Informe Final de Proyecto de Laboratorio

Autor: Wadhy Zahir Callan

Título del Proyecto:

Vehículo de Exploración y Recolección de Datos en Terrenos de Difícil Acceso

Propósito y Funcionalidades del Prototipo

El objetivo principal de este proyecto es desarrollar un vehículo autónomo y controlado de forma remota que pueda recolectar datos ambientales en terrenos y ambientes de difícil acceso, donde el ser humano no puede llegar fácilmente. Este vehículo está diseñado específicamente para tareas de exploración y reconocimiento.

Funcionalidades principales del prototipo:

1. Recolección de datos ambientales:
   - Medición de temperatura y humedad mediante un sensor DHT11.
   - Detección de niveles de monóxido de carbono (CO) con un sensor MQ9.
2. Almacenamiento de datos:
   - Los datos recopilados se registran y almacenan en una tarjeta SD utilizando un módulo lector de tarjeta SD.
3. Sistema móvil autónomo:
   - Implementación de un sistema de control remoto vía Bluetooth (BT).
   - Capacidad para moverse de manera estable en terrenos complejos, utilizando un sistema de motores y sensores para interactuar con su entorno.

Selección de Componentes

Placas de Control:

1. Arduino UNO R3 (Primera placa):
   - Controla los motores a través de un shield L293D.
   - Gestiona la comunicación Bluetooth.
2. Arduino UNO R3 (Segunda placa):
   - Gestiona la adquisición de datos mediante los sensores DHT11 y MQ9.
   - Almacena los datos en una tarjeta SD utilizando el módulo lector correspondiente.

Sensores y Actuadores:

1. Sensor DHT11: Mide la temperatura y la humedad.
2. Sensor MQ9: Detecta niveles de monóxido de carbono (CO).
3. Motores:
   - Cuatro motores DC de 3 a 6 V con caja reductora.

Codigo Fuente:

1) Auto control remoto:

```
#include <AFMotor.h>
#include <SoftwareSerial.h>

// Configuración de los pines para el Bluetooth
int bluetoothTx = 2; // TX del Arduino conectado al RXD del HC-05
int bluetoothRx = 3; // RX del Arduino conectado al TXD del HC-05
SoftwareSerial bluetooth(bluetoothTx, bluetoothRx);

// Inicialización de los motores en el shield
AF_DCMotor motor1(1, MOTOR12_1KHZ); // Motor delantero izquierdo
AF_DCMotor motor2(2, MOTOR12_1KHZ); // Motor delantero derecho
AF_DCMotor motor3(3, MOTOR34_1KHZ); // Motor trasero izquierdo
AF_DCMotor motor4(4, MOTOR34_1KHZ); // Motor trasero derecho

void setup() {
 // Configuración inicial
 Serial.begin(9600);       // Monitor serial para depuración
 bluetooth.begin(9600);    // Configuración del módulo Bluetooth

 // Configurar velocidades iniciales para los motores
 motor1.setSpeed(200); // Velocidad de 0 a 255
```

```
  motor2.setSpeed(200);
  motor3.setSpeed(200);
  motor4.setSpeed(200);

  // Inicializar todos los motores en estado detenido
  detener();
  Serial.println("Esperando comandos Bluetooth...");
}

void loop() {
  // Verificar si hay datos disponibles desde el Bluetooth
  if (bluetooth.available()) {
    char command = bluetooth.read(); // Leer el comando enviado desde la app
    Serial.print("Comando recibido: ");
    Serial.println(command);

    // Evaluar el comando recibido y ejecutar la acción correspondiente
    switch (command) {
      case 'F': // Avanzar
        adelante();
        break;
      case 'B': // Retroceder
        atras();
        break;
      case 'L': // Girar a la izquierda
        izquierda();
```

```
      break;
    case 'R': // Girar a la derecha
      derecha();
      break;
    case 'S': // Detener
      detener();
      break;
    default:
      detener(); // Por seguridad, detener si el comando no es reconocido
      break;
   }
  }
}

// Funciones para controlar los motores
void adelante() {
  motor1.run(FORWARD);
  motor2.run(FORWARD);
  motor3.run(FORWARD);
  motor4.run(FORWARD);
  Serial.println("Avanzando hacia adelante...");
}

void atras() {
  motor1.run(BACKWARD);
  motor2.run(BACKWARD);
```

```
  motor3.run(BACKWARD);
  motor4.run(BACKWARD);
  Serial.println("Retrocediendo...");
}

void izquierda() {
  motor1.run(FORWARD);
  motor2.run(FORWARD);
  motor3.run(BACKWARD); // Motores traseros giran en direcciones opuestas para girar
  motor4.run(FORWARD);
  Serial.println("Girando a la izquierda...");
}

void derecha() {
  motor1.run(FORWARD);
  motor2.run(FORWARD);
  motor3.run(FORWARD); // Motores traseros giran en direcciones opuestas para girar
  motor4.run(BACKWARD);
  Serial.println("Girando a la derecha...");
}

void detener() {
  motor1.run(RELEASE);
  motor2.run(RELEASE);
  motor3.run(RELEASE);
```

```
  motor4.run(RELEASE);
  Serial.println("Motores detenidos.");
 }
```

2) Arduino recolector de datos ambientales:

```
#include <Adafruit_Sensor.h>
#include <DHT.h>
#include <DHT_U.h>
#include <SD.h>

// Configuración del sensor DHT
#define DHTPIN A0     // Pin al que está conectado el sensor DHT
#define DHTTYPE DHT11 // Tipo de sensor: DHT11 o DHT22

DHT dht(DHTPIN, DHTTYPE); // Inicializar el sensor DHT

// Configuración del sensor MQ-9
#define MQ9PIN A1 // Pin al que está conectado el sensor MQ-9

const int chipSelect = 10; // Pin CS de la tarjeta SD

void setup() {
  Serial.begin(9600);
  Serial.println("Iniciando...");
```

```
  dht.begin(); // Iniciar el sensor DHT

  // Inicializar la tarjeta SD
  if (!SD.begin(chipSelect)) {
    Serial.println("Error al inicializar la tarjeta SD.");
    while (true); // Detener el programa si falla
  }
  Serial.println("Tarjeta SD inicializada correctamente.");
}

void loop() {
  // Leer datos del DHT11
  float hum = dht.readHumidity();    // Leer humedad
  float temp = dht.readTemperature(); // Leer temperatura

  if (isnan(hum) || isnan(temp)) {
    Serial.println("Error al leer el sensor DHT.");
    return;
  }

  // Leer datos del MQ-9
  int gasValue = analogRead(MQ9PIN); // Leer el valor analógico del MQ-9

  // Imprimir datos en el monitor serial
  Serial.print("Temperatura: ");
```

```
  Serial.print(temp);
  Serial.print(" °C, Humedad: ");
  Serial.print(hum);
  Serial.print(" %, Gas: ");
  Serial.println(gasValue);

  // Guardar datos en la tarjeta SD
  File dataFile = SD.open("datos.txt", FILE_WRITE);
  if (dataFile) {
    dataFile.print("Temperatura: ");
    dataFile.print(temp);
    dataFile.print(" °C, Humedad: ");
    dataFile.print(hum);
    dataFile.print(" %, Gas: ");
    dataFile.println(gasValue);
    dataFile.close();
    Serial.println("Datos guardados correctamente en la tarjeta SD.");
  } else {
    Serial.println("Error al abrir el archivo en la tarjeta SD.");
  }

  delay(1000); // Breve demora de 1 segundo
}
```