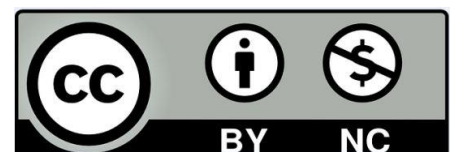


Guía

VelociBot CBR





Índice.

- .- Introducción
- .- Material necesario
- .- Montaje y conexiones
- .- Programación:
 - .- sketch 1. Test motor derecho.
 - .- sketch 2. Movimientos básicos del VelociBotCBR
 - .- sketch 3. Movimientos básicos con subrutinas del VelociBotCBR
 - .- sketch 4. Sensor de infrarrojos
 - .- sketch 5. Seguidor de líneas básico



Introducción.

VelociBotCBR es un robot de la categoría de velocistas que ha sido diseñado por alumn@s de Tecnología Industrial de 2º de Bachillerato del IES Valentín Turienzo de Colindres, Cantabria.

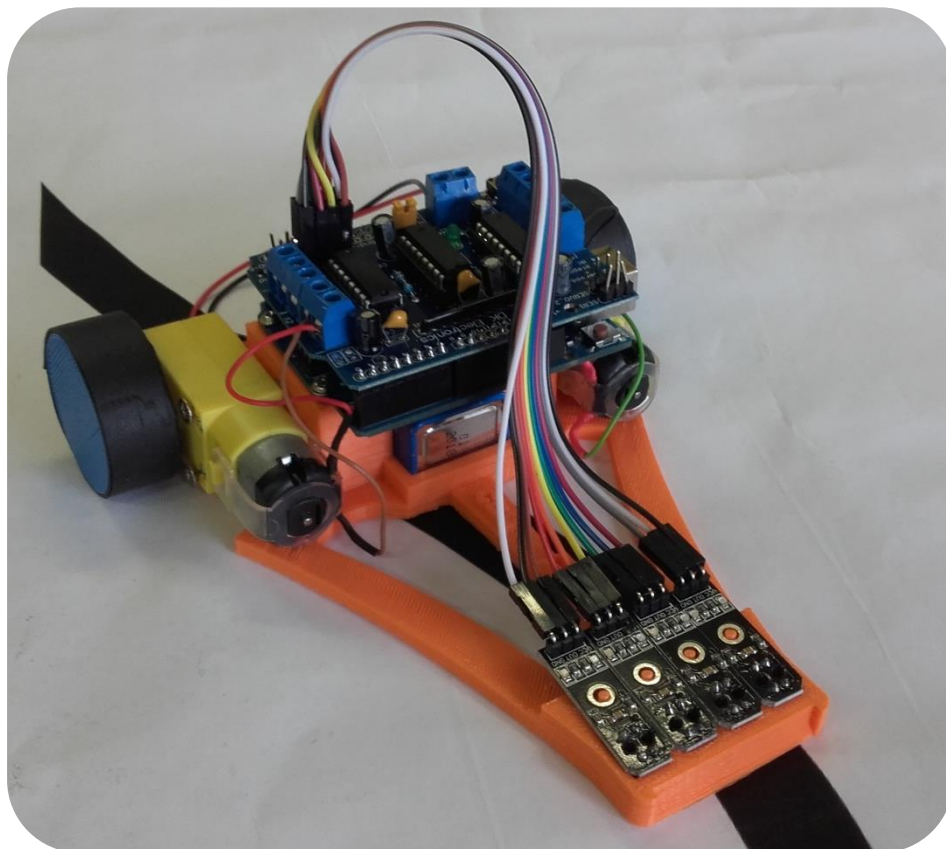
Las principales premisas en su diseño fueron las siguientes:

- Sencillez de montaje, utilizando el menor número de tuercas, tornillos, adhesivos,...
- Menor coste de los componentes y material y facilidad en su adquisición.
- Control realizado con Arduino.
- Chasis impreso en 3D y diseñado con SketchUp.

El objetivo principal de este proyecto es que cualquier persona que quiera construir este robot, profesores, estudiantes, aficionados,... lo pueda hacer en muy poco tiempo, con las mínimas complicaciones y el mínimo coste.

Descarga los archivos .stl para imprimir [aquí](#).

Descarga los archivos .skp por si quieres modificar y personalizar en robot.



Material necesario

Listado de componentes VelociBotCBR

Artículo	Imagen
1 x Arduino UNO	
1 x Shield L293 control de motores	
2 x Motorreductor DC120	
4 x Módulo seguidor IR Sensor TCRT5000	
40 x Cables dupot hembra-hembra 20 cm	
Conector y pila 9 v	
Chasis y cubierta impreso en 3D	
Ruedas impresas en 3D	
Cinta "tapafugas" para las ruedas	

El coste total aproximado del robot completo no supera los 45 €.

Montaje y Conexiones.

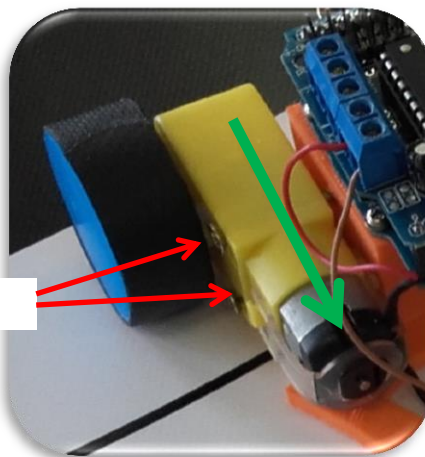
MONTAJE:

Empezaremos montando los dos motorreductores CC en el chasis utilizando dos tornillos de 3 x 30 mm de cabeza hexagonal por cada motor. Previamente se deben realizar las conexiones de los cables a los terminales del motor.

Si el motorreductor tuviera doble eje como el de la imagen, sería necesario cortar uno de los ejes.

IMPORTANTE

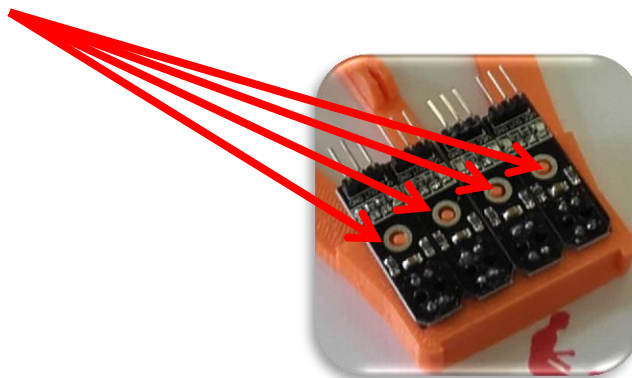
NO APRETAR DEMASIADO LOS TORNILLOS. Puede llegar a presionar la reductora del motor y hacer que este se quede trabado.



Tornillos 3x30

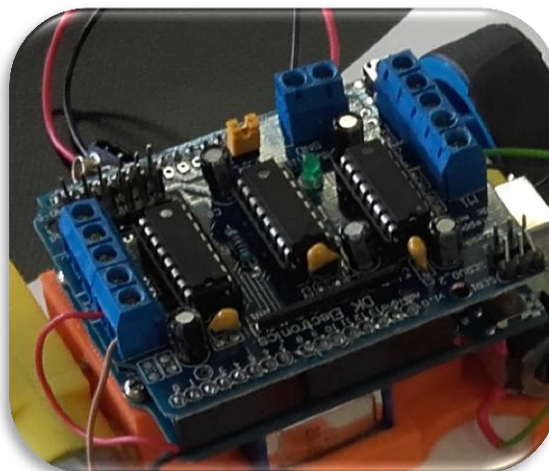
El motorreductor debe ir colocado en la posición de la flecha verde.

A continuación colocaremos los cuatro sensores TCRT5000. En un primer momento su alojamiento está diseñado para colocarlos simplemente por presión, pero se podría sustituir esta sujeción por tornillos de 3 x 6 mm.



Seguidamente colocaremos la Arduino UNO sobre la estructura sujetándola con tres tornillos, y sobre ella colocaremos la Shield L293 en la parte superior, tal y como se ve en la foto.

En la parte inferior está el alojamiento para la batería de 9 v.



IMPORTANTE

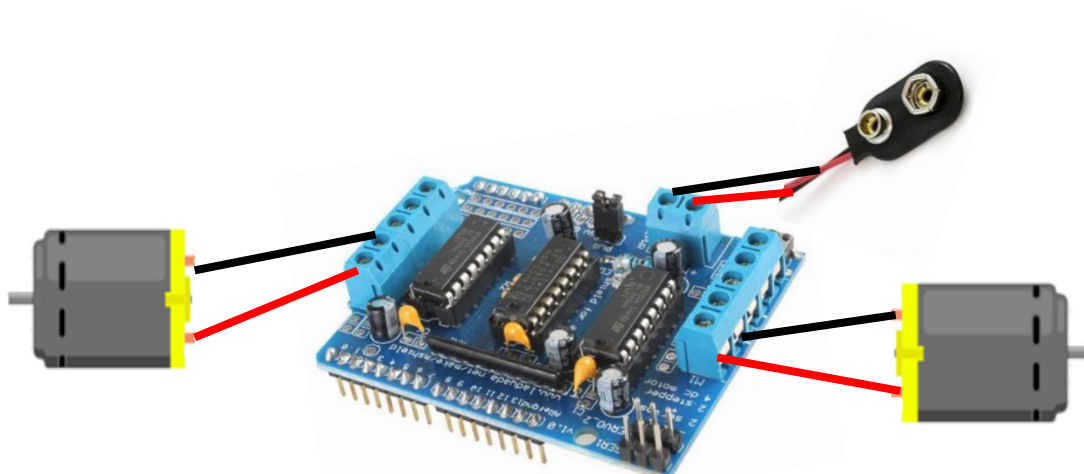


La Shield de motores L293 no tienen soldados los pines A0-5 (donde van conectados los sensores) por lo que es necesario soldarlos previamente.

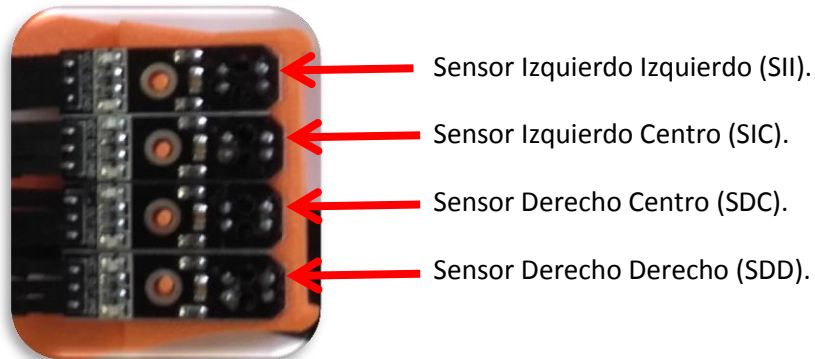
Conexionado:

El siguiente paso consiste en conectar los motorreductores a la Shield. Visto el robot desde arriba, el motor Izquierdo (MI) lo conectaremos a los terminales **M1** y el motor Derecho (MD) a los terminales **M4**.

***Cuando se realice el Test de motores habrá que revisar su polaridad.**



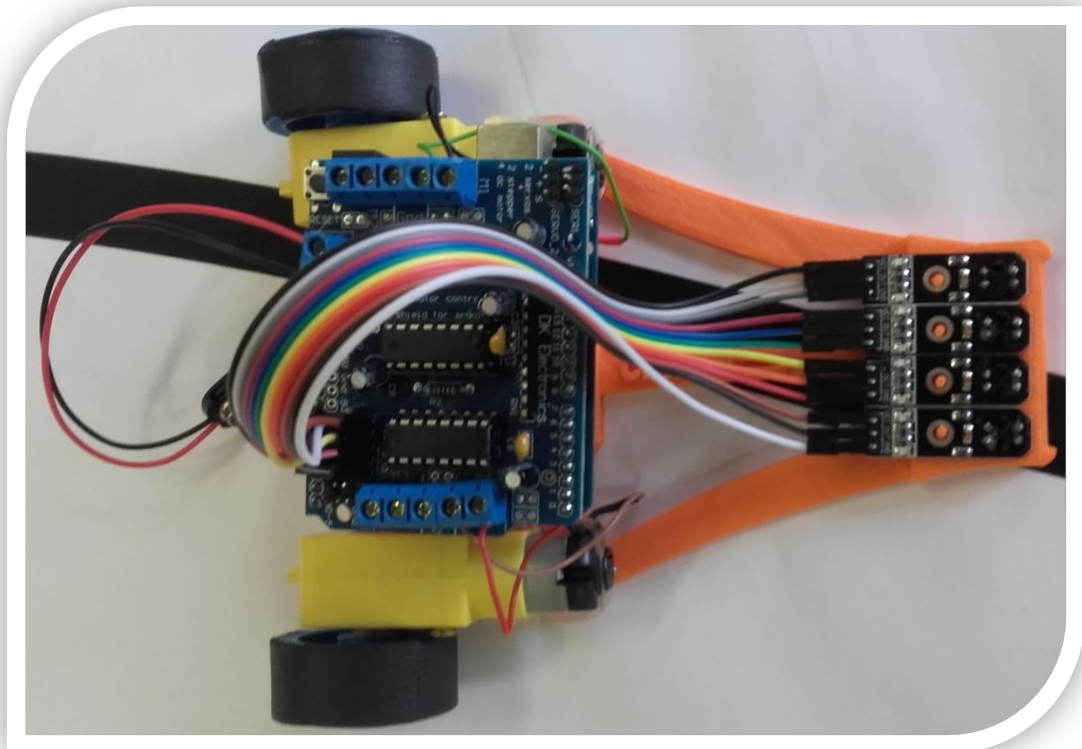
Los cuatro sensores TCRT5000 los vamos a llamar de la siguiente manera:



Su conexión a la shield es la siguiente:

Sensor	Pines de la Shield
Sensor Izquierdo Izquierdo (SII)	A3 (PIN 17)
Sensor Izquierdo Centro (SIC)	A2 (PIN 16)
Sensor Derecho Centro (SDC)	A1 (PIN 15)
Sensor Derecho Derecho (SDD)	A0 (PIN 14)

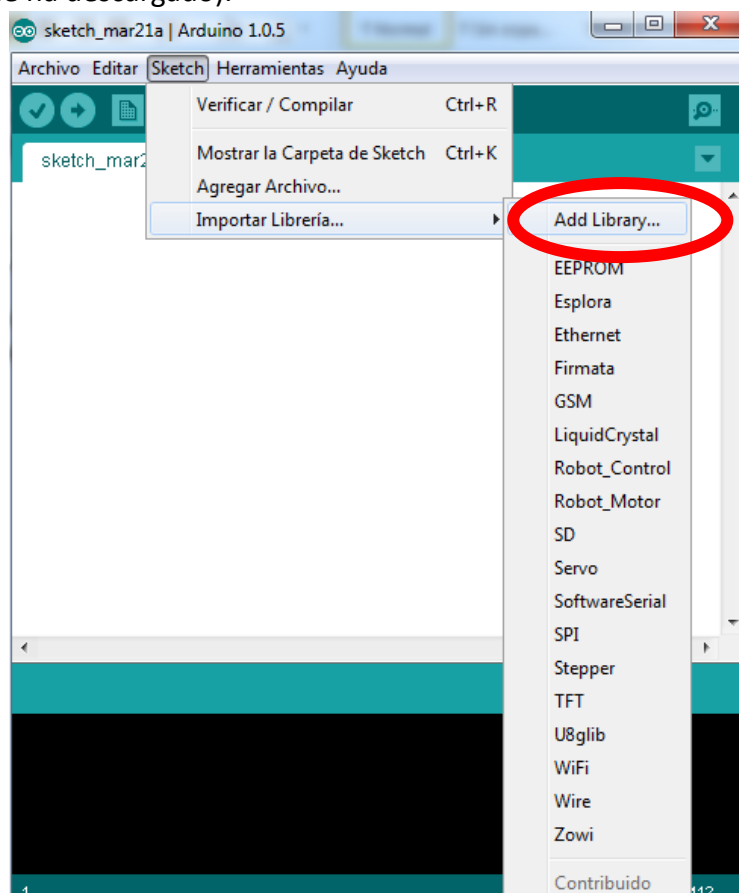
Finalmente, el VelociBotCBR quedará como la imagen:



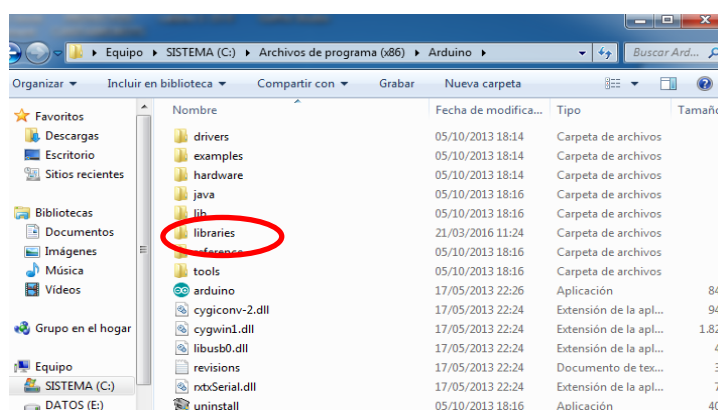
PROGRAMACIÓN

Antes de empezar con la programación, debemos instalar la librería [AFMotor.h](#). Podemos realizar una de las siguientes opciones:

.- **OPCIÓN 1:** Abrimos el **IDE de Arduino**, Pinchamos en **Sketch**, **Importar en librería** y finalmente en **Add Library**. En este punto añadiríamos la librería [AFMotor.h](#) (la carpeta entera en formato ZIP tal y como se ha descargado).



.- **OPCIÓN 2:** Si la opción anterior nos diera algún problema, podemos meter directamente la carpeta descomprimida en **C:\archivos de programa (x86)\arduino\libraries**



Sketch 1. Test motor DERECHO

Vamos a comenzar con la programación y para ello realizaremos un sencillo programa para controlar el motor DERECHO (MD). Haremos que gire hacia delante durante 1 seg, que se pare 2 seg y que giren hacia atrás 1 seg.

Con esta práctica se pretende comprobar el correcto conexionado del motorreductor, conocer su sentido de giro y su control.

El Sketch correspondiente sería este:

```
// TEST DE MOTORES DE VelociBOTCBR
// MOTOR IZQUIERDO (MI) es el M1
// MOTOR DERECHO (MD) es el M4
// SENSOR IZQUIERDO IZQUIERDO en el A3
// SENSOR IZQUIERDO CENTRO en el A2
// SENSOR DERECHO CENTRO en el A1
// SENSOR DERECHO DERECHO en el A0
// Es necesario instalar la librería AFMotor.h para que funcione correctamente este Sketch

//Test motor Derecho ADELANTE, PARO, ATRAS.

#include <AFMotor.h>

AF_DCMotor motorMD(4);    // Definimos el Motor Derecho en la conexión M4

void setup() {

    motorMD.setSpeed(250);    // Set de velocidad MD fijada en 250 (esta velocidad se puede variar
                             // desde 0 (parado) a 255 (velocidad máxima) por la salida PWM)

    motorMD.run(RELEASE);    // Motor Derecho preparado
}

void loop() {

    motorMD.run(FORWARD);    // Motor Derecho hacia delante (si gira al revés cambiamos la
                             // polaridad en la placa)
    motorMD.setSpeed(255);    // Set de velocidad MD fijada en 255
    delay(1000);              // Se mantiene durante 1 s.

    motorMD.run(FORWARD);    // Motor Derecho hacia delante
    motorMD.setSpeed(0);     // Set de velocidad MD fijada en 0. El motorD se para.
    delay(2000);              // Se mantiene durante 2 s.
```



```
motorMD.run(BACKWARD); // Motor Derecho hacia atrás (si gira al revés cambiamos la polaridad en la placa)
```

```
motorMD.setSpeed(255); // Set de velocidad MI fijada en 255  
delay(1000);           // Se mantiene durante 1 s.  
}
```

**Recuerda, Si el motor DERECHO no girara en el sentido que le indicamos debemos cambiar su polaridad en la Shield.*

Ejercicio:

Realiza los cambios necesarios para Testear el motor IZQUIERDO y comprobar su sentido de giro

Sketch 2. Movimientos Básicos del VelociBotCBR

Los movimientos básicos del VelociBotCBR son los siguientes; Adelante, Paro, Atrás, Giro Derecha, Giro Izquierda. Ahora vamos a trabajar con los dos motores a la vez. Para realizar los giros debemos parar el motor que esté en el sentido que queramos girar, por ejemplo, si queremos girar a la izquierda debemos parar el motor izquierdo y si queremos girar a la derecha, debemos parar el motor derecho.

```
// TEST DE MOTORES DE VelociBOTCBR  
// MOTOR IZQUIERDO (MI) es el M1  
// MOTOR DERECHO (MD) es el M4  
// SENSOR IZQUIERDO IZQUIERDO en el A3  
// SENSOR IZQUIERDO CENTRO en el A2  
// SENSOR DERECHO CENTRO en el A1  
// SENSOR DERECHO DERECHO en el A0  
// Es necesario instalar la librería AFMotor.h para que funcione correctamente este Sketch
```

```
//MOVIMIENTOS BÁSICOS DEL ROBOT ADELANTE, ATRAS, GIRO DERECHA, GIRO IZQUIERDA.
```

```
#include <AFMotor.h>
```

```
AF_DCMotor motorMD(4); // Definimos el Motor Derecho en la conexión M4  
AF_DCMotor motorMI(1); // Definimos el Motor Izquierdo en la conexión M1
```

```
void setup() {
```

```
    motorMD.setSpeed(250); // Set de velocidad MD fijada en 250 (esta velocidad se puede variar desde 0 (parado) a 255 (velocidad máxima) por la salida PWM)  
    motorMI.setSpeed(250); // Set de velocidad MI fijada en 250 (esta velocidad se puede variar desde 0 (parado) a 255 (velocidad máxima) por la salida PWM)
```





```
motorMD.run(RELEASE);    // Motor Derecho preparado
motorMI.run(RELEASE);    // Motor Izquierdo preparado
}

void loop() {

    motorMD.run(FORWARD);    // Motor Derecho hacia delante (si gira al revés cambiamos la
polaridad en la placa)
    motorMI.run(FORWARD);    // Motor Izquierdo hacia delante (si gira al revés cambiamos la
polaridad en la placa)
    motorMD.setSpeed(255);    // Set de velocidad MD fijada en 255
    motorMI.setSpeed(255);    // Set de velocidad MI fijada en 255
    delay(1000);            // Se mantiene durante 1 s.

    motorMD.run(FORWARD);    // Motor Derecho hacia delante
    motorMI.run(FORWARD);    // Motor Izquierdo hacia delante
    motorMD.setSpeed(0);    // Set de velocidad MD fijada en 0. El motorD se para.
    motorMI.setSpeed(0);    // Set de velocidad MI fijada en 0. El motorI se para.
    delay(2000);            // Se mantiene durante 2 s.

    motorMD.run(FORWARD);    // Motor Derecho hacia delante (si gira al revés cambiamos la
polaridad en la placa)
    motorMI.run(FORWARD);    // Motor Izquierdo hacia delante (si gira al revés cambiamos la
polaridad en la placa)
    motorMD.setSpeed(255);    // Set de velocidad MD fijada en 255
    motorMI.setSpeed(0);    // Set de velocidad MI fijada en 0. El motorI se para.
    delay(2000);            // Se mantiene durante 2 s.

    motorMD.run(FORWARD);    // Motor Derecho hacia delante (si gira al revés cambiamos la
polaridad en la placa)
    motorMI.run(FORWARD);    // Motor Izquierdo hacia delante (si gira al revés cambiamos la
polaridad en la placa)
    motorMD.setSpeed(0);    // Set de velocidad MD fijada en 0.El motorD se para
    motorMI.setSpeed(255);    // Set de velocidad MI fijada en 255.
    delay(2000);            // Se mantiene durante 2 s.

    motorMD.run(BACKWARD);    // Motor Derecho hacia atrás (si gira al revés cambiamos la polaridad
en la placa)
    motorMI.run(BACKWARD);    // Motor Izquierdo hacia atrás (si gira al revés cambiamos la polaridad
en la placa)
    motorMI.setSpeed(255);    // Set de velocidad MI fijada en 255
    motorMD.setSpeed(255);    // Set de velocidad MD fijada en 255
    delay(2000);            // Se mantiene durante 2 s.
}
```

**Podemos hacer otro tipo de giros más suaves sin necesidad de parar el motor del interior. En lugar de bajar a 0 su setSpeed podemos variar su PWM entre 20 y 100, siempre más bajo que la velocidad del motor exterior. HAZ PRUEBAS.*



Sketch 3. Movimientos Básicos con SUBROUTINAS del VelociBotCBR

Continuamos con nuestra programación y para ello vamos a introducir el concepto de **SUBROUTINAS**. Las subrutinas son unos «miniprogramas», unas funciones, que se ejecutan al ser llamadas desde el programa principal. La declaración de funciones, no va ni en el SETUP, ni en el LOOP, va de forma independiente entre los dos. En esta actividad crearemos 5 subrutinas ADELANTE, ATRAS, PARO, GIRODERECHA y GIROIZQUIERDA.

El Sketch correspondiente sería este:

```
// TEST DE MOTORES DE VelociBOTCBR
// MOTOR IZQUIERDO (MI) es el M1
// MOTOR DERECHO (MD) es el M4
// SENSOR IZQUIERDO IZQUIERDO en el A3
// SENSOR IZQUIERDO CENTRO en el A2
// SENSOR DERECHO CENTRO en el A1
// SENSOR DERECHO DERECHO en el A0
// Es necesario instalar la librería AFMotor.h para que funcione correctamente este Sketch

//MOVIMIENTOS BÁSICOS DEL ROBOT REALIZADOS CON SUBROUTINAS ADELANTE, PARO, GIRO
IZQUIERDA, GIRO DERECHA, ATRAS.

#include <AFMotor.h>

AF_DCMotor motorMD(4);    // Definimos el Motor Derecho en la conexión M4
AF_DCMotor motorMI(1);    // Definimos el Motor Izquierdo en la conexión M1

void setup() {

    motorMD.setSpeed(250);  // Set de velocidad MD fijada en 250 (esta velocidad se puede variar
desde 0 (parado) a 255 (velocidad máxima) por la salida PWM)
    motorMI.setSpeed(250);  // Set de velocidad MI fijada en 250 (esta velocidad se puede variar
desde 0 (parado) a 255 (velocidad máxima) por la salida PWM)

    motorMD.run(RELEASE);   // Motor Derecho preparado
    motorMI.run(RELEASE);   // Motor Izquierdo preparado
}

void loop() {

    ADELANTE();
    delay (1000);
    ATRAS();
    delay (1000);
    PARO();
    delay (1000);
    GIRODERECHA();

    delay (1000);
    GIROIZQUIERDA();
```

```
delay (1000);
}

void ADELANTE(){          // SUBROUTINA ADELANTE.

    motorMD.run(FORWARD);    // Motor Derecho hacia delante (si gira al revés cambiamos la
polaridad en la placa)
    motorMI.run(FORWARD);    // Motor Izquierdo hacia delante (si gira al revés cambiamos la
polaridad en la placa)
    motorMD.setSpeed(255);    // Set de velocidad MD fijada en 255
    motorMI.setSpeed(255);    // Set de velocidad MI fijada en 255
    }

void PARO(){              //SUBROUTINA PARO.
    motorMD.run(FORWARD);    // Motor Derecho hacia delante
    motorMI.run(FORWARD);    // Motor Izquierdo hacia delante
    motorMD.setSpeed(0);      // Set de velocidad MD fijada en 0. El motorD se para.
    motorMI.setSpeed(0);      // Set de velocidad MI fijada en 0. El motorI se para.
    }

void GIROIZQUIERDA(){     //SUBROUTINA GIRO IZQUIERDA.
    motorMD.run(FORWARD);    // Motor Derecho hacia delante (si gira al revés cambiamos la
polaridad en la placa)
    motorMI.run(FORWARD);    // Motor Izquierdo hacia delante (si gira al revés cambiamos la
polaridad en la placa)
    motorMD.setSpeed(255);    // Set de velocidad MD fijada en 255
    motorMI.setSpeed(0);      // Set de velocidad MI fijada en 0. El motorI se para.
    }

void GIRODERECHA(){       //SUBROUTINA GIRO DERECHA.
    motorMD.run(FORWARD);    // Motor Derecho hacia delante (si gira al revés cambiamos la
polaridad en la placa)
    motorMI.run(FORWARD);    // Motor Izquierdo hacia delante (si gira al revés cambiamos la
polaridad en la placa)
    motorMD.setSpeed(0);      // Set de velocidad MD fijada en 0.El motorD se para
    motorMI.setSpeed(255);    // Set de velocidad MI fijada en 255.
    }

void ATRAS(){             //SUBROUTINA ATRAS.
    motorMD.run(BACKWARD);    // Motor Derecho hacia atras (si gira al revés cambiamos la polaridad
en la placa)
    motorMI.run(BACKWARD);    // Motor Izquierdo hacia atras (si gira al revés cambiamos la polaridad
en la placa)
    motorMI.setSpeed(255);    // Set de velocidad MI fijada en 255
    motorMD.setSpeed(255);    // Set de velocidad MD fijada en 255
    }
```


Ejercicio:

Completa el programa realizando SUBROUTINAS de giros más suaves hacia DERECHA e IZQUIERDA cambiando el PWM del motor que queda en el interior.

Ejercicio:

Utilizando las SUBROUTINAS, realiza un programa en el que tu robot describa un cuadrado perfecto, llegando al mismo punto de la salida.

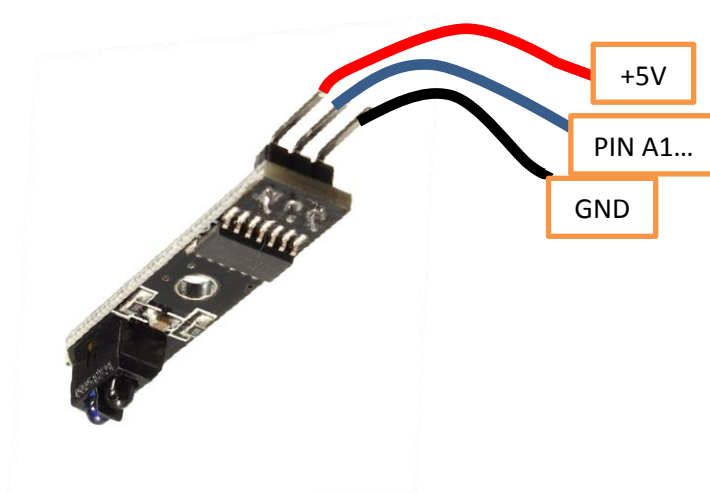
Ya hemos visto cómo controlar los motores DERECHOS e IZQUIERDOS del VelociBotCBR, ahora vamos a ver cómo hacerlo girar aún más rápido. Se pueden realizar tres tipos de giros, por llamarlos de alguna manera, lentos, suaves y rápidos. Los “*lentos*” consistirán en dejar parados los motores del interior del giro haciendo avanzar los del exterior, para los “*suaves*” cambiamos el PWM del motor interior a valores más pequeños que el exterior. Y los “*muy rápidos*” haciendo que los motores del interior giren hacia atrás mientras que los del exterior van hacia adelante. Ver cuadro siguiente:

CASOS DE GIROS		
<p>GIRO A IZQUIERDAS LENTO</p> 	<p>GIRO A IZQUIERDAS SUAVE</p> 	<p>GIRO A IZQUIERDAS RÁPIDO</p> 
<p>GIRO A DERECHAS LENTO</p> 	<p>GIRO A DERECHAS SUAVE</p> 	<p>GIRO A DERECHAS RÁPIDO</p> 

Sketch 4: Sensor Infrarrojos.

En esta práctica vamos a conocer el funcionamiento de los sensores IR. Más concretamente el sensor infrarrojo TCRT5000.

Un sensor infrarrojo es un dispositivo optoelectrónico capaz de medir la radiación electromagnética infrarroja de los cuerpos en su campo de visión. El sensor que nos ocupa está compuesto por un diodo emisor de luz y un fototransistor. Si el fototransistor recibe luz da una señal de cero, si no recibe luz da un 1.



Características:

- Dimensiones: 10mm x 40mm
- Voltaje de operación: 4.5V a 5.5V
- Distancia de operación: Desde 1mm hasta 12mm
- Sensor TCRT5000
- Salida digital
- Posee filtro bloqueador de luz de día
- Voltaje de trabajo: 5VDC
- Posee 3 pines: V_{CC} (+), GND (-), OUT (V_{out})
- Si detecta un objeto color claro (ej. línea blanca) $V_{out} > 4$ V
- Si no detecta nada (ej. línea negra) $V_{out} = 0$ V

[Más información de los sensores IR](#)

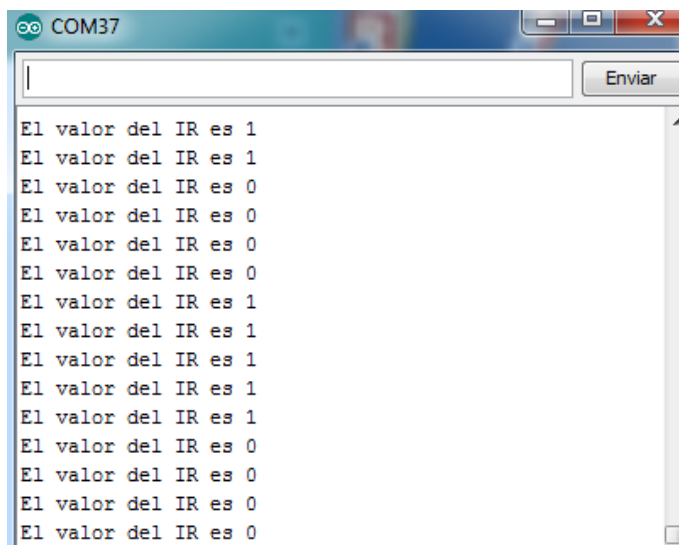
Para conocer los valores del sensor IR vamos a realizar un sencillo Sketch utilizando el Monitor Serial.

```
int IR = 14; //el Pin 14 corresponde con el A0.

void setup() {
  Serial.begin(9600);
  pinMode(IR, INPUT);
}

void loop() {
  IR = digitalRead(14);
  Serial.print("El valor del IR es ");
  Serial.println(IR);

  delay(1000);
}
```



COM37

Enviar

El valor del IR es 1
El valor del IR es 1
El valor del IR es 0
El valor del IR es 0
El valor del IR es 0
El valor del IR es 0
El valor del IR es 1
El valor del IR es 1
El valor del IR es 1
El valor del IR es 1
El valor del IR es 1
El valor del IR es 0
El valor del IR es 0
El valor del IR es 0
El valor del IR es 0

De esta manera observamos que cuando el sensor detecta línea negra su valor es 1 y cuando detecta blanco su valor es 0. (En el propio sensor hay un led rojo que se enciende cuando marca 0).

Ejercicio:

Escribe un nuevo código para comprobar el valor de los 4 sensores con un Serial.print de tal forma que vaya escribiendo que sensor es y su valor, por ejemplo:

El valor de SII es 0

El valor de SIC es 1

El valor de SDC es 1

El valor de SDD es 0...

SOLUCIÓN:

```
// TEST DE SENSORES DE VelociBOTCBR

// MOTOR IZQUIERDO (MI) es el M1

// MOTOR DERECHO (MD) es el M4

// SENSOR IZQUIERDO IZQUIERDO en el A3

// SENSOR IZQUIERDO CENTRO en el A2

// SENSOR DERECHO CENTRO en el A1

// SENSOR DERECHO DERECHO en el A0

// Es necesario instalar la librería AFMotor.h para que funcione correctamente este Sketch

int SDD=A0;

int SDC=A1;

int SIC=A2;

int SII=A3;

void setup() {

    Serial.begin(9600);

    pinMode(SDD,INPUT);

    pinMode(SDC,INPUT);

    pinMode(SIC,INPUT);

    pinMode(SII,INPUT);

}

void loop() {

    int valorSDD=digitalRead(A0);

    if(valorSDD==1){

        Serial.println("valorSDD es blanco");

    }else{

        Serial.println("valorSDD es negro");

    }

}
```

```
delay(1000);

int valorSDC=digitalRead(A1);

if(valorSDC==1){

Serial.println("valorSDC es blanco");

}else{

    Serial.println("valorSDC es negro");

}

delay(1000);

int valorSIC=digitalRead(A2);

if(valorSIC==1){

Serial.println("valorSIC es blanco");

}else{

    Serial.println("valorSIC es negro");

}

delay(1000);

int valorSII=digitalRead(A3);

if(valorSII==1){

Serial.println("valorSII es blanco");

}else{

    Serial.println("valorSII es negro");

}

delay(1000);

}
```


Sketch 5: Seguidor de línea básico.

En esta actividad únicamente utilizaremos los **dos sensores centrales** del VelociBotCBR, el SIC y el SDC, con ello conseguiremos que el robot siga una línea negra de 2 cm de grosor sobre un fondo blanco.

A partir de este momento el funcionamiento del VelociBotCBR se puede mejorar utilizando sus 4 sensores y cambiando las velocidades de los motores, pero eso lo dejamos en tus manos...

```
// SEGUIDOR DE LÍNEA SENCILLO VelociBOTCBR

// MOTOR IZQUIERDO (MI) es el M1

// MOTOR DERECHO (MD) es el M4

// SENSOR IZQUIERDO IZQUIERDO en el A3

// SENSOR IZQUIERDO CENTRO en el A2

// SENSOR DERECHO CENTRO en el A1

// SENSOR DERECHO DERECHO en el A0

// Es necesario instalar la librería AFMotor.h para que funcione correctamente este Sketch

#include <AFMotor.h>

int SII=A3;  // Definimos el SENSOR IZQUIERDO IZQUIERDO conexión A3

int SIC=A2;

int SDC=A1;

int SDD=A0;

AF_DCMotor motorMD(4);    // Definimos el Motor Derecho en la conexión M4

AF_DCMotor motorMI(1);    // Definimos el Motor Izquierdo en la conexión M1

void setup() {

  pinMode(SII, INPUT);

  pinMode(SIC, INPUT);

  pinMode(SDC, INPUT);

  pinMode(SDD, INPUT);
```

```
motorMD.setSpeed(250);    // Set de velocidad MD fijada en 250 (esta velocidad se puede variar
desde 0 (parado) a 255 (velocidad máxima) por la salida PWM)

motorMI.setSpeed(250);    // Set de velocidad MI fijada en 250 (esta velocidad se puede variar
desde 0 (parado) a 255 (velocidad máxima) por la salida PWM)

motorMD.run(RELEASE);    // Motor Derecho preparado

motorMI.run(RELEASE);    // Motor Izquierdo preparado

}

void loop() {

  int SII = digitalRead(A3);

  int SIC = digitalRead(A2);

  int SDC = digitalRead(A1);

  int SDD = digitalRead(A0);

  if ((SIC==LOW) && (SDC==LOW)){    //Si el Sensor SIC detecta negro y el Sensor SDC detecta negro

    ADELANTE();    // Hacia adelante.

  }

  else if ((SIC==LOW) && (SDC==HIGH)){ //Si el Sensor SIC detecta negro y el Sensor SDC detecta
blanco

    GIROIZQUIERDA();    // Gira a la Izquierda

  }

  else if ((SIC==HIGH) && (SDC==LOW)){ //Si el Sensor SIC detecta blanco y el Sensor SDC detecta
negro

    GIRODERECHA();    // Gira a la derecha

  }

  else{    // Si no hace nada de lo anterior

    PARO();    // Paro.

  }

}

void ADELANTE(){    // SUBROUTINA ADELANTE.

  motorMD.run(FORWARD);    // Motor Derecho hacia delante (si gira al revés cambiamos la
polaridad en la placa)
```

```
motorMI.run(FORWARD);    // Motor Izquierdo hacia delante (si gira al revés cambiamos la
polaridad en la placa)

motorMD.setSpeed(255);    // Set de velocidad MD fijada en 255

motorMI.setSpeed(255);    // Set de velocidad MI fijada en 255

}

void PARO(){              //SUBROUTINA PARO.

motorMD.run(FORWARD);     // Motor Derecho hacia delante

motorMI.run(FORWARD);     // Motor Izquierdo hacia delante

motorMD.setSpeed(0);      // Set de velocidad MD fijada en 0. El motorD se para.

motorMI.setSpeed(0);      // Set de velocidad MI fijada en 0. El motorI se para.

}

void GIROIZQUIERDA(){     //SUBROUTINA GIRO IZQUIERDA.

motorMD.run(FORWARD);     // Motor Derecho hacia delante (si gira al revés cambiamos la
polaridad en la placa)

motorMI.run(FORWARD);     // Motor Izquierdo hacia delante (si gira al revés cambiamos la
polaridad en la placa)

motorMD.setSpeed(255);    // Set de velocidad MD fijada en 255

motorMI.setSpeed(0);      // Set de velocidad MI fijada en 0. El motorI se para.

}

void GIRODERECHA(){       //SUBROUTINA GIRO DERECHA.

motorMD.run(FORWARD);     // Motor Derecho hacia delante (si gira al revés cambiamos la
polaridad en la placa)

motorMI.run(FORWARD);     // Motor Izquierdo hacia delante (si gira al revés cambiamos la
polaridad en la placa)

motorMD.setSpeed(0);      // Set de velocidad MD fijada en 0.El motorD se para

motorMI.setSpeed(255);    // Set de velocidad MI fijada en 255.

}

void ATRAS(){             //SUBROUTINA ATRAS.

motorMD.run(BACKWARD);    // Motor Derecho hacia atras (si gira al revés cambiamos la polaridad
en la placa)
```

```
motorMI.run(BACKWARD); // Motor Izquierdo hacia atras (si gira al revés cambiamos la polaridad
en la placa)

motorMI.setSpeed(255); // Set de velocidad MI fijada en 255

motorMD.setSpeed(255); // Set de velocidad MD fijada en 255

}
```

En cantabRobots.es puedes descargar toda la información del VelociBotCBR.

[Pincha aquí.](#)

Las carpetas de documentos son las siguientes:

- .- Diseño. Archivos .skp y .stl
- .- Librería.
- .- Documentación. PDF con los datasheets de driver, sensores...
- .- Todos los sketches del manual.