Right |
|---|---|---|---|
| Near | Turn Right | Turn Left or Right | Turn Left |
| Medium | Turn Left | Straight | Turn Right |
| Far | Turn Left | Straight | Turn Right |

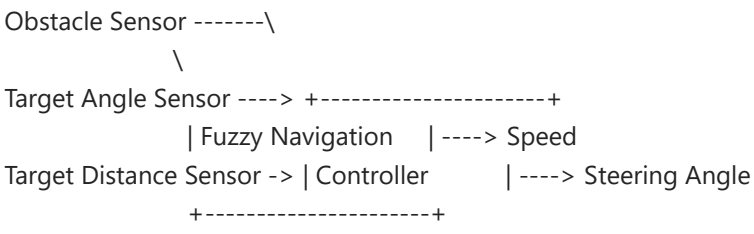
---

## 5.10.6 Rule Table for Speed

| Obstacle Distance / Target Distance | Near | Medium | Far |
|---|---|---|---|
| Near | Stop | Slow | Slow |
| Medium | Stop | Slow | Fast |
| Far | Stop | Fast | Fast |

---

## 5.10.7 Diagram: Fuzzy Robot Navigation System




---

## 5.10.8 Worked Example

Suppose:

> Obstacle distance = Near with membership 0.7
> Target angle = Left with membership 0.8
> Target distance = Far with membership 0.6

Rules:

> IF obstacle is near THEN speed is slow.
> IF obstacle is near AND target is left THEN steering is right.
> IF target distance is far THEN speed is fast.

## Rule Firing Strengths

Rule 1:

> Obstacle near = 0.7

So speed slow strength = 0.7.

Rule 2:

$$\alpha = \min(0.7, 0.8) = 0.7$$

So steering right strength = 0.7.

Rule 3:

> Target far = 0.6

So speed fast strength = 0.6.

The robot should:

> Move carefully and steer right.

---

## 5.10.9 Python Code: Simple Fuzzy Robot Navigation

```python
def robot_navigation(obstacle_distance, target_angle, target_distance):
    # obstacle_distance in meters
    # target_angle in degrees
    # target_distance in meters

    # Speed decision
    if target_distance < 0.5:
        speed = "Stop"
    elif obstacle_distance < 1.0:
        speed = "Slow"
    elif target_distance > 2.0:
        speed = "Fast"
    else:
        speed = "Slow"

    # Steering decision
    if obstacle_distance < 1.0:
        if target_angle < 0:
            steering = "Turn Right to avoid obstacle"
        elif target_angle > 0:
            steering = "Turn Left to avoid obstacle"
        else:
            steering = "Turn Left or Right"
    else:
        if target_angle < -10:
            steering = "Turn Left toward target"
        elif target_angle > 10:
            steering = "Turn Right toward target"
        else:
            steering = "Go Straight"

    return speed, steering
```

```
speed, steering = robot_navigation(
    obstacle_distance=0.8,
    target_angle=-20,
    target_distance=3.0
)

print("Speed:", speed)
print("Steering:", steering)
```

---

## 5.10.10 Improved Fuzzy Navigation Pseudocode

> Start
>
> Read obstacle distance
> Read target angle
> Read target distance
>
> Fuzzify obstacle distance as Near, Medium, Far
> Fuzzify target angle as Left, Straight, Right
> Fuzzify target distance as Near, Medium, Far
>
> Apply fuzzy rules
>
> Combine rule outputs
>
> Defuzzify speed
> Defuzzify steering angle
>
> Send commands to robot motors
>
> Repeat until target is reached
>
> Stop

---

## 5.10.11 Advantages of Fuzzy Robot Navigation

- Works with uncertain sensor values.
- Does not require exact mathematical model.
- Easy to write human-like rules.
- Can handle nonlinear behavior.
- Suitable for real-time decision-making.

---

## 5.10.12 Limitations

- Rule design can become complex.

- Performance depends on quality of membership functions.
- Difficult to guarantee optimal behavior.
- Testing is required in many environments.
- May need tuning for different robots.

---

## Quick Revision Points

- Fuzzy logic helps robots navigate in uncertain environments.
- Inputs include obstacle distance and target angle.
- Outputs include speed and steering.
- Rules are written in simple IF-THEN format.
- Robot navigation combines obstacle avoidance and target following.

---

## Questions and Answers

### Q1. Why is fuzzy logic useful in robot navigation?

Because robot sensors are uncertain, environments are dynamic, and decisions can be described using linguistic rules.

### Q2. Write one rule for obstacle avoidance.

> IF obstacle is near THEN speed is slow.

### Q3. What are two outputs of a fuzzy robot navigation system?

Speed and steering angle.

### Q4. What should the robot do if the target is near?

It should stop or slow down.

---

# Unit 5 Summary

## Important Concepts

| Topic | Key Idea |
|---|---|
| Control system design | Use fuzzy logic when exact models are difficult |
| Control surface | Shows input-output behavior |
| Assumptions | Expert rules, fuzzy variables, useful membership functions |
| Simple FLC | Uses IF-THEN rules for control |
| Design examples | Washing machine, AC, braking system |

| Topic | Key Idea |
|---|---|
| Aircraft landing | Fuzzy controller imitates pilot decisions |
| Fuzzy optimization | Handles vague goals and flexible constraints |
| Fuzzy regression | Handles imprecise data using fuzzy coefficients |
| Robot navigation | Uses fuzzy rules for movement and obstacle avoidance |

# Important Formulas

## Error

$$e(k) = r(k) - y(k)$$

## Change in Error

$$\Delta e(k) = e(k) - e(k-1)$$

## Centroid Defuzzification

$$z^* = \frac{\int z\mu(z)\,dz}{\int \mu(z)\,dz}$$

## Discrete Defuzzification

$$z^* = \frac{\sum z_i \mu(z_i)}{\sum \mu(z_i)}$$

## Fuzzy AND

$$\mu_{A\cap B}(x) = \min[\mu_A(x), \mu_B(x)]$$

## Fuzzy OR

$$\mu_{A\cup B}(x) = \max[\mu_A(x), \mu_B(x)]$$

## Bellman-Zadeh Decision

$$\mu_D(x) = \min[\mu_G(x), \mu_C(x)]$$

## Fuzzy Linear Regression

$$\tilde{Y} = \tilde{A}_0 + \tilde{A}_1 x$$

## Fuzzy Regression Center

$$\hat{y} = a_0 + a_1 x$$

## Fuzzy Regression Spread

$$s = c_0 + c_1|x|$$

# Exam-Oriented Short Questions

## 1. Define fuzzy control system.

A fuzzy control system is a control system that uses fuzzy logic, linguistic variables, and IF-THEN rules to generate control actions.

## 2. What are the main blocks of a fuzzy logic controller?

The main blocks are:

1. Fuzzification
2. Rule base
3. Inference engine
4. Defuzzification

## 3. What is a control surface?

A control surface is a graph showing how controller output changes with input variables.

## 4. What is fuzzy optimization?

Fuzzy optimization is optimization involving fuzzy goals, fuzzy constraints, or uncertain parameters.

## 5. What is fuzzy linear regression?

Fuzzy linear regression is a regression method where coefficients or outputs are represented as fuzzy numbers.

---

# Exam-Oriented Long Questions

## 1. Explain the structure of a fuzzy logic controller with a diagram.

A fuzzy logic controller contains fuzzification, rule base, inference engine, and defuzzification. Crisp inputs are converted into fuzzy values, rules are applied, fuzzy outputs are obtained, and finally crisp output is produced.

Diagram:

> Crisp Input
> |
> v
> Fuzzification
> |
> v
> Rule Base + Inference Engine
> |
> v
> Fuzzy Output
> |
> v
> Defuzzification

```
   |
   v
Crisp Control Output
```

---

## 2. Explain fuzzy optimization with an example.

Fuzzy optimization is used when goals or constraints are not exact. It uses membership functions to represent satisfaction levels. The Bellman-Zadeh method uses:

$$\mu_D(x) = \min[\mu_G(x), \mu_C(x)]$$

The best decision is the one with maximum decision membership.

---

## 3. Explain robot navigation using fuzzy logic.

A fuzzy robot navigation system uses sensor inputs such as obstacle distance, target angle, and target distance. These inputs are fuzzified into terms such as near, far, left, and right. Fuzzy rules decide speed and steering. Finally, defuzzification converts fuzzy decisions into motor commands.

Example rule:

> IF obstacle is near AND target is left THEN turn right.

---

# Final Quick Revision Checklist

Before the exam, make sure you can:

- Define fuzzy controller.
- Draw the fuzzy controller block diagram.
- Explain fuzzification and defuzzification.
- Write fuzzy IF-THEN rules.
- Construct a simple rule table.
- Explain control surface.
- Solve Bellman-Zadeh fuzzy optimization problems.
- Calculate fuzzy regression center and spread.
- Explain aircraft landing control using fuzzy logic.
- Explain robot navigation using fuzzy logic.