**📄 Project Documentation: Automatic Vacuum Cleaner using Arduino**

**✅ Objective**

To design and build a low-cost automatic vacuum cleaner robot using Arduino that can detect obstacles, navigate a room autonomously, and clean the floor using a small vacuum motor.

**🔧 Components Required**

| **Component** | **Quantity** | **Description** |
| --- | --- | --- |
| Arduino UNO | 1 | Microcontroller to control the system |
| Ultrasonic Sensor (HC-SR04) | 1–3 | For obstacle detection |
| Motor Driver Module (L298N) | 1 | To control the motors |
| BO Motors with wheels | 2 | For movement |
| Caster wheel | 1 | For balance |
| Small Vacuum Motor (DC) | 1 | To suck dirt |
| Power Supply (Li-ion Battery) | 1 | Power source |
| Chassis (robot frame) | 1 | Body of the vacuum cleaner |
| Wires and connectors | As needed | For connections |
| Breadboard (optional) | 1 | For prototyping |

**⚙️ Working Principle (How It Works)**

The automatic vacuum cleaner uses Arduino as the brain. It moves around a room using two DC motors. An **ultrasonic sensor** in front detects obstacles. When the sensor detects a wall or obstacle within a certain distance, it instructs the motors to stop and change direction.

Meanwhile, a small **vacuum motor** remains ON during the entire operation, sucking in dirt and dust into a container attached to the bottom of the robot.

The robot repeats this process continuously, covering the area and cleaning it automatically.

**🔌 Circuit Diagram**

Here's a basic overview:

* **Motors (BO Motors)** connected to **Motor Driver Module**
* **Motor Driver** connected to **Arduino (pins 5, 6, 10, 11)**
* **Ultrasonic Sensor (HC-SR04)** connected to Arduino (Trigger - pin 8, Echo - pin 9)
* **Vacuum Motor** connected to Arduino (through relay or transistor, or directly to battery with switch)
* **Battery Pack** powers Arduino and motors

*Note: You can draw or use simulation tools like Tinkercad or Fritzing for an actual diagram.*

**💻 Arduino Code Overview**

cpp

CopyEdit

#define ENA 5 // Motor A speed control

#define IN1 6 // Motor A direction

#define IN2 7

#define ENB 10 // Motor B speed control

#define IN3 11 // Motor B direction

#define IN4 12

#define trigPin 8

#define echoPin 9

long duration;

int distance;

void setup() {

pinMode(trigPin, OUTPUT);

pinMode(echoPin, INPUT);

pinMode(ENA, OUTPUT);

pinMode(IN1, OUTPUT);

pinMode(IN2, OUTPUT);

pinMode(ENB, OUTPUT);

pinMode(IN3, OUTPUT);

pinMode(IN4, OUTPUT);

Serial.begin(9600);

}

void loop() {

// Ultrasonic sensor logic

digitalWrite(trigPin, LOW);

delayMicroseconds(2);

digitalWrite(trigPin, HIGH);

delayMicroseconds(10);

digitalWrite(trigPin, LOW);

duration = pulseIn(echoPin, HIGH);

distance = duration \* 0.034 / 2;

Serial.print("Distance: ");

Serial.println(distance);

if (distance < 20) {

// Obstacle detected, move back and turn

moveBackward();

delay(500);

turnRight();

delay(700);

} else {

// Move forward

moveForward();

}

delay(100);

}

void moveForward() {

digitalWrite(IN1, HIGH);

digitalWrite(IN2, LOW);

analogWrite(ENA, 200);

digitalWrite(IN3, HIGH);

digitalWrite(IN4, LOW);

analogWrite(ENB, 200);

}

void moveBackward() {

digitalWrite(IN1, LOW);

digitalWrite(IN2, HIGH);

analogWrite(ENA, 200);

digitalWrite(IN3, LOW);

digitalWrite(IN4, HIGH);

analogWrite(ENB, 200);

}

void turnRight() {

digitalWrite(IN1, HIGH);

digitalWrite(IN2, LOW);

analogWrite(ENA, 200);

digitalWrite(IN3, LOW);

digitalWrite(IN4, HIGH);

analogWrite(ENB, 200);

}

**⚙️ Features**

* Obstacle avoidance using ultrasonic sensor
* Autonomous movement
* Continuous vacuum cleaning while navigating
* Low power consumption
* Cost-effective and DIY-friendly

**📏 Improvements / Add-Ons (Optional)**

* Use **IR sensors** for edge detection (prevent falling from stairs)
* Use **rotating brushes** for better cleaning
* Add **Bluetooth module** for manual control
* Add **dustbin full detection**
* Add **charging dock and auto-charging feature**

**🧪 Testing**

1. Place the robot on a clean surface.
2. Power it on. It should start moving forward.
3. If it detects an obstacle (within 20 cm), it should stop, reverse, and turn.
4. The vacuum motor should be ON and collecting dust.
5. Observe its ability to avoid objects and navigate a room.

**📚 Conclusion**

This project demonstrates how simple components and Arduino programming can create a useful autonomous device. Though basic, this robot mimics the core functionality of commercial robotic vacuum cleaners like Roomba. With enhancements, it can be a powerful home-cleaning assistant.