Lab name : Fundamentals of Integrated Robotic Car Development

Description : This lab introduces the core concepts behind building an integrated robotic car. Participants will start by assembling the car's physical structure and connecting essential components like motors, sensors, and microcontroller. In the first section, students will learn how to set up and calibrate sensors to detect obstacles. Following this, they will move on to programming motor controls and implementing basic movement functions. Finally, students will combine sensor data and motor commands to create a robotic car capable of navigating obstacles autonomously. By the end of this lab, participants will have a basic working model of a robotic car and an understanding of its key components.

Level : medium

Objective:
Familiarize with assembling the robotic car chassis and programming basic motor control for forward and backward movement.
Materials Needed:

- Robotic car chassis
- Motors with driver module (e.g., L298N)
- Microcontroller (e.g., Arduino Uno or similar)
- Battery pack
- Jumper wires
  Steps:
- **Assemble the Chassis**:
  - Attach wheels and motor mounts to the chassis as instructed in your kit.
  - Ensure the motors are securely fastened to avoid movement issues.
- **Connect Motors to Motor Driver**:
  - Attach each motor to the motor driver module, connecting the motor’s output pins to the motor driver’s OUT pins.
  - Connect the motor driver’s power and ground pins to the microcontroller’s corresponding pins.
- **Program Basic Motor Control**:
  - ```
    const int motorPin1 = 3;
    const int motorPin2 = 4;

    void setup() {
        pinMode(motorPin1, OUTPUT);
        pinMode(motorPin2, OUTPUT);
    }
    ```
    ```

```
void moveForward() {
    digitalWrite(motorPin1, HIGH);
    digitalWrite(motorPin2, LOW);
}

void moveBackward() {
    digitalWrite(motorPin1, LOW);
    digitalWrite(motorPin2, HIGH);
}

void stopCar() {
    digitalWrite(motorPin1, LOW);
    digitalWrite(motorPin2, LOW);
}

void loop() {
    moveForward();
    delay(2000);
    stopCar();
    delay(1000);
    moveBackward();
    delay(2000);
    stopCar();
    delay(1000);
}
```

- **Testing**:
- Power on the robotic car and upload the code.
- Verify the forward, stop, and backward movements are functioning as expected.

Objective:
Combine obstacle detection with autonomous navigation, allowing the car to move forward and stop when an obstacle is detected.
Materials Needed:

- Robotic car with chassis and motors (assembled in Lab Guide 1)
- Ultrasonic sensor
- Microcontroller (e.g., Arduino Uno)
- Jumper wires
  Steps:
- **Mount the Ultrasonic Sensor**:
  Attach the ultrasonic sensor to the front of the robotic car to detect obstacles ahead.
- **Connect the Ultrasonic Sensor**:
  - Connect the sensor's VCC and GND to the microcontroller.
  - Attach the Trigger and Echo pins of the sensor to two digital pins on the microcontroller (e.g., pin 9 for Trigger, pin 10 for Echo).
- **Write Code (C++) for Obstacle Detection**:

```
const int motorPin1 = 3;
const int motorPin2 = 4;
const int trigPin = 9;
const int echoPin = 10;

void setup() {
    pinMode(motorPin1, OUTPUT);
    pinMode(motorPin2, OUTPUT);
```

```
    pinMode(trigPin, OUTPUT);
    pinMode(echoPin, INPUT);
    Serial.begin(9600);
}

int getDistance() {
    digitalWrite(trigPin, LOW);
    delayMicroseconds(2);
    digitalWrite(trigPin, HIGH);
    delayMicroseconds(10);
    digitalWrite(trigPin, LOW);

    long duration = pulseIn(echoPin, HIGH);
    int distance = duration * 0.034 / 2;
    return distance;
}

void loop() {
    int distance = getDistance();
    Serial.print("Distance: ");
    Serial.print(distance);
    Serial.println(" cm");

    if (distance < 15) { // Obstacle within 15 cm
        digitalWrite(motorPin1, LOW);
        digitalWrite(motorPin2, LOW);
        delay(500);  // Stop briefly
    } else {
        digitalWrite(motorPin1, HIGH);
        digitalWrite(motorPin2, LOW);
    }

    delay(100);
}
```

- **Testing Autonomous Navigation**:
  - Power the robotic car, upload the code, and observe its behavior as it encounters obstacles.
  - The car should stop when it detects an object within the set distance (15 cm).
- **Adjustments and Debugging**:
  - If the car isn't stopping as expected, ensure the ultrasonic sensor is correctly wired.
  - Adjust the distance threshold (e.g., `15 cm`) or the car's speed for optimal performance.
