PROYECTO FINAL

SENSOR DE APARCAMIENTO CON ARDUINO, BUZZER Y LED.

LEIRE VIDAL VILLEGAS. CRIMINOLOGÍA E INGENIERÍA INFORMÁTICA.
SISTEMAS EMPOTRADOS Y DE TIEMPO REAL

ÍNDICE:

1. INTRODUCCIÓN

En el desarrollo de este proyecto se implementara un sensor de aparcamiento utilizando un microcontrolador Arduino Uno, un sensor ultrasónico, un LED y un buzzer. El sistema mide la distancia a un objeto que se encuentre frente al sensor y emite una señal luminosa y auditiva para avisar al conductor. La velocidad de los pitidos del buzzer aumenta a medida que el objeto se acerca, mientras que el LED se enciende solo cuando la distancia es crítica (menor o igual a 20 cm).

Este tipo de sistemas es fundamental en la seguridad vial y puede servir como base para desarrollos más avanzados en vehículos autónomos o sistema de asistencia al conductor.

2. OBJETIVOS

Los objetivos del proyecto son:

- Desarrollar un dispositivo electrónico que mida la distancia a un objeto usando un sensor ultrasónico.
- Emitir una señar visual mediante un LED para indicar proximidad crítica.
- Emitir señales sonoras mediante un buzzer que modulen la frecuencia de pitidos según la distancia.
- Configurar el sistema para que las alertas comiencen a partir de 20 centímetros y que la frecuencia sea fija, pero con aumentando la velocidad de los pitidos cuanto más cerca esté el objeto.

3. MATERIALES

Para la realización de esta proyecto se requieren los siguientes componentes:

- ARDUINO UNO. Para poder añadir el código al sistema. Cuenta con pines digitales y analógicos suficientes para este proyecto. Una unidad aproximadamente 20€.
- SENSOR ULTRASÓNICO HC-SR04. Funciona mediante la emisión de ondas ultrasónicas a 40kHz. El sensor tiene dos partes, el transmisor (TRIG) y el receptor (ECO). El sensor envía la señal ultrasónica y luego el ECHO se activa mientras recibe la reflexión de la onda, permitiendo calcular el tiempo de vuelo y, con ello, la distancia. Una unidad aproximadamente 3€.
- LED. Diodo emisor de luz para señalizar visualmente la proximidad. Se conecta en serie con una resistencia para limitar la corriente y evitar que se queme. Una unidad aproximadamente 0.10€.
- BUZZER PIEZOELÉCTRICO. Genera un tono audible controlado por Arduino. Se utiliza para emitir pitidos cuya cadencia se ajusta según la distancia detectada. Una unidad aproximadamente 1€.
- RESISTENCIA DE 220 OHMIOS. Para limitar la corriente que recibe el LED. Una unidad aproximadamente 0.10€.
- PROTOBOARD y CABLES DE CONEXIÓN. Para poder montar el circuito (sin soldadura). Aproximadamente 5€.

Aunque todos los elementos empleados en la implementación de este sistema se encontraban dentro de los materiales proporcionados, se ha añadido un estimado del precio del montaje de este sistema. Precio total aproximado: 29.10€.

4. MONTAJE DEL CIRCUITO

El montaje se realiza en una protoboard realizando las siguientes conexiones:

- SENSOR ULTRASÓNICO HC-SR04.

- VCC se conecta a 5V de Arduino (cable naranja).
- GND se conecta a tierra de Arduino (cable amarillo).
- Pin TRIG conectado a pin digital 10 (cable azul).
- Pin ECHO conectado a pin digital 9 (cable verde).

- LED

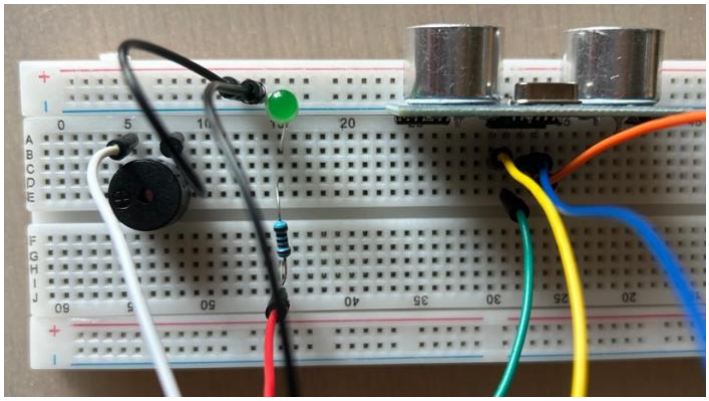
- Ánodo (+) al pin digital 3 a través de la resistencia 220Ω (cables rojo).
- Cátodo (-) a GND (cable negro).

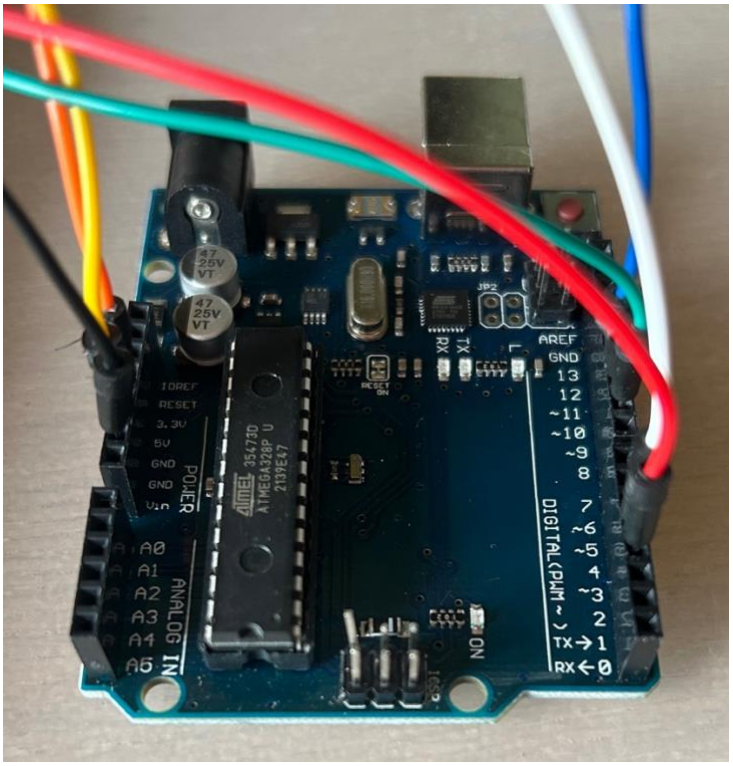
- BUZZER

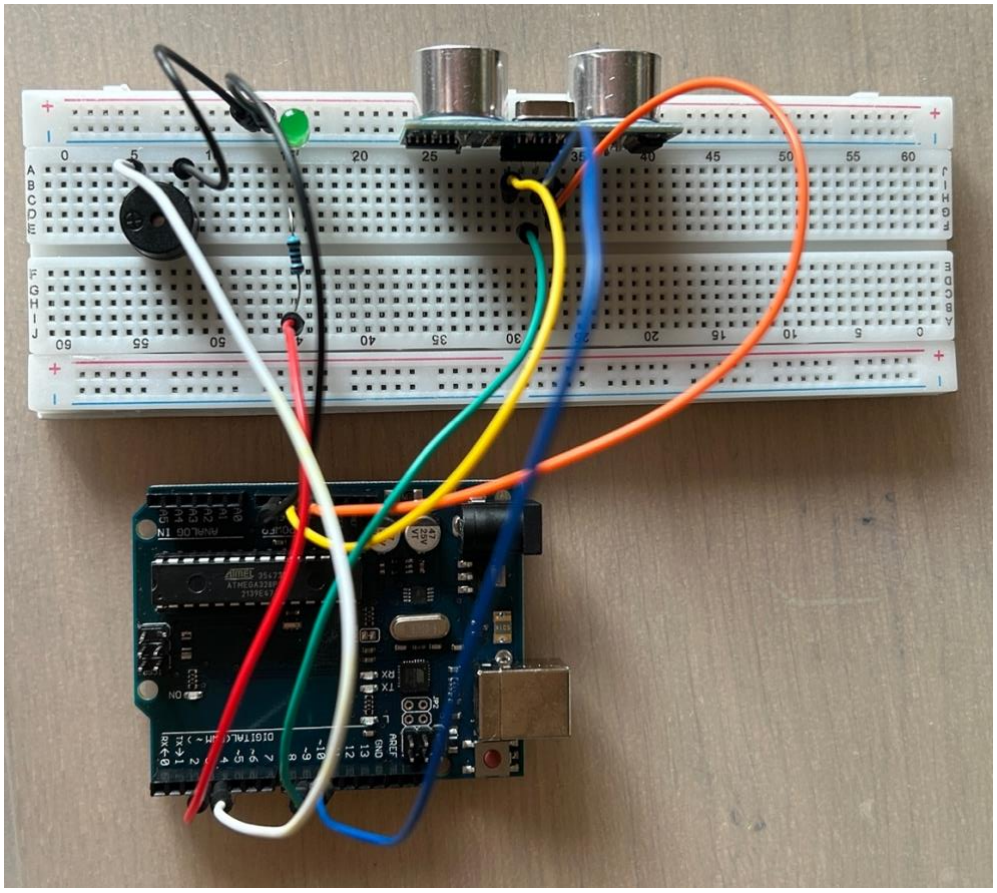
- Terminal positivo al pin digital 4 (cable blanco).
- Terminal negativo a GND (cable negro conectado también para el LED).

El funcionamiento del sistema sigue el siguiente esquema:

- GENERACIÓN DEL PULSO ULTRASÓNICO. Arduino genera un pulso de 10 microsegundos en el pin TRIG para activar el sensor.
- RECEPCIÓN DEL ECO. El pin ECHO queda en HIGH durante el tiempo que tarda el pulso ultrasónico en ir, reflejarse y volver.
- CÁLCULO DE DISTANCIA. Se mide el tiempo del pulso HIGH con *pulseIn()* y, conociendo la velocidad del sonido, se calcula la distancia en centímetros.
- CONTROL DEL LED Y BUZZER.
    - Si la distancia está ENTRE 1 y 20 centímetros. El LED se enciende, el buzzer emite pitidos a 1kHZ y la frecuencia de pitidos es más rápida cuanto menos es la distancia.
    - Si al distancia es MAYOR a 20 centímetros o NO ha detectado el objeto. El LED y el buzzer se quedan apagados.




AREF
GND
13
12
~11
~10
~9
8
7
~6
~5
4
~3
2
TX→1
RX←0
DIGITAL(PWM~)
POWER
ANALOG IN
A0
A1
A2
A3
A4
A5
ON
RESET
TX
RX
JP2



5. CÓDIGO FUENTE

```
// pines

int TRIG = 10;  // pin para disparar el pulso ultrasónico
int ECO = 9; // pin para recibir el eco
int LED = 3; // pin para el LED
int BUZZER = 4; // pin para el buzzer

//variables

int DURACION; // duración del pulso del sensor ultrasónico
int DISTANCIA; // distancia calculada en centímetros

void setup() {
  pinMode(TRIG, OUTPUT); // configura TRIG como salida
  pinMode(ECO, INPUT); // configura ECO como entrada
  pinMode(LED, OUTPUT); // configura LED como salida
  pinMode(BUZZER, OUTPUT); // configura buzzer como salida
  Serial.begin(9600); // inicializa comunicación serial para monitoreo
}
```

```
void loop() {
  // enviar pulso ultrasónico
  digitalWrite(TRIG, LOW);
  delayMicroseconds(2);
  digitalWrite(TRIG, HIGH);
  delayMicroseconds(10);
  digitalWrite(TRIG, LOW);

  // medir duración y convertir a distancia en centimetros
  DURACION = pulseIn(ECO, HIGH);   // leer duración del pulso en el pin ECO
  DISTANCIA = DURACION / 58.2; // calcular distancia: velocidad sonido / 2 (ida y vuelta)
  Serial.println(DISTANCIA); // mostrar distancia por puerto serial
  delay(30);

// comprobar si la distancia está dentro del rango crítico
  if (DISTANCIA > 0 && DISTANCIA <= 20) {
    digitalWrite(LED, HIGH); // encender LED

    // intervalo del pitido: cuanto más cerca el pitido es más rápido (de 30 ms a 1000 ms)
    int beepDelay = map(DISTANCIA, 1, 20, 30, 1000);

    tone(BUZZER, 1000);  // frecuencia fija de 1kHz
    delay(50);           // duración corta del pitido
    noTone(BUZZER);
    delay(beepDelay);    // pausa proporcional a la distancia

  } else {
    digitalWrite(LED, LOW); // apagar LED
    noTone(BUZZER);  // apagar buzzer
  }
}
```

5.1. DECLARACIÓN DE PINES

```
int TRIG = 10;  // pin para disparar el pulso ultrasónico
int ECO = 9; // pin para recibir el eco
int LED = 3; // pin para el LED
int BUZZER = 4; // pin para el buzzer
```

- TRIG. Pin digital que enviará el pulso ultrasónico.
- ECO. Pin que recibe el eco del sensor tras rebotar en un objeto.
- LED. LED de advertencia que se encenderá si hay un objeto cerca.
- BUZZER. Dispositivo que emitirá un sonido de alerta.

5.2. VARIABLES DE MEDICIÓN

```
int DURACION; // duración del pulso del sensor ultrasónico
int DISTANCIA; // distancia calculada en centímetros
```

- DURACION. Tiempo (en microsegundos) que tarda el eco en volver.
- DISTANCIA. Resultado del cálculo de distancia a partir de DURACION.

5.3. FUNCIÓN DE CONFIGURACIÓN

```
void setup() {
  pinMode(TRIG, OUTPUT); // configura TRIG como salida
  pinMode(ECO, INPUT); // configura ECO como entrada
  pinMode(LED, OUTPUT); // configura LED como salida
  pinMode(BUZZER, OUTPUT); // configura buzzer como salida
  Serial.begin(9600); // inicializa comunicación serial para monitoreo
}
```

- Se definen los modos de los pines.
- Se inicializa la comunicación serial a 9600 baudios para ver datos en el Monitor Serial.

5.4. FUNCIÓN PRINCIPAL

Esta función se repite mientras Arduino este encendido.

- ENVIO DEL PULSO ULTRASÓNICO

```
digitalWrite(TRIG, LOW);
delayMicroseconds(2);
digitalWrite(TRIG, HIGH);
delayMicroseconds(10);
digitalWrite(TRIG, LOW);
```

- Se envía un pulso de 10 microsegundos para activar el sensor.
- Primero se pone a LOW, luego a HIGH durante 10 microsegundos y, finalmente, de nuevo a LOW.

- MEDIR DURACIÓN Y CALCULAR LA DISTANCIA

```
DURACION = pulseIn(ECO, HIGH);   // leer duración del pulso en el pin ECO
DISTANCIA = DURACION / 58.2; // calcular distancia: velocidad sonido / 2 (ida y vuelta)
Serial.println(DISTANCIA); // mostrar distancia por puerto serial
delay(30);
```

- pulseIn(ECO,HIGH) mide cuanto tiempo el pin ECO está en alto, es decir, el tiempo que tardó en recibir el eco.
- La fórmula DISTANCIA = DURACION / 58.2 convierte el tiempo en distancia en centímetros, ya que el sonido recorre aproximadamente 58.2 microsegundos por centímetro (en ida y vuelta).
- Se muestra el resultado en el monitor.
- Se espera (30 microsegundos) antes de hacer la siguiente medición.

- OBSTÁCULO A 20 CENTÍMETROS O MENOS

```
if (DISTANCIA > 0 && DISTANCIA <= 20) {
  digitalWrite(LED, HIGH); // encender LED

  // intervalo del pitido: cuanto más cerca el pitido es más rápido (de 30 ms a 1000 ms)
  int beepDelay = map(DISTANCIA, 1, 20, 30, 1000);

  tone(BUZZER, 1000);  // frecuencia fija de 1kHz
  delay(50);           // duración corta del pitido
  noTone(BUZZER);
  delay(beepDelay);    // pausa proporcional a la distancia

} else {
  digitalWrite(LED, LOW); // apagar LED
  noTone(BUZZER);  // apagar buzzer
}
```

- Si hay un objeto entre 1 y 20 centímetros se enciende el LED.
- map() transforma el rango [1, 20] en un rango de tiempo [30, 1000] milisegundos.
  - A 1 centímetro la espera será de 30 milisegundos. Pitidos más rápidos.
  - A 20 centímetros la espera será de 1000 milisegundos. Pitidos más lentos.
- tone(BUZZER, 1000), el buzzer emite un tono de 1000 Hz (1kHz).
- delay(50), el sonido dura solo 50 milisegundos.
- noTone(Buzzer), se apaga el buzzer.
- delay(beepDelay), espera antes del próximo pitido.

- NO HAY OBJETO CERCA

```
} else {
  digitalWrite(LED, LOW); // apagar LED
  noTone(BUZZER);  // apagar buzzer
}
```

- Si la distancia es mayor a 20 centímetros o no se detecta, se apaga el LEC y el buzzer.

6. LIMITACIONES Y POSIBLES MEJORAS

6.1. LIMITACIONES

El rango es bastante limitado, alertando solo a partir de los 20 centímetros. De la misma manera, la frecuencia del buzzer es fija, no cambiando de tonalidad. Esto podría hacer la alerta menos intuitiva.

De la misma manera, al estar formada solo por un sensor, no se cubren áreas laterales o ángulos diferentes, tampoco contemplando ambientes ruidosos donde el sonido del buzzer peude pasar desapercibido.

6.2. POSIBLES MEJORAS

Entre las mejoras se podría cambiar la frecuencia del buzzer para que el tono sea más agudo cuando más cerca esté el objeto, aumentando la señal sonora. También se pueden agregar más LEDs para indicar diferentes rangos de distancias.

Por otro lado, se puede incluir una pantalla LCD para mostrar la distancia en centímetros en tiempo real o usar múltiples sensores ultrasónicos que puedan cubrir un rango más amplio, detectando obstáculos en carios puntos.

Otras incorporaciones que se podrían realizar serían vibraciones o alertas hápticas para realizar señales más perceptibles en ambientes ruidosos o implementar un modo de ahorro de energía que apague el sistema cuando no haya ningún objeto cerca.