CHAPTER 17

Debugging & Testing

Learning Outcomes

- Introduction to Debugging
- Testing
- Life Cycle of Testing

In robotics, writing a program is just the beginning. The real challenge comes when the robot doesn't work as expected. Maybe the wheels don't move, the sensor gives the wrong reading, or the robot doesn't respond at all. This is where **debugging and testing** become important. **Debugging** means finding and fixing errors or mistakes in the code or circuit. **Testing** means checking if the robot is working correctly and performing the tasks it is supposed to do. By learning debugging and testing, you'll be able to build smarter and more reliable robots that work the way you want.

What is Debugging?

Debugging is the process of finding and fixing errors (called *bugs*) in your code or hardware setup. These bugs might stop your robot from working properly. Think of debugging like fixing a car engine that won't start—you have to figure out what's wrong and fix it.

What is Testing?

Testing means checking if your program works correctly in real conditions. It also helps make sure your robot does the right thing **every time**, and not just once by luck.

Types of Errors in Robotics

Type of Error	What It Means	Example
Syntax Error	The code is written incorrectly	Missing ; or incorrect spelling
Logic Error	The code runs but does the wrong thing	Turning right instead of left
Hardware Error	The robot's parts are not connected right	Wrong wiring to motors or sensors
Runtime Error	The program crashes during execution	Division by zero or undefined variable

How to Debug Your Program

1. **Check your code line by line**
   Look for missing semicolons, wrong variable names, or incorrect logic.

2. **Use the Serial Monitor (Arduino)**
   Print messages like "Sensor value = 450" to see what your robot is reading.
3. **Test in small steps**
   Don't write the whole program at once. Test each part separately (like just turning on the motor).
4. **Check wiring and connections**
   Make sure all cables, pins, and sensors are correctly placed.
5. **Use LEDs or Buzzers**
   Use them to check if signals are being sent properly.

How to Test Your Robot

Step	What to Do
Component Testing	Test each part (sensor, motor, LED) on its own first
Unit Testing	Test small parts of your code separately
Full System Testing	Run the full program on the robot and observe behavior
Edge Case Testing	Try unusual or extreme conditions (e.g., no light, wall very close)

Tools Used in Debugging and Testing

- **Serial Monitor** (in Arduino IDE)
  See what values the robot is reading or sending.
- **Multimeter**
  To check voltage, current, and connections in circuits.
- **LEDs or Debugging Lights**
  Use as indicators to show whether parts of the program are running.

Tips for Efficient Debugging

- Always keep your code clean and well-commented
- Try changing only one thing at a time and testing again
- If you get stuck, take a break or ask someone for help
- Save working versions of code before making big changes

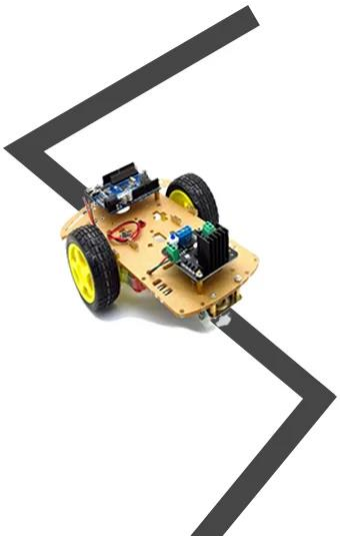
Final Checklist Before You Say "It Works!"

- Does the robot do what it's supposed to every time?
- Does it work in different lighting or movement conditions?
- Does it recover from small problems (like battery voltage drop)?
- Is the code neat, readable, and efficient?

Testing and Debugging a Line-Following Robot

What is the Robot's Task?

This robot is designed to follow a black line on a white surface using three IR sensors.



Components Used:

- Arduino Uno
- Two DC motors with motor driver (L298N)
- IR sensor array (3 sensors: Left, Middle, Right)
- Chassis with wheels
- Battery pack

Step-by-Step Testing:

1. Check the Setup

- Make sure all wires are connected properly.
- Upload the code to Arduino.
- Open the serial monitor to see what values the sensors are giving.

2. Test the Sensors

- Place the robot on the black line.
- The IR sensors should give these results:
  - On black line → Sensor value = LOW (0)
  - On white surface → Sensor value = HIGH (1)

3. Test the Motors

- Run a simple code to check motor movement:
- digitalWrite(Motor1A, HIGH);
- digitalWrite(Motor1B, LOW); // Moves forward
- Check if both motors are rotating in the correct direction.

Debugging the Robot

Problem: Robot turns in circles instead of following the line.

a) Wrong Sensor Reading

- Use Serial.print() to see sensor values.
- Maybe the sensor is not detecting the line correctly.
- Fix: Adjust the sensor or update the code to better read the surface.

b) Motor Wires Reversed

- If robot turns left instead of right, check motor wiring.
- Fix: Swap the wires or change motor direction in the code.

c) Mistake in the Code

- Maybe the if-else conditions are wrong.
- Fix: Print sensor values to check which block is running, then fix the logic.

Example debugging code:

```
if(leftSensor == LOW && middleSensor == HIGH && rightSensor == LOW) {
  Serial.println("Going Forward");
```

```
  // Move forward
} else if(leftSensor == LOW) {
  Serial.println("Turning Left");
  // Turn left
} else if(rightSensor == LOW) {
  Serial.println("Turning Right");
  // Turn right
}
```

Final Testing

- Place the robot on the track again.
- Check if it now follows the line smoothly.
- Make small changes to motor speed or turning if needed.

Chapter Highlights

- **Debugging** means finding and fixing errors in the robot's code or hardware.
- **Testing** is the process of checking if the robot works properly under all conditions.
- **Syntax errors** happen when the code is not written in the correct format.
- **Logic errors** give wrong results even though the code runs.
- **Hardware errors** occur due to loose wires, wrong connections, or damaged parts.
- **Runtime errors** happen while the program is running, often causing the system to crash.
- Some **useful debugging tips** include:
  - Checking the code line by line to spot mistakes.
  - Testing one part at a time instead of the full robot.
  - Verifying all wire connections and pin numbers.
- In the **final check**, make sure that the robot works correctly every time it is used. It performs well in different types of situations. The code is neat, easy to understand, and runs efficiently.

Exercise

Multiple-Choice Questions (MCQs)

1. What is the main purpose of debugging?
   a) Writing programs
   b) Fixing hardware
   c) Fixing errors in the program
   d) Charging the robot
2. Which tool helps you see sensor readings in Arduino?
   a) Multimeter
   b) Serial Monitor
   c) Timer
   d) Motor Driver
3. A **logic error** means:
   a) The robot won't start
   b) The code does not compile
   c) The robot does the wrong task
   d) The sensor is disconnected
4. Which error occurs if a semicolon is missing?
   a) Runtime error
   b) Hardware error
   c) Logic error
   d) Syntax error
5. Which of the following is used to test voltage or current?
   a) LED

b) Serial Monitor
c) Multimeter
d) Flowchart

6. What should be tested first in robot testing?
   a) Full system
   b) Software only
   c) Each component separately
   d) Battery
7. What is the first step in debugging?
   a) Uploading code
   b) Taking a break
   c) Checking code line by line
   d) Asking for help
8. What does a runtime error mean?
   a) The code is perfect
   b) The robot behaves correctly
   c) The program crashes while running
   d) There is a loose wire
9. A tool to display messages like "Sensor value = 500" is:
   a) Serial Monitor
   b) LED
   c) Flowchart
   d) Logic analyser
10. What does edge-case testing mean?
    a) Ignoring sensor values
    b) Testing with wrong inputs
    c) Testing in extreme situations
    d) Skipping testing

True or False

1. Debugging helps improve the robot's appearance.
2. A logic error causes the program to compile but behave incorrectly.
3. You should test the full system before testing individual components.
4. Serial Monitor helps track what the robot is doing.
5. Debugging is only needed in complex robots.

Fill in the blanks

1. _____ is the process of finding and fixing errors in code or hardware.
2. A _____ error happens when the program runs but does the wrong thing.
3. The _____ tool in Arduino IDE is used to print values for checking.
4. Testing each small part of code is called _____ testing.
5. A _____ helps measure voltage in a circuit.

Assertion and Reason

1. **Assertion (A):** Debugging helps in making robots more reliable.
   Reason (R): It identifies and fixes errors in code and hardware.
   a) Both A and R are true, and R is the correct explanation of A
   b) Both A and R are true, but R is not the correct explanation of A
   c) A is true, but R is false
   d) A is false, but R is true
2. **Assertion (A):** Syntax errors are difficult to detect.
   Reason (R): Syntax errors are caused by logical mistakes in thinking.
   a) Both A and R are true, and R is the correct explanation of A
   b) Both A and R are true, but R is not the correct explanation of A
   c) A is true, but R is false
   d) A is false, but R is true
3. **Assertion (A):** A robot should always be tested under different conditions.
   Reason (R): Different conditions help identify edge-case failures.
   a) Both A and R are true, and R is the correct explanation of A
   b) Both A and R are true, but R is not the correct explanation of A
   c) A is true, but R is false
   d) A is false, but R is true
4. **Assertion (A):** The serial monitor is used to power the robot.
   Reason (R): Serial monitor helps display sensor and program data.
   a) Both A and R are true, and R is the correct explanation of A
   b) Both A and R are true, but R is not the correct explanation of A
   c) A is true, but R is false
   d) A is false, but R is true
5. **Assertion (A):** Logic errors cause a robot to stop working completely.
   Reason (R): Logic errors are due to wrong programming logic.
   a) Both A and R are true, and R is the correct explanation of A
   b) Both A and R are true, but R is not the correct explanation of A
   c) A is true, but R is false
   d) A is false, but R is true

Short Answer Questions

1. What is debugging in robotics?
2. List any two types of errors in robot programming.
3. How is the serial monitor useful in debugging?
4. Why is unit testing important?
5. Name any two tools used for testing a robot.

Long Answer Questions

1. Explain the steps you follow while debugging a robot program.
2. Describe any four types of errors commonly found in robotics.
3. How do testing and debugging help in building reliable robots?
4. Write a checklist to ensure your robot program is fully tested.
5. Compare and explain unit testing, system testing, and edge-case testing in robotics.