UNIVERSIDAD DE GUADALAJARA

CENTRO UNIVERSITARIO DE CIENCIAS EXACTAS E INGENIERÍA

SEMINARIO DE PROBLEMAS DE PROGRAMACIÓN DE SISTEMAS EMBEBIDOS

ACTIVIDAD 8. CARRITO SEGUIDOR DE LUZ

PROFESOR: EHECATL JOEL CHÁVEZ MARTÍNEZ

ALUMNOS:

- FABIO ENRIQUE ALMANZA CASTAÑEDA
- LUIS FERNANDO CASTILLO HERNÁNDEZ
- DAMIÁN ALEJANDRO LIRA ALAMO

GUADALAJARA, JALISCO, 2025

Objetivo:

Diseñar e implementar un robot autónomo que detecte y se desplace hacia una fuente de luz utilizando sensores LDR, un ESP32 como controlador principal y motores DC controlados mediante un puente H.

MATERIALES:

CANDTIDAD	COMPONENTE
1	Placa ESP32
4	Motores DC (TT)
1	Driver L298N
2	Fotorresistencias(LDR)
1	Chasis para Robot
1	Fuente de Alimentación
2	Resistencias 10K
Cables	Duopont y Alimentacion

Descripción de la práctica:

Un seguidor de luz es un robot móvil capaz de detectar la dirección de mayor intensidad luminosa y moverse en esa dirección. Este comportamiento se logra con sensores de luz (como LDRs), que disminuyen su resistencia al aumentar la luz. Usando un divisor de voltaje, es posible obtener una señal analógica proporcional a la luz percibida.

El ESP32, como microcontrolador principal, lee los valores de los LDRs a través de entradas analógicas. Según la diferencia de intensidad entre ambos sensores, decide cómo activar los motores a través del puente H, lo que permite avanzar, girar o detenerse.

Pasos a seguir:

1. Conexión de sensores LDR: o Cada LDR forma un divisor de voltaje con una resistencia de 10kΩ. o Se conectaron al ESP32 en los pines ADC (por ejemplo: GPIO 34 y GPIO 35).

2. Control de motores: o Se utilizaron 4 motores reductores conectados a un puente H. o Los pines de control del puente H se conectaron a salidas digitales PWM del ESP32.

3. Lógica de control: o El ESP32 compara los valores leídos por los dos sensores LDR. o Si hay más luz a la izquierda, gira hacia la izquierda. Si hay más luz a la derecha, gira hacia la derecha. Si ambos reciben luz similar, avanza.

4. Pruebas: o Se probaron distintas fuentes de luz. o Se ajustaron los umbrales de diferencia de luz para mejorar la precisión de movimiento.

Explicación del código:

1. Declaración de Pines

```
// Pines de sensores de luz
const int LDR_IZQ = 34;  // ADC1
const int LDR_DER = 35;  // ADC1


// Pines del motor (con driver L298N o similar)
const int IN1 = 25; // Motor Izquierdo adelante
const int IN2 = 26; // Motor Izquierdo atrás
const int IN3 = 27; // Motor Derecho adelante
const int IN4 = 14; // Motor Derecho atrás
```

Se definen los pines de entrada para los sensores LDR (izquierdo y derecho), conectados a pines analógicos del ESP32 (GPIO 34 y 35).

También se definen los pines de salida que se conectan al driver de motor, que controla el movimiento del motor izquierdo y derecho.

2. Configuración inicial

```
void setup() {
  Serial.begin(115200);


  // Configurar pines de motor como salida
  pinMode(IN1, OUTPUT);
  pinMode(IN2, OUTPUT);
  pinMode(IN3, OUTPUT);
  pinMode(IN4, OUTPUT);
}
```

Serial.begin(115200); inicia la comunicación serial para poder ver los valores en el monitor serial (útil para depuración).

pinMode(..., OUTPUT); configura los pines de control del motor como salida digital, para poder enviarles señales de encendido o apagado.

3. Lectura de sensores y lógica de control

```
void loop() {
  // Leer valores de los LDR
  int luzIzq = analogRead(LDR_IZQ);
  int luzDer = analogRead(LDR_DER);


  Serial.print("Luz Izq: "); Serial.print(luzIzq);
  Serial.print(" | Luz Der: "); Serial.println(luzDer);


  // Comparar valores
  int diferencia = abs(luzIzq - luzDer);
  int tolerancia = 100; // Ajustable: sensibilidad a la diferencia.
```

```
if (diferencia < tolerancia) {
   // Luz pareja: avanzar recto
   adelante();
 } else if (luzIzq > luzDer) {
   // Más luz a la izquierda: girar izquierda
   girarIzquierda();
 } else {
   // Más luz a la derecha: girar derecha
   girarDerecha();
 }
 delay(100);
}
```

- Se leen los valores de luz captados por los LDR usando analogRead(). Cuanta más luz incide sobre un LDR, menor es su resistencia, y mayor será el valor leído.
- Se imprime en consola para monitorear.
- Se calcula la diferencia entre ambos valores. Si la diferencia es pequeña (menor a la tolerancia), se considera que la luz es pareja y el carrito avanza recto.
- Si hay más luz en el LDR izquierdo, el carrito gira a la izquierda; si hay más en el derecho, gira hacia allá.
- El delay(100); añade una pausa de 100 milisegundos para evitar lecturas demasiado rápidas y erráticas.

4. Funciones de movimiento del carrito

```
void adelante() {
  digitalWrite(IN1, HIGH);
  digitalWrite(IN2, LOW);
  digitalWrite(IN3, HIGH);
  digitalWrite(IN4, LOW);
}

void girarIzquierda() {
  digitalWrite(IN1, LOW);
  digitalWrite(IN2, HIGH);
  digitalWrite(IN3, HIGH);
  digitalWrite(IN4, LOW);
}

void girarDerecha() {
  digitalWrite(IN1, HIGH);
  digitalWrite(IN2, LOW);
  digitalWrite(IN3, LOW);
  digitalWrite(IN4, HIGH);
}
```

- Estas funciones controlan el sentido de giro de los motores usando el driver de motor.
- En cada función se establecen los niveles HIGH (encendido) o LOW (apagado) de los pines conectados al driver.

CONCLUCIONES:

• Se logró implementar correctamente un sistema autónomo sencillo para el seguimiento de luz. Este proyecto refuerza conocimientos sobre sensores analógicos, control de motores con puente H y programación básica en microcontroladores como el ESP32. Además, demuestra cómo se puede lograr un comportamiento reactivo mediante electrónica básica y lógica de comparación.

• El ajuste de frecuencia PWM es esencial para obtener un control preciso en motores de baja potencia.

• La estructura del código permite una lógica clara y extensible, ideal para prácticas básicas de robótica.

Fotografías: