 **UNIVERSIDAD POLITECNICA DEL**

**VALLE DE TOLUCA.**

Proyecto: Sensor de proximidad (Arduino UNO)

Asignatura: Programación Avanzada

PRESENTA:

Maldonado García Brayan Uriel

Jiménez López Benjamín

PROFESORA:

Monserrath Ponce Cruz

**Julio, 2019**

**Introducción**

Arduino es una plataforma electrónica en código abierto que se basa en una placa input/output y un entorno de desarrollo (IDE) basado lenguaje de programación.

En relación a esto, el microcontrolador usado para el desarrollo de este proyecto es una placa controladora ARDUINO UNO en conjunto de utilización de inteligencia artificial puede realizar funciones y procesos multifuncionales capaces de tomar decisiones de acuerdo a las condiciones previamente cumplidas o por otra parte no cumplidas, la inteligencia artificial a logrado grandes avances dentro del campo tecnológico y la globalización.

En el desarrollo de la tecnología de los microcontroladores tenemos una gran variedad de tipos y modelos para cada uno de los problemas e innovaciones de las industrias y empresas en forma general, sin embargo, muchas soluciones resultan muy costosas para nuestro medio, por lo que es necesario buscar otras alternativas y herramientas tecnológicas.

Al realizar el sistema de proximidad utilizaremos la tarjeta “Arduino uno”, el cual esta basado principalmente en el sistema de estacionamiento de los automóviles modernos en específico el sensor de proximidad el cual se encuentra localizado en la parte trasera y delantera del automóvil.

**Objetivo**

En los últimos años los sensores de proximidad que se encuentran en modelos como el Hyundai Elantra han obtenido mayor importancia, sobre todo en los automóviles en los que la tecnología es la base de su funcionamiento. Una de las características que mayor atención ha logrado es la identificación del peligro inminente, que permite saber cuándo el vehículo está demasiado cerca de un objeto. A pesar de esta característica muchos aún no saben que son los sensores de proximidad o cómo funcionan realmente.

Antes, la publicidad estaba enfocada en los nuevos diseños o en la capacidad de sus motores. Ahora, los fabricantes promocionan sus nuevos vehículos por otras características, no disponibles hasta hace muy poco.

Desde la asistencia para estacionar como la que posee el Hyundai Elantra, que ayuda a los conductores a completar algunas de las maniobras más complejas, el parqueo paralelo hasta el control de cruce adaptado (ACC), que permite a tu vehículo disminuir o incrementar la velocidad dependiendo del tráfico alrededor tuyo. La manera en la que los automóviles son construidos, usados y vendidos está siendo revolucionada.

 Estas capacidades, y otras como esta son muy importantes para el desarrollo de los nuevos automóviles, y no podría ser posible sin los sensores de proximidad, pero que es lo que son, y ¿cómo funcionan?

El presente proyecto trata de simular el sistema que utilizan los coches modernos al momento de estacionarse, nos es útil para no colisionar con los otros vehículos u objetos, según nos acercamos a un objeto el pitido va subiendo en frecuencia.

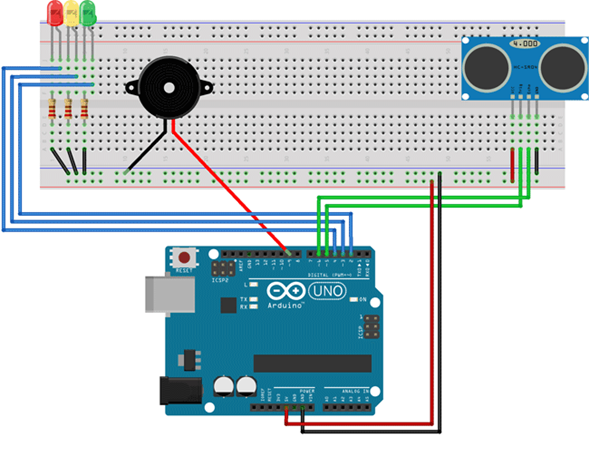


Figura 1. Diagrama del circuito (Fuente: Elaboración propia).

**Componentes**

* Arduino UNO
* Protoboard donde conectaremos los componentes
* Cables para hacer las conexiones
* 3 resistencias de 220 Ω
* 1 LED verde
* 1 LED amarillo
* 1 LED rojo
* 1 sensor ultrasónico Arduino (HC-SR04)
* 1 buzzer

LEDs también ayudan a saber la cercanía o lejos que estamos del objeto en cuestión.

Si se encuentra lejos, el sonido se emite con una frecuencia baja y se enciende el LED verde.

Si estamos en una posición intermedia, el pitido aumenta la frecuencia y se encenderá el LED amarillo.

Por último, si estamos demasiado cerca el pitido será muy agudo, una frecuencia alta, y se encenderá el LED rojo.

**Justificación**

El presente proyecto tiene como finalidad explicar el funcionamiento de un sensor de proximidad controlado a través de la herramienta Arduino y así proporcionar un esquema (ver Figura 1.) acerca de la construcción de este sensor y cualquier persona pueda implementarlo o adecuarlo a sus actividades cotidianas.

**Desarrollo**

El funcionamiento es muy sencillo. El sensor envía una onda ultrasónica a través del disparador o trigger, revota contra el objeto y el receptor o echo detecta la onda. Sabiendo cuánto ha tardado en viajar dicha onda, podemos saber la distancia.

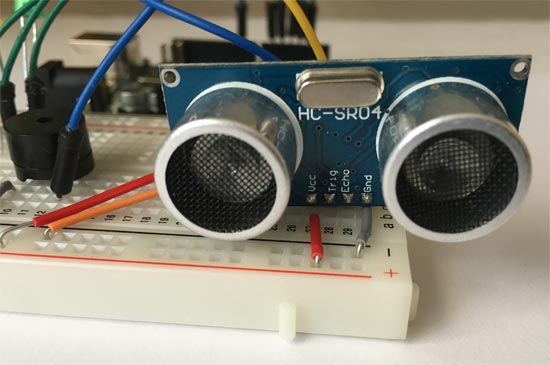


Imagen 2. Sensor ultrasonico (Fuente, Arduino.Org)

Su función es avisar de lo cerca o lejos que estamos de otro vehículo u objeto a la hora de aparcar. Nos avisa con un sonido y con unas luces que te indican lo cerca o lejos que estás de colisionar.

Para simular correctamente el sensor de distancia vamos a utilizar un buzzer Arduino. Estos componentes utilizan la piezoelectricidad, un fenómeno físico que afecta en determinados cristales (el cuarzo es el más común). Al someter un cristal de este tipo, se deforman y vibran. Si conseguimos que esa vibración tenga una frecuencia dentro del espectro audible, conseguiremos un sonido.

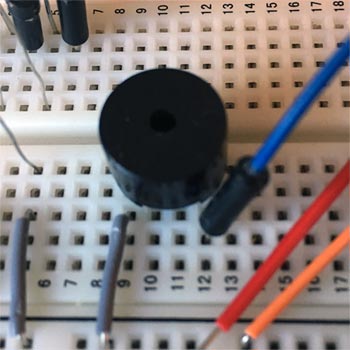


Imagen 3. Buzzer (Fuente, Arduino.Org)

**Código para la programación del dispositivo**

#define LEDVERDE 2

#define LEDAMARILLO 3

#define LEDROJO 4

#define TRIGGER 5

#define ECHO 6

#define BUZZER 9

const float sonido = 34300.0; // Velocidad del sonido en cm/s

const float umbral1 = 30.0;

const float umbral2 = 20.0;

const float umbral3 = 10.0;

void setup() {

Serial.begin(9600);

pinMode(LEDVERDE, OUTPUT);

pinMode(LEDAMARILLO, OUTPUT);

pinMode(LEDROJO, OUTPUT);

pinMode(ECHO, INPUT);

pinMode(TRIGGER, OUTPUT);

pinMode(BUZZER, OUTPUT);

apagarLEDs();

}

void loop() {

iniciarTrigger();

float distancia = calcularDistancia();

apagarLEDs();

if (distancia < umbral1)

{

alertas(distancia);

}

}

void apagarLEDs()

{

digitalWrite(LEDVERDE, LOW);

digitalWrite(LEDAMARILLO, LOW);

digitalWrite(LEDROJO, LOW);

}

void alertas(float distancia)

{

if (distancia < umbral1 && distancia >= umbral2)

{

digitalWrite(LEDVERDE, HIGH);

tone(BUZZER, 2000, 200);

}

else if (distancia < umbral2 && distancia > umbral3)

{

digitalWrite(LEDAMARILLO, HIGH);

tone(BUZZER, 2500, 200);

}

else if (distancia <= umbral3)

{

digitalWrite(LEDROJO, HIGH);

tone(BUZZER, 3000, 200);

}

}

float calcularDistancia()

{ caso a HIGH

unsigned long tiempo = pulseIn(ECHO, HIGH);

float distancia = tiempo \* 0.000001 \* sonido / 2.0;

Serial.print(distancia);

Serial.print("cm");

Serial.println();

delay(500);

return distancia;

}

void iniciarTrigger()

{

digitalWrite(TRIGGER, LOW);

delayMicroseconds(2);

digitalWrite(TRIGGER, HIGH);

delayMicroseconds(10);

digitalWrite(TRIGGER, LOW);

}

**MARCO CONCEPTUAL**

Dado que el tema central de esta investigación estará puesto en un Sistema de Proximidad utilizando la herramienta de ARDUINO UNO y la Programación Concurrente, será necesario plantear algunos conceptos que sirvan como apoyo en la lectura.

**1 inteligencia artificial**

La inteligencia artificial es un concepto atractivo para muchas partes interesadas del mundo empresarial, científico y gubernamental. Desde el punto de vista económico, resulta muy atractivo que las máquinas puedan realizar tareas para las que se solía requerir la intervención humana. Una solución de inteligencia artificial eficaz puede "pensar" más rápido y procesar más información que cualquier cerebro humano. La inteligencia artificial también presenta el potencial de ampliar las capacidades humanas a lugares donde las personas tienen dificultades para llegar, por ejemplo, el espacio exterior o ubicaciones remotas en la Tierra donde podrían resultar útiles habilidades humanas como los conocimientos médicos.

* + 1. **Algoritmos evolutivos**

Los algoritmos evolutivos son métodos de optimización y búsqueda de soluciones basados en los postulados de la [evolución biológica](http://es.wikipedia.org/wiki/Evoluci%C3%B3n_biol%C3%B3gica). En ellos se mantiene un conjunto de entidades que representan posibles soluciones, las cuales se mezclan, y compiten entre sí, de tal manera que las más aptas son capaces de prevalecer a lo largo del tiempo, evolucionando hacia mejores soluciones cada vez.

Los algoritmos evolutivos, y la [computación evolutiva](http://es.wikipedia.org/wiki/Computaci%C3%B3n_evolutiva), son una rama de la [inteligencia artificial](http://es.wikipedia.org/wiki/Inteligencia_artificial). Son utilizados principalmente en problemas con [espacios de búsqueda](http://es.wikipedia.org/wiki/Espacio_de_b%C3%BAsqueda) extensos y no lineales, en donde otros métodos no son capaces de encontrar soluciones en un tiempo razonable.

**1.1 Programación**

En los últimos años el rol de la programación se ha hecho vital para los procesos de sistematización de tareas y manejo de información. De una manera más específica, la programación se encuentra involucrada en el desarrollo de industrias bien sean de entretenimiento y comunicaciones hasta las compañías en la industria musical, siendo todas estas desarrolladoras de software.

Cuando hablamos de programación nos referimos al proceso por el cual podemos diseñar, codificar, limpiar y proteger el código fuente de programas computacionales.

* + 1. **Programación Avanzada**

La programación avanzada es un proceso que se utiliza para idear y ordenar las acciones que se realizarán en el marco de un proyecto; al anuncio de las partes que componen un acto o espectáculo; a la preparación de máquinas para que cumplan con una cierta tarea en un momento determinado; a la elaboración de programas para la resolución de problemas mediante ordenadores, y a la preparación de los datos necesarios para obtener una solución de un problema.

* + 1. **Programación Concurrente**

Se conoce por programación concurrente a la rama de la informática que trata de las técnicas de programación que se usan para expresar el paralelismo entre tareas y para resolver los problemas de comunicación y sincronización entre procesos.

* + 1. **Programación estructurada**

La programación estructurada es un paradigma de programación orientado a mejorar la claridad, calidad y tiempo de desarrollo de un programa de computadora recurriendo únicamente a subrutinas y tres estructuras básicas: secuencia, selección (if y switch) e iteración (bucles for y while).

Asimismo, se considera innecesario y contraproducente el uso de la instrucción de transferencia incondicional (GOTO), que podría conducir a código espagueti, mucho más difícil de seguir y de mantener, y fuente de numerosos errores de programación.

* + 1. **Programación Lineal**

Se conoce como programación lineal a la técnica de la matemática que permite la optimización de una función objetivo a través de la aplicación de diversas restricciones a sus variables. Se trata de un modelo compuesto, por lo tanto, por una función objetivo y sus restricciones, constituyéndose todos estos componentes como funciones lineales en las variables en cuestión.

**1.1.3 Lenguaje de Programación**

Un lenguaje de programación es un lenguaje formal que proporciona una serie de instrucciones que permiten a un programador escribir secuencias de órdenes y algoritmos a modo de controlar el comportamiento físico y lógico de una computadora con el objetivo de que produzca diversas clases de datos. A todo este conjunto de órdenes y datos escritos mediante un lenguaje de programación se le conoce como programa.

**1.1.3.1 Lengua de programación C**

C es un lenguaje de programación de propósito general: originalmente desarrollado por Dennis Ritchie entre 1969 y 1972 en los Laboratorios Bell, como evolución del anterior lenguaje B, a su vez basado en BCPL.

Se trata de un lenguaje de tipos de datos estáticos, débilmente tipificado, de medio nivel, ya que dispone de las estructuras típicas de los lenguajes de alto nivel pero, a su vez, dispone de construcciones del lenguaje que permiten un control a muy bajo nivel. Los compiladores suelen ofrecer extensiones al lenguaje que posibilitan mezclar código en ensamblador con código C o acceder directamente a memoria o dispositivos periféricos.

**1.1.3.2 C++**

C++ es un lenguaje de programación diseñado en 1979 por Bjarne Stroustrup. La intención de su creación fue extender al lenguaje de programación C mecanismos que permiten la manipulación de objetos. En ese sentido, desde el punto de vista de los lenguajes orientados a objetos, el C++ es un lenguaje híbrido.

**1.1.3.3 PHP**

PHP, acrónimo recursivo en inglés de PHP: Hypertext Preprocessor (preprocesador de hipertexto), es un lenguaje de programación de propósito general de código del lado del servidor originalmente diseñado para el preprocesado de texto plano en UTF-8. Posteriormente se aplicó al desarrollo web de contenido dinámico, dando un paso evolutivo en el concepto de app online, por su caracter de servicio.

Fue uno de los primeros lenguajes de programación del lado del servidor que se podían incorporar directamente en un documento HTML en lugar de llamar a un archivo externo que procese los datos. El código es interpretado por un servidor web con un módulo de procesador de PHP que genera el texto plano en formato UTF-8, ampliamente reconocido por el estándar HTML, dando como resultado, en los exploradores, una salida al usuario perfectamente entendible.

**1.1.3.4 Java Script**

JavaScript (abreviado comúnmente JS) es un lenguaje de programación interpretado, dialecto del estándar ECMAScript. Se define como orientado a objetos, ​ basado en prototipos, imperativo, débilmente tipado y dinámico.

Se utiliza principalmente en su forma del lado del cliente (client-side), implementado como parte de un navegador web permitiendo mejoras en la interfaz de usuario y páginas web dinámicas, aunque existe una forma de JavaScript del lado del servidor (Server-side JavaScript o SSJS). Su uso en aplicaciones externas a la web, por ejemplo, en documentos PDF, aplicaciones de escritorio (mayoritariamente widgets) es también significativo.

* 1. **ARDUINO UNO**

Arduino Uno es una placa electrónica basada en el microcontrolador ATmega328. Cuenta con 14 entradas/salidas digitales, de las cuales 6 se pueden utilizar como salidas PWM (Modulación por ancho de pulsos) y otras 6 son entradas analógicas. Además, incluye un resonador cerámico de 16 MHz, un conector USB, un conector de alimentación, una cabecera ICSP y un botón de reseteado. La placa incluye todo lo necesario para que el microcontrolador haga su trabajo, basta conectarla a un ordenador con un cable USB o a la corriente eléctrica a través de un transformador.

* 1. **SENSOR ULTRASONICO HC-SR04**

El sensor HC-SR04 es un sensor de distancia de bajo costo que utiliza ultrasonido para determinar la distancia de un objeto en un rango de 2 a 450 cm. Destaca por su pequeño tamaño, bajo consumo energético, buena precisión y excelente precio. El sensor HC-SR04 es el más utilizado dentro de los sensores de tipo ultrasonido, principalmente por la cantidad de información y proyectos disponibles en la web. De igual forma es el más empleado en proyectos de robótica como robots laberinto o sumo, y en proyectos de automatización como sistemas de medición de nivel o distancia.

El sensor HC-SR04 posee dos transductores: un emisor y un receptor piezoeléctricos, además de la electrónica necesaria para su operación. El funcionamiento del sensor es el siguiente: el emisor piezoeléctrico emite 8 pulsos de ultrasonido(40KHz) luego de recibir la orden en el pin TRIG, las ondas de sonido viajan en el aire y rebotan al encontrar un objeto, el sonido de rebote es detectado por el receptor piezoeléctrico, luego el pin ECHO cambia a Alto (5V) por un tiempo igual al que demoró la onda desde que fue emitida hasta que fue detectada, el tiempo del pulso ECO es medido por el microcontrolador y así se puede calcular la distancia al objeto. El funcionamiento del sensor no se ve afectado por la luz solar o material de color negro (aunque los materiales blandos acústicamente como tela o lana pueden llegar a ser difíciles de detectar).

La distancia se puede calcular utilizando la siguiente formula:

Distancia(m) = {(Tiempo del pulso ECO) \* (Velocidad del sonido=340m/s)}/2

* 1. **BUZZER**

Un buzzer pasivo o un altavoz son dispositivos que permiten convertir una señal eléctrica en una onda de sonido. Estos dispositivos no disponen de electrónica interna, por lo que tenemos que proporcionar una señal eléctrica para conseguir el sonido deseado.

En oposición, los buzzer activos disponen de un oscilador interno, por lo que únicamente tenemos que alimentar el dispositivo para que se produzca el sonido.