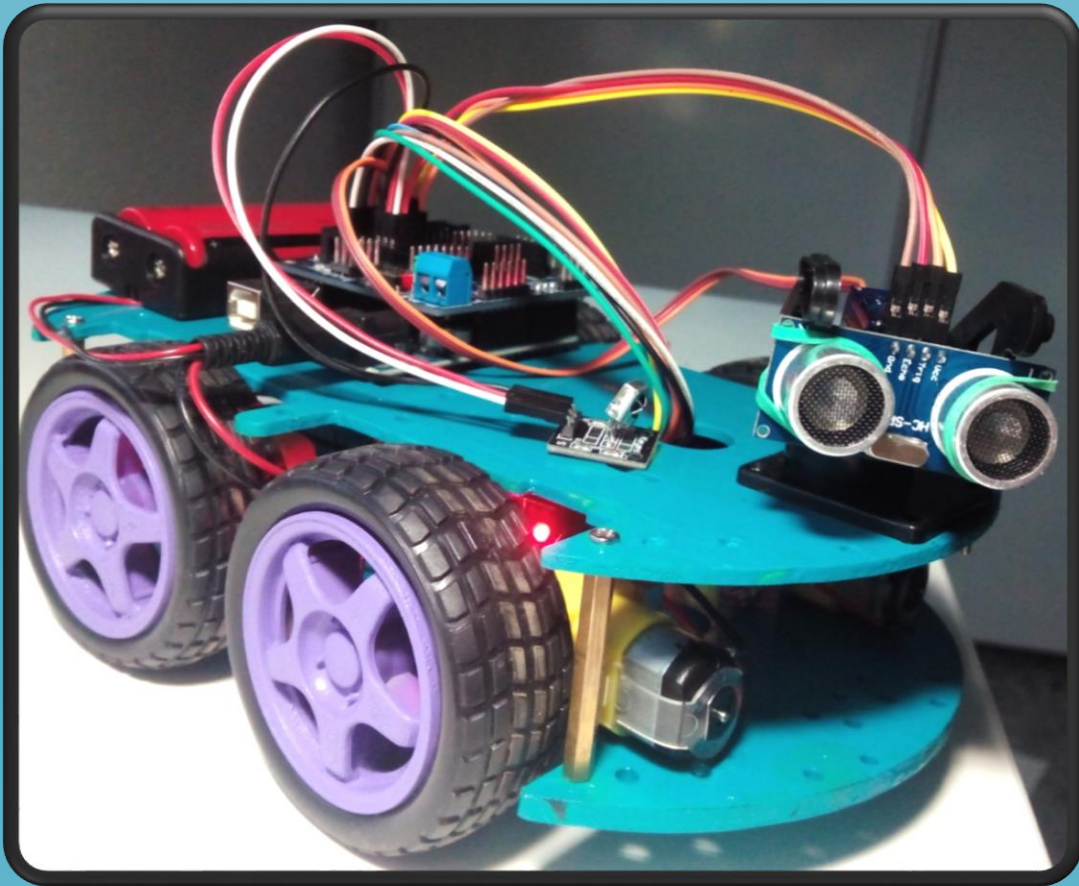


ROBOT PROGRAMADO CON ARDUINO



Autor: David Sánchez Cozar

Tutor: Pau Cedrón

Centro: Escola Goar

Curso: 2016/2017

ÍNDICE

1. AGRADECIMIENTOS	3
2. INTRODUCCIÓN.....	4
3. PLANIFICACIÓN	6
4. ¿QUÉ ES LA PROGRAMACIÓN? ¿QUÉ ES ARDUINO?	8
5. COMPONENTES	9
5.1. Placa Arduino UNO.....	9
5.2. Sensor shield	11
5.3. Driver	12
5.4. Motores.....	13
5.5. Chasis.....	14
5.6. Ruedas	14
5.7. Batería	15
5.8. Sensor ultrasónico	16
5.9. Servomotor	18
5.10. Control remoto	20
6. CONSTRUCCIÓN DEL ROBOT	21
7. PROGRAMACIÓN	26
8. PRESUPUESTO	32
9. PROBLEMAS	33
10. CONCLUSIONES.....	34
11. BIBLIOGRAFÍA.....	36
12. BIBLIOGRAFÍA FOTOGRÁFICA	38
13. ANEXOS	40

1. AGRADECIMIENTOS

Esta parte es la más importante de todo mi proyecto porque si no hubiera sido por todas las personas que me han ayudado, aconsejado, animado y estado a mi lado en todos estos momentos de dedicación, no hubiera sido posible poder realizar este trabajo.

El primero de todos a los que quiero agradecer es a mi tutor Pau Cedrón porque ha estado siempre conmigo, dándome información con diferentes páginas webs y aconsejándome en todo momento tanto en el trabajo teórico como en el práctico para llevarlo a buen fin.

Después a mi padre que siempre me ha animado a que desarrollara esta idea y me ha dado su punto de vista sobre la importancia que tiene un proyecto de segundo de Bachillerato. Me gustaría agradecer a mi hermano y a mi madre por darme siempre su apoyo y por su colaboración económica ya que el presupuesto no estaba al alcance de mi bolsillo. En definitiva, me gustaría darle las gracias a todo aquel que ha hecho posible que yo pudiera conseguir que este trabajo fuera una realidad.

2. INTRODUCCIÓN

Mi trabajo de investigación trata sobre la robótica y la programación con Arduino. La motivación para realizar este trabajo fue gracias a mi interés por la tecnología, más concretamente, por la programación. Me llama mucho la atención la parte en que puedes programar un robot para que haga la función que quieras y cuando quieras.

No tenía claro qué trabajo quería hacer, pero gracias a las guías que me dio mi tutor comentándome que consultara diferentes links relacionados con la robótica, me decidí a crear un robot comprando un kit en Internet. Como era de esperar, esta idea inicial no tenía ningún sentido, ya que iba a coger unas piezas que me venían en un pack y montarlas sin ningún tipo de aliciente y ya tendría el robot construido. Mi tutor ya me comentó que esto no iba a valorarse con buena nota, así que finalmente me decanté por crear un robot programado con Arduino, con la diferencia de que en vez de venirme todas las piezas juntas, investigaré por Internet qué piezas debería utilizar para poder llevarlo a cabo. Otro punto positivo que me comentó mi tutor de elaborar un robot por mi cuenta es la motivación, ya que voy a ser yo mismo el que me desafíe a conseguir que mi robot funcione y a investigar cómo poder mejorarlo. Gracias a esto, al final del trabajo, mis conocimientos sobre la robótica y la programación serán mucho más amplios y habré mejorado de una forma sustancial.

Para poder realizar este robot, tuve que invertir muchas horas de estudio y dedicación en verano porque los conceptos sobre robótica que yo tenía en ese momento eran muy básicos. Como todo en la vida, el principio es lo más difícil. La programación fue la parte más complicada de todas, ya que comenzaba de cero, pero a base de ilusión y ganas de aprender fui capaz de realizarla, de lo cual me siento muy orgulloso.

Mi trabajo está distribuido en tres partes:

- La descripción de cada uno de los accesorios que he utilizado, donde pongo las principales características y cómo funcionan. Esta sería la parte teórica.
- El montaje de mi robot, donde explico detalladamente cómo he creado el robot. Esta sería la parte práctica.

- La creación de mi programa que he subido a la placa Arduino UNO.

Aparte de esta división, también realicé un video para la plataforma YouTube, de cómo construí mi robot porque, de esta forma, se ve claro todos los pasos que he seguido. Una de las experiencias más entretenidas ha sido esta grabación porque me ha hecho ver que para que un video en YouTube o en cualquier otra plataforma esté bien hecho, hay que dedicarle mucho tiempo, ya sea grabando el video o editándolo para que el sonido y la imagen sean de calidad.

Los objetivos que me he marcado para desarrollar este proyecto son:

1. Aprender a programar con Arduino.
 2. Aumentar mis conocimientos sobre la robótica.
 3. Hacer que funcione correctamente el robot.
-
1. Quería aprender a programar la placa Arduino UNO poco a poco, empezando desde lo más básico encendiendo un LED hasta algo más complicado como controlarlo con control remoto.
 2. La robótica es un tema bastante famoso y ya tenía algunos conocimientos básicos, pero gracias a este proyecto los he mejorado considerablemente.
 3. El objetivo final era hacer funcionar el robot e intentar mejorarlo lo máximo posible.

3. PLANIFICACIÓN

Una de las partes más importantes de mi trabajo fue la planificación. En mi opinión, para que un trabajo se pueda realizar correctamente debe haber una planificación detrás para situar en el tiempo las diferentes tareas y así tenerlo todo más controlado.

Mi proyecto está dividido en seis fases en que las tres primeras duran tres meses y las otras tres seis meses aproximadamente:

- En la primera fase, el proyecto se divide en las partes principales que compondrán el trabajo: introducción, objetivos, conclusiones, etc.
- En la segunda fase, se completa la parte de objetivos y justificación.
- En la tercera fase, se elabora una planificación de las tareas futuras, un cronograma y un presupuesto del robot.
- En la cuarta fase, se desarrolla la parte escrita y la parte práctica y se envía al tutor. Esta parte es la más importante ya que ocupa tres meses y se sitúa en verano, es decir, que hay mucho tiempo libre y hay que aprovecharlo para intentar acabar el trabajo.
- En la quinta fase, se corrigen todos los errores comentados por el tutor y se muestra cómo funciona el robot.
- En la sexta fase, se acaba de maquetar todo el trabajo y se hace referencia al canal de YouTube que se ha creado. Finalmente, se envía todo el proyecto en PDF al tutor con una semana de antelación y se corrigen todos los errores que haya marcado.

También llevé a cabo un cronograma para situar las tareas en el tiempo. Este cronograma me ha servido mucho a la hora de hacer el proyecto ya que llevaba todo el trabajo bien controlado, sabiendo que labor tenía que hacer en cada momento.

PLANIFICACIÓN DE TAREAS A REALIZAR							
	ACCIONES	FECHA PREVISTA	TIEMPO EMPLEADO	FORMACIÓN	LUGAR DE COMPRA	AYUDA EXTERNA	COSTE ECONÓMICO
1	Decisión/elección sobre el proyecto a realizar	16 de abril de 2016	2 horas	Familia y tutor	/	Internet	/
2	Trabajo en conjunto con el tutor	Proceso continuo	1 hora por semana	/	/	/	/
3	Información sobre robótica	Junio-Julio-Agosto	2 horas por semana	Personal	/	Internet	/
4	Hacer un índice de cómo estará dividido mi trabajo	1 de septiembre de 2016	2 horas	Personal y tutor	/	Internet	/
5	Investigar cómo debo empezar a hacer el robot	3 de junio de 2016	4 horas	Ayuda del tutor	/	Internet	/
6	Información de cómo localizar piezas y herramientas	Junio-Julio-Agosto	4 horas	Personal y familia	Internet y tienda	Internet	/
7	Comprar las piezas necesarias	Junio-Julio-Agosto	3 Horas	Personal y ayuda del tutor	Internet y tienda	Internet	100 euros
8	Trastear e ir probando las piezas en el robot	Julio-Agosto-Septiembre	1 hora por día	Personal y ayuda del tutor	/	Internet	/
9	Formación sobre soldadura	Agosto	3 horas	Personal y familia	Leroy Merlin	Internet	20 euros
10	Formación sobre programación	Julio-Agosto	5 horas por semana	Personal	/	Internet	/
11	Formación sobre control remoto	Agosto-Septiembre	2 horas por semana	Personal	Internet	Internet	10 euros
12	Formación sobre compatibilidad con Android	Septiembre	1 hora por semana	Personal	Internet	Internet	5 euros
13	Montaje de mi robot en vídeo	10 de septiembre de 2016	8 horas	Personal	/	Internet	/
14	Presentar a mi tutor mi parte escrita	Proceso continuo	1 hora	Ayuda del tutor	/	/	/
15	Corregir errores que me ha explicado el tutor	Proceso continuo	3 horas	Ayuda del tutor	/	/	/
16	Hacer un dossier con la parte escrita	5 de diciembre de 2016	2 horas por semana	/	Copistería	/	10 euros
17	Realizar una presentación de prueba	25 de diciembre de 2016	3 horas	Personal y ayuda del tutor	/	/	/
18	Precio total del trabajo	/	/	/	/	/	145 euros
19	Trabajo de investigación (parte escrita)	20 de noviembre de 2016	3 horas por semana	Ayuda del tutor	/	Internet	10 euros
20	Trabajo de investigación (parte práctica)	15 de septiembre de 2016	1 hora por día	Personal y ayuda del tutor	/	Internet	135 euros

Imagen 1: Cronograma.

4. ¿QUÉ ES LA PROGRAMACIÓN? ¿QUÉ ES ARDUINO?

La programación es el proceso por el cual una persona desarrolla un programa valiéndose de una herramienta que le permita escribir el código (el cual puede estar en uno o varios lenguajes, como Java o Arduino, en este caso) y de otra que sea capaz de “traducirlo” a lo que se conoce como lenguaje de máquina que puede ser entendido por un microprocesador.

Este último paso se conoce como compilación y es necesario para que el código pueda ser ejecutado por la plataforma para la cual haya sido creado.

De una forma más técnica, la programación se realiza mediante el uso de algoritmos, que son secuencias finas y ordenadas de instrucciones que deben seguirse para realizar una acción.

Arduino es una plataforma de programación de código abierto basado en hardware y software flexibles y fáciles de usar. Puede haber diferentes entradas que son las que reciben la información del exterior como un sensor ultrasónico, y las salidas que son las que transmiten el mensaje que llevan a través de la programación como un motor.

Escogí esta opción de programación por las siguientes características:

- **El precio:** las placas Arduino son relativamente baratas comparadas con otras plataformas microcontroladoras.
- **Multiplataforma:** el software de Arduino se ejecuta en diferentes sistemas operativos como Windows y Linux.
- **Código abierto y software extensible:** el software de Arduino está publicado como herramientas de código abierto, disponible para poder hacer extensiones por programadores experimentados.

5. COMPONENTES

En este apartado se habla de los componentes del coche: cómo son, qué función tienen en el robot y de qué se componen.

Los componentes son: la placa Arduino UNO, que es la clave en todo el robot ya que es dónde se compila la programación, la Sensor Shield, que es dónde se pone todo el cableado, el driver, que sirve para dar dirección a los motores, los motores, que sirven para dar movimiento a las ruedas, el chasis, para tener bien colocados los accesorios, las ruedas, para que el robot pueda desplazarse, la batería, para dar energía al robot, el sensor ultrasónico, para detectar los obstáculos, el servomotor, que sirve para hacer girar el sensor ultrasónico y finalmente el control remoto, para controlarlo a distancia.

5.1. Placa Arduino UNO

¿Qué es?

Se trata de un microcontrolador, una placa, un pequeño sistema de procesamiento.

Partes

- **Pines de entrada y salida digitales:** los pines enumerados del 2 al 13 en la parte superior derecha son entradas/salidas digitales. Esto significa que podemos mandar que salga voltaje o que estén apagados si los hemos definido por el programa como salida (OUTPUT) para el caso de que por ejemplo queramos encender un LED. Por otro lado los podemos definir como entrada (INPUT), con lo cual podremos consultar si estos han recibido o no cierto voltaje.

En los números 3, 5, 9, 10 y 11 junto al número se encuentra este símbolo “~”, lo cual indica que, a diferencia de los demás pines, estos son capaces de producir una salida moduladora de ancho de pulso (PWM). La diferencia entre ambos se encuentra en que un pin sin salida modulada puede enviar o no una corriente de 5 voltios, pero los que pueden ondular la corriente, en lugar de mandar una corriente continua,

en su lugar mandan pulsos eléctricos de corta duración seguido de otro corto momento en que está apagada la corriente.

- **Botón reset:** permite reiniciar la tarjeta, lo cual permite volver a leer el programa desde la primera secuencia.
- **Entrada USB:** es la conexión entre el ordenador y placa Arduino UNO. Es utilizado para cargar los programas al Arduino, además de que puede ser utilizado en otros casos como entrada de alimentación para la placa ya que proporciona un voltaje de 5 volts.
- **Microcontrolador:** es un circuito integrado digital que puede ser usado para diversos propósitos debido a que es programable. Está compuesto por una unidad central de proceso, memorias y líneas de entrada y salida.
- **Entrada de alimentación externa:** acepta un voltaje entre 7 y 12 volts. La ventaja de Arduino es que debido a su voltaje y entradas puede trabajar con baterías y no forzosamente debe estar conectado a una corriente eléctrica. En mi caso yo uso una batería con dos pilas de 3,7 volts cada una.
- **Pines de entradas y salidas analógicas:** mientras que las entradas/salidas digitales están a 5 voltios o a cero voltios, es decir, que solo tienen estas dos opciones que pueden interpretarse como apagado/encendido, las entradas o salidas analógicas aceptan valores intermedios, es por esto que son las entradas adecuadas para leer las medidas dadas por un instrumento analógico, como un sensor ultrasónico, que mide la distancia a la cual tiene un objeto. En cuanto a su trabajo como salidas son ciertamente similares a las salidas PWM, pero en este caso controlan el voltaje y no la duración del impulso.

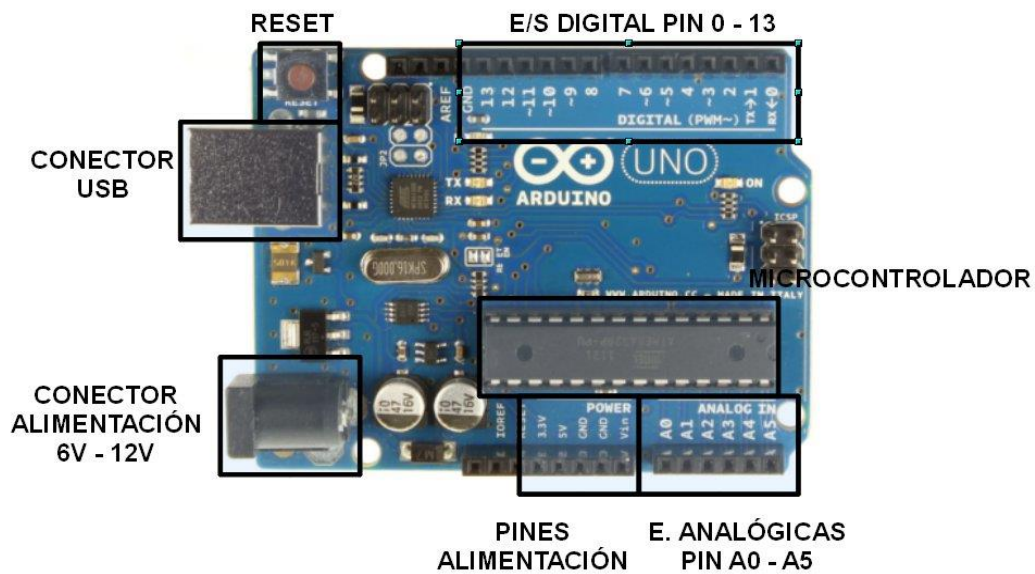


Imagen 2: Placa Arduino UNO.

¿Cómo hacer uso de Arduino?

1. **Programación:** el Arduino debe ser programado dependiendo de las tareas que se le van a encargar.
2. **Verificación del programa:** el programa te permite comprobar que todo está bien y si hay errores te los señala para que puedas modificarlos, de lo contrario no te dejará pasar el programa completamente a la placa.
3. **Cargar el programa al Arduino:** una vez se ha verificado que el programa está correcto, este se compila, es decir, pasa de ser un lenguaje de código humano a un lenguaje de máquina para que la placa Arduino UNO pueda interpretarlo. Una vez compilado el programa, se conecta la placa, vía USB, para poder enviarle el archivo y este lo almacena en su procesador para que finalmente, una vez cargado, lo ejecutemos en el proyecto que estemos realizando.

5.2. Sensor shield

¿Qué es?

Es una placa cuyo objetivo es que sea más cómodo conectar los cables y los dispositivos a los pines de Arduino correctos. No es un dispositivo activo, simplemente conecta los pines del Arduino a muchos conectores que están

listos para usar y conectar varios artilugios como servomotores o sensores ultrasónicos.

Características

- Controlador de secuencia de interfaz analógica y digital.
- Interfaz de comunicación del módulo Bluetooth.
- Interfaz APC220 WIRELESS.
- LCD (Liquid Cristal Display) serie y paralelo.
- Interfaz sensor de ultrasonidos.
- 6 entradas analógicas.
- Interfaz controladora de 32 servos.

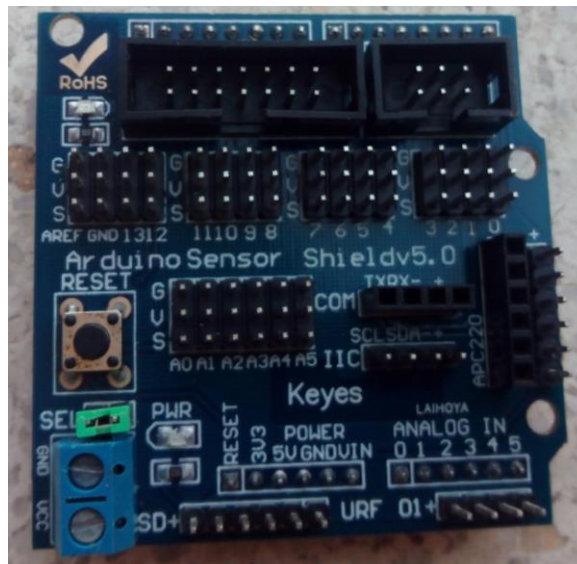


Imagen 3: Sensor Shield.

5.3. Driver

¿Qué es y qué función tiene?

Este módulo permite controlar cuatro motores de corriente continua de hasta 2 amperios.

Cuenta con jumpers de selección para habilitar cada una de las salidas del módulo (A y B). La salida A esta conformada por IN1 y IN2 y la salida B por IN3 y IN4. Los pines de habilitación son ENA y ENB respectivamente.

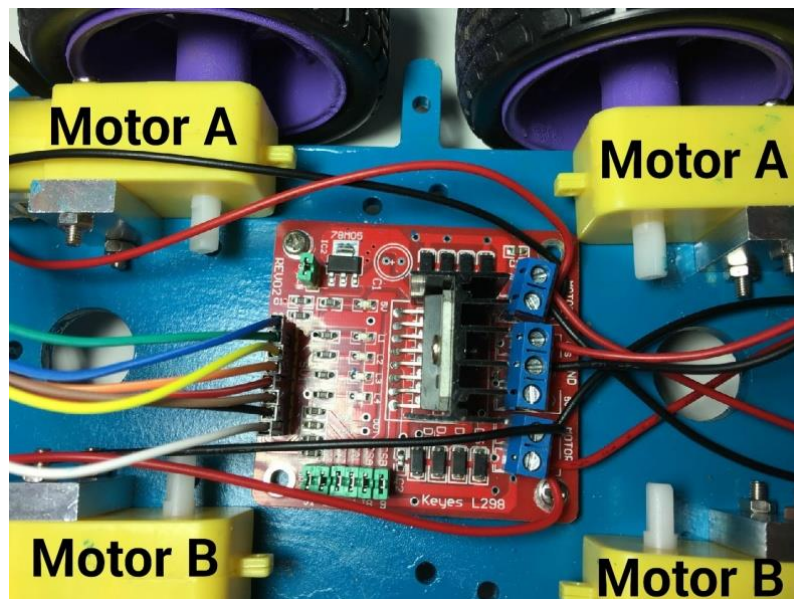


Imagen 4: Driver.

5.4. Motores

¿Qué es?

Un motor es la parte del robot que es capaz de hacer funcionar el sistema, transformando energía eléctrica, que es la que proporciona la batería, en energía mecánica para realizar un trabajo. En este caso, el motor recibe la energía eléctrica a través de los cables los soldados que hacen la conexión con el driver.



Imagen 5: Motor con sus cables soldados.

Tipos de motores

Existen diversos tipos de motores, pero se dividen en dos grandes grupos: motores térmicos, que son máquinas térmicas que transforman el calor en trabajo mecánico, y motores eléctricos, que transforman la energía eléctrica en energía mecánica.

Características del motor

- Tensión: 3V - 6v
- Corriente: 100 mA - 120 mA
- Rpm (con neumáticos): 100 - 240
- Peso del motor: 29 gr (cada uno)
- Tamaño del motor: 70 mm x 22 mm x 18 mm

5.5. Chasis

¿Qué es?

Es el elemento estructural, encargado de soportar los esfuerzos estáticos y dinámicos que tiene el vehículo.

Características

- Un mismo tipo de chasis puede adaptarse a varios tipos de carrocería.
- Un mismo tipo de chasis puede alargarse o cortarse.
- Es totalmente duro y rígido.
- Suele estar construido con diferentes materiales, dependiendo de la rigidez, costo y forma necesaria. Los más habituales son de acero pero el mío está hecho de vidrio acrílico.

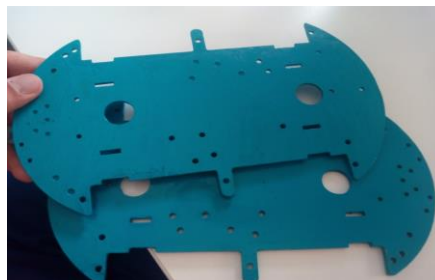


Imagen 6: Chasis del robot.

5.6. Ruedas

Una rueda es un objeto mecánico que tiene forma de disco y que se instala en un eje para que gire a su alrededor. En la construcción de mi robot, utilicé cuatro ruedas, las cuales van conjuntas a los motores ya que solamente hay este tipo para estos motores.

Detalles

- Diámetro de las llantas: 55 mm
- Diámetro del neumático: 26 mm
- Peso: 10 gr
- Abertura del centro: 5 mm x 3 mm



Imagen 7: Ruedas.

5.7. Batería

¿Qué es una batería?

Una batería es un dispositivo que permite producir energía eléctrica.

Todas las baterías poseen dos terminales. Una de ellas suele estar marcada con un signo positivo “+” mientras que la otra tiene un signo negativo “-”. Al hacer la conexión entre las dos terminales, los electrones fluyen. En este caso conectaremos el motor a la batería.

¿Qué pilas he utilizado?

Las pilas correspondientes de este robot son las ultrafire de 3,7 volts cada una. He usado esta batería porque me permite conectar dos pilas y se enlaza directamente al driver a través de los cables.



Imagen 8: Batería y pilas.

5.8. Sensor ultrasónico

¿Qué son y qué función tienen?

Los sensores ultrasónicos son accesorios que detectan objetos a distancias que van desde pocos centímetros hasta varios metros. Estos sensores funcionan de forma muy parecida a los murciélagos; emite un sonido y mide el tiempo que la señal tarda en regresar. Estos reflejan en un objeto y el sensor recibe el eco producido y lo convierte en señales eléctricas, las cuales son elaboradas. Estos sensores trabajan solamente en el aire, y pueden detectar objetos con diferentes formas, diferentes colores, superficies y de diferentes materiales. Los materiales pueden ser sólidos, líquidos o polvorientos, sin embargo han de ser deflectores de sonido. Los sensores trabajan según el tiempo de transcurso del eco, es decir, se valora la distancia temporal entre el impulso de emisión y el impulso del eco.



Imagen 9: Sensor ultrasónico.

Ventajas y desventajas

- Funcionan en presencia de polvo, humo o vapor.
- No se ven afectados por el color ni translucidez.
- Alcance sobresaliente.
- Poco desgaste.
- No hay necesidad de contacto.
- La precisión disminuye según el material y la forma del objeto.
- No pueden ser utilizados en el vacío.
- Pueden sufrir de interferencias de otros dispositivos que trabajen a la misma frecuencia.

Principio de funcionamiento

El sensor tiene un disco piezoeléctrico montado en una superficie que produce ondas de sonido de alta frecuencia. Cuando los pulsos transmitidos pegan con un objeto reflector de sonido produce un eco, así la duración del pulso reflejado es evaluado en el transductor. Cuando el objetivo entra dentro del rango de operación, la salida del interruptor cambia de estado, cuando sale del rango, la salida regresa a su estado original.

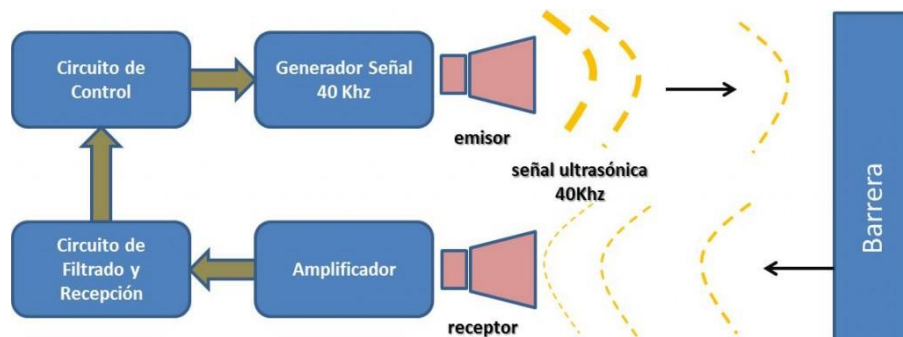


Imagen 10: Funcionamiento del sensor.

Partes

1. Cabeza sensora.
2. Transductor o acondicionador.
3. Circuito de procesamiento.
4. Salida.

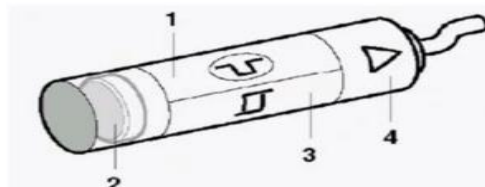


Imagen 11: Partes del sensor.

5.9. Servomotor

¿Qué es y cuál he usado?

Los servomotores son un tipo de motor que tienen la ventaja de que se puede controlar su posición y su velocidad a través de la programación.

Yo he utilizado el servomotor “tower pro sg90” porque es bastante pequeño y muy fácil de manejar. La razón por la que he decidido usar un servomotor, es para hacer un barrido con el sensor ultrasónico ya que, gracias a este artilugio, el sensor ultrasónico tiene una vista de 180°.

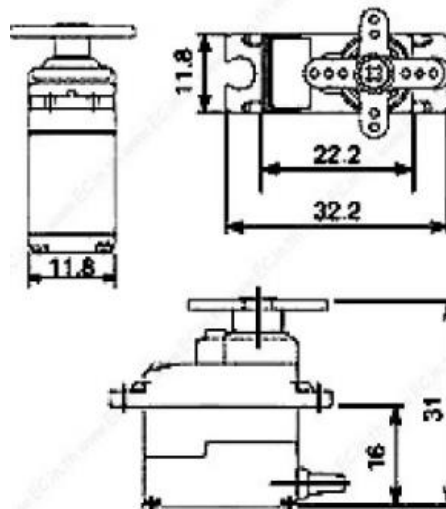


Imagen 12: Medidas del servomotor (mm).

Características

- **Peso:** 9 gr.
- **Dimensiones:** 22.2 mm x 11.8 mm x 31 mm
- **Velocidad:** 0,1 seg/60°
- **Voltaje de funcionamiento:** 3.0 v-7.2 v
- **Ángulo de rotación:** 180°
- **Longitud de cable:** 24,5 cm



Imagen 13: Servomotor.

Partes

- **Carcasa:** está formado por una carcasa que lo protege de cualquier golpe.
- **Motor de corriente continua:** es el componente esencial que hará mover los engranajes. Tiene un motor de corriente continua de poco

voltaje y, si debe soportar mucha carga, ya no se mueve porque tiene baja potencia de trabajo, es decir, tiene un par motor muy bajo.

- **Caja reductora:** en la caja reductora, los servomotores tienen una serie de engranajes que tienen como función reducir la velocidad y aumentar la fuerza con la que se mueve.
- **Sistema de control:** el control de los servomotores se basa en el sistema de modulación por ancho de pulsos o PWM. El ancho nos indica el ángulo de posición, cuanto más ancho sea, más milisegundos durará, por tanto será mayor y cuanto menos ancho sea, menos milisegundos durará y en este caso, el ángulo será menor. Los servomotores tienen un amplificador de error que lo que hace es calcular el error de posición que es la diferencia de posición real del motor y la deseada. Para conseguir que el error sea cero, el motor deberá continuar girando hasta alcanzar la posición deseada y una vez lo consiga, dejará de girar. Si queremos que el motor no sea movido por ninguna fuerza externa de esta posición, deberemos continuar enviándole información para mantener su posición. A continuación hay una imagen que muestra cómo van variando los ángulos según el ancho de los pulsos. Se puede ver que con un milisegundo el ancho del pulso es muy pequeño y consecuentemente el ángulo es cero. El valor de los pulsos del servo oscila entre 1ms y 2ms.

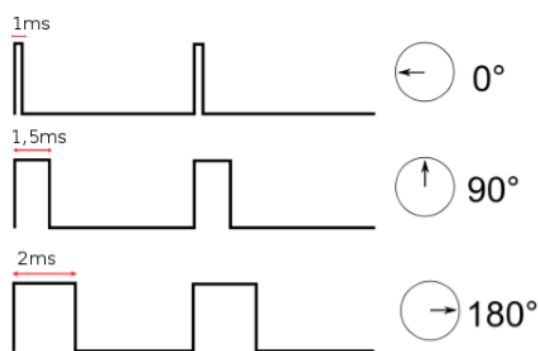


Imagen 14: Funcionamiento del ancho de pulso.

5.10. Control remoto

¿Qué es y cómo funciona?

Un control remoto es un dispositivo electrónico usado para realizar una operación remota sobre una máquina.

Yo utilicé un mando que hará de transmisor y un módulo receptor de infrarrojos que actuará como un receptor. Al presionar un botón del mando, se envía una señal de luces infrarrojas al receptor, es por esto que si hay entre medio un objeto que no deja pasar esta señal, el módulo no recibe ninguna señal.



Imagen 15: Mando

El control remoto, por dentro, tiene un chip o un microcontrolador que al presionar un botón, sabe qué señales debe enviar. El chip, a diferencia de un mando de la televisión, no está programado. Nosotros programamos el módulo receptor de infrarrojos para que, según la señal que nos envíe el transmisor, el robot haga una acción u otra.

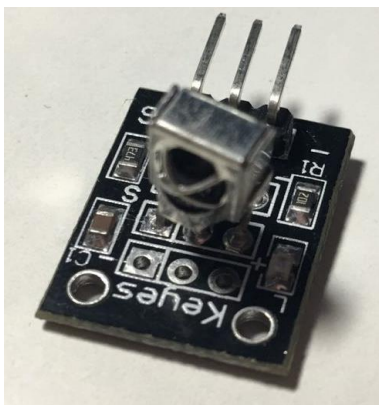


Imagen 16: Módulo receptor.

Una de las dudas más curiosas es la siguiente: ¿cómo puede ser que el control remoto emita una luz y nosotros no podamos verla? Esta pregunta se responde con la física. La física cuántica dice que todo lo que podemos ver en el universo es vibración, es decir, que todo, incluidos nosotros, somos ondas. Lo que hace que cada cuerpo tenga diferente velocidad en la que vibra una y otra. Y esto pasa con la luz. El ojo humano tan solo detecta ciertas frecuencias de luz.

6. CONSTRUCCIÓN DEL ROBOT

Paso 1. En primer lugar, comencé atornillando la placa Arduino UNO en medio y la batería en la parte de atrás del chasis. Las puse bien juntas porque la batería debía hacer la conexión con la placa Arduino UNO y los cables no eran de una gran longitud. También añadí una plataforma que consistía en aguantar el servomotor, aunque esto lo comentaré detalladamente más adelante.

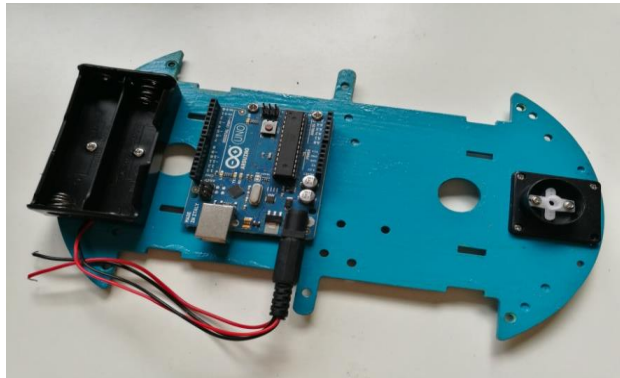


Imagen 17: Accesorios atornillados.

A continuación, acoplé los cuatro motores de corriente continua al chasis de abajo. Estos motores venían ya con sus cables soldados pero, al ir pasando el tiempo, se fueron despegando uno a uno, así que tuve que ir a comprar un soldador y estaño y soldarlos personalmente.

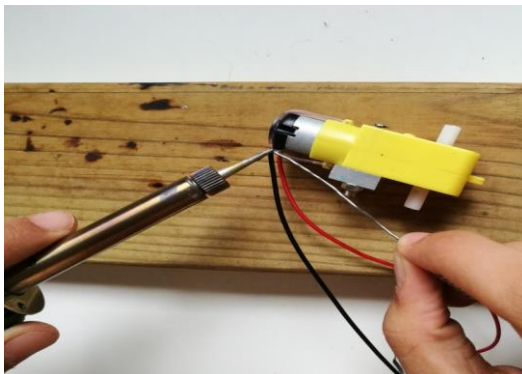


Imagen 18: Soldadura.

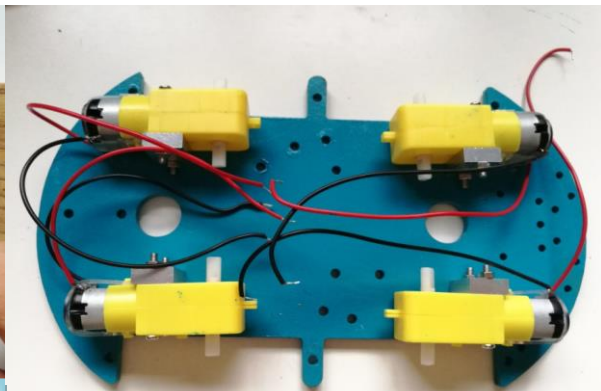


Imagen 19: Motores atornillados.

Paso 2. En segundo lugar, coloqué el driver entre medio de los motores. Las conexiones entre el driver y los motores se hicieron de la siguiente forma: Los motores que están en la parte de abajo se llaman "B" y los motores de arriba "A". Para hacer correctamente las conexiones, investigué cómo funcionaba el driver con estos motores y llegué a la conclusión de que los cables rojos hacían la conexión en los extremos y los negros en el interior, cada uno en su respectiva entrada. También acoplé las ruedas a los motores.

Además, conecté los cables unifilares al driver en sus respectivos pines. No importa el color de cada cable, sino dónde lo colocamos en la Sensor shield, pero eso lo veremos más adelante.

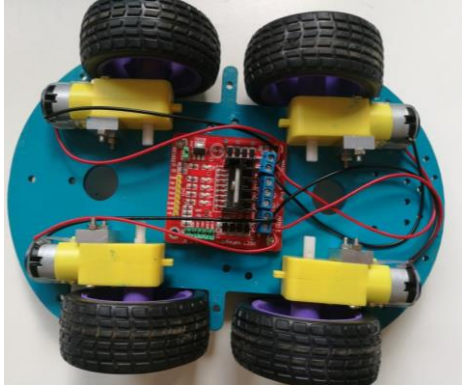


Imagen 20: Motores y driver.

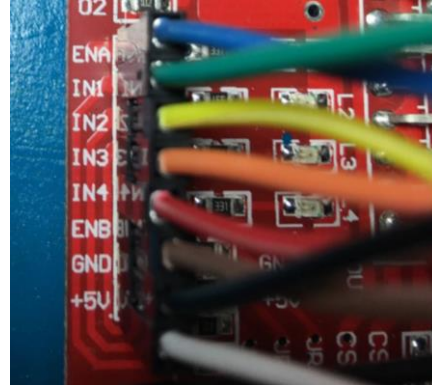


Imagen 21: Cables conectados.

Paso 3. En este paso acoplamos los dos chasis con diferentes tornillos. Entre medio de este proceso, hicimos la conexión de la batería con el driver ya que sino el robot no se movería. Los cables de la batería y los motores eran los mismos, por tanto realizaremos el mismo proceso que antes. A diferencia de la conexión de los motores con el driver, aquí los enlacé a las entradas del medio, que son GND (cable negro) y VMS (cable rojo). Finalmente, acabé de unir los dos chasis.



Imagen 22: Batería-driver.

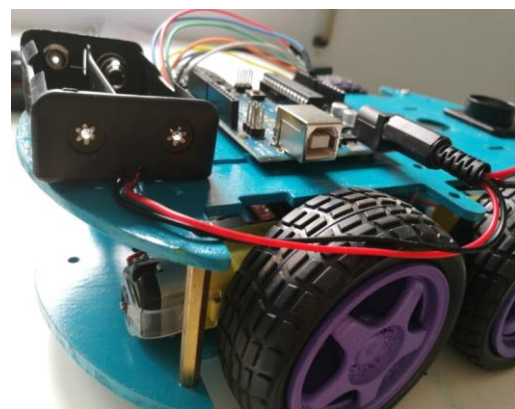


Imagen 23: Chasis unidos por los tornillos.

Paso 4. El cuarto paso se basa en juntar el servomotor y el sensor ultrasónico. Para llevar a cabo este proceso, utilicé unas placas de plástico. Para poder hacer el barrido con el sensor, necesité que este estuviera unido al servomotor y así ir girando a la par del servomotor. Estas placas me sirvieron para tener el servomotor bien protegido y además para poder mantener al sensor ultrasónico bien unido con una goma, como se puede apreciar en la foto.



Imagen 24: Placas de plástico.

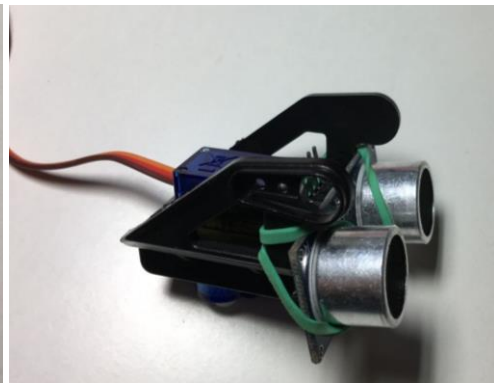


Imagen 25: Sensor ultrasónico unido al servomotor.

El uso de la plataforma que atornillamos en el primer paso es muy básico. Simplemente, colocaremos el servomotor en esa abertura y, como estará enganchado, podrá hacer su respectivo movimiento de 180°.

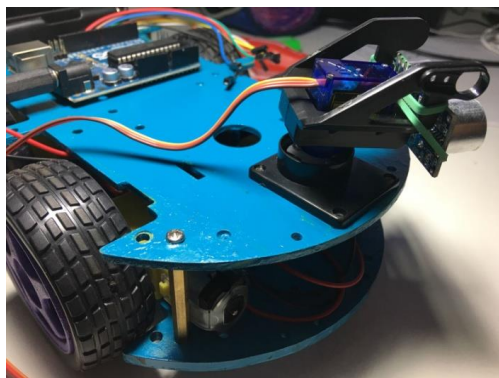


Imagen 26: Servomotor y sensor ultrasónico acoplados en el chasis.

Paso 5. En este paso realizaré las conexiones con el Arduino. En primer lugar, acoplé la sensor shield con el Arduino y después conecté el driver con la sensor shield. Enlacé los cables de la siguiente manera:

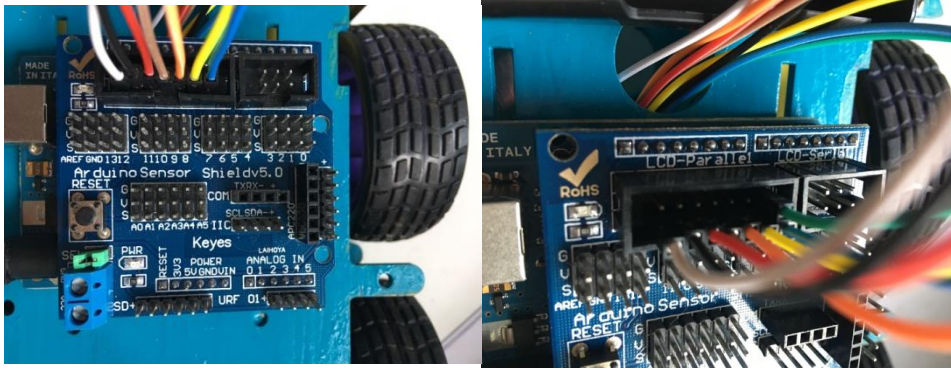


Imagen 27: Conexión entre el driver y la sensor shield.

A continuación, conecté los cables a los pines del sensor ultrasónico y los empalmé en la sensor shield. Como vemos en la sensor shield, tenemos Ground (G), Voltaje (V) y señal (S) y también los números que son los pines.

El sensor ultrasónico lo coloqué de la siguiente forma:

- Gnd lo coloqué en el pin 8 con Ground.
- Vcc lo coloqué en el pin 8 con el voltaje.
- Echo lo coloqué en el pin 10 con la señal.
- Trig lo coloqué en el pin 8 con la señal.

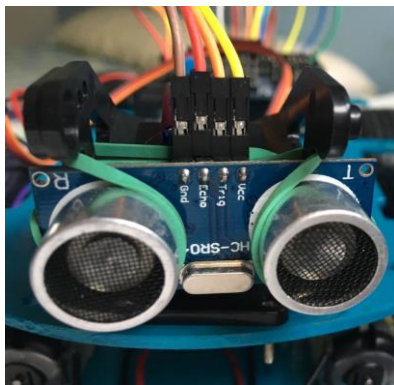


Imagen 28: Cables conectados al sensor.

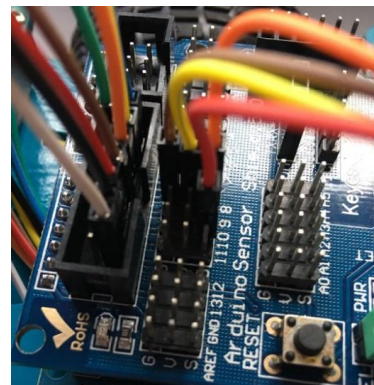


Imagen 29: Conexión ultrasónico-sensor shield.

El siguiente accesorio que conecté fue el servomotor. El cable marrón del servomotor es ground, el rojo es voltaje y el naranja es señal y los enlacé en el pin 9.

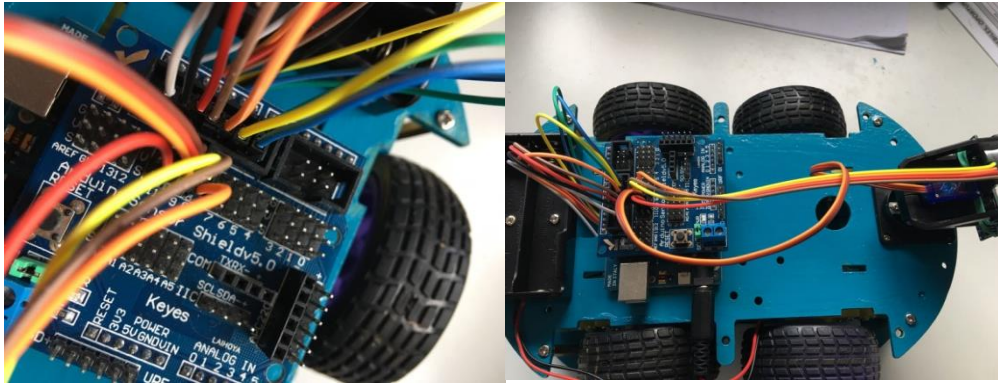


Imagen 30: Conexión entre el servomotor y la sensor shield.

El último componente que añadí fue el módulo receptor de infrarrojos. Investigando por Internet llegué a la conclusión de que el pin del cable negro es ground, el rojo es voltaje y el blanco es señal y lo acoplé en el pin 11. Finalmente conecté las pilas. Para comprobar si he hecho bien las conexiones entre el driver, la batería y la sensor shield, se debe encender una luz roja en el driver como vemos en la foto.

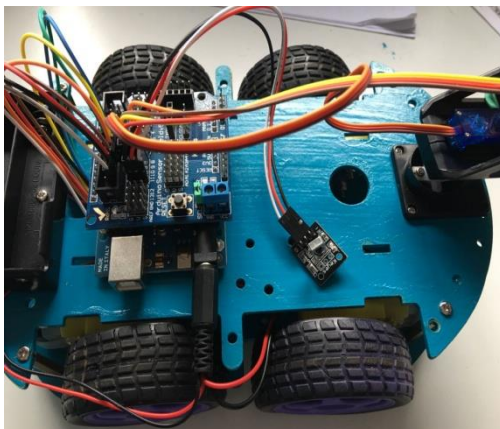


Imagen 31: Conexión módulo-sensor Shield.

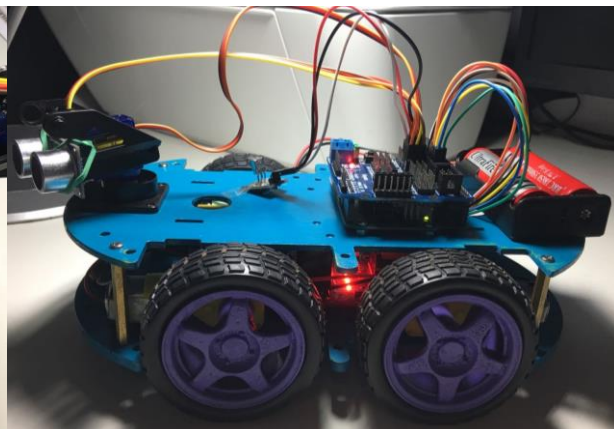


Imagen 32: Robot finalizado.

7. PROGRAMACIÓN

La programación con Arduino hace referencia a la información que contiene la placa Arduino UNO. Esta, se ha hecho con el programa "Arduino" que te permite programar la placa a tu gusto ya que hay miles de códigos y librerías.

Sin duda, esta ha sido la parte más difícil del trabajo, ya que mis conocimientos sobre programación eran muy básicos. Otra razón de la dificultad de este apartado es la de poner correctamente cada letra o símbolo ya que si alguno no es correcto, el programa no funcionará, por eso hay que estar concentrado en todo momento para saber qué se está haciendo.

¿Qué son las librerías?

Una librería es un fichero de código que se instala en el programa y se utiliza para llevar a cabo la programación, con la finalidad de que realice una tarea concreta.

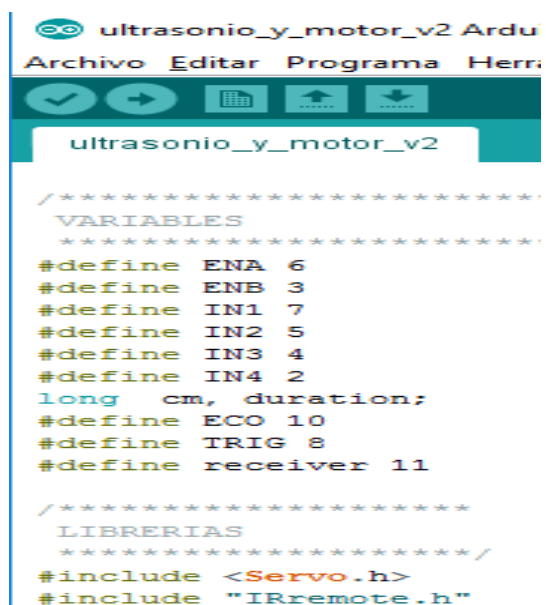
Existen multitud de librerías que permiten dotar a Arduino de nuevas funcionalidades, como por ejemplo crear un GPS, enviar mensajes SMS o manejar pantallas LCD.

Se pueden crear librerías para no reescribir el código o bien usar otras creadas por terceras personas, que es lo que yo he hecho.

7.1. Programación con Arduino

Paso 1. Para comenzar, puse las variables que iba a utilizar en el programa y las librerías que iba a necesitar de los distintos accesorios. Las primeras seis variables son las del driver que usaré más adelante para cambiar de dirección el robot. Las otras cuatro están relacionadas con el control del sensor ultrasónico.

Las librerías que utilicé son las del servomotor y la del control remoto.



```
ultrasonio_y_motor_v2 Ardu
Archivo Editar Programa Herr
ultrasonio_y_motor_v2

/*****
VARIABLES
*****/
#define ENA 6
#define ENB 3
#define IN1 7
#define IN2 5
#define IN3 4
#define IN4 2
long cm, duration;
#define ECO 10
#define TRIG 8
#define receiver 11

/*****
LIBRERIAS
*****/
#include <Servo.h>
#include "IRremote.h"
```

Imagen 33: Variables y librerías.

A continuación, declaro al servomotor y defino la variable para su control. La función de servomotor la haré "Servo", que es el nombre que le he puesto al servomotor en el programa (se puede poner cualquier nombre), y la variable "angulo = 90", que la defino de esta forma para más adelante. También declaro el control remoto con el nombre de "irrecv (receiver)" i los resultados de este "results", que también los emplearé.

```
/*declaro el servomotor*/  
Servo myservo;  
int angulo = 90, Boton;  
  
/*Declaro control remoto*/  
IRrecv irrecv(receiver);  
decode_results results;
```

Imagen 34: Servomotor y control remoto.

Paso 2. En el "void setup()" se representa la inicialización de todos los parámetros del sistema. Esto solo se ejecuta una vez al inicio.

Defino en que pin está conectado el servomotor (9). Utilizo la función "pinMode" para configurar los pines del driver como salidas, donde controlaré la dirección de los motores. También hago uso de "digitalWrite" que es otra función donde introduzco que los motores A y B estén en "HIGH", es decir, que su voltaje será de 5 voltios. Finalmente activo el receptor del control remoto.

```
void setup()  
{  
    myservo.attach(9);  
    //Definición control de motores  
    pinMode(ENA, OUTPUT);  
    pinMode(ENB, OUTPUT);  
    pinMode(IN1, OUTPUT);  
    pinMode(IN2, OUTPUT);  
    pinMode(IN3, OUTPUT);  
    pinMode(IN4, OUTPUT);  
    // Habilitar Motor A, Motor B  
    digitalWrite(ENA, HIGH);  
    digitalWrite(ENB, HIGH);  
    // Serial communication  
    Serial.begin(9600);  
    irrecv.enableIRIn(); /*Activación del receptor  
                           del control remoto*/  
}
```

Imagen 35: Servomotor (9) y motores.

Paso 3. En el “void loop()” se hace lo contrario que en el “void setup()”. Tiene la función de que se repita constantemente lo que tenga a continuación, es decir, que entre en bucle.

Aquí utilizo la estructura “while”, que significa mientras, y los códigos “!=”, que significa que no es igual a lo que tenga a continuación, y “&&” que significa “and” en inglés. Entonces, la primera expresión significa lo siguiente: mientras no pulse el botón 5 ni el botón 6, el robot se quedará en reposo esperando a recibir una señal e irá consultando si la ha recibido cada 100 milisegundos.

```
void loop()
{
    while(Boton != 5 && Boton != 6)
    {
        delay(100);
        if (irrecv.decode(&results))
        {
            translateIR();
            irrecv.resume();
        }
    }
}
```

Imagen 36: Iniciación del “void loop”.

Ahora vemos qué pasa si pulsamos el botón 5.

“If” se utiliza para realizar una acción u otra. Si he presionado el botón 5, el robot entrará en el modo automático, es decir, que con un sensor ultrasónico detectamos un obstáculo y el robot se desplazará a un lado o a otro.

En primer lugar definimos una variable “cm = Distancia()”. Si la distancia es menor a 35 centímetros, defino unas variables (“verif_dist_derecha” y “verif_dist_izquierda”) para guardar las distancias que hay a la derecha y a la izquierda y se llevará a cabo lo siguiente:

El motor se parará y el servomotor se moverá para enfocar el sensor a la izquierda a través de la función “for”, que sirve para producir un incremento del ángulo poniendo “++” al final o una disminución poniendo “-”.

A continuación, el sensor lee la distancia y la guarda en la variable “verif_dist_izquierda”. Se vuelve a mover el servomotor para enfocar el sensor

a la derecha y volver a leer la distancia, guardarla en la variable “verif_dist_derecha” y finalmente recolocar el sensor mirando al frente.

```
if(Boton == 5)
{
    cm = Distancia();
    if(cm <= 35)
    {
        long verif_dist_derecha, verif_dist_izquierda;
        MotorAB_Brake(1000);
        for (angulo = 0; angulo <= 180; angulo++) {
            myservo.write(angulo);
            delay(15);
        }
        verif_dist_izquierda = Distancia();
        for (angulo = 180; angulo >= 0; angulo--) {
            myservo.write(angulo);
            delay(15);
        }
        verif_dist_derecha = Distancia();
        myservo.write(90);
        delay(200);
    }
}
```

Imagen 37: Modo automático.

Después de todo esto, el robot se pregunta, ¿dónde tengo más espacio? Si hay menos espacio a la izquierda, el robot girará a la derecha. Pero para ejecutar este movimiento, hay que declarar qué ruedas son las que avanzan y cuáles no.

```
if(verif_dist_derecha > verif_dist_izquierda)
{
    digitalWrite(IN1, LOW); // Girar derecha
    digitalWrite(IN2, HIGH);
    digitalWrite(IN3, HIGH);
    digitalWrite(IN4, LOW);
    delay(250);
}
else
{
    digitalWrite(IN1, HIGH); // Girar izquierda
    digitalWrite(IN2, LOW);
    digitalWrite(IN3, LOW);
    digitalWrite(IN4, HIGH);
    delay(250);
}
```

Imagen 38: Direcciones que tomará el robot.

Para elaborar este movimiento, he utilizado otra vez

“digitalWrite” y he declarado qué motores son los que no se van a activar.

He llamado IN1 a la rueda derecha delantera, IN2 a la rueda delantera izquierda, IN3 a la rueda trasera izquierda y IN4, a la rueda trasera derecha. Como se puede apreciar en la fotografía, si gira a la izquierda, se ponen “HIGH” IN1 y IN4 y en “LOW” IN2 y IN3 y si gira a la derecha, al revés. Después de empezar el giro, a los 250 milisegundos, el robot vuelve a avanzar hacia delante y repite todo el proceso.

La estructura "else" funciona para cuando no hay una distancia menor de 35 centímetros. Si el sensor no detecta ningún obstáculo, el robot avanzará, sin parar, en línea recta.

```

    }
    else
    {
        digitalWrite(IN1, HIGH);
        digitalWrite(IN2, LOW);
        digitalWrite(IN3, HIGH);
        digitalWrite(IN4, LOW);
    }
    delay(100);
}

```

Imagen 39: Desplazamiento en línea recta.

Paso 4. Si he pulsado el número 6, que es el modo manual, el robot irá leyendo las instrucciones enviadas por el mando IR para saber a qué dirección debe moverse o detenerse. El receptor leerá el código enviado por el mando cada 50 milisegundos, y según el botón que pulse cambiará de dirección.

Para poder realizar este proceso hago uso de la estructura "switch case", que se utiliza para agilizar la toma de decisiones múltiples, y se utiliza de una forma muy parecida al "if" o "if else". Pero si esta parte de la programación con el mando la hubiese puesto con "if" habrían muchísimos, y no quedaría muy claro, en cambio, con esta estructura se pueden tener las funciones mucho más claras. En el

```

else if(Boton == 6)
{
    if (irrecv.decode(&results))
    {
        translateIR();
        irrecv.resume();
        delay(50);
    }
    switch(Boton)
    {
        case 1:
            digitalWrite(IN1, HIGH);
            digitalWrite(IN2, LOW);
            digitalWrite(IN3, HIGH);
            digitalWrite(IN4, LOW);
            break;
        case 2:
            digitalWrite(IN1, HIGH); //
            digitalWrite(IN2, LOW);
            digitalWrite(IN3, LOW);
            digitalWrite(IN4, HIGH);
            break;
        case 3:
            digitalWrite(IN1, LOW); //
            digitalWrite(IN2, HIGH);
            digitalWrite(IN3, HIGH);
            digitalWrite(IN4, LOW);
            break;
        case 4:
            digitalWrite(IN1, HIGH); //
            digitalWrite(IN2, HIGH);
            digitalWrite(IN3, HIGH);
            digitalWrite(IN4, HIGH);
            break;
        default:
            digitalWrite(IN1, HIGH); //
            digitalWrite(IN2, HIGH);
            digitalWrite(IN3, HIGH);
            digitalWrite(IN4, HIGH);
    }
}

```

Imagen 40: Modo manual.

caso 1, el robot avanzará hacia delante, en el caso 2, el robot girará a la izquierda, en el caso 3, girará a la derecha y en el caso 4, el robot se detendrá.

Paso 5. En este último paso declararé las partes del sensor ultrasónico como salidas o entradas. Esta rutina, cuando es llamada, devuelve la distancia en función del tiempo en que tardan en rebotar las ondas. Para realizar este proceso, declararé “TRIG” como salida, ya que envía las ondas, y “ECO” como entrada, porque recibe las ondas. Según lo que hayan tardado estas ondas en volver, se sabrá a qué distancia está cada objeto. Finalmente definimos los códigos del mando IR que recibirá la placa Arduino.

```
long Distancia()
{
    pinMode(TRIG, OUTPUT);
    digitalWrite(TRIG, LOW);
    delayMicroseconds(2);
    digitalWrite(TRIG, HIGH);
    delayMicroseconds(5);
    digitalWrite(TRIG, LOW);
    pinMode(ECO, INPUT);
    duration = pulseIn(ECO, HIGH);
    return duration / 29 / 2;
}

switch(results.value)
{
    case 0xFF629D: Boton = 1; break; // ADELANTE
    case 0xFF22DD: Boton = 2; break; // IZQ
    case 0xFFC23D: Boton = 3; break; // DER
    case 0xFFA857: Boton = 4; break; // ABAJO
    case 0xFF6897: Boton = 5; break; // MODO 1
    case 0xFF9867: Boton = 6; break; // MODO 2
    default:
        Serial.println(" other button ");
}
delay(500);
```

Imagen 41: Declaración de las partes del sensor ultrasónico.

Imagen 42: Definición del control remoto.

8. PRESUPUESTO

En este apartado mostraré cuál ha sido el precio de cada uno de los accesorios y el precio total del robot. Los componentes han sido comprados a través de las siguientes páginas de Internet:

- **Placa Arduino UNO:** <https://es.aliexpress.com/>
- **Sensor shield:** <https://www.amazon.es/>
- **Motores:** <https://es.aliexpress.com/>
- **Chasis:** <https://es.aliexpress.com/>
- **Ruedas:** <https://es.aliexpress.com/>
- **Batería:** <https://es.aliexpress.com/>
- **Servomotor:** <https://es.aliexpress.com/>
- **Control remoto:** <http://www.ebay.es/>
- **Sensor ultrasónico:** <https://www.amazon.es/>
- **Driver:** <https://www.amazon.es/>

PRESUPUESTO DE MI ROBOT		
	ACCESORIOS	COSTE ECÓNOMICO
1	Placa Arduino UNO	23 €
2	Sensor shield	15 €
3	Driver	13 €
4	Motores	6 €
5	Chasis	6 €
6	Ruedas	2 €
7	Batería	4 €
8	Sensor ultrasónico	15 €
9	Servomotor	10 €
10	Control remoto	12 €
	Coste total económico	106 €

Tabla 1: Presupuesto del robot.

9. PROBLEMAS

En esta sección quiero hacer constar que no ha sido todo tan fácil como parece ya que he tenido varios problemas que he tenido que ir solucionando durante todo el proyecto.

El primer problema que tuve fue la selección del tema a desarrollar porque no tenía bien claro que era lo que quería hacer. Gracias a mi tutor y a la información que me proporcionó me decanté por realizar este proyecto.

Otro problema que tuve fue el tiempo en que tardaban en enviarme los accesorios. No había comprado antes nada por Internet y me preocupaba bastante que no me llegaran las compras pero finalmente todo llegó, aunque no en la fecha prevista.

También tuve problemas con los accesorios. Cuando compré los motores, tenían sus cables perfectamente soldados pero fue pasando el tiempo y estos cables fueron despegándose. Yo no había soldado nunca pero, gracias a mi padre que fue quien me aconsejó que comprase un soldador en el Leroy Merlin, aprendí a realizar una soldadura con estaño.

Sin duda, lo que me causó más problemas fue la programación. Mis conocimientos eran muy básicos para intentar hacer lo que yo me propuse, así que tuve que trabajar duro si quería sacar adelante el proyecto.

El último problema que tuve fue a la hora de la grabación del video. Para que el video fuera de calidad, tuve que editarlo. Esto no fue una tarea sencilla ya que no sabía cómo se editaban videos así que investigué por Internet mirando tutoriales.

10. CONCLUSIONES

Mi objetivo a conseguir era crear un robot programado con Arduino que fuese capaz de tener un modo automático en el cual tuviese incluido un sensor ultrasónico que hiciera un barrido de 180° y fuese capaz de decidir a qué dirección le correspondería girar para no chocar. Este proceso no ha sido un camino de rosas ya que la programación fue uno de los puntos más dificultosos de todo el trabajo, sin embargo, cuando entendí sus partes esenciales, fui capaz de llevarla a cabo. Es cierto que fue bastante complicado el aprendizaje del lenguaje de Arduino, pero la satisfacción que sentí cuando el robot se movió por primera vez compensó todo el esfuerzo realizado anteriormente. Como mi tutor me comentó, en el proyecto hay que ir poco a poco subiendo de nivel, es decir, realizar lo más básico para evolucionar continuamente. Gracias a este consejo, me decidí a añadirle algo más al robot, otra función en la que se pudiese controlar a distancia el robot. Esta parte también fue muy laboriosa pero, ya tenía la base de la programación anterior así que fue más sencillo de lo que esperaba. Sin duda, estoy convencido de que he conseguido mi objetivo principal haciendo que funcione correctamente el robot e incluso he llegado a más de lo que esperaba.

Una de las cosas más importantes de este proyecto ha sido la dedicación y el esfuerzo empleado durante la época de verano porque fueron meses que, por no tener otras responsabilidades, he podido dedicarle mucho más tiempo al proyecto. Para poder realizar un trabajo de investigación con un buen resultado, no puede faltar una buena planificación desde el principio porque así se puede tener todo más controlado desde el primer momento.

Este trabajo de investigación me ha ayudado bastante a aprender y a desarrollar nuevos conceptos. Después de estos meses investigando los diferentes accesorios que necesitaría para completar el robot, he aprendido a hacer conexiones entre diferentes placas para poder llegar a un resultado. Otro de los puntos a resaltar de este trabajo es cómo afrontar los problemas. Si tú no dominas desde el principio el tema es obvio que habrá problemas y tendrás que resolverlos y este proyecto te obliga a solucionarlos porque sino el trabajo estaría incompleto.

Seguramente, mi trabajo no tiene una acción directa en la vida cotidiana pero sí que se puede utilizar como base para otro proyecto de universidad mejorándolo con muchas más aplicaciones e incluso podría llegar a ser una forma de entretenimiento. Mi parte práctica ha ido mejorando poco a poco. Eso es lo más entretenido de crear tu propio robot ya que puedes ir progresando, sin límites. Un objetivo para el futuro que tengo pendiente de realizar es crear una app en android para que en vez de controlarlo con un mando se pudiera controlar con el propio móvil.

Todas estas experiencias vividas me han hecho disfrutar de momentos muy buenos, afrontar momentos difíciles y hacerme ver que con esfuerzo y dedicación se puede conseguir todo lo que uno se propone y esa es, en mi opinión, la base de este proyecto. Por tanto, repetiría otra vez la experiencia y recomendaría este tema a todas las personas que tengan ganas de iniciarse en la robótica y en la programación.

11. BIBLIOGRAFÍA

Julián Pérez Porto y Ana Gardey, Definición de programación, Definición.DE, <http://definicion.de/programacion/>, 2 de agosto de 2016.

Luis Thayer Ojeda, ¿Qué es Arduino?, Arduino.cl, <http://arduino.cl/que-es-arduino/>, 2 de agosto de 2016.

Jorge Chávez, Arduino UNO: ¿cómo funciona la famosa placa de Arduino?, TREEGIC, <http://www.treegicmagazine.com/geek/arduino-uno-como-funciona-la-famosa-placa-de-arduino/>, 5 de agosto de 2016.

Daniel Gallardo García, Apuntes de ARDUINO, http://educacionadistancia.juntadeandalucia.es/profesorado/pluginfile.php/2882/mod_resource/content/1/Apuntes_ARDUINO_nivel_PARDILLO.pdf, 10 de agosto de 2016.

Arduino Sensor Shield V5.0 (126-127), Picaxe.es, <http://www.picaxe.biz/tienda/producto/gx-800/1/arduino-sensor-shield-v5-0-126-127->, 13 de agosto de 2016.

Andrés Cruz, Tutorial: Uso de Driver L298N para motores DC y paso a paso con Arduino, <http://electronilab.co/tutoriales/tutorial-de-uso-driver-dual-l298n-para-motores-dc-y-paso-a-paso-con-arduino/>, 17 de agosto de 2016.

MARCO FIDEL OROZCO, Reductores y motorreductores, monografías.com, <http://www.monografias.com/trabajos13/reducty/reducty.shtml>, 21 de agosto de 2016.

CJ, Yahoo, <https://ar.answers.yahoo.com/question/index?qid=20090404093040AAppaye>, 23 de agosto de 2016.

CSCAZORLA, Xataka, ¿Cómo funciona una batería?, <http://www.xatakaciencia.com/sabias-que/como-funciona-una-bateria>, 26 de agosto de 2016.

Wikipedia, https://es.wikipedia.org/wiki/Sensor_ultras%C3%B3nico, 26 de agosto de 2016.

Nazly López, Catherine Vargas, Laura Beltrán y Diego García, SENSORES ULTRASONICOS, Prezi, <https://prezi.com/rzik8ow5d4ee/sensores-ultrasonicos/>, 26 de agosto de 2016.

Micro Servo 9g SG90 TowerPro, Electronilab, <http://electronilab.co/tienda/micro-servo-9g-towerpro/>, 27 de agosto de 2016.

Wikipedia, control remoto, https://es.wikipedia.org/wiki/Control_remoto, 10 de agosto de 2016 Prometec, módulo receptor de infrarrojos, <http://www.prometec.net/infrarrojos/>, 29 de agosto de 2016.

Joaquín V. Álvarez Martín, Añadir y usar librerías con Arduino, <http://explicandotecnologia.blogspot.com.es/2012/06/anadir-y-usar-librerias-con-arduino.html>, 31 de agosto de 2016.

Lenguaje de Arduino, Arduino Home, <https://www.arduino.cc/en/Reference/HomePage>, 3 de septiembre de 2016

12. BIBLIOGRAFÍA FOTOGRÁFICA

Imagen 1. Elaboración propia.

Imagen 2. César Martínez Manuel Hidalgo, CONTROL Y ROBÓTICA CON EduBasica, GRUPO COLABORATIVO - ROBÓTICA EDUCATIVA, <http://platea.pntic.mec.es/~mhidalgo/edubasica/01arduino/arduino01.html>, 2 de agosto de 2016.

Imagen 3. Elaboración propia.

Imagen 4. Elaboración propia.

Imagen 5. Elaboración propia.

Imagen 6. Elaboración propia.

Imagen 7. Elaboración propia.

Imagen 8. Elaboración propia.

Imagen 9: Elaboración propia.

Imagen 10. Rubén Lorenzo Araujo, Funcionamiento de un medidor ultrasónico de distancia, Portal de ingeniería y gestión de mantenimiento, <http://www.ingenieriamantenimiento.org/medida-de-distancia-por-ultrasonidos/>, 26 de agosto de 2016.

Imagen 11. Nazly López, Catherine Vargas, Laura Beltrán y Diego García, SENSORES ULTRASONICOS, Prezi, <https://prezi.com/rzik8ow5d4ee/sensores-ultrasonicos/>, 26 de agosto de 2016.

Imagen 12. SG90 9 g Micro Servo, <http://www.micropik.com/PDF/SG90Servo.pdf>, 27 de agosto de 2016.

Imagen 13. Elaboración propia.

Imagen 14. Servomotor de modelismo, Wikipedia, https://es.wikipedia.org/wiki/Servomotor_de_modelismo, 28 de agosto de 2016.

Imagen 15. Elaboración propia.

Imagen 16. Elaboración propia.

Imagen 17. Elaboración propia.

Imagen 18. Elaboración propia.

Imagen 19. Elaboración propia.

Imagen 20. Elaboración propia.

Imagen 21. Elaboración propia.

Imagen 22. Elaboración propia.

Imagen 23. Elaboración propia.

Imagen 24. Elaboración propia.

Imagen 25. Elaboración propia.

Imagen 26. Elaboración propia.

Imagen 27. Elaboración propia.

Imagen 28. Elaboración propia.

Imagen 29. Elaboración propia.

Imagen 30. Elaboración propia.

Imagen 31. Elaboración propia.

Imagen 32. Elaboración propia.

Imagen 33. Elaboración propia.

Imagen 34. Elaboración propia.

Imagen 35. Elaboración propia.

Imagen 36. Elaboración propia.

Imagen 37. Elaboración propia.

Imagen 38. Elaboración propia.

Imagen 39. Elaboración propia.

Imagen 40. Elaboración propia.

Imagen 41. Elaboración propia.

Imagen 42. Elaboración propia.

Imagen 43. Elaboración propia.

Tabla 1. Elaboración propia.

13. ANEXOS

Anexo 1

Programa:

```
#define ENA 6
```

```
#define ENB 3
```

```
#define IN1 7
```

```
#define IN2 5
```

```
#define IN3 4
```

```
#define IN4 2
```

```
long cm, duration;
```

```
#define ECO 10
```

```
#define TRIG 8
```

```
#define receiver 11
```

```
#include <Servo.h>
```

```
#include "IRremote.h"
```

```
Servo myservo;
```

```
int angulo = 90, Boton;
```

```
IRrecv irrecv(receiver);
```

```
decode_results results;
```

```
void setup()
```

```
{
```

```
    myservo.attach(9);
```

```
    //Definición control de motores
```

```
    pinMode(ENA, OUTPUT);
```

```
    pinMode(ENB, OUTPUT);
```

```
    pinMode(IN1, OUTPUT);
```

```
    pinMode(IN2, OUTPUT);
```

```
    pinMode(IN3, OUTPUT);
```

```
    pinMode(IN4, OUTPUT);
```

```
    // Habilitar Motor A, Motor B
```



```

    digitalWrite(ENA, HIGH);
    digitalWrite(ENB, HIGH);
    // Serial communication
    Serial.begin(9600);
    irrecv.enableIRIn

}

void loop()
{

    while(Boton != 5 && Boton != 6)
    {
        delay(100);
        if (irrecv.decode(&results))
        {
            translateIR();
            irrecv.resume();
        }
    }
    if(Boton == 5)
    {
        cm = Distancia();
        if(cm <= 35)
        {
            long verif_dist_derecha, verif_dist_izquierda;
            MotorAB_Brake(1000);
            for (angulo = 0; angulo <= 180; angulo++) {
                myservo.write(angulo);
                delay(15);
            }
            verif_dist_izquierda = Distancia();
            for (angulo = 180; angulo >= 0; angulo--) {
                myservo.write(angulo);

```

```

        delay(15);
    }
    verif_dist_derecha = Distancia();
    myservo.write(90);
    delay(200);
    if(verif_dist_derecha > verif_dist_izquierda)
    {
        digitalWrite(IN1, LOW);
        digitalWrite(IN2, HIGH);
        digitalWrite(IN3, HIGH);
        digitalWrite(IN4, LOW);
        delay(250);
    }
    else
    {
        digitalWrite(IN1, HIGH);
        digitalWrite(IN2, LOW);
        digitalWrite(IN3, LOW);
        digitalWrite(IN4, HIGH);
        delay(250);
    }
}
else
{
    digitalWrite(IN1, HIGH);
    digitalWrite(IN2, LOW);
    digitalWrite(IN3, HIGH);
    digitalWrite(IN4, LOW);
}
delay(100);
}
else if(Boton == 6)
{
    while(Boton != 7)

```

```

{
    if (irrecv.decode(&results))
    {
        translateIR();
        irrecv.resume();
        delay(50);
    }
    switch(Boton)
    {
        case 1:
            digitalWrite(IN1, HIGH);
            digitalWrite(IN2, LOW);
            digitalWrite(IN3, HIGH);
            digitalWrite(IN4, LOW);
            break;
        case 2:
            digitalWrite(IN1, HIGH);
            digitalWrite(IN2, LOW);
            digitalWrite(IN3, LOW);
            digitalWrite(IN4, HIGH);
            break;
        case 3:
            digitalWrite(IN1, LOW);
            digitalWrite(IN2, HIGH);
            digitalWrite(IN3, HIGH);
            digitalWrite(IN4, LOW);
            break;
        case 4:
            digitalWrite(IN1, HIGH);
            digitalWrite(IN2, HIGH);
            digitalWrite(IN3, HIGH);
            digitalWrite(IN4, HIGH);
            break;
        default:

```

```

        digitalWrite(IN1, HIGH);
        digitalWrite(IN2, HIGH);
        digitalWrite(IN3, HIGH);
        digitalWrite(IN4, HIGH);
    }
}
}
}

long Distancia()
{
    pinMode(TRIG, OUTPUT);
    digitalWrite(TRIG, LOW);
    delayMicroseconds(2);
    digitalWrite(TRIG, HIGH);
    delayMicroseconds(5);
    digitalWrite(TRIG, LOW);
    pinMode(ECO, INPUT);
    duration = pulseIn(ECO, HIGH);
    return duration / 29 / 2;
}

{
    switch(results.value)
    {

        case 0xFF629D: Boton = 1; break; // ADELANTE
        case 0xFF22DD: Boton = 2; break; // IZQ
        case 0xFFC23D: Boton = 3; break; // DER
        case 0xFFA857: Boton = 4; break; // ABAJO
        case 0xFF6897: Boton = 5; break; // MODO 1
        case 0xFF9867: Boton = 6; break; // MODO 2
        case 0xFF52AD: Boton = 7; break; // # PAUSA;

        default:

```

```
Serial.println(" other button ");  
  
}  
  
delay(500);  
}
```

Anexo 2

Para aclarar más el montaje del robot, grabé un vídeo de cómo construí mi robot paso a paso en la plataforma YouTube.

Link del video: <https://www.youtube.com/watch?v=WE-1f7Y0Jbo>

