"MANEJO DE SENSORES Y CONVERSIÓN A/D EN ARDUINO"

Trabajo Preparatorio N°7
Laboratorio de Sistemas Embebidos

Melanny Cecibel Dávila Pazmiño
Ingeniería en Telecomunicaciones
Facultad de Eléctrica y Electrónica
Quito, Ecuador
melanny.davila@epn.edu.ec

***Abstract*—En el siguiente trabajo preparatorio se presentará el uso de sensores mediante la realización de conversión de señales analógicas a digitales; todo esto se hará mediante el uso del microcontrolador Arduino.**

***Index Terms*—Arduino, sensor, pines, digitalización, muestreo.**

I. Introducción

Es importante aprender sobre la conversión AD en Arduino ya que no todas las señales serán de origen digital, por lo que será necesario realizar una conversión en ellas. Este proceso empieza midiendo la señal analógica, posteriormente se debe muestrear la señal y que esta tenga n componentes. Luego se asignan valores discretos a las muestras, obteniéndose un error de cuantización. Manejar sensores en la placa Arduino permite crear diferentes aplicaciones con propósitos cada vez más aplicados a la realidad por ejemplo, a la crisis sanitaria que se esta viviendo actualmente, gracias al uso de un sensor de temperatura y otros en complemento es posible generar un dispensador de gel automático.

II. Objetivos

- Relacionar al estudiante con el uso y manejo de sensores en Arduino.
- Relacionar al estudiante con el uso y manejo del conversor Analógico Digital de la Placa Arduino Uno.
- Utilizar diferentes sensores para resolver problemas cotidianos mediante sistemas de control [2].

III. Cuestionario

A. Consultar las siguientes características de funcionamiento para un conversor analógico – digital:

- *Resolución:* Cantidad de valores discretos en los que un ADC puede traducir una señal análoga a digital. usualmente es medida en base al número de bits que el ADC tiene a su salida. El número de bits representa la cantidad máxima de valores discretos o de pasos que un ADC puede tener. Con 10 bits se tendrían 1024 valores posibles [1].
- *Niveles de Cuantización:* Niveles de cuantización permiten obtener o saber el número de valores distintos que una señal puede tomar en su valor una señal analógica convertida en digital [3].
- *Error de cuantificación:* Se obtiene en base a la diferencia entre el valor de la señal analógica y el valor de la señal digital que fue producida por dicha señal analógica, también es conocido como ruido de cuantización.
- *Linealidad:* Desviación de una línea recta a la curva que representa la magnitud de salida en función de la magnitud de entrada.
- *Precisión:* Es una medida de similitud, se refiere a la dispersión del conjunto de valores obtenidos de mediciones repetidas de una magnitud. Cuanto menor es la dispersión mayor la precisión.
- *Tiempo de asentamiento:* Es el tiempo que se requiere para que la curva de respuesta alcance un rango alrededor del valor final del tamaño especificado por el porcentaje absoluto del valor final.

B. Consultar y describir brevemente el proceso de conversión de señales analógicas a digitales en Arduino Uno. Incluir los comandos necesarios para su implementación.

El microcontrolador de Arduino UNO contiene internamente un conversor analógico a digital de 6 canales. El conversor tiene una resolución de 10 bits, devolviendo enteros entre 0 y 1023.

Para esto se debe usar un elemento, sensor que reciba señales analógicas, este lee la señal y va asignándole diferentes valores discretos, llamados niveles de cuantización. El proceso puede ser mejor comprendido con la siguiente figura.

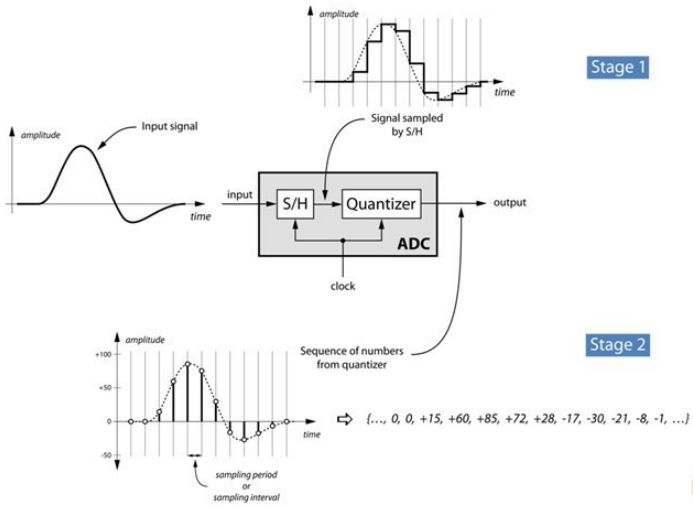

Fig. 1. Esquema de conversión AD.[1]

A continuación, se presentan los comandos que permiten el control de los pines digitales de Arduino:

1) **AnalogRead()**
   Lee el valor del pin analógico especificado. Las placas Arduino contienen un convertidor analógico al digital multicanal de 10 bits. Esto significa que asignará los voltajes de entrada entre 0 y el voltaje de funcionamiento (5V o 3.3V) en valores enteros entre 0 y 1023.

```
analogRead(pin)
```

Fig. 2. Sintaxis de analogRead().[4]

   Devuelve un tipo de dato entero, y se debe especificar el pin de entrada analógica
2) **AnalogReference()**
   Configura el voltaje de referencia utilizado para la entrada analógica (es decir, el valor utilizado como la parte superior del rango de entrada). En Arduino Uno se tiene las siguientes tipos de opciones:
   - DEFAULT: por defecto es 5V o 3.3V.
   - INTERNAL: referencia incorporada de 2.56 Voltios.
   - INTERNAL1V1: referencia incorporado de 1.1V.
   - INTERNAL2V56: referencia incorporada de 2.56V.
   - ETERNAL: voltaje aplicado en el pin AREF(0-5v).

```
analogReference(type)
```

Fig. 3. Sintaxis de analogReference.[5]

3) **AnalogWrite()**
   Escribe un valor analógico (onda PWM) en un pin. Se puede usar para encender un LED con diferentes brillos o conducir un motor a varias velocidades. Después de una llamada a analogWrite(), el pin generará una onda rectangular constante del ciclo de trabajo especificado hasta la próxima llamada

```
analogWrite(pin, value)
```

Fig. 4. Sintaxis de analogWrite.[6]

   Se debe especificar el pin al que se desea escribir un valor de 0 a 255.
4) **Map()**
   Vuelve a mapear un número de un rango a otro. Es decir, un valor de fromLow se asignaría a toLow , un valor de fromHigh a toHigh , valores intermedios a valores intermedios, etc.

```
map(value, fromLow, fromHigh, toLow, toHigh)
```

Fig. 5. Sintaxis de map.[7]

   Los parametros recibidos son value que es el numero a mapear, fromLow que es el limite inferior del rango actual del valor, fromHigh que es el limite superior del rango actual del valor, toLow que es el limite inferior del rango objetivo y toHigh que es el limite superior del rango objetivo.
5) **Constrain()**
   Restringe un número para estar dentro de un rango

```
constrain(x, a, b)
```

Fig. 6. Sintaxis de Constrain.[8]

   X es el número para restringir, a es el extremo inferior del rango, b es el extremo superior del rango.

C. Consultar y describir el principio de funcionamiento y uso de los siguientes sensores: sensor IR, fotoresistor, sensor de Temperatura/Humedad, sensor de Presión, Anemómetro, Radiación UV, Sensor de pluviosidad. Para cada uno de los sensores especificar un dispositivo comercial que realice esta función y obtener la hoja de datos.

- *Sensor IR:* Los sensores de proximidad por infrarrojos IR para evitar obstáculos están compuestos por un transmisor que emite energía de infrarrojos IR y un receptor que detecta la energía IR reflejada por la presencia de cualquier obstáculo en la parte frontal del módulo. El módulo tiene el potenciómetro que permite al usuario ajustar el rango de detección.
  Su principio de funcionamiento se encuentra basado en un detector de infrarrojos, en el espectro electromagnético, la porción infrarroja se divide en tres regiones: cerca de la región infrarroja, la región de infrarrojo medio y la región del infrarrojo lejano.[9]
  El dispositivo comercial es el sensor FC-51, que se basa en un comparador LM393, del cual se obtendrá la hoja de datos.

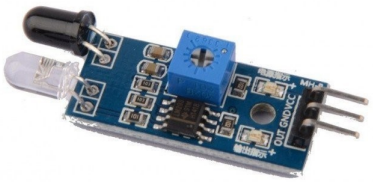

Fig. 7. Sensor FC-51.[9]

- *Fotoresistor:* Es un componente electrónico cuya resistencia varía en función de la luz. Se trata de un sensor que actúa como una resistencia variable en función de la luz que capta. A mayor intensidad de luz, menor resistencia: el sensor ofrece una resistencia de 1M ohm en la oscuridad, alrededor de 10k ohm en exposición de luz ambiente, hasta menos de 1k ohm expuesto a la luz del sol. Su principio de funcionamiento consiste en actuar gracias a un divisor de voltaje.
  Uno se los elementos comerciales que se puede encontrar en esta sección, es la fotoresistencia LDR (GL55)

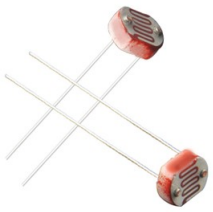

Fig. 8. Fotoresistencia LDR GL55.[10]

- *Sensor de Temperatura/Humedad:* Los sensores DHT11 y DHT22 son sensores digitales de Temperatura y Humedad, fáciles de implementar con cualquier microcontrolador. Utiliza un sensor capacitivo de humedad y un termistor para medir el aire circundante y solo un pin para la lectura de los datos. Tal vez la desventaja de estos es la velocidad de las lecturas y el tiempo que hay que esperar para tomar nuevas lecturas (nueva lectura después de 2 segundos), pero esto no es tan importante puesto que la Temperatura y Humedad son variables que no cambian muy rápido en el tiempo.
  En este caso se tomará como elemento comercial al sensor DHT11.

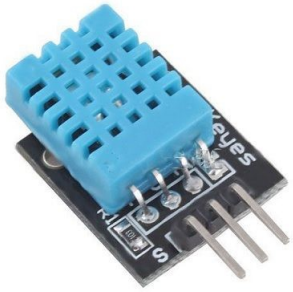

Fig. 9. Sensor DHT11.[11]

- *Sensor de Presión:* En este caso se hará referencia a un sensor de presión diferencial, estos son usados para medir nivel de líquidos, entre otras aplicaciones tiene la señal acondicionada, compensación de temperatura y salida es analógica. Para el caso comercial se analizará al sensor mpx5050dp.

Fig. 10. Sensor de presión diferencial.[12]

- *Anemómetro:* El anemómetro o anemógrafo es un aparato meteorológico que se usa para la predicción del clima y, específicamente, para medir la velocidad del viento. Este instrumento es muy común en las estaciones meteorológicas. Mide fácilmente la velocidad del viento, el sensor entrega un voltaje analógico que es proporcional a la velocidad del viento m/s (metros/segundo).
  En este caso encontrar un sensor comercial, resulta difícil ya que no existe uno “estandarizado”, tras realizar una búsqueda se pudo encontrar al sensor SEN-08942.

Fig. 11. Sensor SEN-08942. [13]

- *Radiación UV:* Este sensor UV, se utiliza para detectar el índice de intensidad ultravioleta (UV). Esta forma de radiación electromagnética tiene longitudes de onda más cortas que la radiación visible y son esas longitudes cortas las que detecta este sensor.
  Este módulo se basa en el sensor UVM-30A, que tiene una amplia gama espectral de 200nm hasta 370nm. La señal eléctrica de salida del módulo es de tipo analógica, que varía respecto a la intensidad de los rayos UV. El nanómetro (nm), es la unidad de longitud que equivale a una mil millonésima parte de un metro. Se utiliza para medir la longitud de onda de la radiación ultravioleta, radiación infrarroja y la luz. [15]

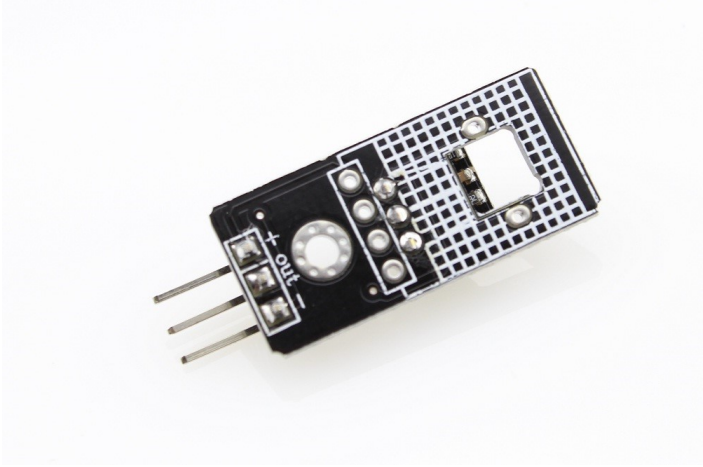

Fig. 12. Sensor de radiación UV. [14]

- *Sensor de pluviosidad:* El sensor en sí son unas placas metálicas recorridas por una corriente. Cuando el agua se deposita encima de las placas hay una conducción de corriente al unirse ambas placas por el agua, un medio conductor. Así se determina si hay agua o no.
  Luego, según la conducción que hay se puede saber si hay mucha agua o poca. Es por eso que uno de los pines ofrece una señal analógica. Al leer esta señal analógica por los pines analógicos de Arduino se puede obtener una señal de 0 (la placa está seca y no hay conducción de corriente) a 1023 (la placa está empapada y ofrece el máximo de corriente que puede dar). [8]
  En la parte comercial se tiene al Sensor FC-37 el cual se encuentra basado en un comparador LM393.

Fig. 13. Sensor FC-37. [15]

D. Consultar las siguientes características relacionadas al uso de sensores en la tarjeta de desarrollo Arduino.

- *Rango de la variable física:* Es un rango en el que la magnitud medida por un sensor puede aplicarse.
- *Rango dinámico de la lectura:* El rango dinámico es la relación del voltaje máximo al voltaje mínimo que el ADC puede convertir.
- *Precisión:* Es una medida de la desviación de la salida analógica a partir del valor previsto para el caso ideal.
- *Sensibilidad:* Mínima magnitud en la señal de entrada requerida para generar una determinada magnitud en la señal de salida.
- *Resolución:* Mínima variación de la magnitud de entrada que puede ser detectada en la salida.
- *Tiempo de respuesta:* Es el tiempo que transcurre desde que se aplica una entrada constante hasta que el sensor produce una respuesta en consecuencia a la salida correspondiente.
- *Tipo de entrada:* En función si es un tipo de sensor que necesita una entrada analógica o digital.
- *Protocolo de comunicación:* Permite que Arduino se comunique con dispositivos serie, para este cometido en ocasiones se hace uso de los pines digitales 0 y 1.

E. En un centro comercial se desea implementar un sistema embebido que controle la temperatura de las personas que ingresan. El circuito que se pretende diseñar estará conformado por dos sensores de temperatura (TMP36), un sensor PIR, una tarjeta Arduino Uno, un arreglo de 4 displays, un buzzer y un servomotor que dispensará el gel antiséptico. El sistema debe diseñarse para cumplir con las siguientes condiciones: para ingresar al centro comercial, el usuario se aproxima al sensor PIR el cual detectará la presencia, esta acción inicia el proceso de medición de su temperatura y la temperatura del ambiente. El registro de temperatura corporal se compara con la temperatura máxima permitida, en caso de que se detecte una temperatura inferior a los 37 °C el sistema activará la máquina que dispensará de gel antiséptico, además aumentará en 1 el número de ingresos correctos. Caso contrario, si la persona registra una temperatura corporal igual o superior a 37°C, se activa una alarma sonará y se mostrará mediante el monitor serial la frase "Ingreso Denegado". En todos los casos, el circuito mediante el arreglo de displays mostrará la temperatura y un mensaje que contiene el total de personas detectadas, el total de ingresos no asintomáticos y el total de accesos denegados se enviará a través de puerto para ser presentado en el monitor serial. El display mostrará la temperatura ambiente cuando no se ha detectado la presencia de una persona.

A continuación, se presenta el esquemático realizado en el software de simulación Tinkercad junto con el código fuente que permite implementar lo solicitado.

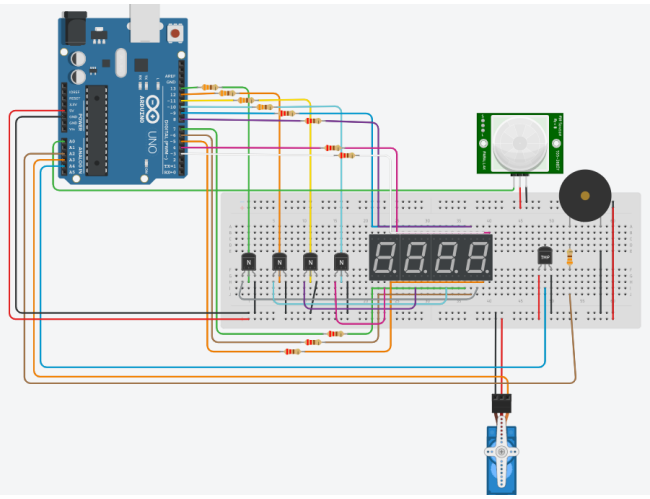

Fig. 14. Esquemático implementado

```
//Trabajo Preparatorio 7: control de ingreso a
    un centro comercial

//Servomotor
#include <Servo.h>

//Display en hexadecimal
int display7c[10]={0x3f,0x06,0x5b,0x4f,0x66,0x6d
    ,0x7d,0x07,0x7f,0x67};
byte a=3;
byte b=4;
byte c=5;
byte d=6;
byte e=7;
byte f=8;
byte g=9;

//Pines de los TBJs
byte t1=10;  //miles
byte t2=11;  //centenas
byte t3=12;  //decenas
byte t4=13;  //unidades
int tempor = 0;//Variable para el barridoo


//Variable auxiliares
float temperatura = 0,aux;
int Personas = 0;
int Ingresos = 0;
int Denegados = 0;
int Medida;
Servo Gel;


void setup()
{
  //Lazo for para la configuracion de los 8
    pines digitales como salida
  for(int i=a;i<=t4;i++)
  {
    pinMode(i,OUTPUT);
  }
  //Declaracioon de los  pines como entradas
  pinMode(A0,INPUT);//Sensor PIR
  pinMode(A1,INPUT);//Sensor Temperatura cliente
  pinMode(A4,INPUT);//Sensor Temperatura cliente
  //Pines de salida
  pinMode(A2,OUTPUT);//Buzzer
  pinMode(2,OUTPUT);//Punto decimal de la
    temperatura
  //Inicio el monitor serial
  Serial.begin(9600);
  //Inicio el servomotor
  Gel.attach(A3);//Pin del servomotor
  Gel.write(0);//Estado inicial del servomotor
}



//Funci n principal
void loop()
{
  //Medicion constante
  Medir();
}

//Funci n que coloca en el puerto de salida los
    bits comenzando
//desde el pin de inicio hasta el pin fin
void puerto(int bits, int ini,int fin)
{
  for(int i=ini;i<=fin;i++)
  {
    digitalWrite(i,bitRead(bits,i-ini));
```

```
  }
 }

//Funcion mostrar
//Se encarga de la multiplexaci n de los displays para
//mostrar el numero tempor
void mostrar()
{
  int dig[4];
  //digito millar
  dig[0]=tempor/1000;
  //digito centena
  dig[1]=(tempor-dig[0]*1000)/100;
  //digito Decena
  dig[2]=(tempor-dig[0]*1000-dig[1]*100)/10;
  //digito Unidad
  dig[3]=(tempor-dig[0]*1000-dig[1]*100-dig[2]*10);

  //Rutina de multiplexacion
  for(int i=t1;i<=t4;i++){
    puerto(display7c[dig[i-t1]],a,g);
    digitalWrite(i,HIGH);  //Encende el display de unidades
    delay(1);
    digitalWrite(i,LOW);   //Apaga el display de unidades
  }
}



//Funcion Mensaje
//Mensaje que se muestra por el monitor serial cuando
//se detecta un nueva persona sensada
void Mensaje()
{
  Serial.println("Sistema de Deteccion de Ingresos");
  Serial.print("Personas detectadas: ");
  Serial.println(Personas);
  Serial.print("Ingresos no asintomaticos: ");
  Serial.println(Ingresos);
  Serial.print("Accesos denegados: ");
  Serial.println(Denegados);
  Serial.println();
}

void Error()
{
  temperatura = 45;
  //Mostramos la temperatura
  tempor = temperatura * 100 ;
  Serial.println("Lectura extrema!!!");
  Serial.println("Llamando a emergencias");
}


//Calculo de la temperatura
void temp(int pin)
{
    //Lectura de la temperatura
    Medida = analogRead(pin);
    //Paso a  C
    aux =  (Medida * 5.0)/1023;
    temperatura = (aux * 100) - 50;
    //Mostramos la temperatura
    tempor = temperatura * 100 ;
}


void Valido()
{
  //se colocar al dispensador en posicion para entregar gel
  Gel.write(90);
  //lazo para mostrar la temperatura mientras haya movimiento
  while(digitalRead(A0) == HIGH)
  {
    digitalWrite(2,HIGH);
    mostrar();
  }
  //Se envia el mensaje de control
  Mensaje();
  tempor = 0;
  //Cierro el Gel
  Gel.write(0);
}

void NoValido()
{
  //Indico que la persona no debe ingresar
  Serial.println("Ingreso Denegado");
  Serial.println();
  //Aumento la cuenta de ingresos denegados
  Denegados = Denegados + 1;
  Personas = Personas + 1;
}



//Funcion Medir
//Se activa cada vez que se detecta movimiento
void Medir()
{

  mostrar();
  //Si se detecta movimeinto se procede a sensar
  if (digitalRead(A0) == HIGH)
  {
    temp(A1);
    //Si la temperatura es menor a 37 C
    if(temperatura < 37)
    {

      //Se aumentan los contadores
      Personas = Personas + 1;
      Ingresos = Ingresos + 1;

      Valido();
    }
    //en caso de que la temperatura sea mayor a 37 grados
    //y no se tenga una alarma activa
    else if (temperatura>=45)
    {
      Error();
      while(digitalRead(A0) == HIGH)
      {
        //Prendo la alarma
        tone(A2,330);
        //Prendo el punto de los decimales
        digitalWrite(2,HIGH);
        mostrar();
      }
      tempor = 0;
      //Apago la alarma
      noTone(A2);

    }
    else if (temperatura>=37)
    {
      NoValido();
      //lazo para mostrar la temperatura
```

```
    mientras haya movimiento
      while(digitalRead(A0) == HIGH)
      {
        //Prendo la alarma
        tone(A2,330);
        //Prendo el punto de los decimales
        digitalWrite(2,HIGH);
        mostrar();
      }
      //se envia el mensaje de control
      Mensaje();
      tempor = 0;
      //Apago la alarma
      noTone(A2);
    }
  }
  else
  {
    temp(A4);
    if(temperatura>=45)
    {
      Error();
      tone(A2,330);
    }
    else
    {
      noTone(A2);
    }
  }
}
```

Código 1: Implementación del sistema de control de temperatura

REFERENCES

[1] '"Conversor Analógico Digital – Aprendiendo Arduino", Aprendiendo Arduino, 2020. [Online]. Disponible en: https://aprendiendoarduino.wordpress.com/tag/conversor-analogico-digital/. [Acced: 18- Enero- 2021].

[2] E. Espinosa, E. Tatayo, "MANEJO DE SENSORES Y CONVERSIÓN A/D EN ARDUINO". C.P. SISTEMAS EMBEBIDOS, Accedido: ene. 18, 2020. [En línea].

[3] Metas.com.mx, 2020. [Online]. Disponible en: http://www.metas.com.mx/guiametas/La-Guia-MetAs-08-01-linealidad.pdf. [Accedido: 18- Enero- 2021].

[4] W. ATmega32A? and M. Atef, "What is the difference between ADC single conversion mode and free running mode in ATmega32A?", Electrical Engineering Stack Exchange, 2020. [Online]. Available: https://electronics.stackexchange.com/questions/451436/what-is-the-difference-between-adc-single-conversion-mode-and-free-running-mode. [Accedido: 18- Enero- 2021].

[5] "analogRead() - Arduino Reference", Arduino.cc, 2020. [Online]. Disponible en: https://www.arduino.cc/reference/en/language/functions/analog-io/analogread/. [Acced: 18- Enero- 2021].

[6] "analogReference() - Arduino Reference", Arduino.cc, 2020. [Online]. Disponible en: https://www.arduino.cc/reference/en/language/functions/analog-io/analogreference/. [Accedido: 18- Enero- 2021].

[7] "analogWrite() - Arduino Reference", Arduino.cc, 2020. [Online]. Disponible en: https://www.arduino.cc/reference/en/language/functions/analog-io/analogwrite/. [Accedido: 18- Enero- 2021].

[8] "map() - Arduino Reference", Arduino.cc, 2020. [Online]. Disponible en: https://www.arduino.cc/reference/en/language/functions/math/map/. [Accedido: 18- Enero- 2021].

[9] "constrain() - Arduino Reference", Arduino.cc, 2020. [Online]. Disponible en: https://www.arduino.cc/reference/en/language/functions/math/constrain/. [Accedido: 18- Enero- 2020].

[10] J. Ahedo, "Como usar el módulo sensor de Infrarrojos IR FC-51 para evitar obstáculos con Robot Arduino/Genuino - Web-Robótica.com", Web-robotica.com, 2020. [Online]. Disponible en: https://www.web-robotica.com/arduino/conceptos-basicos-arduino/como-usar-el-modulo-sensor-de-infrarrojos-ir-fc-51-para-evitar-obstaculos-con-robot-arduinogenuino. [Accedido: 18- Enero- 2021].

[11] "Medir nivel de luz con Arduino y fotoresistencia LDR (GL55)", Luis Llamas, 2020. [Online]. Disponible en: https://www.luisllamas.es/medir-nivel-luz-con-arduino-y-fotoresistencia-ldr/. [Accedido: 18- Enero- 2021].

[12] "Tutorial sensor de temperatura y humedad DHT11 y DHT22", Naylampmechatronics.com, 2020. [Online]. Disponible en: https://naylampmechatronics.com/blog/40Tutorial-sensor-de-temperatura-y-humedad-DHT1.html. [Accedido: 18- Enero- 2021].

[13] m. diferencial, "mpx5050dp sensor presion diferencial - DynamoElectronics", DynamoElectronics, 2020. [Online]. Disponible en: https://www.dynamoelectronics.com/tienda/mpx5050dp-sensor-presion-diferencial/. [Accedido: 18- Enero- 2021].

[14] Agspecinfo.com, 2020. [Online]. Disponible en: http://www.agspecinfo.com/pdfs/S/SEN08942.PDF. [Accedido: 18- Enero- 2021].

[15] V. Ventura, "Sensor de radiación ultravioleta con Arduino y UVM30A", polaridad.es, 2020. [Online]. Disponible en: https://polaridad.es/sensor-radiacion-ultravioleta-arduino-indice-uv-uvm30a-guva-s12sd/. [Accedido: 18- Enero- 2021].

[16] "Detector de lluvia con Arduino y sensor FC-37 o YL-83", Luis Llamas, 2020. [Online]. Disponible en: https://www.luisllamas.es/arduino-lluvia/. [Accedido: 18- Enero- 2021].

[17] "Sensores – Aprendiendo Arduino", Aprendiendo Arduino, 2020. [Online]. Disponible en: https://aprendiendoarduino.wordpress.com/category/sensores/. [Accedido: 18- Enero- 2021]