Obstacle-Avoiding Robot Using Arduino

Moruţan Maria
Group: 30433
20.01.2026

Contents

1. Problem Statement

Autonomous mobile robots are increasingly used in environments where direct human control is impractical or impossible. One of the most fundamental challenges in autonomous navigation is **obstacle avoidance**: the ability of a robot to move through an environment without colliding with objects in its path.

The problem addressed by this project is the following:

Design and implement an autonomous robot capable of detecting obstacles in its environment and avoiding them in real time, without human intervention.

The robot must:

- Continuously move forward in an unknown environment
- Detect obstacles ahead using distance sensors
- Decide an alternative direction when an obstacle is encountered
- Resume forward motion after avoiding the obstacle
- Operate autonomously using embedded logic

This problem is representative of real-world applications such as:

- Robotic vacuum cleaners
- Warehouse automation
- Autonomous vehicles (at a simplified scale)

2. Proposed Project

The proposed project is an **Arduino-based obstacle-avoiding robot** that uses:

- An **ultrasonic distance sensor** for obstacle detection
- Two **DC motors** for locomotion
- A **motor driver** for directional control
- A **servo motor** to rotate the ultrasonic sensor
- Embedded C++ logic running on an Arduino microcontroller

The robot navigates autonomously by:

1. Moving forward continuously
2. Measuring the distance to objects in front
3. Stopping and reversing when an obstacle is detected
4. Scanning the environment to the left and right
5. Choosing the safest direction and turning accordingly
6. Resuming forward movement

Each motion operation is encapsulated into a function:

- `moveForward()` activates both motors in the forward direction.

- `moveBackward()` reverses both motors.
- `moveStop()` disables all motor outputs.
- `turnLeft()` and `turnRight()` pivot the robot by driving motors in opposite directions.
- Pulse-width modulation (`analogWrite`) is used to control motor speed.

This improves readability, reusability, and debugging.

When an obstacle is detected:

1. The robot stops.
2. It moves forward briefly to reposition.
3. The ultrasonic sensor scans right and left using the servo motor.
4. Distances are compared.
5. The robot turns toward the direction with more free space.
6. The robot stops again before resuming normal operation.
7. If no obstacle is detected, the robot is commanded to move forward

Although the software logic instructs the robot to **move forward** when no obstacle is detected, the robot physically moves **straight backward**.

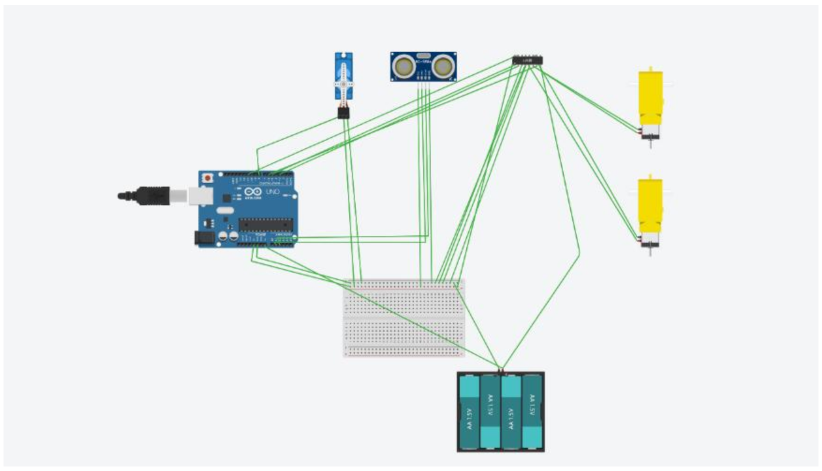
Cause:

- One or both motors are wired in reverse polarity.
- As a result, the electrical "forward" command produces mechanical backward rotation.
- Since both motors are reversed in the same way, the robot still moves straight, but in the opposite direction.

Why Turning Still Works:

- During turns, the motors are driven in opposite directions.
- Even with reversed wiring, the relative motion remains correct.
- This masks the wiring issue during rotation but reveals it during straight motion.

3. Circuit schematic



4. Picture of the project