Práctica - Sensores II

Daniela Alejandra Córdova Porta

Richard Junior Mercado Correa

UNIVERSIDAD COMPLUTENSE DE MADRID

Madrid, enero de 2021

Resumen

Este documento expone el procedimiento y análisis realizado en la práctica 4 - Sensores II de la asignatura Robótica en la Universidad Complutense de Madrid. En primer lugar, se establecen los objetivos de la práctica y su fundamento teórico. Luego, se establece el procedimiento para realizarla; explicando las conexiones y código necesarios. Finalmente se discuten los resultados obtenidos y se exponen los errores cometidos.

Índice

1. Objetivos

- Montaje de los sensores que se usarán en el robot.
- Uso y manejo del conversor A/D MCP3008 con la RaspBerry Pi
- Uso básico de sensores con la RaspBerry Pi.

2. Fundamento Teórico

Los sensores son dispositivos que permiten medir variables internas o externas del robot. Esto permite que el comportamiento de nuestro robot cambie según los valores detectados por estos sensores. Estos dispositivos pueden ser internos, si miden variables propias del robot, o externos, si detectan variables ajenas al robot. En esta práctica se utilizarán sensores externos de proximidad.

Los sensores utilizados son:

2.1. Sensor Analógico (GP2D12 o GD2Y0a41SK0F)

En sensor GP2D12 o GD2Y0a41SK0F es un sensor que mide las distancias mediante infrarrojos y devuelve con una salida analógica dependiendo de la distancia medida. Su salida es de forma continua y se obtienen valores cada 32 ms. Necesita como voltaje 5 voltios y el valor analógico es devuelto en su salida Vo (que corresponde al cable amarrillo). Se utiliza como sensor de detector de obstáculos muy próximos.

2.2. Sensor digital (CNY70)

El CNY70 es un sensor óptico reflexivo de infrarrojos de corto alcance (de 0,3 a 10 mm) basado en un diodo emisor de luz infrarroja y un receptor formado por un fototransistor, ambos apuntando a la misma dirección. Su funcionamiento se basa en la capacidad de reflexión del objeto, y la detección del rayo reflejado por el receptor.

Tiene un alcance muy corto, por lo que no se suele utilizar para detectar obstáculos. Se utilizará este sensor para la detección y seguimiento de líneas.

El circuito implementado consiste en que cuando el sensor está sobre la línea negra, el sensor permanece en corte, devolviendo un valor muy alto a la salida en el colector, y si se encuentra sobre un fondo blanco, el haz infrarrojo se refleja y el fototransistor se satura entregando a la salida del circuito un nivel bajo.

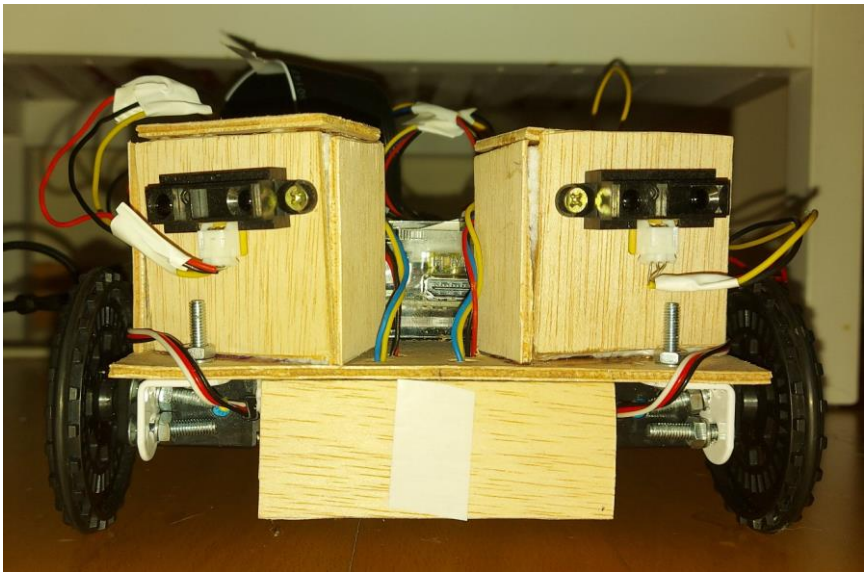
3. Procedimiento

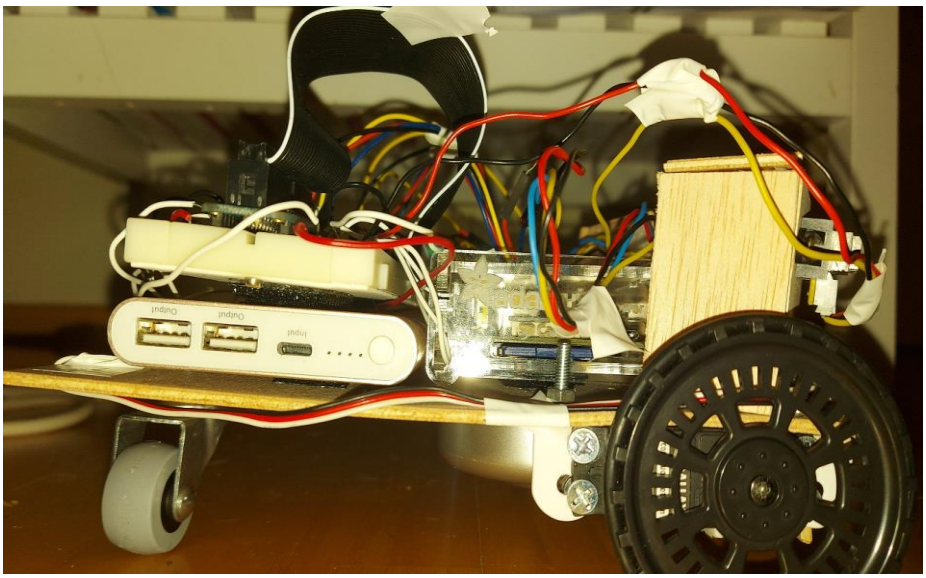
Conectamos la RaspBerry Pi a través del cable Ethernet y usamos el VNC viewer para poder realizar la práctica.

3.1. Montaje de los sensores en el robot.

Se montaron los sensores usando el dibujo plasmado en el archivo de la práctica sensores II.

Siendo el resultado el siguiente:

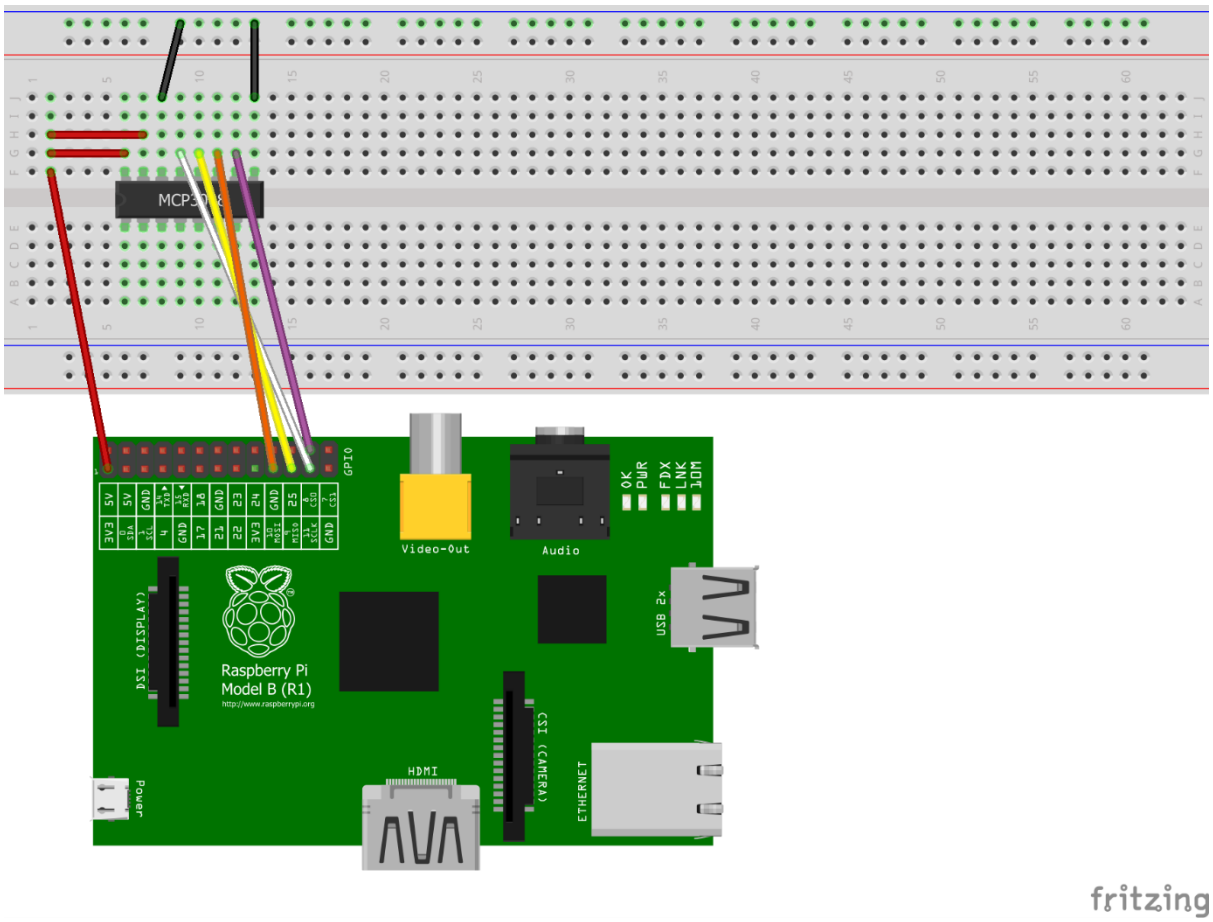




Los sensores infrarrojos CNY70 se encuentran dirigiendo la luz infrarroja hacia el piso. Los dos tipos de sensores (GP2D12 y CNY70) están encajados en cubos de poliestireno cubiertos con madera.

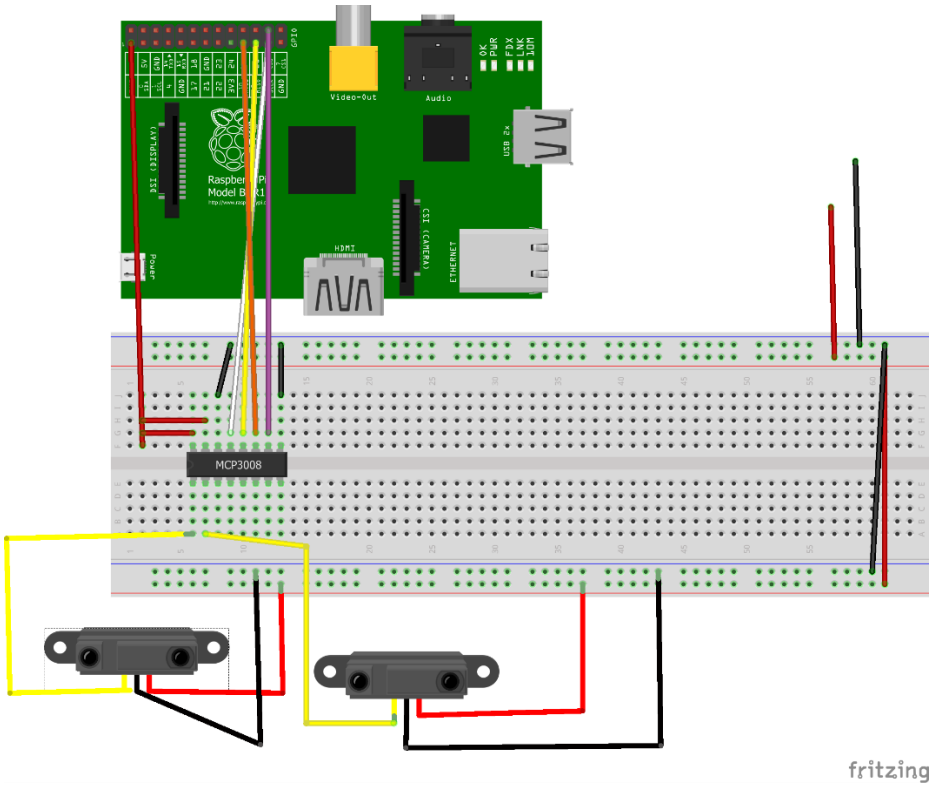
3.2. Sensores en el robot.

Para los dos sensores del robot es necesario realizar el siguiente circuito para el convertidor analógico MCP3008



3.2.1. Sensor Analógico (GP2D12 o GD2Y0a41SK0F):

Construimos el siguiente circuito y conectamos la salida del cable amarillo del sensor del lado derecho con el canal CHO del MCP3008 y el sensor del lado izquierdo con el canal CH1.

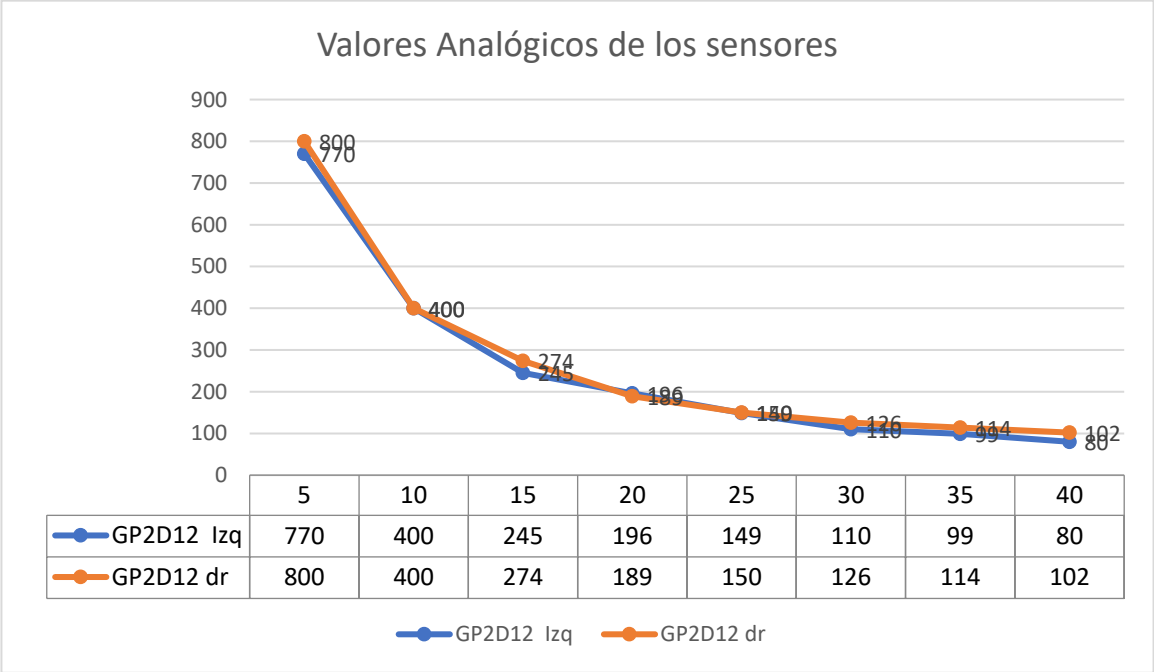


3.2.1.1. Programa para obtener los valores:

Luego se programó un código que obtenga los valores analógicos usando como entrada el pin 100 para el sensor derecho y el pin 101 para el sensor izquierdo. Luego con la función analogRead de la librería se pasan estos números de pin y el valor que devuelve cada llamada es el que se imprime en la pantalla.

```
while(1){
   valor1=analogRead (PIN_GP2_izq) ;
   valor2=analogRead (PIN_GP2_dr) ;
  printf("Valor iz: %d\n", valor1);
 printf("Valor dr: %d\n", valor2);
}
```

3.2.1.2. Valores obtenidos:

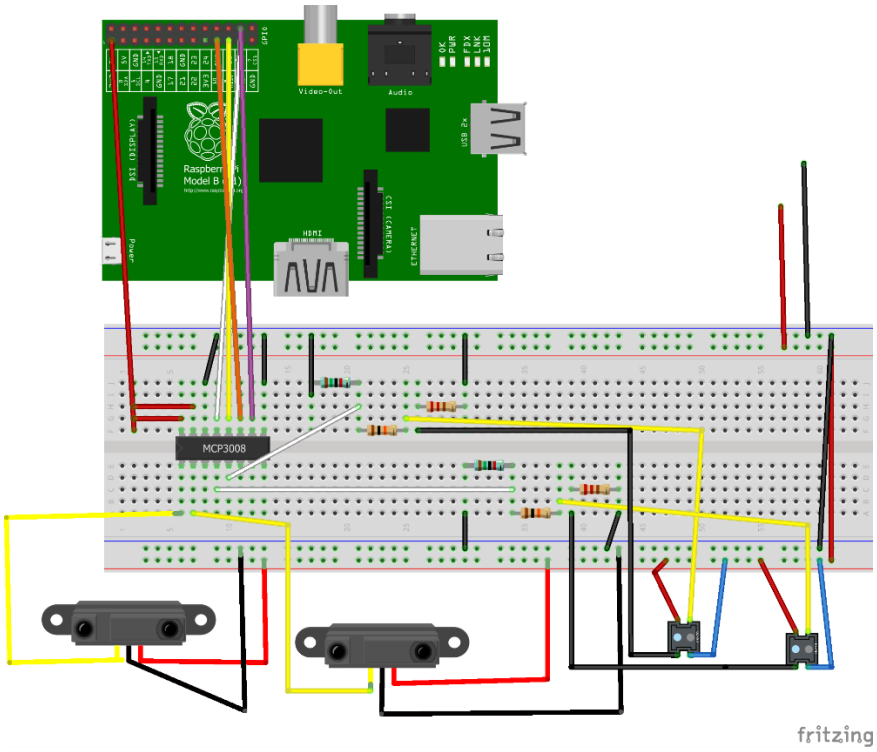


	5	10	15	20	25	30	35	40
GP2D12 Izq	770	400	245	196	149	110	99	80
GP2D12 dr	800	400	274	189	150	126	114	102

GP2D12 Izq GP2D12 dr

3.2.2. Sensor digital (CNY70):

Como en el anterior circuito, queremos analizar los valores que se obtienen con el sensor CNY70 a través del convertidor analógico MCP3008. Conectamos el sensor con las resistencias necesarias y la salida entre las resistencias 10K ohm y 15K ohm con uno de los canales del convertidor. El sensor del lado izquierdo se conecta con el canal CH3 y el lado derecho con el canal CH2.



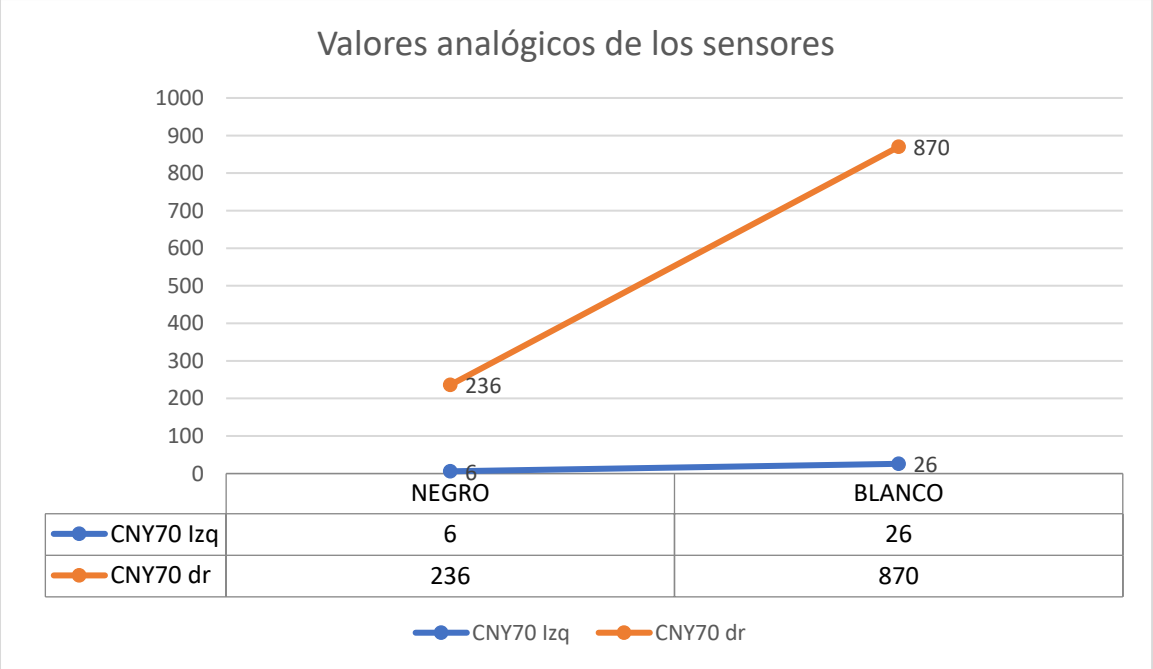
3.2.2.1. Programa para obtener los valores:

Luego se programó un código que obtenga los valores analógicos usando como entrada el pin 102 para el sensor derecho y el pin 103 para el sensor izquierdo. Luego con la función analogRead de la librería se pasan estos números de pin y el valor que devuelve cada llamada es el que se imprime en pantalla.

```
while(1){
  int valorDR=analogRead (PIN_CNY_dr) ;
  int valorIZ=analogRead (PIN_CNY_izq) ;

  printf("Valor IZQ: %d\n", valorIZ);
  printf("Valor DR: %d\n", valorDR);
  }
```

3.2.2.2. Valores obtenidos:



4. Pruebas y análisis de resultados

Con los programas ubicados en el apéndice, siguiendo la misma idea que usamos para obtener los valores en los apartados anteriores, se hacen las siguientes pruebas:

4.1. Sensores CNY70

Si alguno de los dos sensores no está recibiendo señal (está en el aire), recibe valores analógicos muy pequeños, menores a 4. Luego para estos valores se coloca que, si es para los dos, los dos motores avancen. Si uno de ellos recibe señal y el otro no, el que recibe señal detiene su motor correspondiente. La señal es cualquier valor superior a 4 pero en estos sensores, el sensor izquierdo tiene un rango pequeño, siendo este de [0, 30]. Por ello decimos que para cualquier valor superior a 10, está recibiendo una señal. En el caso del derecho, el rango es mucho más mayor [0, 900]. En este sensor, apenas recibe una señal, recibe un valor analógico mayor a 850; por lo cual establecimos ese valor como aquel que define que obtiene una señal.

Ejecutando el algoritmo se obtiene el siguiente resultado que se puede ver en el video:
https://drive.google.com/file/d/1YB_zlSTDAyOn8VL8UjguiuOMpI4UnUIe/view?usp=sharing

4.2. Sensores GP2xx

Usando los valores obtenidos anteriormente, se hizo un array de struct con dos valores:

- **Key:** corresponde a los valores analógicos que se obtienen
- **Values**: es la distancia que corresponde con ese valor.

En el algoritmo obtenemos los valores analógicos y luego con búsqueda binaria recorremos el array para obtener el índice en el array donde se ubica este valor y así saber cuál es la distancia para ese valor analógico. Esto se hace para los dos sensores, cada sensor tiene su array. Si alguno de estos tiene una distancia menor o igual a 20, paramos los motores.

Ejecutando el algoritmo se obtiene el siguiente resultado que se puede ver en el video:
https://drive.google.com/file/d/1YByqHGj4y6G3wlDjYHTGzYhwN2vyrKa8/view?usp=sharing

5. Discusión y conclusiones

5.1. Conclusiones:

- En el sensor GP2xx, a mayor distancia entre el obstáculo, menores valores analógicos devuelve
- En el sensor CNY70, si no recibe señal o incide en el color negro, los valores son muy pequeños. A mayor claridad al reflejarse, mayor es el valor analógico obtenido.

5.2. Errores:

No tuvimos errores en la realización de este experimento, pero reconocemos que pudo haber surgido varios problemas si no conectamos las tierras correctamente o si leíamos dónde no se debe los datos de los sensores. Igualmente, los sensores deben ser ubicados apropiadamente ya que si no los valores no son los mismos, el robot no funcionaría adecuadamente.

6. Bibliografía

- PRÁCTICA: Programación de sensores en el robot. (2020). *Práctica – Sensores II*.

7. Apéndice:

7.1. Programa para sensores CNY70

```
#include <stdlib.h>
#include <softPwm.h>
#include <stdio.h>
#include <errno.h>
#include <string.h>
#include <wiringPi.h>
#include <softPwm.h>
#include <wiringPiSPI.h>
#include <mcp3004.h>

#define BASE 100
#define SPI_CHAN 0
#define RANGE 100

#define PIN0 0
#define PIN1 1

#define PIN_CNY_izq 103
#define PIN_CNY_dr 102

int main(void){


   mcp3004Setup (BASE, SPI_CHAN); // 3004 and 3008 are the same 4/8 channels

   wiringPiSetup();

   softPwmCreate(PIN0, 0, RANGE);
   softPwmCreate(PIN1, 0, RANGE);

   while(1){
   int valorDR=analogRead (PIN_CNY_dr) ;
   int valorIZ=analogRead (PIN_CNY_izq) ;

   printf("Valor IZQ: %d\n", valorIZ);
   printf("Valor DR: %d\n", valorDR);
```

```
  if(valorDR<=10 && valorIZ <=2){
     softPwmWrite (0, 10) ;
     softPwmWrite (1, 35) ;
  }
  else if(valorDR>10 && valorIZ >2){
        softPwmWrite (0, 0) ;
     softPwmWrite (1, 0) ;
  }
  else if (valorIZ <=2 && valorDR>10){

     softPwmWrite (0, 10) ;  //Izq
     softPwmWrite (1, 0) ;

  }
  else if(valorDR<=10&&valorIZ >2 )
  {

     softPwmWrite (0, 10) ;
     softPwmWrite (1, 35) ;   //Dr

  }
  else
  {
     softPwmWrite (0, 0) ;
     softPwmWrite (1, 0) ;   //Dr

  }
  }

  return 0;
}
```

7.2. Programa para sensores GP2xxx

```
#include <stdlib.h>
#include <stdio.h>
#include <errno.h>
#include <string.h>

#include <wiringPi.h>
#include <softPwm.h>
#include <wiringPiSPI.h>
#include <mcp3004.h>

#define BASE 100
#define SPI_CHAN 0
#define RANGE 100

#define PIN0 0
#define PIN1 1

#define PIN_GP2_izq 101
#define PIN_GP2_dr 100
#define SIZE_OF_MAP 20


struct Map {
   int key;
   int value;
};
struct Map mapDr[SIZE_OF_MAP]= {{52, 60}, {98, 50}, {102, 40}, {114, 35}, {126, 30}, {150, 25}, {1
80, 20}, {230, 17}, {274, 15},
          {332, 13}, {400, 10}, {624, 7}, {674, 6}, {800, 5}, {860, 4}, {950, 3}};
struct Map mapIz[SIZE_OF_MAP]= {{52, 60}, {68, 50}, {80, 40}, {99, 35}, {110, 30}, {149, 25}, {196,
20}, {220, 17}, {245, 15},
          {332, 13}, {400, 10}, {624, 7}, {674, 6}, {770, 5}, {860, 4}, {950, 3}};


int binarySearch(struct Map arr[], int l, int r, int x, int n) ;
int main(void){
```

```
    int n = sizeof(map1) / sizeof(map1[0]);

    mcp3004Setup (BASE, SPI_CHAN); // 3004 and 3008 are the same 4/8 channels

    wiringPiSetup();

    softPwmCreate(PIN0, 0, RANGE);
    softPwmCreate(PIN1, 0, RANGE);
    int valor1, valor2, indexIz=0, indexDr=0;

    while(1){
    valor1=analogRead (PIN_GP2_izq) ;
    valor2=analogRead (PIN_GP2_dr) ;


    indexIz = binarySearch(mapIz, 0, n-1, valor1, n);
    indexDr = binarySearch(mapDr, 0, n-1, valor2, n);
    //printf("Valor iz: %d\n", valor1);
    //printf("Valor dr: %d\n", valor2);

    printf("valorIzq: %d\n", mapIz[indexIz].value);
    printf("valorDr: %d\n", mapDr[indexDr].value);
    if(mapIz[indexIz].value<= 20 || mapDr[indexDr].value<= 20){
       softPwmWrite (0, 0) ;
        softPwmWrite (1, 0) ;
    }
    else {
     softPwmWrite (0, 10) ;
     softPwmWrite (1, 38) ;
    }

    }

    return 0;
}

int binarySearch(struct Map arr[], int l, int r, int x, int n)
{
```

```
    if (r >= l) {
        int mid = l + (r - l) / 2;

        // If the element is present at the middle
        // itself
        if (arr[mid].key <= x && mid<n && arr[mid+1].key > x)
            return mid;
        else if (arr[mid].key >=x && mid==0)
            return mid;
        else if (arr[mid].key<=x && mid==n-1)
            return mid;
        // If element is smaller than mid, then
        // it can only be present in left subarray
        if (arr[mid].key > x)
            return binarySearch(arr, l, mid - 1, x, n);

        // Else the element can only be present
        // in right subarray
        return binarySearch(arr, mid + 1, r, x,n);
    }

    // We reach here when element is not
    // present in array
    return -1;
}
```