Facultad
de
Ciencias

Diseño e implementación de un vehículo programable con Arduino

(Design and implementation of a programmable vehicle with Arduino)

Trabajo de Fin de Grado
para acceder al

GRADO EN INGENIERIA INFORMÁTICA

Autor: Pedro Lastra Machín

Director: Fernando Vallejo

Febrero - 2020

GRADUADO EN INGENIERÍA INFORMÁTICA

CALIFICACIÓN DEL TRABAJO FIN DE GRADO

Realizado por: Pedro Lastra Machín
Director del TFG: Fernando Vallejo
Título: "Diseño e implementación de un vehículo programable con Arduino"
Title: "Design and implementation of a programmable vehicle with Arduino "

Presentado a examen el día:

para acceder al Título de

INGENIERO EN INFORMÁTICA

Composición del Tribunal:

Presidente:
Secretario:
Vocal:

Este Tribunal ha resuelto otorgar la calificación de: .....

Fdo.: El Presidente **Fdo.: El Secretario**

Fdo.: El Vocal **Fdo.: El Director del TFG**
(sólo si es distinto del secretario)

Vº Bº del Subdirector **Trabajo Fin de Grado N.º**
(a asignar por Secretaría)

Agradecimientos

Me gustaría reconocer en este trabajo el apoyo y la confianza de todos aquellos que me han acompañado estos años y también de los que lo harán en el futuro.

A mis profesores de la facultad de ciencias por su compañía durante los meses en los que se ha desarrollado este trabajo, en especial a Fernando por su ayuda sin la cual no hubiese sido posible la realización de este.

Y sobre todo a la familia y amigos, aquellos que con su apoyo y compañía hacen que uno tenga motivos para seguir trabajando.

Resumen:

En este proyecto se va a diseñar y construir un vehículo autónomo programable controlado por un microcontrolador basado en Arduino uno. El objetivo es que el vehículo sea capaz de ejecutar una serie de movimientos previamente almacenados en su memoria de forma autónoma, además se le podrá dotar de mecanismos de seguridad que eviten la colisión con obstáculos que se encuentren en su trayectoria.

La construcción del vehículo se llevará a cabo mediante fichas LEGO y su desarrollo contará con unas pruebas hardware de los diferentes componentes y la conexión entre estos, la búsqueda del diseño para adaptarse al mayor número de situaciones, la interacción del vehículo con el medio, los diferentes problemas mecánicos, sus materiales y como tema principal, el desarrollo software donde se hablará del manejo del vehículo, funciones utilizadas, tratamiento de errores y posibles mejoras.

El control del vehículo se realizará con un microcontrolador basado en Arduino al que se le incorporarán los periféricos necesarios (motores, sensores, botoneras, pantalla, etc.). La programación de estos dispositivos se intentará realizar, en su mayoría, al más bajo nivel evitando el uso de drivers disponibles en internet.

Palabras clave: Arduino, microcontrolador, driver, vehículo

Abstract:

In this project I will design and build an autonomous programmable vehicle controlled by a microcontroller based on Arduino one. The objective is to have a vehicle capable of executing a series of movements previously stored in its memory in an autonomous way. It will also be able to be equipped with safety mechanisms that avoid collision with obstacles that are in its trajectory.

The construction of the vehicle will be carried out by means of LEGO cards and its development will include hardware tests of the different components and the connection between them, the search for the design to adapt to the greatest number of situations, the interaction of the vehicle with the environment, the different mechanical problems, its materials and, as the main topic, the development of software where the handling of the vehicle, functions used, treatment of errors and possible improvements will be discussed.

The control of the vehicle will be done with a microcontroller based on Arduino to which the necessary peripherals will be incorporated (engines, sensors, button panels, screen, etc.). The programming of these devices will try to be done, mostly, at the lowest level avoiding the use of drivers available on the internet.

Keywords: Arduino, microcontroller, driver, vehicle

Índice de contenido

1 - Introducción y objetivos 9
1.1 Motivación 9
1.2 Objetivo 9
1.3 Estructura de la memoria 10
2 – Descripción funcional 12
2.1 Descripción física del vehículo 12
2.2 Descripción funcional del vehículo 13
3.1 Diseño físico del vehículo 15
3.2 Diseño de la electrónica 16
3.2.a Arduino UNO 16
3.2.b Sistema motriz 18
3.2.c Pantalla LCD 22
3.2.d LED de estado 24
3.2.e Sensor de distancia 25
3.2.f Mando a distancia 27
3.3 Diseño del software 29
3.3.a Rutina de inicialización 30
3.3.b Funciones 31
3.3.c Bucle 33
3.3.d Comportamiento ante obstáculos 34
3.3.e Tratamiento de errores 34
4 - Evaluación y pruebas 35
4.1 Pruebas individuales de funcionamiento de cada módulo 35
4.2 Pruebas de funcionamiento global 36
5 – Conclusiones y trabajos futuros 38
6 – Bibliografía 40

Índice de figuras

Fig. 1 Mando infrarrojo **12**
Fig. 2 Componentes del vehículo **13**
Fig. 3 Diseño estructural **15**
Fig. 4 Esquema de la placa Arduino **17**
Fig. 5 Motor con reductora **19**
Fig. 6 Placa fundumoto **20**
Fig. 7 Técnica PWM **21**
Fig. 8 Pantalla LCD **22**
Fig. 9 I2C PCF8574 ..... **23**
Fig. 10 Esquema I2C **24**
Fig. 11 Esquema bombilla LED **25**
Fig. 12 Esquema sensor ultrasónico **26**
Fig. 13 Esquema de pulsos **27**
Fig. 14 Onda en el protocolo NEC **28**
Fig. 15 Estructura bloque de datos protocolo NEC **29**
Fig. 16 Mando IR **29**

1 - Introducción y objetivos

Inicialmente se hablará sobre la motivación que impulsa a la realización de este trabajo, así como los objetivos planteados al inicio de este. Por último, se indica la estructura que se sigue en esta memoria.

1.1 Motivación

La llegada a la sociedad actual de dispositivos inteligentes, como electrodomésticos o vehículos autónomos, ha incrementado el uso de microprocesadores que, controlando diversos sensores y actuadores, permiten realizar numerosas tareas de manera autónoma con el menor esfuerzo posible. Ha sido tanta la inserción de estos dispositivos en la vida diaria que nos los podemos encontrar casi en cualquier ámbito de aplicación, desde grandes procesos industriales hasta los nuevos juguetes futuristas que permiten a los más pequeños aprender los conceptos de las nuevas tecnologías mientras desarrollan sus primeros juegos.

La idea de profundizar en los procesos de automatización ha llevado a desarrollar este proyecto. Para ello se ha buscado una plataforma sencilla sobre la que integrar diverso hardware y permitir realizar tareas de control. Para poder desarrollar un proyecto en base a esta tecnología, se ha decidido realizar un móvil programable, al estilo de ciertos juguetes que actualmente se comercializan para chavales de a partir de los 4 o 5 años como el *Anki Cozmo Vector* [6].

1.2 Objetivo

Un vehículo autónomo programable es aquel vehículo que por medio de una configuración previamente establecida puede realizar de manera independiente tareas y movimientos tomando decisiones por si solo sin que sea necesario la intervención de ninguna acción humana.

Partiendo de esta definición, se pretende diseñar y crear un vehículo autónomo programable el cual sea capaz de, recibiendo una serie de comandos de movimiento elegidos por el usuario, intentar lleva a cabo una ruta esquivando los obstáculos que encuentre en su trayectoria dando media vuelta cuando

aparezca uno de estos para poder continuar, así como informar al usuario en todo momento de la situación del vehículo mediante luces led y una pantalla.

Sus objetivos serán los siguientes:

- Ejecutar una serie de movimientos introducidos por el usuario sin colisionar con ningún elemento.

- Ser capaz de ejecutar tres tipos de movimientos diferentes; movimientos en línea recta, tanto hacia delante como hacia atrás, giros de 90 grados y cambios de dirección.

- Ser capaz de cambiar la dirección de la ruta en caso de que no sea posible realizarla por la aparición de un obstáculo.

- Informar al usuario en todo momento de la situación del vehículo.

Para conseguir estos objetivos, el proyecto ha pasado por tres fases de diseño: el diseño y construcción de un vehículo, fase en la cual se elegirá la mejor disposición de los elementos para conseguir adaptarse mejor al entorno y ser capaz de responder a los obstáculos más rápidamente; el diseño e implementación de la electrónica, fase en la que, tras probar varios posibles mecanismos para llevar a cabo el movimiento, la dirección, la recepción de datos y el muestreo de los eventos, se elegirán los componentes más adecuados para el funcionamiento; y la programación del sistema, donde una vez elegidos los componentes y la disposición de estos se harán funcionar de forma óptima cumpliendo los objetivos planteados.

1.3 Estructura de la memoria

La memoria realizada presenta una estructura dividida en cuatro partes principales para poder explicar con exactitud cómo funciona, por qué funciona de la manera en que lo hace y cómo ha evolucionado hasta el punto final. Sus partes son las siguientes:

1. Descripción funcional: Donde se hará una pequeña descripción física del vehículo explicando los distintos elementos que se encontrará un usuario que desee utilizar el vehículo. En segundo lugar, se detallará como se interactúa con el vehículo, programando su movimiento y las distintas indicaciones que nos mostrará el vehículo. Por último, se hará una explicación exhaustiva de cómo actúa el vehículo ante diferentes eventos,

tanto del usuario como los encontrados en el entorno durante su movimiento.

2. Diseño y desarrollo del vehículo: Para hablar de diseño es necesario hablar del diseño físico del vehículo, el diseño de la electrónica y el diseño del software. En el diseño del vehículo se describirá como se ha diseñado el vehículo, la instalación de los módulos y la disposición de estos. En el diseño de la electrónica se hablará de cada uno de los módulos que componen la electrónica del vehículo explicando con detalle cómo funciona cada uno de estos, que tecnología utilizan y el papel que cumplen dentro del circuito, así como la manera de conectarse a la placa central. Por último, en el diseño del software, se hablará sobre la metodología de diseño de software utilizada, así como el planteamiento de un diseño modular, describiendo el algoritmo de software y las diferentes funciones y partes del código que hacen funcionar al vehículo de forma eficiente.

3. Evaluación y pruebas: En esta parte se hablará sobre las pruebas unitarias de funcionamiento de cada módulo que se han realizado por separado para, una vez vistos los resultados de cada uno de estos, llevar a cabo unas pruebas de funcionamiento globales. Estas pruebas otorgarán los resultados que se muestran al hacer funcionar los módulos en conjunto en diferentes situaciones. Así se hará una valoración de si el vehículo es totalmente operativo y puede responder a todo tipo de eventos.

4. Conclusiones y trabajos futuros: Por último, se hablará brevemente de las conclusiones obtenidas una vez finalizado con éxito la construcción y diseño del vehículo, así como de posibles trabajos futuros para mejorar algunas de las funcionalidades que este presentará.

2 – Descripción funcional

En este apartado se presentará de forma genérica el proyecto explicando sus principales funciones, objetivos, disposición y diseño.

2.1 Descripción física del vehículo

Como se explicará más adelante en el capítulo 3, se ha realizado un prototipo con piezas LEGO en el que, sin embargo, se pueden apreciar los distintos componentes necesarios para su correcto funcionamiento en las Figuras 1 y 2. Estos componentes son los siguientes:

- Ruedas: encargadas de realizar los diferentes movimientos entre los que destacan los desplazamientos en línea recta, los giros y los cambios de dirección. Además, proporcionan la estabilidad necesaria para poder llevar a cabo estos movimientos sin peligro de vuelco ni accidente durante la marcha, sosteniendo sobre ellas el peso de todo el vehículo. Dos de estas son las ruedas motoras y la restante es una rueda auxiliar.

- Entrada de datos: la cual se llevará a cabo a través de un sensor infrarrojo el cual recibirá los datos de un mando a distancia y un sensor de ultrasonidos para determinar la distancia a los obstáculos.

Figure 1 Mando Infrarrojo

- Salida de información: a través de una pantalla LCD y una bombilla led se mantendrá al usuario informado de los diferentes eventos sucedidos durante la ejecución del programa.

- La batería: haciendo uso de una pila alcalina de 9V se alimentará todo el circuito.

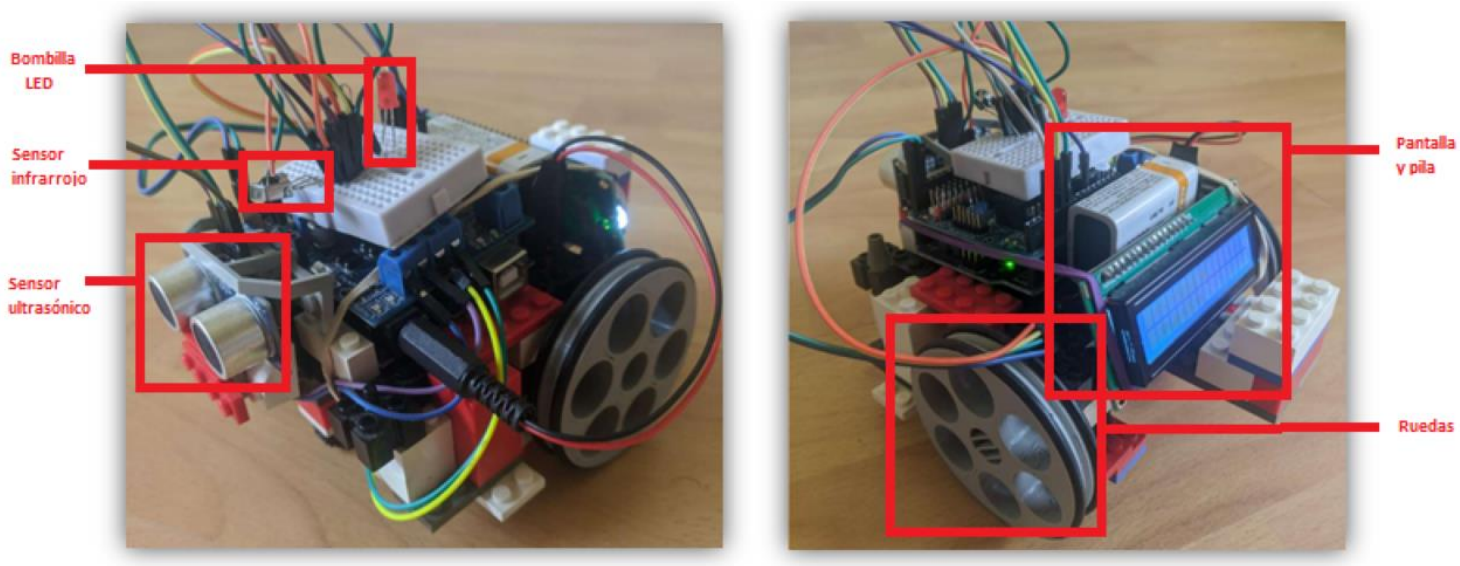


Figure 2: Componentes del vehículo

Este vehículo será capaz de funcionar en superficies lisas sin demasiada rugosidad evitando lugares con escalones o alturas ya que su estructura actual solo permitirá la detección de obstáculos, pero no de orificios o huecos.

2.2 Descripción funcional del vehículo

A continuación, se explicará el funcionamiento que lleva a cabo el vehículo autónomo construido y cómo responde a diferentes eventos, ya sean aquellos que favorecen el funcionamiento con fluidez como aquellos otros que hacen saltar algún tipo de excepción o impiden que se realice el movimiento correctamente. En este último caso se explicará cómo responde el vehículo y cómo continua la ejecución una vez tratado el error.

Para encender el vehículo es necesario únicamente enchufarlo a la corriente que ofrece la pila. En ese momento la pantalla se enciende e indica lo que el usuario debe hacer. Inicialmente, el vehículo se queda en modo de espera hasta que el usuario introduzca a través del mando a distancia los comandos deseados para el funcionamiento. Estos comandos podrán ser cuatro; marcha adelante, recorriendo una distancia de unos 30 centímetros manteniendo la dirección; marcha atrás, recorriendo de nuevo unos 30 centímetros; giro a la izquierda y giro a la derecha, de 90 grados correspondiendo respectivamente a los números 2, 8, 4 y 6 del mando los cuales una vez pulsados en el orden deseados son almacenados en la memoria del vehículo pudiendo almacenar hasta 50 comandos diferentes para más tarde ser ejecutados. Se ha decidido hacer los giros pivotando

sobre la rueda que se encuentra en la dirección del giro, es decir, frenando el motor de dicha rueda y aplicando velocidad en la rueda contraria para así cambiar la dirección del vehículo.

Cuando el usuario pulsa alguno de estos botones, se muestra por pantalla el comando introducido para verificar la recepción de este. Los comandos de marcha adelante y marcha atrás se realizarán a una velocidad lo suficientemente segura para en caso de obstáculo, dar tiempo a los motores a detener el vehículo por completo.

En el caso de que el usuario introduzca un comando desconocido o una combinación de botones, se parará la recepción de comandos, se mostrará un mensaje de error por pantalla informando del evento sucedido y a continuación se continuará con la recepción de comandos sin guardar dicho comando erróneo.

El usuario podrá finalizar el proceso de introducción de comandos pulsando el botón 5. La introducción de comandos también finaliza si se ha alcanzado el máximo de comandos permitidos. Al finalizar el proceso de introducción de comandos, el vehículo pasa al modo de ejecución de comandos. Aquí, una vez mostrado el mensaje de "Ejecutando..." se comienzan a ejecutar los comandos que previamente introdujo el usuario. Si uno de estos comandos es una marcha atrás, este se ejecutará mientras la luz led roja está encendida a modo de alerta, indicando así que la dirección ha cambiado y que en ese momento el vehículo no está capacitado para detectar obstáculos ya que la detección de estos solo se puede dar cuando este realiza un movimiento hacia adelante.

En el proceso de ejecución de comandos, si el vehículo se topa con algún obstáculo a menos de 30 centímetros desde la frontal, este mostrará por pantalla un mensaje indicando que ha encontrado un obstáculo, encenderá su luz led de forma intermitente y tras dar un giro de 180 grados sobre su propio eje, continua la marcha en el punto donde se quedó. Este evento puede darse tantas veces como obstáculos encuentre y al finalizar la serie se mostrará el número de veces que el vehículo tuvo que dar la vuelta por uno de estos obstáculos.

Una vez ejecutados todos los movimientos almacenados, se muestra un mensaje por pantalla de fin y se pregunta al usuario si quiere empezar de nuevo el proceso introduciendo nuevos comandos. Para volver a empezar el proceso es necesario que el usuario pulse el botón 5 del mando. Si este lo pulsa, el proceso se reiniciará volviendo a pedir al usuario el envío de nuevos comandos, pero si por el contrario pulsa otro botón se mostrará un mensaje para reintentar la recepción del botón 5 del mando. Si el usuario no pulsa ningún botón, se da por finalizada la ejecución y el programa se queda parado a la espera de ser desconectado o recibir una instrucción de reinicio.

3 Diseño y desarrollo del vehículo autónomo programable

Para poder describir el diseño del vehículo de forma completa, es necesario dividirlo en tres partes explicando con exactitud la distribución de sus elementos, la manera en la que estos funcionan y como se han unificado en un único sistema para poder llevar a cabo un objetivo general.

3.1 Diseño físico del vehículo

El prototipo presentado realizado con LEGO pretende ser lo más funcional posible sin presentar una apariencia parecida a la de un vehículo clásico y la disposición de los módulos, así como la correcta selección de estos, hacen esto posible. En la parte inferior se encuentran las ruedas, dos de estas colocadas en la parte trasera conectadas a una placa Fundumoto, encargada de controlar sus motores y conectada directamente sobre una placa Arduino, y una rueda auxiliar en la parte delantera. Sobre estas, se encuentran la placa Arduino UNO y la placa Fundumoto.

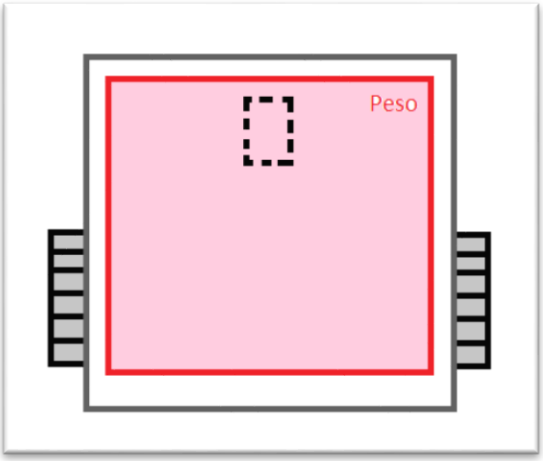


Figure 3: Diseño estructural

El uso de tres ruedas proporciona estabilidad y mayor maniobrabilidad respecto a las 4 ruedas.

Para poder asegurar que el vehículo puede detectar y esquivar los obstáculos, presenta en la parte delantera un módulo de ultrasonidos conectado directamente a la placa Arduino y orientado hacia la frontal del dispositivo.

Por otra parte, con la intención de mantener al usuario informado tanto de las diferentes fases en las que se encuentra el vehículo como de posibles errores u obstáculos encontrados, se encuentran en la parte superior una pantalla LCD y una bombilla led conectados a puertos digitales de la placa Arduino, así como el receptor IR el cual debe estar colocado lo más al descubierto posible para recibir correctamente los comandos sin pérdidas.

3.2 Diseño de la electrónica

En el diseño de la electrónica hablaremos sobre los componentes que incorpora el vehículo explicando uno a uno su funcionamiento, las técnicas que utilizan, cómo se han conectado cada uno de ellos y como se ha diseñado el sistema de movimiento. Toda esta electrónica está basada en Arduino UNO.

3.2.a Arduino UNO

Arduino en una plataforma de creación de electrónica de código libre basada en hardware y software libre, flexible y fácil de utilizar para creadores y desarrolladores. Arduino ofrece la plataforma Arduino IDE [1], entorno de programación con el que cualquier usuario puede crear aplicaciones para las placas Arduino y realizar todo tipo de utilidades.

Arduino es un proyecto y no un modelo concreto de placa, lo que quiere decir que compartiendo su diseño básico te puedes encontrar con diferentes tipos de placas.

La placa elegida para este proyecto es una placa Arduino UNO [7]caracterizada por un microcontrolador AVR ATmega16U2. Tiene 14 pines digitales los cuales se podrán configurar como pines de entrada o de salida y 6 pines analógicos únicamente de entrada aunque pueden hacer la función de I/O digital.

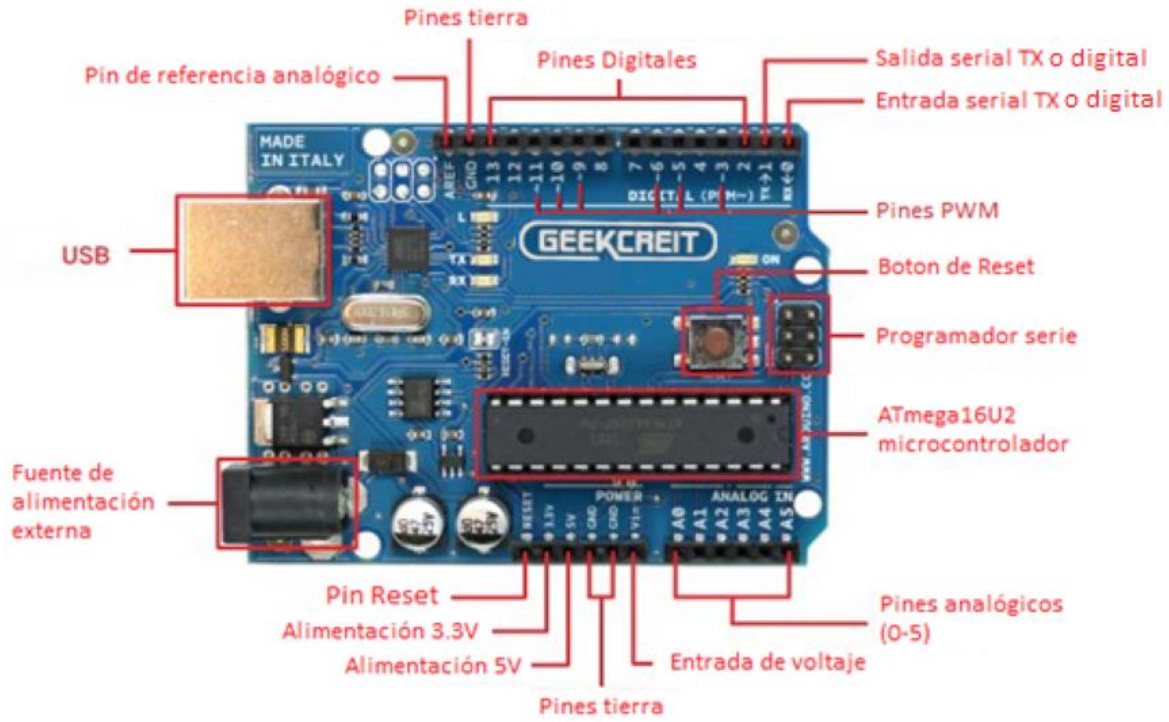


Fig. 4 Esquema de la placa Arduino

Cuando hablamos de microcontrolador [8] hablamos de un circuito integrado programable el cual nos permite ejecutar órdenes guardadas en su propia memoria. Este contiene una unidad central de procesamiento o CPU, las unidades de memoria donde almacena los registros con los cuales operaran las instrucciones en lenguaje ensamblador, en este caso tanto RAM como ROM, los puertos de entrada y salida y los periféricos.

Si hablamos específicamente del microcontrolador AVR ATmega16U2 [9] tenemos que nombrar sus principales características que lo diferencian del resto de microcontroladores. Utiliza una arquitectura AVR de 32 registros de 8 bits cada uno, una memoria flash de 16KB, 512 bytes de memoria EEPROM y 512 bytes de memoria SRAM y se caracteriza por tener un bajo consumo. Además, dispone de un oscilador interno para la sincronización de instrucciones, 22 pines de entradas y salidas, una interface UART, 2 de SPI y 5 pines PWM.

La placa del Arduino la podremos alimentar por medio de un cable USB o mediante una batería externa de 9 voltios, aunque acepta voltajes entre 7 y 20 voltios.

Dentro de los puertos de entrada y salida funcionalmente encontramos los siguientes:

- Pines digitales: Son 13 pines que sirven como E/S de un bit y nos pueden valer para leer sensores y escribir en actuadores. Seis de estos pines hacen

uso de la técnica PWM lo cual permite emular salidas analógicas. Además, presentan diversos protocolos de comunicación (I2C, SPI, UART).

- Pines analógicos: Son 6 pines, únicamente de entrada y usan un conversor analógico/digital. Sirven para leer sensores analógicos como sondas de temperatura.

- Otras entradas/salidas: Existen otros pines que nos dan diferentes funcionalidades como el USB y el programador serie.

Los pines digitales pueden servir tanto de entrada como de salida, con un uso sencillo y unos valores discretos fuera de rangos. Su lógica se basa en dos estados, alto y bajo representada alternativamente por ceros y unos con los cuales conseguiremos transiciones de nivel bajo a alto o viceversa llamados flancos. Los pines analógicos en cambio sirven como pines de entrada aunque también pueden funcionar como entradas o salidas digitales.

Además, cada uno de estos puertos tendrá a su vez tres registros; El registro DDR, determina si el pin es una entrada o una salida (1 salida, 0 entrada); el registro PORT controla si el pin está en nivel alto (1) o en nivel bajo (0) y el registro PIN permite leer el estado de un pin. (solo lectura). Cada pin puede tener múltiples funciones, como la generación de PWM, o las capacidades de ADC, los pines 6 y 7 del PORTB son los pines de entrada para el oscilador de cristal, y el pin 6 del PORTC corresponde al botón de reinicio. [10].

Para el manejo de las entradas analógicas serán necesarios los siguientes registros; ADMUX, selector del canal del multiplexor del ADC y el voltaje de referencia; ADCSRA y ADCSRB, para el control del ADC y su estado y ADCL y ADCH, para depositar el resultado al finalizar la conversión. En este proyecto, estos pines analógicos no serán utilizados ya que haciendo uso exclusivamente de los pines digitales podemos hacer funcionar todos los módulos.

3.2.b Sistema motriz

Los motores utilizados para este proyecto son dos motores con reductora controlados por medio de la placa Fundumoto, acoplada directamente a la placa Arduino, la cual permite dar independencia tanto de sentido como de velocidad a cada uno de los motores. Estos motores serán los encargados de alimentar las dos ruedas motrices del vehículo.

Estas ruedas han sido impresas mediante una impresora 3D para encajar a la perfección con los motores y tienen la altura suficiente para no rozar con ningún componente del circuito. Además, están reforzadas con dos gomas en cada rueda para ganar grip y así poder realizar mejor los diferentes tipos de movimientos.

Los motores se caracterizan por incluir varias ruedas dentadas o engranajes entre la cabeza del motor y la rueda, con las cuales se consigue un aumento considerable de la potencia perdiendo a cambio velocidad máxima lo cual permite el funcionamiento del sistema de forma segura y eficiente. Estas ruedas dentadas están acopladas al motor dentro de una unidad cerrada.

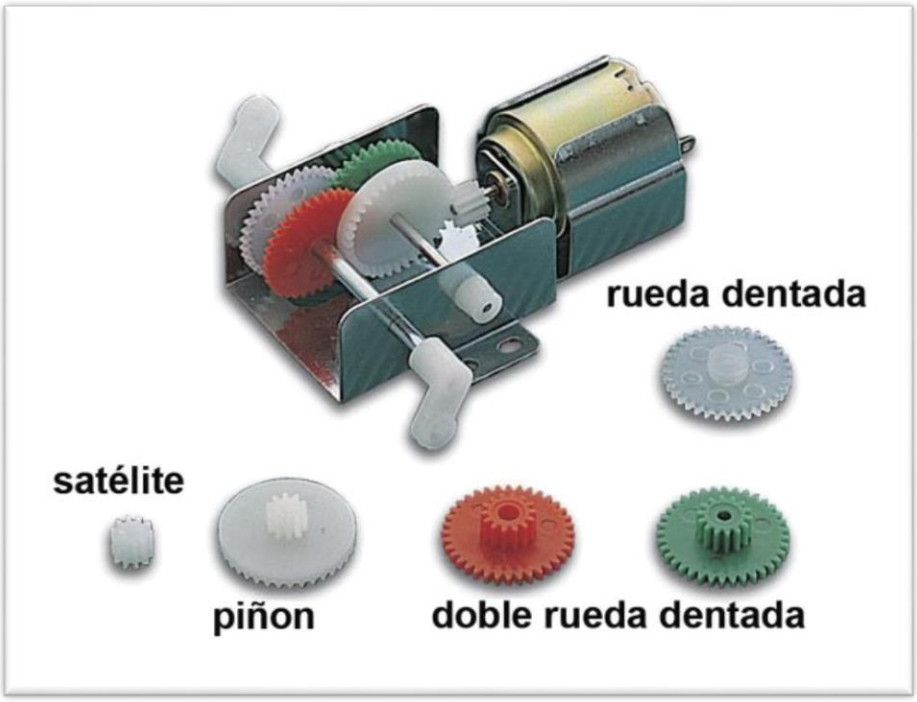


Fig. 5 Motor con reductora

Algunas de sus ventajas son una mayor eficacia para la transmisión de la potencia proporcionada por el motor, una mayor seguridad en la transmisión para reducir el coste en el mantenimiento y una reducción de los tiempos para su instalación.

No obstante, entre sus inconvenientes se encuentran su fuerte ruido cuando están en funcionamiento, su elevado coste y la necesidad de llevar a cabo un mantenimiento para conservarse en óptimas condiciones de funcionamiento.

Su Placa Fundumoto es un controlador de motores de corriente continua que utiliza un chip de motor de alta potencia, L298P, capaz de controlar 2 motores

al mismo tiempo. Al tener los mismos pines libres que la placa Arduino, ha permitido conectar los módulos que he necesitado a Arduino, cosa que otro tipo de placas no permite. La figura 7 muestra la estructura de esta placa.

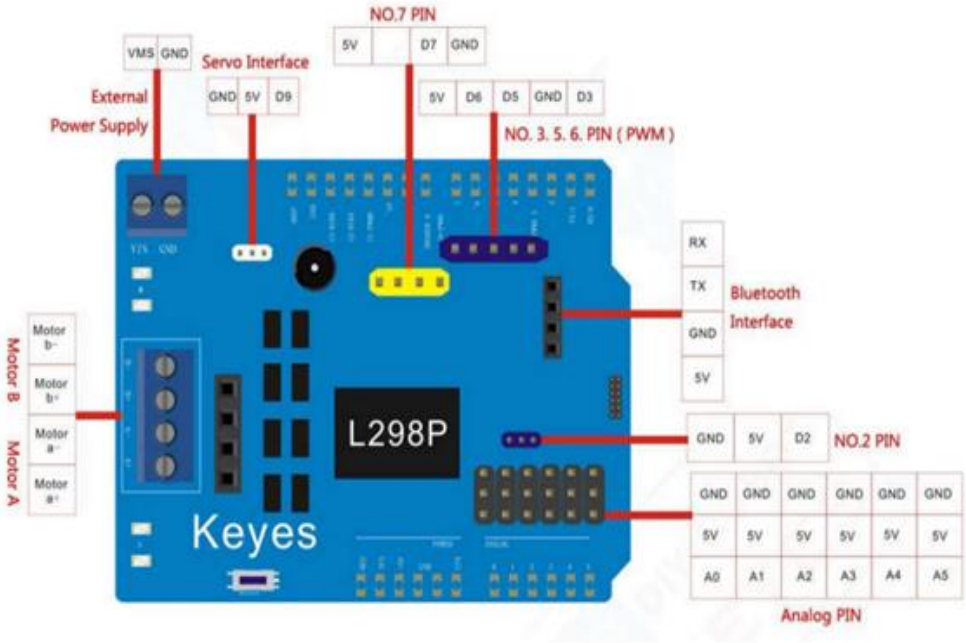


Fig. 6 Placa fundumoto

Entre sus características encontramos su chip L298P que utiliza directamente el puerto IO digital de la placa principal (D10 D11 D12 D13), un buzzer integrado en la placa y colocado en el pin D4, siete pines digitales totalmente libres de la placa principal (Pines: D2, D3, D5, D6, D7, D8 y D9) y seis pines analógicos y por último, y aunque permite una alimentación independiente, es posible realizar dicha alimentación mediante la pila de la placa Arduino a la cual está conectada.

Mediante su chip controlaremos ambos motores, así el motor A será controlado mediante los pines 12 para su dirección y 10 para la velocidad. Para controlar la velocidad es necesaria una señal analógica la cual obtenemos usando la técnica PWM, la cual se explicará más adelante. La velocidad será proporcional al nivel de tensión de la señal analógica. El motor B lo controlaremos de igual manera mediante los pines 13 y 11 [11].

Como se ha comentado anteriormente, el movimiento del vehículo se llevará a cabo por medio de sus dos ruedas motrices alimentadas por dos motores con reductora los cuales aplican la fuerza que el usuario estime a las ruedas pudiendo aplicarse de forma independiente entre estas. Esta independencia motriz permitirá, sin necesidad de cambiar el ángulo de dichas ruedas, realizar giros aplicando diferentes velocidades a cada una de estas.

Para poder mover estos motores será necesario aplicar una velocidad X y una dirección Y a cada uno de estos por sus diferentes pines de salida. El que proporciona la velocidad es un pin digital PWM lo cual permite introducir valores numéricos de entre 0 y 255 siendo 255 la velocidad máxima posible ya que no existen pines analógicos de salida. Dependiendo de que comando se elija se aplicará la misma velocidad y sentido en ambos (avanzar recto), diferentes velocidades, pero mismo sentido (giros) o diferentes sentidos, pero mismas velocidades (giro de 180 grados).

Técnica PWM

PWM son las siglas de una técnica muy utilizada en el mundo electrónico y significa: Pulse Width Modulation.

Consiste en activar una salida digital durante un tiempo y mantenerla apagada durante el resto del periodo. Como se puede ver en la figura 4, el promedio de la tensión de salida, a lo largo del tiempo, será igual al valor analógico deseado. En esta modulación se mantiene constante la frecuencia (es decir, el tiempo entre disparo de pulsos), mientras que se hace variar la anchura del pulso [7].

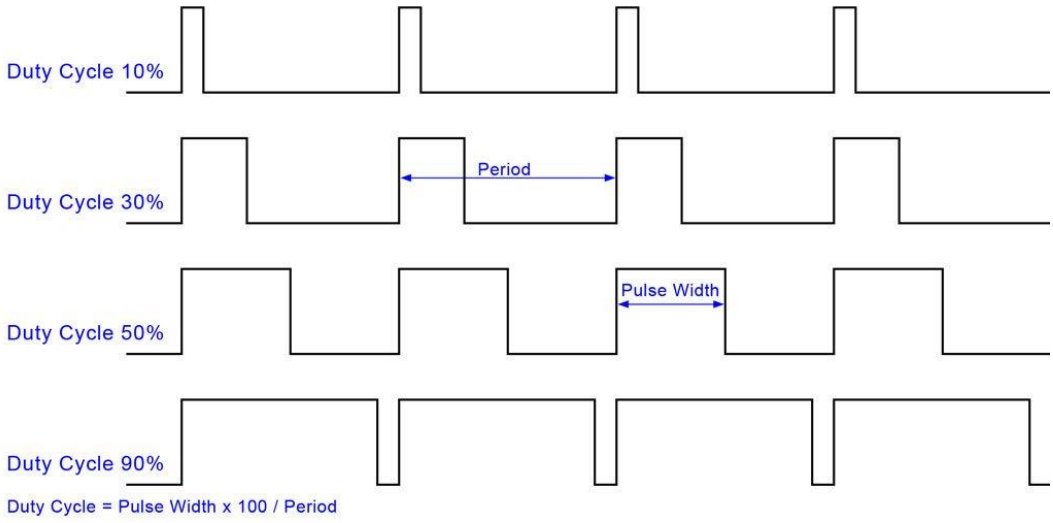


Fig. 7 Técnica PWM

Duty cycle: La proporción de tiempo que está encendida la señal, respecto al total del ciclo.

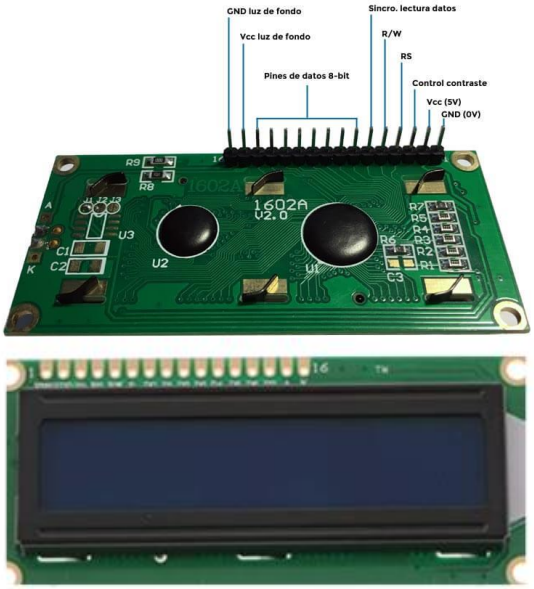
Para nuestro proyecto, disponer de 6 pines PWM como los que nos proporciona esta placa, nos permite llevar a cabo el movimiento de los motores con diferentes velocidades. Esto se conseguirá introduciendo por uno de estos pines un valor numérico (la velocidad) el cual más tarde se conseguirá enviar al motor variando el Duty Cycle como previamente se ha explicado.

De esta forma, se conseguirá aplicar diferentes velocidades en ambos motores de manera sencilla como si de un pin analógico se tratase.

3.2.c Pantalla LCD

La pantalla del vehículo servirá para estar informados en todo momento de la situación del vehículo. Esta podrá mostrar los diferentes estados, la detección de obstáculos o mensajes explicativos para informar al usuario de cómo trabajar con el sistema, así como posibles errores. Para ello se utilizará una función que permitirá mostrar mensajes por pantalla recibidos como parámetros. Se admitirán hasta 2 mensajes de hasta 16 caracteres cada uno.

Para poder informar al usuario en todo momento de la situación del vehículo, he decidido incorporar una pantalla LCD [13] la cual permite mostrar mensajes de texto en dos líneas de hasta 16 caracteres por línea. Esto es posible gracias a su luz polarizada que permite mostrar la información en pantalla a partir de una serie de filtros.



PIN	FUNCIÓN
1	GND (Tierra)
2	5 Voltios
3	Control de contraste pantalla
4	RS – Selector entre comandos y datos
5	RW – Escritura y lectura de comandos y datos
6	Sincronización de lectura de datos
7-14	Pines de datos de 8-bit
15	Alimentación luz de fondo (5V)
16	GND (Tierra) luz de fondo (0V)

Fig. 8 Pantalla LCD

No obstante, una de las mayores desventajas de este componente es que requiere un gran número de pines para hacer funcionar la pantalla LCD debido a su uso de un bus paralelo para comunicarse. Es por ello que ha sido necesario incorporar un adaptador PCF8574 que permite hacer la comunicación con la pantalla LCD por medio de dos líneas digitales a través del bus I2C.

Para ello es necesario el uso de los siguientes componentes: la propia pantalla, el controlador I2C-LCD PCF8574 y el interfaz I2C de la placa Arduino.

El controlador I2C PCF8574 [13] es un expansor de entradas y salidas digitales controlado por I2C. Su funcionamiento, al igual que cualquier dispositivo I2C, se basa en el uso de las líneas SDA, línea serie para los datos y la línea SCL para la señal del reloj. El microcontrolador es el que está configurado como maestro y él módulo PCF8574 que está conectado al bus se configura como esclavo, el maestro es el que realiza las peticiones de lectura o escritura sobre el módulo y controla la señal de reloj (SCL). La transferencia de datos puede ser inicializada solo cuando el bus no está ocupado y las tramas de datos transmitidas tendrán la siguiente estructura:

- Bit de inicio de comunicación
- Byte con la dirección del esclavo + bit de lectura/escritura.
- Datos y número de bytes de estos.
- ACK de reconocimiento para cada byte transmitido
- Bit de parada

Además, nos permite controlar el contraste de los dígitos girando el potenciómetro que se encuentra en el módulo. La luz de fondo se controla principalmente por software desde el Arduino, pero el módulo también permite desconectar el led de la luz de fondo removiendo un jumper LED.

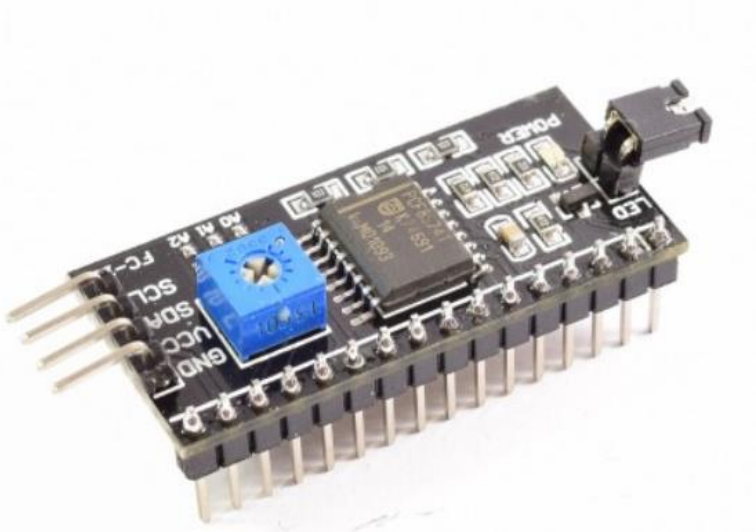


Fig. 9 I2C PCF8574

Para todo ello fue necesario localizar los pines del bus I2C en el Arduino correspondientes a los pines analógicos A4 y A5 utilizándolos como I/O digital, pudiendo utilizar también los pines SDA y SCL.

Bus I2C

El bus I2C o TWI (Two Wire Interface) es un sistema de comunicación maestro-esclavo (la transferencia de datos es siempre inicializada por un maestro; el esclavo reacciona) que requiere únicamente dos cables para su funcionamiento, uno para la señal de reloj (SCL) y otro para el envío o recepción de datos (SDA), lo cual es una ventaja frente al bus SPI pudiendo alcanzar hasta 3.4Mbps de velocidad [4].

Por contra, su funcionamiento es un poco más complejo, así como la electrónica necesaria para implementarla.

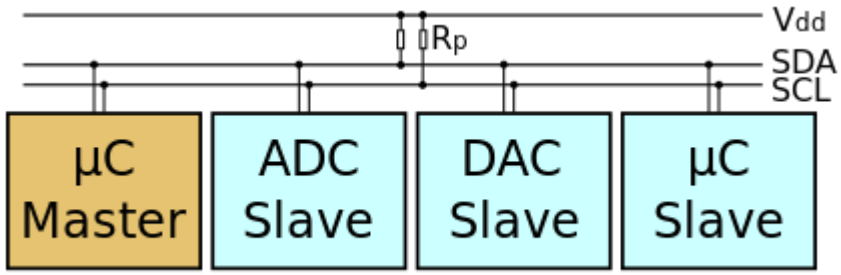


Fig. 10 Esquema I2C

Su proceso de comunicación comienza cuando el maestro envía una secuencia de bits llamada start condition la cual pone a los esclavos en espera. El maestro envía un byte con la dirección del dispositivo esclavo con el que se quiere comunicar, así como la operación a realizar, lectura o escritura y esta dirección es comparada con la de cada esclavo. Por cada byte leído o escrito el maestro debe enviar un bit de ACK para confirmar la recepción o envío de la información. Por último, al acabar la comunicación, el maestro transmite una señal de stop condition [2].

En nuestro proyecto utilizaremos el bus I2C para realizar la comunicación entre la placa y la pantalla LCD. Esta pantalla actuará como esclavo, esperando para recibir la información enviada por la señal SDA por el microcontrolador que hace de maestro a través de un pin digital. Este esclavo, como he descrito previamente recibirá byte a byte la información que se desea mostrar por pantalla enviando ACKs por medio del bus.

3.2.d LED de estado

Un led es un diodo emisor de luz y se basa en la unión de dos materiales semiconductores con dopados distintos. Tienen polaridad o lo que es lo mismo,

solo dejan pasar la corriente en un sentido compone de dos patillas, una larga que va conectada al voltaje positivo (ánodo) y una corta que va al voltaje negativo (cátodo) [3].

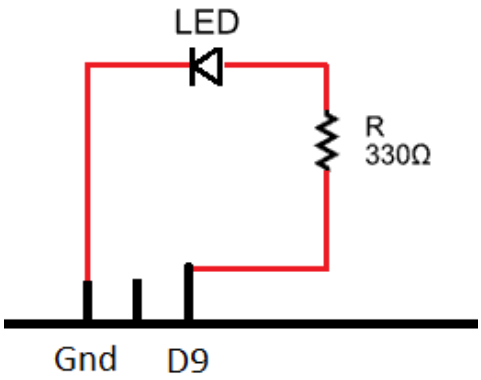


Fig. 11 Esquema bombilla LED

El funcionamiento de los leds es el más sencillo de todos. Para su correcto funcionamiento solo será necesario conectarlo a un pin digital genérico de la placa a través de una resistencia y a tierra. Encenderlo y apagarlo es posible aplicando señales de entrada HIGH y LOW en el pin al cual está conectado.

La resistencia es necesaria para controlar la intensidad que pasa por el LED y por tanto su brillo. Si pasa demasiada intensidad el LED se funde. El valor de dicha resistencia lo podremos obtener aplicando la Ley de Ohm la cual sostiene que $V=I_{nominal}*R$ siendo V la resta de la tensión de alimentación menos la tensión de polarización directa del led e I la corriente nominal del led.

En la parte superior se presenta una luz led de color rojo. Esta servirá para informar de la detección de obstáculos, de la recepción de comandos y de la finalización de la ruta, siendo complementaria a la pantalla. Además, parpadeará si existe algún fallo en el sistema. El resto del tiempo esta permanecerá apagada.

Su uso se llevará a cabo mediante una simple función que permita encender el led durante un tiempo *n* activando su puerto digital de salida correspondiente.

3.2.e Sensor de distancia

El módulo de ultrasonidos colocado en la parte frontal del dispositivo tiene como función detectar los obstáculos con los que se topa el vehículo y así poder esquivarlos evitando choques que puedan dañar la estructura.

Los módulos de ultrasonidos sirven para medir distancias enviando un pulso de alta frecuencia el cual rebota en el objeto cercano y es reflejado hacia el sensor. Para su uso solo es necesario el propio módulo HC-SR04 [14] el cual dispone de un transmisor y un receptor. Conociendo la velocidad del sonido, se medirá el tiempo entre la emisión del pulso y su recepción de su eco y estima la distancia del objeto cuya superficie impacta el impulso de ultrasonidos.

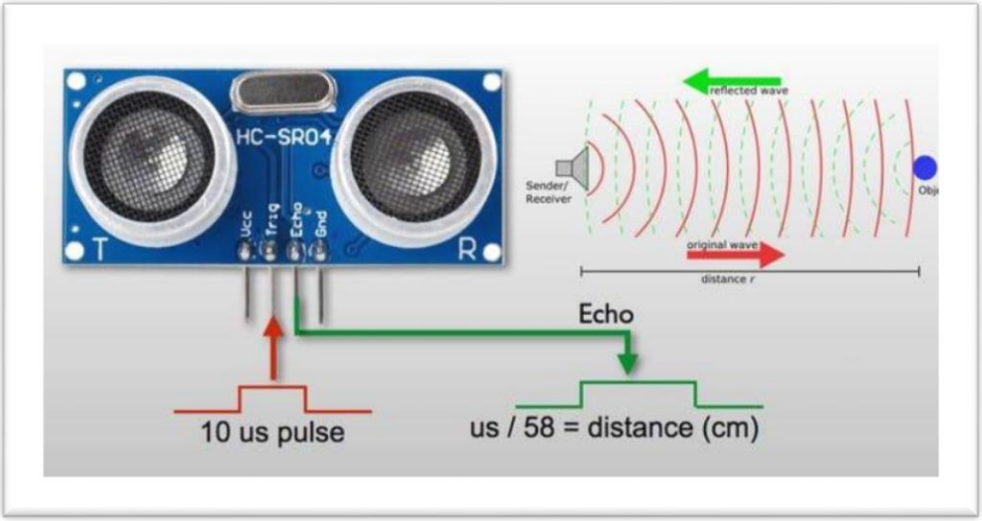


Fig. 12 Esquema sensor ultrasónico

Su funcionamiento se basa en medir el tiempo entre envío y la recepción del pulso y conociendo la velocidad del sonido en condiciones normales (343m/s) obtenemos que el sonido tarda 29,2 microsegundos en recorrer un centímetro. Por lo tanto, podemos deducir lo siguiente:

$$Distancia(cm) = \frac{Tiempo(\mu s)}{29{,}2 * 2}$$

El motivo de dividir entre dos el tiempo es porque se mide el tiempo que tarda el pulso en ir y volver, por lo que la distancia recorrida por el pulso es el doble de la que quiero medir.

Su conexión se lleva a cabo a través de dos pines digitales cualquiera, uno para el emisor del pulso y otro para para leer el tiempo del eco, el cual es un pulso de duración igual al tiempo que ha tardado la señal en ir y volver.

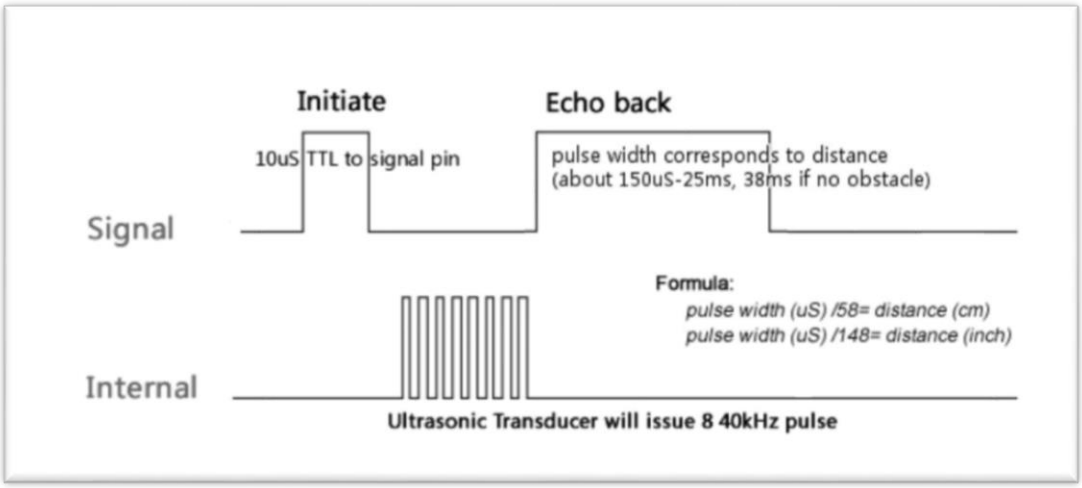


Fig. 13 Esquema de pulsos

Este módulo podrá detectar los obstáculos cuando estén a menos de dos metros, pero no se tomarán en cuenta hasta no acercarse a los 30 centímetros, momento en el cual, tras hacer una doble comprobación para evitar falsos positivos, se realizará un frenado y su consiguiente giro de 180 grados.

Para conseguir esta detección utilizaremos una función que realice dichas comprobaciones enviando un pulso por el pin de salida y esperando a recibir por el pin de entrada el pulso de respuesta para más tarde calcular la distancia en centímetros.

3.2.f Mando a distancia

Los módulos IR son utilizados para recibir señales remotas de un mando a distancia mediante una señal infrarroja la cual se emite desde el mando emisor hasta el receptor, que está colocado directamente en el circuito.

Su mando a distancia transmite un cierto mensaje al receptor empleando luz infrarroja como sistema de transmisión. La luz empleada típicamente está en el rango de 940 nm.

Como en cualquier transmisión el mensaje tiene que seguir unas determinadas normas (forma de los pulsos, duración, contenido…) que deben ser conocidas tanto por el emisor como por el receptor para que la comunicación sea correcta.

Por otro lado, el mensaje nunca se envía directamente como un pulso, sino que se envía modulada sobre una onda portadora. Esto se hace para mejorar el rechazo al ruido y a la luz ambiental la señal de control se envía modulada en una onda portadora. La frecuencia de la onda portadora depende de protocolo

empleado, pero, en general, varía entre 36-50 kHz, siendo el más habitual en torno a los 38 kHz.

Uno de los protocolos más habituales, que es el que emplearemos con Arduino, es el protocolo NEC, que emplea una onda portadora de 38 kHz y modulación por distancia de pulsos (PDM Pulse Distance Modulation). [5]

La onda portadora tiene un periodo de 26µs, y la señal transmitida distingue entre 0 y 1 por la duración de los pulsos, siendo.

- Logical 0 – Un pulso de 562.5µs seguido por un espacio de 562.5µs.
- Logical 1 – Un pulso de 562.5µs seguido por un espacio de 1.675µs.

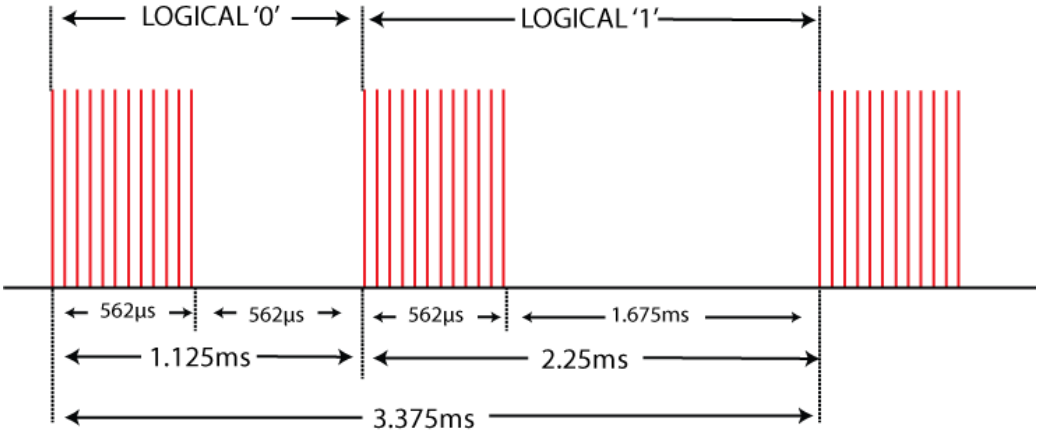


Fig. 14 Onda en el protocolo NEC

En cuanto al contenido del mensaje enviado, el protocolo NEC envía una dirección 8 bits y un comando de 8 bits. Esto significa que pueden controlarse hasta 256 dispositivos sin interferir entre ellos, y enviar hasta diferentes 256 comandos.

Una particularidad del protocolo NEC es que se envía cada bloque dos veces, una normal y una negada, como sistemas de prevención de errores. La transmisión comienza con una señal de 9ms, seguido de un espacio de 4.5ms.

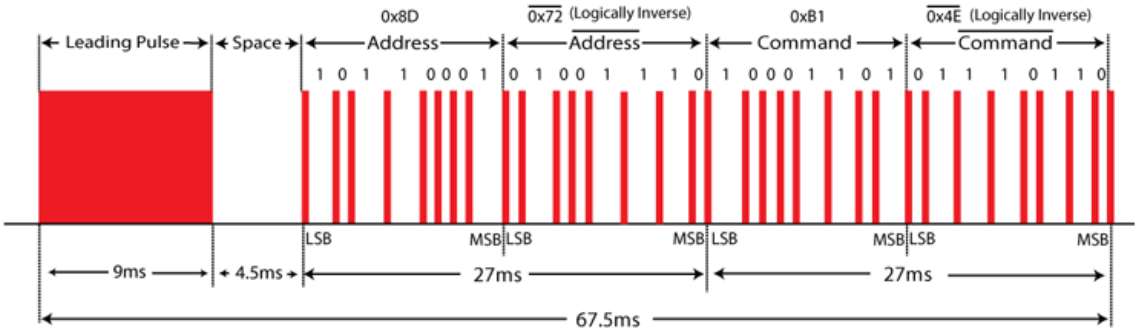


Fig. 15 Estructura bloque de datos protocolo NEC

Por tanto, la señal completa se compone de:

- Pulso 9ms + espacio 4.5ms
- 8 bits, dirección del receptor
- 8 bits, dirección del receptor invertida
- 8 bits, comando enviado
- 8 bits, comando enviado invertido

Su conexión también es aparentemente sencilla, solo necesita estar conectado a un pin digital cualquiera, por el cual recibirá los mensajes enviados por el mando.

Este módulo recibe una señal mediante un led infrarrojo de un mando para más tarde ser interpretada por el procesador como los diferentes comandos de movimiento o finalización. Cuando estos se reciban y se identifiquen, se almacenarán en la memoria del dispositivo.

Fig. 16 Mando IR

3.3 Diseño del software

El software desarrollado presenta un modelo de diseño de software en cascada lo cual conlleva un diseño secuencial. [15] En este modelo, en cada iteración de añaden nuevas funcionalidades al software, teniendo desde el inicio un sistema operativo, pero con operatividad limitada. Para ello se ha llevado a cabo una completa planificación presentando las diferentes etapas necesarias para el funcionamiento desde el punto inicial al final siendo necesario acabar cada una de estas para avanzar a la siguiente. Estas etapas son las siguientes: Planteamiento del problema, análisis de materiales y del entorno, diseño y construcción de los algoritmos unitarios, pruebas unitarias e implementación del algoritmo final como unificación de todos los elementos una vez se sabe que funcionan de forma unitaria.

El algoritmo está formado por un proceso de inicialización en primer lugar y un bucle. En cada vuelta de este se comprueba en qué modo se encuentra y se opera según el modo. Dentro del bucle encontramos varios casos o modos dependiendo de una variable que nos indica el modo en el que se encuentra el vehículo y en cada repetición de este bucle se comprobará el modo en el que se encuentra para operar en función de este. Dentro de estos modos encontramos las llamadas a las funciones y a los diferentes drivers que controlan los módulos. Entre ellas

encontramos funciones para recibir y almacenar los comandos, funciones para ejecutar los movimientos o funciones para responder a diferentes errores.

```
Rutina de inicialización {
  Inicializacion de la comunicación

  Inicialización del ultrasonidos

  Inicialización de la pantalla LCD

  Definimos el pin del led como pin de salida

  Inicialización del receptor IR

  Definimos los pines de los motores como pines de salida
}
```

```
loop {
  switch de modo{
    modo 1:
        ...
        Bucle de recepción de comandos

        Si hay error
            modo = 4
        ...
    modo 2:
        ...
        Bucle de ejecución de comandos

        Si hay obstáculos
            funcion de obstaculo
        Si hay error
            modo = 4
        ...
    modo 3:
        ...
        Finalización de comandos

        reintentar?
            modo = 1
        ...
    modo 4:
        ...
        Tratamiento de errores
        ...
    Otro modo:
      modo = 1
  }

}
```

Cada una de estas tiene un objetivo y una estructura aceptando diferentes parámetros de entrada, así como devolviendo diferentes tipos de datos según más convenga. Dentro de nuestro código estas funciones realizan tareas tales como encender la luz led, calcular la distancia del módulo ultrasónico o recibir y ejecutar los diferentes movimientos.

Para realizar estas tareas haremos uso de estructuras condicionales como el if, el if-else o el switch y bucles como los for o los while. También usaremos variables auxiliares como contadores incrementales y decrementales, y activadores o flags.

3.3.a Rutina de inicialización

En primer lugar, en la rutina de inicialización, rutina de primer nivel, se habilitan los dispositivos que requieren inicialización y que serán utilizados

posteriormente. Estos son el módulo de ultrasonidos, la pantalla LCD, la bombilla LED y el receptor IR. Algunos de ellos requerirán un modo de OUTPUT como el led, mientras que otros como el módulo de ultrasonidos necesitarán hacer uso del modo de INPUT para poder leer la información obtenida por el módulo. Esta rutina, ejecutada únicamente una vez al inicio del código, hace uso de funciones de los dispositivos.

3.3.b Funciones

Una vez habilitados los módulos del circuito e iniciado el funcionamiento del sistema, será necesario el uso de funciones para hacer funcionar todos estos módulos:

- Recepción y lectura de los comandos de movimiento: La función de recepción y lectura de los comandos (boolean recibeComandos()) rutina de primer nivel con llamadas a otras funciones para el manejo de dispositivos, es una función sin parámetros de entrada basada en un bucle donde se espera la recepción de los diferentes comandos representados como un conjunto de dígitos. Cada conjunto de dígitos tiene asignado un movimiento.

  Una vez recibido un comando, este se convertirá a un movimiento y se guardará en un array para más tarde ser ejecutado. Una vez recibido el comando de finalización se activará un flag que finalizará el bucle. En el caso de que se reciba un comando no existente, la función devolverá un error en forma de booleano que será mostrado por pantalla.

- Encendido y apagado de la luz led: Esta función (void enciende(double ms)), siendo la más básica de todas, es una rutina de último nivel correspondiendo al driver del led y tendrá como parámetro de entrada el tiempo en milisegundos el cual está encendida la bombilla led. Esta función será utilizada para avisar de un obstáculo, indicar marcha atrás o avisar de la finalización de la cadena de comandos una vez ejecutados.

- Cálculo de la distancia a un obstáculo: La función long distancia(), driver del dispositivo y rutina de último nivel, es una función sin parámetros de entrada que devuelve en forma de número en coma flotante con doble precisión la distancia en centímetros al obstáculo más cercano.

  Una vez enviado el trigger o pulso de voltaje de 10 microsegundos a través del transmisor, esperaremos a recibir dicho pulso rebotado en el obstáculo por el receptor. La distancia al objeto la obtendremos multiplicando el

tiempo entre emisión y recepción por la velocidad del sonido entre dos, ya que, si no haríamos esta última división, estaríamos obteniendo la distancia del transmisor al objeto + la del objeto al receptor.

- Salida de un mensaje por pantalla: Esta función (void mensaje(String a, String b) rutina de segundo nivel es un driver de dispositivo con dos parámetros de entrada que hace uso de los drivers del I2c y la pantalla. Es la encargada de mostrar por la pantalla LCD todos los mensajes informativos del sistema. Tiene dos parámetros de entrada para distinguir el texto a mostrar en la primera línea y el de la segunda línea y para ello será necesario colocar, como si de un array se tratase, el cursor en el punto inicial para comenzar a escribir. Aunque este cursor pueda colocarse en un punto diferente al inicial, en este programa solo se colocará al principio de cada línea para escribir con el mayor espacio posible.

- Aviso de un obstáculo: Apoyándose en la función distancia(), la función void obstáculo(), rutina de segundo nivel, muestra por pantalla el aviso de que se ha encontrado un obstáculo, enciende los leds, da media vuelta al vehículo, indica el número de obstáculos encontrados por pantalla y continua la marcha.

- Retomar marcha: Esta función de primer nivel (boolean retomar()), llamada una vez se ha finalizado la ejecución de todos los comandos, tiene como labor pedir al usuario que se pulse el botón “0” para retomar la marcha si este lo creyese conveniente. Consta de un bucle sencillo donde se espera la recepción de comandos y se comprueba que se ha pulsado el botón correcto.

- Ejecución de comandos de movimiento: La función boolean ejecuta(int x) de primer nivel, recibe como parámetro uno de los movimientos deseados para ser ejecutados y en función de cuál sea, lleva a cabo uno de los movimientos, pudiendo ser, marcha adelante, marcha atrás, giro izquierdo, giro derecha y media vuelta. Cabe destacar que cuando se ejecute el movimiento de marcha atrás y el de media vuelta, el led se encenderá. Al acabar cada uno de los movimientos, se incluirá una pequeña pausa para comprobar si existe algún obstáculo. Pausa apenas perceptible pero necesaria para el correcto funcionamiento global.

  Si el comando recibido como parámetro es un comando inválido, se enviará un “false” activando así un error, el cual se mostrará por pantalla para avisar del evento al usuario.

3.3.c Bucle

El bucle del programa es la parte más importante de este. Constituye una secuencia que se ejecuta repetidas veces para realizar las diferentes acciones tomando diferentes caminos en función de los comandos introducidos o de las circunstancias. Desde este, se llama a las diferentes funciones para hacer funcionar el circuito de manera ordenada.

En primer lugar, si la bandera o flag de modo está activada en el modo inicial o de recepción, representado con el número uno, se muestra un mensaje por pantalla animando al usuario a introducir los comandos deseados e inmediatamente, manteniendo el mensaje en pantalla, se llama a la rutina de recepción de comandos para guardarlos en una lista. Una vez comprobado que la recepción ha sido satisfactoria, se muestra otro mensaje para informar de dicho evento y se cambia la bandera a modo ejecución de comandos. Si este proceso no se completaría correctamente, la bandera no cambiará y se repetiría el proceso en la siguiente iteración del bucle.

Una vez almacenados los movimientos en la primera fase y con el flag en modo ejecución de comandos representado con el número dos, llamaremos tantas veces como comandos hallan en la lista a la función que tras leer un comando concreto lleva a cabo su movimiento. Mientras estos movimientos se llevan a cabo, se llamará repetidamente a la función que calcula la distancia a un obstáculo para comprobar que se puede realizar dicho movimiento sin tener un accidente. Esta llamada se hará antes, durante y después y si se detectase cualquier objeto, se cambiaría la flag a modo emergencia o modo número tres.

En el caso de entrar al modo de emergencia, se mostraría de nuevo un mensaje por pantalla avisando de la situación, se llamaría a la función de encendido del led y se procedería a continuación a realizar un giro de 180 grados para continuar con la ejecución de los movimientos, cambiando de nuevo la flag al modo ejecución de comandos.

Una vez finalizada la lista de comandos, el flag pasará a modo fin o modo tres, se mostrará por pantalla un mensaje de fin, así como las instrucciones para retomar el proceso desde el punto inicial y se activará de forma intermitente el led para avisar del final. Si el usuario sigue las instrucciones para volver a empezar el proceso, la flag volverá a pasar a modo inicial.

3.3.d Comportamiento ante obstáculos

Como he comentado anteriormente, el programa está preparado para actuar en consecuencia a un obstáculo encontrado durante su ejecución. El uso del módulo de ultrasonidos sirve para detectar estos obstáculos y tras comprobar dos veces que efectivamente impiden realizar los movimientos, para asegurarnos de que no sea un falso positivo, el programa avisa de este evento.

Si esto pasase el dispositivo está preparado para poder realizar un giro de 180 grados y continuar su rutina de movimientos, entendiendo que, en su parte trasera, que es el área desde el que se aproxima al obstáculo, está libre para poder continuar. No obstante, previo a continuar la marcha, se vuelve a comprobar si existe un nuevo obstáculo, en cuyo caso el dispositivo se encontraría en una zona donde no sería posible continuar y se avisaría del error.

3.3.e Tratamiento de errores

A modo de captura de excepciones el programa incorpora dentro del bucle un cuarto modo donde en caso de error del sistema se notifique del error y se avise del modo en el que se detectó este. Es por ello que todas las funciones del programa, en caso de no obtener un resultado esperado, podrán activar este modo de error.

Esto irá acompañado de la activación del led de forma fija, así como de las instrucciones para continuar el proceso en el punto donde se quedó la ejecución o al principio, lo cual se indicará mediante el mando infrarrojo.

4 - Evaluación y pruebas

Para hacer posible el funcionamiento de todos los módulos conectados y su funcionamiento dependiente entre ellos, se ha llevado a cabo una serie de pruebas unitarias para conocer en profundidad como actúan ante diferentes eventos, como van conectados y cuáles son sus limitaciones.

4.1 Pruebas individuales de funcionamiento de cada módulo

Las pruebas realizadas con los diferentes módulos ordenadas de menor a mayor dificultad han sido las siguientes:

- Bombilla led: Conectando únicamente el led a la placa se ha probado a apagar y encender este, cada 2 segundos. Al ser el elemento más sencillo, ha servido además como indicador para otras pruebas unitarias.

- Pantalla LCD por I2C: Conectando únicamente el módulo I2C de la pantalla a la placa se han llevado a cabo pruebas encendiendo y apagando la pantalla, intentando mostrar caracteres especiales y jugando con las 2 líneas de la pantalla.

- Motores: Conectando estos a la placa Fundumoto se han realizado varias pruebas para comprobar su funcionamiento.

  En primer lugar, se han probado los motores sin acoplarse al vehículo, para así probar su funcionamiento con diferentes velocidades y direcciones incorporando una rueda en cada uno para poder ver con mayor facilidad estas características.

  Una vez comprobado su correcto funcionamiento en solitario, se han añadido al vehículo para así comprobar cómo trabajan estas cuando tienen que mover un peso. Para ello, ha sido necesaria una calibración jugando con las velocidades para conseguir la misma velocidad que al principio una vez tenemos el peso del vehículo sobre estas. Al hacer esto sus velocidades se reducen y su tiempo de frenado aumenta. En último lugar se han realizado varias pruebas para calcular las velocidades y tiempos que se deben aplicar para conseguir movimientos precisos y poder llevar a cabo la lista al completo sin giros demasiado pronunciados, movimientos que requieran demasiado espacio, o excesos de velocidad.

- Ultrasonidos: Conectado a la placa Arduino, junto con la bombilla led, se ha llevado a cabo una prueba en la que comprobando periódicamente la distancia cada medio segundo, cuando el módulo se acercaba a un obstáculo a menos de 30 centímetros, la bombilla led se encendía. Además, esta distancia se mostraba por pantalla para comprobar la exactitud de la distancia medida por el sensor.

- Receptor IR: Este módulo ha sido el que más dificultad ha traído a la hora de recoger y procesar los códigos enviados por el mando a distancia. Para sus pruebas unitarias, se conectó a la placa Arduino directamente y una vez se enviaba un código desde el mando, el receptor leía los datos que recibía por el puerto serie comprobando que dichos códigos coincidían con la tecla correspondiente. Inicialmente, para conocer los códigos de cada botón, se ha pulsado repetidamente cada uno de ellos por separado para cerciorarse de su correspondencia.

4.2 Pruebas de funcionamiento global

Una vez realizadas las pruebas de los módulos uno a uno se ha procedido a unificar todos los elementos y hacerlos funcionar conjuntamente. Únicamente se hablará de aquellos módulos cuyo funcionamiento ha cambiado notablemente desde sus pruebas unitarias:

- Pantalla LCD por I2C: Tras probar sencillos mensajes en las pruebas unitarias, se ha procedido a mostrar mensajes con una frecuencia más alta y utilizando y adecuando variables para mostrar al usuario, como el número de obstáculos encontrados o el comando introducido. Ha sido necesario limpiar a menudo la pantalla ya que arrastraba caracteres de mensajes anteriores y hacia incomprensible la lectura de mensajes.

- Motores: Una vez comprobado su correcto funcionamiento en solitario, se han añadido al vehículo para así comprobar cómo trabajan estas cuando tienen que mover un peso. Al hacer esto sus velocidades se reducen y su tiempo de frenado aumenta. En último lugar se han realizado varias pruebas para calcular las velocidades y tiempos que se deben aplicar para conseguir movimientos precisos y poder llevar a cabo la lista al completo sin giros demasiado pronunciados, movimientos que requieran demasiado espacio, o excesos de velocidad.

- Receptor IR: Una vez obtenidos los códigos de los botones necesarios para mover el vehículo, se ha incorporado el módulo al vehículo donde

comparando los códigos se han guardado los números equivalentes a los movimientos que los motores deberán realizar.

Una vez concluidas las pruebas de los módulos ya incorporados al vehículo, se han realizado pruebas con diferente número de comandos en primer lugar sin obstáculos y más tarde aumentando el número de obstáculos progresivamente. Además, también se han probado los casos negativos para probar los errores que impiden continuar la trayectoria.

5 – Conclusiones y trabajos futuros

Como conclusión del proyecto podemos destacar los nuevos conocimientos aprendidos sobre la plataforma Arduino, las limitaciones que esta tiene y todas las posibilidades que esta ofrece. Entre las virtudes de las placas Arduino destacamos la posibilidad de hacer funcionar muchos módulos al tiempo con una alimentación bastante sencilla, y su tamaño y peso los cuales ofrecen grandes posibilidades a la hora de construir nuevos proyectos. Por otro lado, algunas de sus limitaciones son el no poder hacer uso de puertos analógicos de salida teniendo que usar puertos digitales PWM o su escasa memoria lo cual no permite crear grandes programas ni almacenar grandes cantidades de datos. Se han aprendido nuevos conocimientos de electrónica, necesarios para la construcción del proyecto, así como para la resolución de ciertos errores que impedían hacer funcionar el circuito y se han reforzado todos los conocimientos previos de programación aprendidos en la carrera además de un aprendizaje de nuevas funciones propias solo de la plataforma Arduino.

Respecto al resultado del proyecto, podemos deducir que se ha conseguido el propósito inicial a pesar de que tanto el diseño como sus componentes han ido variando a medida que se observaban las limitaciones del vehículo con el entorno. No obstante, estos cambios han ayudado a aprender el funcionamiento de un mayor número de componentes y de cómo aspectos físicos que a simple vista no se tienen en cuenta, como puede ser el espacio necesario para realizar un giro, limitado en gran parte por la separación de las ruedas y por el sistema de dirección elegido. Es por ello que, tras estos cambios, se ha obtenido un resultado esperado y correcto que cumple su función esperada.

Una vez finalizado el proyecto podemos destacar que existen cambios que hubiesen hecho mucho más sencillo la creación de este y que si este se volviese a realizar desde cero se cambiarían, como puede ser un diferente sistema de control para el usuario, debido a que este mando IR en específico, trae consigo limitaciones importantes como las posibles pérdidas de datos, las recepciones de datos erróneas o las pulsaciones prolongadas que no corresponden con el código de una pulsación corta para un mismo botón. Esto viene dado por que los mandos utilizados para realizar proyectos de Arduino presentan unos materiales baratos con un corto alcance los cuales requieren solucionar ciertos problemas de software. No obstante, este cambio se puede llevar a cabo en una posible mejora del vehículo.

Como posibles trabajos futuros se han planteado nuevas opciones como el seguimiento de líneas, añadiendo un sensor de infrarrojos; el modo automático como si de una aspiradora autónoma se tratase pudiendo generar un mapa virtual

del espacio recorrido almacenando la posición de los obstáculos; nuevos modos de control, ya sea a través de mandos físicos incorporados en la estructura del vehículo o aplicaciones WIFI o bluetooth; la detección de agujeros o escalones para evitar caídas en áreas con diferentes alturas; una nueva programación de comandos donde se pueda introducir comandos más complejos en los que no solo se hable de dirección y sentido sino de distancia a recorrer o nuevas funcionalidades tales como sensores de vuelco o control por voz.

6 – Bibliografía

[1] Crespo, E. (2018). *Video. Preparación IDE Arduino para ESP8266* .
http://www.aprendiendoarduino.com/tag/ide/ Fecha de la consulta: Mayo de 2020

[2] García, A. *Protocolo I2C y Arduino.*
https://www.digilogic.es/protocolo-i2c-arduino-hmc5883l/ Fecha de la consulta: Febrero de 2020

[3] Llamas, L. (2015). *Encender un led con Arduino*
https://www.luisllamas.es/encender-un-led-con-arduino/#:~:text=Un%20LED%20es%20un%20diodo,entre%20dispositivos%20el%C3%A9ctricos%20y%20electr%C3%B3nicos. Fecha de la consulta: Marzo de 2020

[4] Llamas, Luis (2016) *El bus I2C en Arduino*
https://www.luisllamas.es/arduino-i2c/ Fecha de la consulta: Febrero de 2020

[5] Llamas, L. (2016). *Controlar un Arduino con un mando a distancia infrarrojo*
https://www.luisllamas.es/arduino-mando-a-distancia-infrarrojo/ Fecha de la consulta: Abril de 2020

[6] Savvides, L. (2018). *Conoce a Vector, el minirobot casero de Anki que tiene una gran personalidad* .
https://www.cnet.com/es/noticias/conoce-a-vector-el-mini-robot-casero-de-anki-con-una-gran- personalidad/
Fecha de la consulta: Abril de 2020

[7] Yúbal, FM. (2018). *Qué es Arduino, cómo funciona y qué puedes hacer con uno*
https://www.xataka.com/basics/que-arduino-como-funciona-que-puedes-hacer-uno Fecha de la consulta: Abril de 2020

[8] Colaboradores de Wikipedia. *Microcontrolador*
https://es.wikipedia.org/wiki/Microcontrolador Fecha de la consulta: Abril de 2020

[9] *Microchip ATmega16U2*
https://www.microchip.com/wwwproducts/en/ATmega16u2 Fecha de la consulta: Abril de 2020

[10] *Puertos digitales Arduino avanzado*
https://aprendiendoarduino.wordpress.com/2017/09/03/puertos-digitales-arduino-avanzado/ Fecha de la consulta: Abril de 2020

[11] *Entradas y Salidas Analógicas Arduino. PWM.*
https://aprendiendoarduino.wordpress.com/category/pwm/ Fecha de la consulta: Enero de 2020

[12] *Fundumoto. Un motor shield para Arduino.*
https://brunetica.wordpress.com/2017/02/21/fundumoto-una-motor-shield-para-arduino/ Fecha de la consulta: Marzo de 2020

[13] *Tutorial LCD con I2C, controla un LCD con solo dos pines*
https://www.naylampmechatronics.com/blog/35_Tutorial--LCD-con-I2C-controla-un-LCD-con-so.html Fecha de la consulta: Marzo de 2020

[14] *Tutorial de Arduino y sensor ultrasónico HC-SR04*
https://naylampmechatronics.com/blog/10_Tutorial-de-Arduino-y-sensor-ultras%C3%B3nico-HC-S.html
Fecha de la consulta: Abril de 2020

[15] *El modelo en cascada: desarrollo secuencial de software*
https://www.ionos.es/digitalguide/paginas-web/desarrollo-web/el-modelo-en-cascada/ Fecha de la consulta: Junio de 2020