

Robot móvil

Diseño de Computadores Empotrados

Juan Pedro Rodríguez Gracia
Jesús Rodríguez Heras
Gabriel Fernando Sánchez Reina

6 de febrero de 2018

Índice general

I Memoria	7
1. Introducción	9
1.1. Objetivo	9
1.1.1. Movimientos	9
1.1.2. Detección	9
1.1.3. Recogida de información	9
1.1.4. Algoritmo de resolución del laberinto	10
1.1.5. Monitorización de la información	10
1.2. Hardware empleado	10
1.2.1. Arduino Leonardo	10
1.2.2. Ordenador portátil	10
1.2.3. Sensores	11
1.2.4. Actuadores	11
1.2.5. Elementos de comunicación	11
1.2.6. Alimentación	11
1.3. Software empleado	12
1.3.1. Sketchup 2017	12
1.3.2. Makerbot	12
1.3.3. Arduino IDE	12
1.3.4. SublimeText 3	12
1.3.5. Python 2.7 IDE	12
1.3.6. Serial Bluetooth Terminal	12

2. Especificación de requisitos	13
2.1. Requisitos funcionales	13
2.2. Requisitos no funcionales	13
3. Planificación	15
4. Presupuesto	17
5. Análisis	19
5.1. Caso de uso	19
5.2. Diagrama de flujo	20
6. Diseño	21
6.1. Estructura	21
6.1.1. Programa principal	21
6.1.2. Celdas	21
6.1.3. Sensores	21
6.1.4. Actuadores	21
6.2. Plan de pruebas	21
6.2.1. Primeras pruebas	21
6.2.2. Pruebas finales	21
7. Implementación	23
7.1. Librerías	23
7.1.1. Propias	23
7.1.2. Públicas	28
7.2. Apuntes sobre el código	28
8. Montaje	29
8.1. Chasis	29
8.2. Cableado	29
8.3. Imágenes del proyecto	29
9. Pruebas	35

10. Mejoras**37****II Anexo de códigos** **39****11. Códigos** **41**

11.1. Algoritmo	41
11.2. Pruebas	47
11.2.1. Primeras pruebas	47
11.2.2. Pruebas finales	51
11.3. Monitorización	53

Parte I

Memoria

Capítulo 1

Introducción

1.1. Objetivo

El objetivo del proyecto es la creación de un robot móvil capaz de recorrer un laberinto de 5x5 celdas tratando de encontrar la salida y volviendo a la casilla inicial por el camino más rápido una vez que se llega a la casilla final. Las acciones llevadas a cabo por el robot para conseguir su objetivo son:

1.1.1. Movimientos

- Adelante: Se hace girar a los dos motores en el mismo sentido y a la misma velocidad para obtener un movimiento recto en la misma dirección.
- Atrás: Se hace girar a los dos motores en sentido contrario respecto al movimiento .^adelantez a la misma velocidad.
- Izquierda: Se hace girar el motor derecho del robot hacia delante y el izquierdo hacia atrás, ambos a la misma velocidad.
- Derecha: Se hace girar el motor izquierdo del robot hacia delante y el derecho hacia atrás, ambos a la misma velocidad.

1.1.2. Detección

Para la detección de obstáculos se utilizan sensores situados en la parte de delante (ultrasonidos) y a ambos lados del robot (infrarrojos), dejando la parte trasera sin sensores. En la parte de abajo del robot encontramos otros tres sensores (CNY) para la detección de la transición de las casillas del laberinto, dos en cada esquina delantera y uno más en la esquina derecha trasera.

1.1.3. Recogida de información

Para esta tarea se ha implementado una pila en la que se insertan los movimientos que va realizando el robot mediante su recorrido. Al llegar a la casilla final, los movimientos van saliendo de dicha pila (en orden inverso al que entraron) para que el robot llegue de nuevo a la casilla inicial.

1.1.4. Algoritmo de resolución del laberinto

El **algoritmo** que sigue el robot para establecer la prioridad de dirección sigue el algoritmo de la mano derecha estableciendo el orden de prioridad derecha, delante, izquierda y atrás.

1.1.5. Monitorización de la información

Cada vez que el robot llega a una celda envía al portátil la información sobre la misma (mediante bluetooth) y éste la traslada a la **interfaz gráfica** diseñada en python.

1.2. Hardware empleado

Para este proyecto hemos empleado el siguiente hardware.

1.2.1. Arduino Leonardo

Utilizaremos una placa con un microprocesador Arduino Leonardo que cuenta con las siguientes características:

- Microcontrolador Atmega32u4.
- Voltaje de entrada: 7-12V.
- Voltaje de trabajo: 5V.
- Corriente por pin I/O: 40mA.
- 20 pines digitales I/O.
- 7 canales PWM.
- 12 ADC.
- 16MHz de velocidad de reloj.
- Memoria Flash: 32 KB (ATmega32u4) de los cuales 4 KB son usados por el bootloader.
- Memoria SRAM: 2.5 KB (ATmega32u4).
- Memoria EEPROM: 1KB (ATmega32u4).
- Dimensiones: 68.6 x 53.3mm.
- Peso: 20g.

1.2.2. Ordenador portátil

El portátil que se usará para la conexión mediante bluetooth con el robot será un Toshiba Satellite C850 que cuenta con un procesador Intel i3 2310M @2.1GHz x64.

1.2.3. Sensores

Utilizaremos tres tipos de sensores para poder resolver el problema:

1. UltrasonidoS HC-SR03:

- Dimensiones: 45 * 20 * 15mm
- Rango: 2cm - 4m
- Ángulo: 30°
- Ciclo de respuesta: 100ms

2. Infrarrojos SHARP GP2Y0A21YK0F:

- Dimensiones: 29,5 * 13 * 13,5
- Rango: 10 - 80cm
- Voltaje de salida: 0 - 3,3V

3. Óptico reflexivo CNY70:

- Dimensiones: 7 * 7 * 6mm
- Rango: 0 - 2mm

1.2.4. Actuadores

Como actuadores usaremos dos motores micro metal con reductora 30:1.

1. Motores:

- Dimensiones: 24 * 10 * 12mm
- Torque: 0,3kg/cm
- Velocidad de giro sin carga: 440rpm
- Voltaje: 3 - 9V
- Peso: 10 gramos.

1.2.5. Elementos de comunicación

Para la comunicación con el ordenador portátil se usará un módulo bluetooth HC-05 actuando como esclavo que irá dentro del robot.

1. HC-05:

- Voltaje¹ de trabajo: 1,8 - 3,6V I/O.
- Rango: 9m

1.2.6. Alimentación

Para la alimentación del robot usaremos 6 pilas alcalinas tipo AA de 1,5V obteniendo un total de 9V.

¹Debido a este voltaje, nos vemos obligados a usar un circuito reductor de tensión, el cual se implementa con una resistencia y un diodo zener.

1.3. Software empleado

En cuanto a software, hemos usado los siguientes programas:

1.3.1. Sketchup 2017

Es un programa de diseño 3D que nos permite exportar archivos a formato ".stl". Con él hemos diseñado las piezas que alojarán a los sensores y darán un chasis al robot.

1.3.2. Makerbot

Es un programa usado que convierte los archivos ".stl." en instrucciones que seguirá la impresora 3D con el fin de obtener las piezas anteriormente diseñadas con Sketchup.

1.3.3. Arduino IDE

Es un entorno de desarrollo que hemos usado para programar el algoritmo que sacará al robot del laberinto y en general, toda la lógica del programa.

1.3.4. SublimeText 3

Es un editor de texto utilizado para crear las **bibliotecas** que darán soporte al código de Arduino.

1.3.5. Python 2.7 IDE

Es un entorno de desarrollo en el cual implementaremos la monitorización del robot, mostrando a través de una interfaz gráfica el laberinto y el progreso del robot cruzándolo.

1.3.6. Serial Bluetooth Terminal

Es una aplicación para android disponible en Play Store que nos permite tener una conexión serie con una consola en nuestro móvil a través de la antena bluetooth del mismo.

Capítulo 2

Especificación de requisitos

2.1. Requisitos funcionales

El robot es capaz de avanzar hacia delante, situarse en el centro de una casilla, realizar giros de 90º hacia izquierda y derecha.

También es capaz de realizar movimientos en función de la información recabada por los sensores en cada celda. Por lo tanto, está capacitado para salir del laberinto con la información proporcionada por sus propios sensores.

A la vez que va recorriendo el laberinto, el robot va enviando información al ordenador portátil sobre los obstáculos de cada celda, la trayectoria ejecutada y el ordenador portátil es capaz de mostrar una gráfica del laberinto.

La interfaz del ordenador es capaz de recopilar toda la información recabada por el robot durante su recorrido.

2.2. Requisitos no funcionales

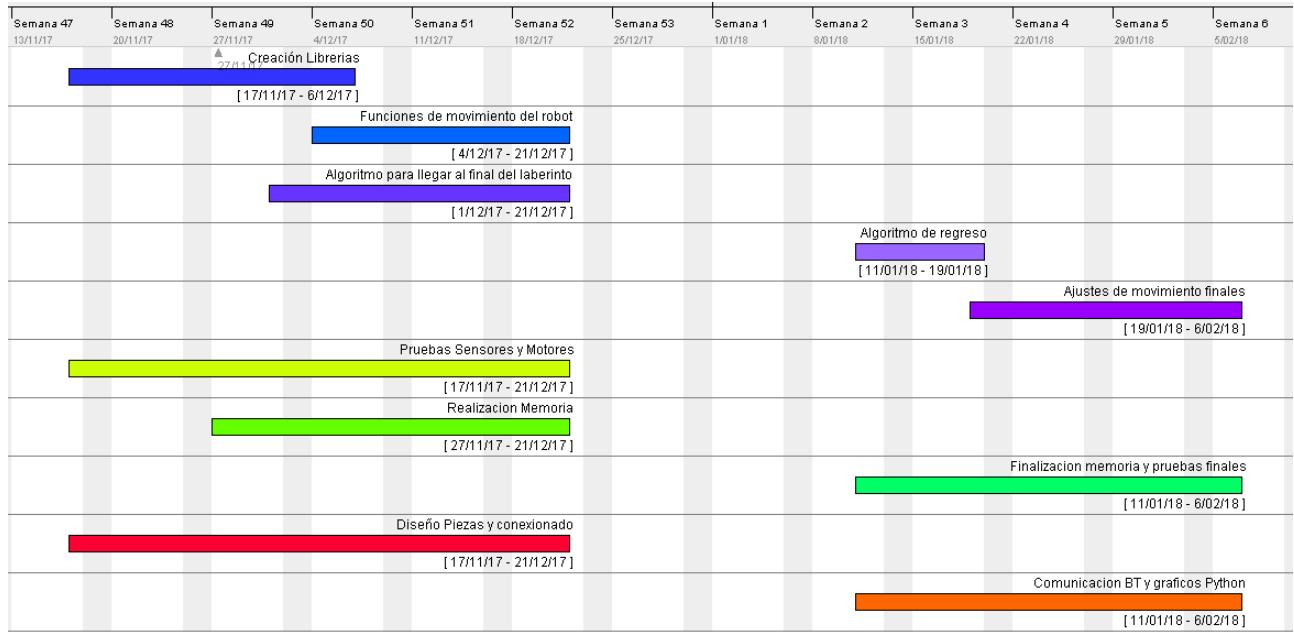
El robot tiene las siguientes medidas: 9.5 x 9.5 x 10cm (largo, ancho, alto).

Cuando el robot se queda sin batería, observamos que se apaga directamente, por lo que es hora de cambiarle las pilas.

Capítulo 3

Planificación

En el siguiente diagrama de Gantt podemos observar la planificación del proyecto desde su inicio hasta su final:



Capítulo 4

Presupuesto

Componente	unidades	Precio total (€)
Arduino Leonardo	1	18
Placa Shield PCB	1	25
Sensor CNY70	3	5.10
Sensor HC-SR04	1	1.49
Cables	Varios	1.20
Módulo bluetooth GW040	1	7.73
Motores Micro Metal	2	27.80
Sensor SHARP	2	21.58
Plástico	20g	1.46
Pilas	6	7.33
Horas	180	20€/hora

Tabla 4.1: Presupuesto del proyecto.

El presupuesto total ascendería a 3716,69€.

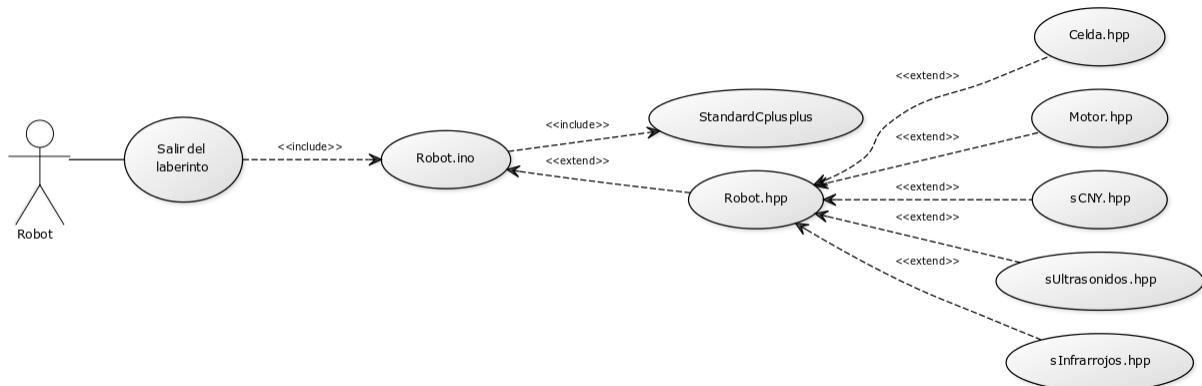
Capítulo 5

Análisis

5.1. Caso de uso

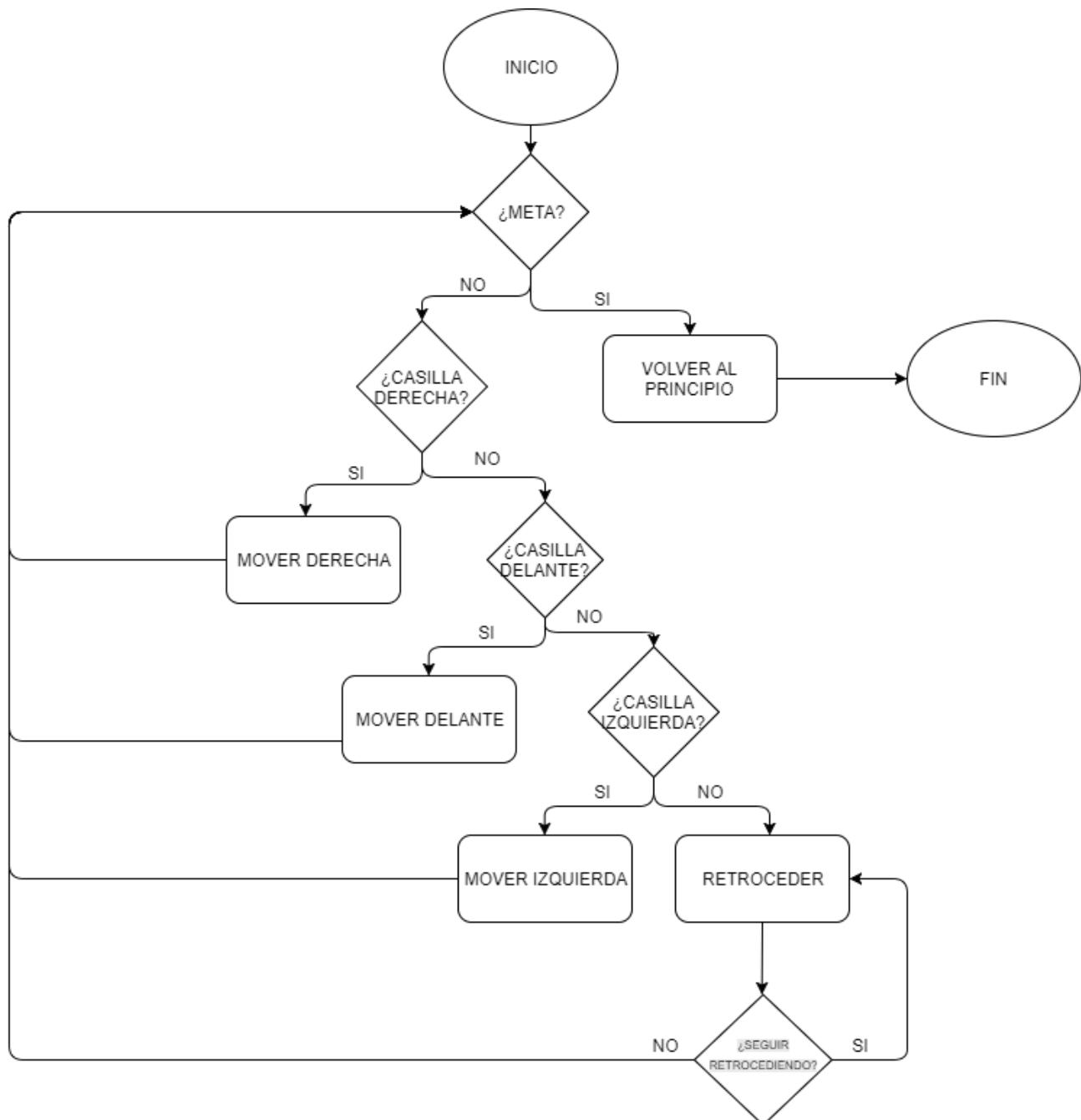
Tenemos un operador, el robot, que necesita salir del laberinto. Para ello, utiliza el archivo `Robot.ino`, que incluye `StandardCplusplus` y extiende de `Robot.hpp`, que a su vez, extiende de las librerías usadas para la implementación del código:

- **Celda.hpp**: De esta librería se extiende la implementación de las casillas y el manejo de las mismas.
- **Motor.hpp**: De esta librería se extiende el control sobre los motores.
- **sCNY.hpp**: De esta librería se extiende el control de los sensores CNY.
- **sUltrasonidos.hpp**: De esta librería se extiende el control del sensor de ultrasonidos delantero.
- **sInfrarrojos.hpp**: De esta librería se extiende el control de los sensores de infrarrojos laterales.



5.2. Diagrama de flujo

En el siguiente diagrama de flujo se ilustra el algoritmo que se ha implementado para intentar salir del laberinto.



Capítulo 6

Diseño

6.1. Estructura

6.1.1. Programa principal

Para el desarrollo de los códigos, seguimos una estructura tal que, a partir del archivo `Robot.ino`, se acceda al resto de librerías que controlarán el robot.

En la librería `Robot.hpp` se encuentra todo lo necesario para definir los pines y las funciones necesarias para el control de los sensores y motores.

6.1.2. Celdas

Para el mapeado de las celdas utilizamos la librería `celda.hpp`.

6.1.3. Sensores

Para el control de los sensores, utilizamos una librería por cada tipo de sensor, de tal forma que tenemos tres librerías: `sUltrasonidos.hpp`, `sInfrarrojos.hpp` y `sCNY.hpp`.

6.1.4. Actuadores

Para el control de los motores hemos usado la librería `motor.hpp`.

6.2. Plan de pruebas

6.2.1. Primeras pruebas

Al principio comenzamos con las **pruebas individuales** de cada sensor, de los motores y de la conexión bluetooth usando los códigos de las prácticas de la asignatura para comprobar su correcto funcionamiento.

6.2.2. Pruebas finales

Una vez que los sensores estuvieron calibrados, pasamos a hacer **pruebas en conjunto** con todos los sensores activos para ir corrigiendo los pequeños fallos que iban surgiendo.

Capítulo 7

Implementación

7.1. Librerías

7.1.1. Propias

Las bibliotecas usadas para dar soporte al código de Arduino son las siguientes:

Robot.hpp

```
1 #ifndef _ROBOT_HPP_
2 #define _ROBOT_HPP_
3
4 #include "celda.hpp"
5 #include "sUltrasonidos.hpp"
6 #include "sInfrarrojos.hpp"
7 #include "sCNY.hpp"
8 #include "motor.hpp"
9
10 // Default values for Robot pins
11
12 #define ULTRA A3
13 #define INFRA_IZQ A4
14 #define INFRA_DCHA A8
15 #define CNY_AT A5
16 #define CNY_IZQ A1
17 #define CNY_DCHA A0
18 #define M_11 10
19 #define M_12 9
20 #define M_21 6
21 #define M_22 5
22
23 class Robot {
24
25 public:
26     Robot(int ultra = ULTRA, int infraIzq = INFRA_IZQ, int infraDcha = INFRA_DCHA, int
```

```

28     int CNYDcha = CNY_DCHA, int M11 = M_11, int M12 = M_12, int M21 = M_21, int M22 = M_22;
29
30     float distanciaDelante() const;
31     float distancialIzq() const;
32     float distanciaDcha() const;
33
34     bool negroAtras() const;
35     bool negroIzq() const;
36     bool negroDcha() const;
37
38 // Testing
39     int numeroAtras() const;
40     int numeroIzq() const;
41     int numeroDcha() const;
42
43     void motorIzqDelante(unsigned v);
44     void motorDchaDelante(unsigned v);
45     void motorIzqDetras(unsigned v);
46     void motorDchaDetras(unsigned v);
47     void motorIzqParar();
48     void motorDchaParar();
49
50
51 private:
52
53
54
55 // Connected sensors and motors
56 sUltrasonidos ultrasonidos;
57 sInfrarrojos sharpIzq;
58 sInfrarrojos sharpDcha;
59
60 sCNY CNYAtras;
61 sCNY CNYIzq;
62 sCNY CNYDcha;
63
64 motor MIzq;
65 motor MDcha;
66 };
67
68 inline Robot::Robot(int ultra, int infraIzq, int infraDcha, int CNYAt, int CNYIzq, int CNYDcha)
69 {
70     ultrasonidos{ultra},
71     sharpIzq{infraIzq},
72     sharpDcha{infraDcha},
73     CNYAtras{CNYAt},
74     CNYIzq{CNYIzq},
75     CNYDcha{CNYDcha},
76     MIzq{M11,M12},
77     MDcha{M21,M22}
78     {}
79
80     inline float Robot::distanciaDelante() const { return ultrasonidos.distancia(); }
81     inline float Robot::distancialIzq() const { return sharpIzq.distancia(); }
82     inline float Robot::distanciaDcha() const { return sharpDcha.distancia(); }

```

```

82
83 inline bool Robot::negroAtras() const { return CNYAtras.negro(); }
84 inline bool Robot::negroIzq() const { return CNYIzq.negro(); }
85 inline bool Robot::negroDcha() const { return CNYDcha.negro(); }
86
87 //Used for testing purposes
88 inline int Robot::numeroAtras() const { return CNYAtras.numero(); }
89 inline int Robot::numeroIzq() const { return CNYIzq.numero(); }
90 inline int Robot::numeroDcha() const { return CNYDcha.numero(); }
91
92 inline void Robot::motorIzqDelante(unsigned v) { MIZQ.delante(v); }
93 inline void Robot::motorDchaDelante(unsigned v) { MDCHA.delante(v); }
94 inline void Robot::motorIzqDetras(unsigned v) { MIZQ.detras(v); }
95 inline void Robot::motorDchaDetras(unsigned v) { MDCHA.detras(v); }
96 inline void Robot::motorIzqParar() { MIZQ.parar(); }
97 inline void Robot::motorDchaParar() { MDCHA.parar(); }
98
99 #endif

```

celda.hpp

```

1 #ifndef _CELDA_HPP_
2 #define _CELDA_HPP_
3
4 //True = there is wall
5 //False = there is no wall
6
7 struct celda
8 {
9     celda(bool del, bool det, bool izq, bool dech);
10
11    celda(const celda&) = default; //Copy constructor
12    celda& operator=(const celda&) = default;
13    celda(celda&&) = default; //Movement constructor1
14    celda& operator=(celda&&) = default;
15
16    bool delante;
17    bool detras;
18    bool izquierda;
19    bool derecha;
20 };
21 //Default cell has no walls around it
22 inline celda::celda(bool del = false, bool det = false, bool izq = false, bool dech = false)
23
24 #endif

```

sUltrasonidos.hpp

```

1 #ifndef _SULTRASONIDOS_HPP_
2 #define _SULTRASONIDOS_HPP_
3
4 class sUltrasonidos {
5
6 public:

```

```

7         sUltrasonidos(int p);
8
9         float distancia();      // returns measured distance in cm
10
11    private:
12        int pin;
13    };
14
15
16    inline sUltrasonidos::sUltrasonidos(int p): pin{p} {}
17
18    inline float sUltrasonidos::distancia() {
19
20        pinMode(pin,OUTPUT);
21
22        digitalWrite(pin,LOW);
23        delayMicroseconds(5);
24
25        digitalWrite(pin,HIGH);
26        delayMicroseconds(10);
27        digitalWrite(pin,LOW);
28
29        pinMode(pin,INPUT);
30
31        return 0.017f * pulseIn(pin,HIGH);
32    }
33
34 }
35
36 #endif

```

sInfrarrojos.hpp

```

1  #ifndef _SINFRARROJOS_HPP_
2  #define _SINFRARROJOS_HPP_
3
4  class sInfrarrojos {
5
6  public:
7
8      sInfrarrojos(int p);
9
10     float distancia();
11
12    private:
13        int pin;
14    };
15
16
17    inline sInfrarrojos::sInfrarrojos(int p): pin{p} { pinMode(p, INPUT); }
18    inline float sInfrarrojos::distancia() {
19
20        float V = analogRead(pin) * 0.00488f;
21

```

```

22         if(V > 0.5f && V < 2.7f)
23             return 12.0f/(V - 0.2f) + 0.42f;
24         else if(V <= 0.5f)
25             return 60.0f;
26         else if(V >= 2.7f)
27             return 8.0f;
28     }
29 }
30
31 #endif
```

sCNY.hpp

```

1 #ifndef _SCNY_HPP_
2 #define _SCNY_HPP_
3
4 #define REF_NEGRO 750 //values above this value are treated as black colour
5
6 class sCNY {
7
8     public:
9
10    sCNY(int p);
11
12    bool negro(); //returns true when the black line is below it
13    int numero(); //testing
14
15     private:
16
17     int pin;
18
19 };
20
21 inline sCNY::sCNY(int p): pin{p} { pinMode(p, INPUT); }
22
23 inline bool sCNY::negro() { return (analogRead(pin) > REF_NEGRO); }
24
25 //testing
26 inline int sCNY::numero() { return analogRead(pin); }
27
28
29 #endif
```

motor.hpp

```

1 #ifndef _MOTOR_HPP_
2 #define _MOTOR_HPP_
3
4 class motor {
5
6     public:
7
8     motor(int p1, int p2);
```

```

10      void delante(unsigned v);           //v is a value between 0 and 255
11      void detras(unsigned v);          //0 = stopped, 255 = full speed
12      void parar();
13
14  private:
15
16      int pin1;
17      int pin2;
18  };
19
20 inline motor::motor(int p1, int p2): pin1{p1}, pin2{p2} { pinMode(p1,OUTPUT); pinMode(p2,OUTPUT);
21
22 inline void motor::delante(unsigned v) {
23
24     if(v > 255)
25         v = 255;
26
27     digitalWrite(pin1,LOW);
28     analogWrite(pin2,v);
29 }
30
31 inline void motor::detras(unsigned v) {
32
33     if(v > 255)
34         v = 255;
35
36     digitalWrite(pin2,LOW);
37     analogWrite(pin1,v);
38 }
39
40 inline void motor::parar() {
41
42     digitalWrite(pin1,LOW);
43     digitalWrite(pin2,LOW);
44 }
45
46 #endif

```

7.1.2. Públicas

StandardCplusplus

La librería StandardCplusplus.hpp la hemos descargado de Internet porque nos hacía falta para la implementación del TAD (tipo abstracto de dato) pila que nos ayudará en la vuelta a la casilla inicial.

7.2. Apuntes sobre el código

En la fase de salir del laberinto, entra en un bucle donde escanea la celda en la que se encuentra y luego va aplicando el algoritmo de la mano derecha hasta llegar a la casilla final.

Una vez que alcanza la meta deshace en orden inverso los movimientos que ha hecho, sin tener en cuenta los retrocesos, para volver a la casilla inicial por el camino más corto posible.

Capítulo 8

Montaje

8.1. Chasis

Para el diseño del chasis, hemos usado una impresora 3D. Las piezas fueron diseñadas en Sketchup 2017 y luego fueron impresas mediante una impresora Makerbot Replicator 2X.

8.2. Cableado

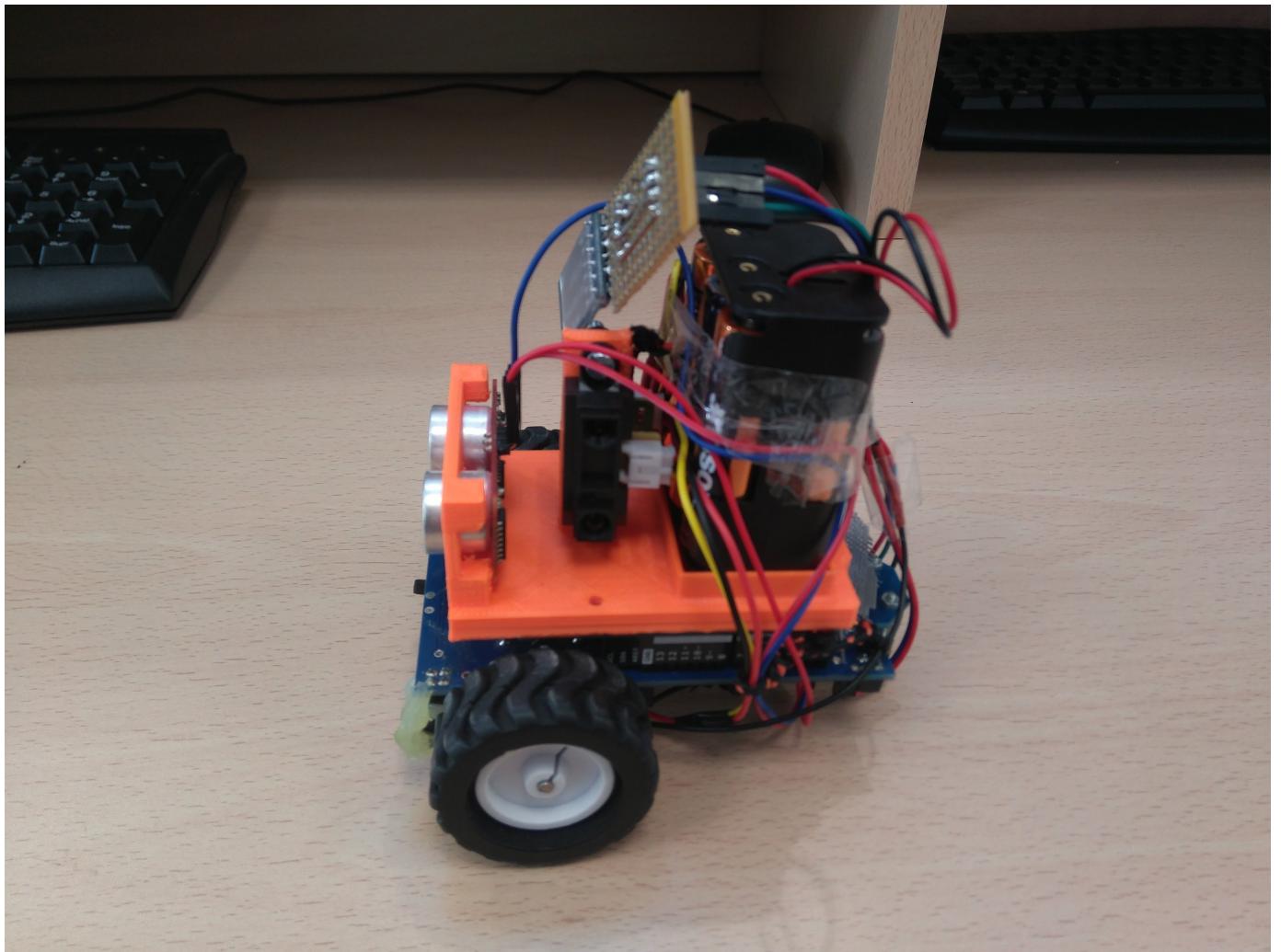
Para el cableado de los sensores hemos usado cables hembra-hembra que conectan los sensores con la placa shield, salvo los CNY70, que ya estaban conectados a la shield. Los motores, al igual que los CNY70, ya estaban conectados a la shield.

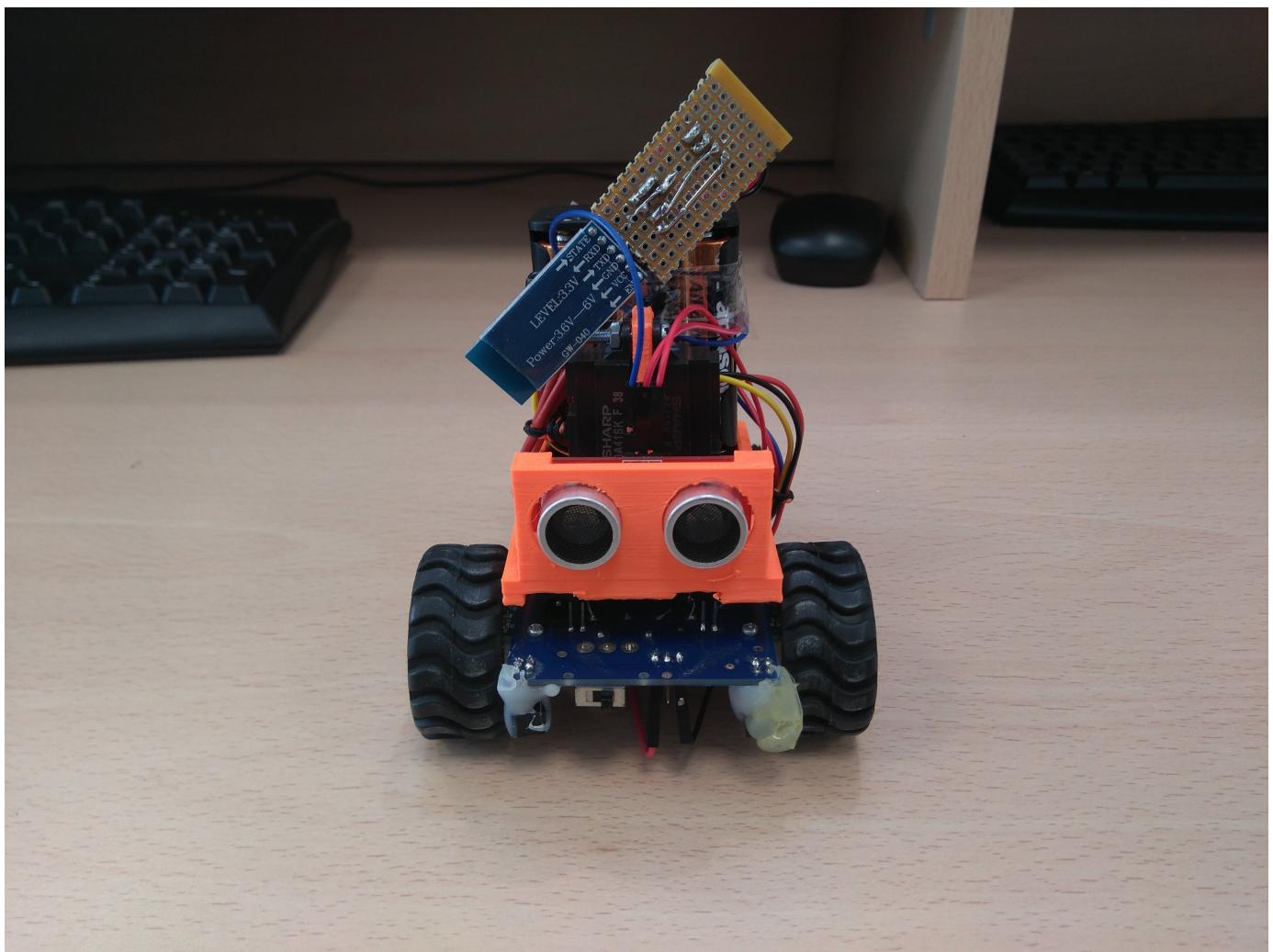
Sensor/Actuador	Pin en shield	Pin en Arduino
Ultrasonidos	J23	A3
Infrarrojos izquierdo	J24	A4
Infrarrojos derecho	J8	A8
CNY70 izquierdo	J15	A1
CNY70 derecho	J19	A0
CNY70 trasero	J11	A5
Motor izquierdo	M21 y M22	10 y 9
Motor derecho	M11 y M12	6 y 5

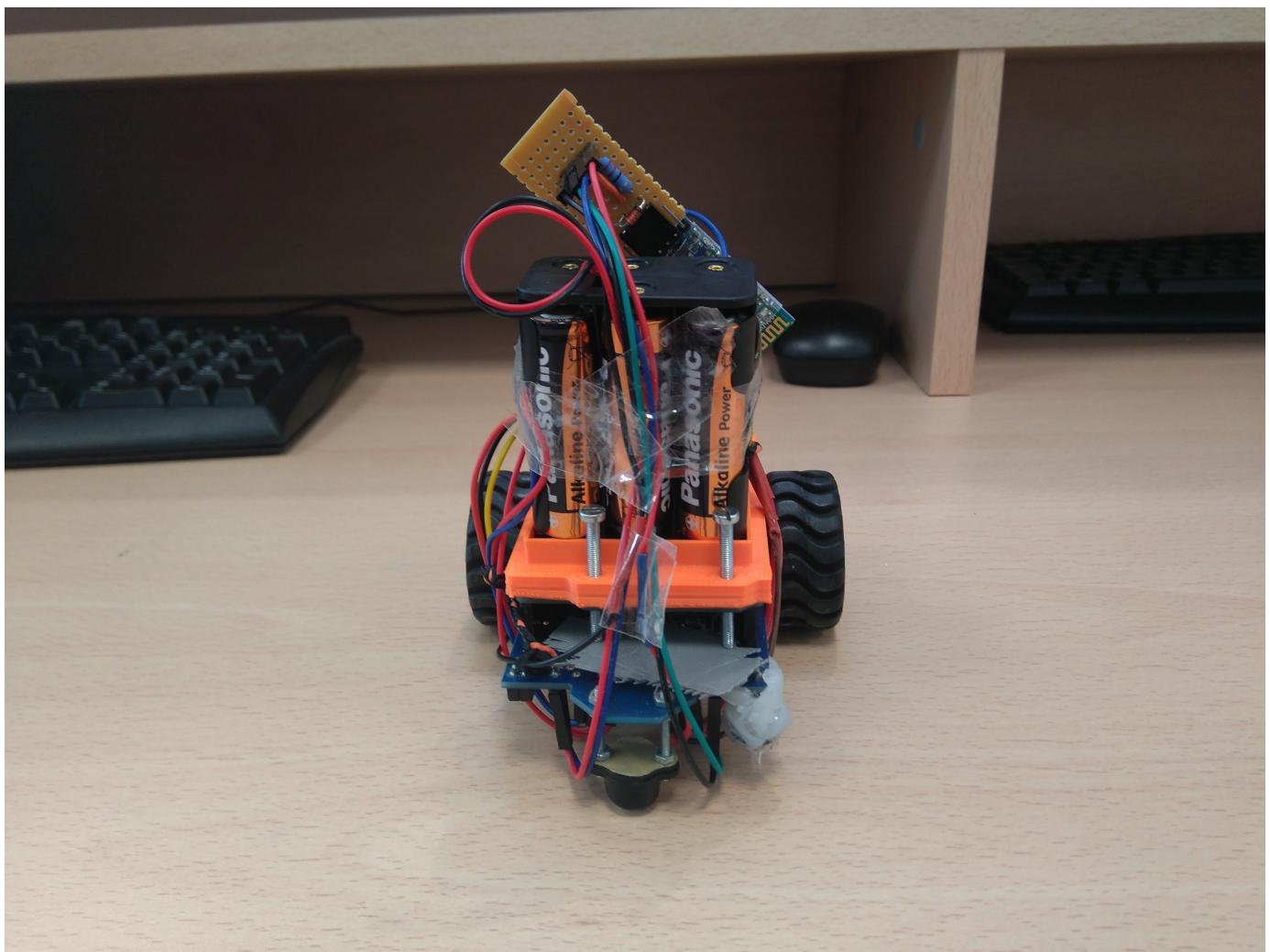
Tabla 8.1: Conexión de pines.

8.3. Imágenes del proyecto

Debido a que el diseño no es lo más imprescindible, hemos optado por un estilo minimalista y funcional antes que un diseño demasiado estético que pudiera dificultar incluso el trabajo del robot.









Capítulo 9

Pruebas

Para las pruebas hemos usado el laberinto del laboratorio haciendo mediciones de los sensores y viendo las velocidades de ambos motores para corregir su trayectoria.

En las **primeras pruebas** se probaron los motores, tanto el avance y retroceso, como los giros a izquierda y derecha. Para la prueba de los sensores usamos la antena bluetooth conectada al móvil mediante la aplicación serial bluetooth terminal por la que recibíamos las mediciones tomadas por los sensores.

Luego, en las **pruebas finales**, pasamos a la interacción del robot con los sensores para probar el algoritmo de salida del laberinto.

Por último, solo faltó algunos arreglos en los parámetros del robot, para que no diera fallos en su trayectoria ni en la medición de los sensores.

Capítulo 10

Mejoras

Mejoras a tener en cuenta:

- Implementación de un diferencial para coger las curvas mejor.
- Medición a tiempo real de las celdas y del recorrido completo para ver que hay en cada lado del robot por cada instante de tiempo.

Parte II

Anexo de códigos

Capítulo 11

Códigos

11.1. Algoritmo

El algoritmo usado para la resolución del laberinto se encuentra en el archivo Robot.ino y es el siguiente:

```
1 #include "Robot.hpp"
2 #include <StandardCplusplus.h>
3 #include <stack>
4 #include <vector>
5
6 // Global vars
7 // ****
8
9 Robot R;                                //The robot itself
10 bool ultimoMuroDelante = true;            //Since we cannot tell if
11                                         //there's a wall behind us,
12                                         //whenever the robot moves
13                                         //forward, the next scan
14                                         //will assign this value to
15                                         //wall behind
16 bool vueltaDelante, vueltaIzquierda, vueltaDerecha; //If it moved behind last
17                                         //time, we don't want to
18                                         //move forward until
19                                         //there's a change of
20                                         //direction, same with left
21                                         //and right
22 std::stack<char, std::vector<int>> pila;    //Stores movements, used
23                                         //when going back to the
24                                         //starting cell
25 bool alcanzada, fin;                      //Wheter the robot has
26                                         //reached the end, and
27                                         //whether it came back
28
29 // ****
30
31 // Functions and procedures declarations
32 //-----
```

```

33
34 //Scans the cell around the robot, returning a "celda" object with information
35 //about where are the walls around it id any.
36 celda escanear(const Robot& R);
37
38 //Returns true if the robot reached the final (fully black) cell
39 bool meta(const Robot& R);
40
41 //Moves the robot until it reaches the next cell, and stops it in the middle
42 void moverDelante(Robot& R);
43 void moverDetras(Robot& R);
44
45 //Turns the robot 90 degrees right/left
46 void girarDerecha(Robot& R);
47 void girarIzquierda(Robot& R);
48
49 //-----
50
51 void setup() {
52
53     Serial1.begin(9600); //Enabling Bluetooth
54
55     vueltaDelante = false;
56     vueltaIzquierda = false;
57     vueltaDerecha = false;
58     ultimoMuroDelante = true;
59     alcanzada = false;
60     fin=false;
61
62     char e = 0;
63
64     while(e != 'e')
65         if(Serial1.available() != 0)
66             e = Serial1.read();
67
68     Serial1.print("e");
69 }
70
71 void loop() {
72
73     if (!alcanzada && !meta(R)){
74
75         celda c = escanear(R,ultimoMuroDelante);
76
77         if (!c.delante)
78             Serial1.print("O");
79         else
80             Serial1.print("X");
81
82         if (!c.izquierda)
83             Serial1.print("O");
84         else
85             Serial1.print("X");
86

```

```

87     if (!c.derecha)
88         Serial1.print("O");
89     else
90         Serial1.print("X");
91
92     if (!c.detras)
93         Serial1.print("O");
94     else
95         Serial1.print("X");
96
97     //Movement priorities: right > front > left > back
98     if (!c.derecha && !vueltaDerecha) {
99         ultimoMuroDelante = c.derecha;
100        girarDerecha(R);
101        moverDelante(R);
102        Serial1.print("2");
103        vueltaIzquierda = false;
104        vueltaDerecha = false;
105        vueltaDelante = false;
106        pila.push('R');
107    }
108    else if (!c.delante && !vueltaDelante) {
109        moverDelante(R);
110        Serial1.print("0");
111        ultimoMuroDelante = c.delante;
112        vueltaIzquierda = false;
113        vueltaDerecha = false;
114        vueltaDelante = false;
115        pila.push('F');
116    }
117    else if (!c.izquierda && !vueltaIzquierda){
118        ultimoMuroDelante = c.izquierda;
119        girarIzquierda(R);
120        moverDelante(R);
121        Serial1.print("1");
122        vueltaIzquierda = false;
123        vueltaDerecha = false;
124        vueltaDelante = false;
125        pila.push('L');
126    }
127    else if (!c.detras){
128        moverDetras(R);
129
130        char mov = 'F';
131
132        if (!pila.empty()){
133            mov = pila.top();
134            pila.pop();
135        }
136
137        if (mov == 'F'){
138            vueltaDelante = true;
139            vueltaIzquierda = false;
140            vueltaDerecha = false;

```

```

141         Serial1.print("3");
142     }
143     if(mov == 'R'){
144         girarIzquierda(R);
145         vueltaDerecha = true;
146         vueltaDelante = false;
147         vueltaIzquierda = false;
148         Serial1.print("4");
149     }
150     if(mov == 'L'){
151         girarDerecha(R);
152         vueltaIzquierda = true;
153         vueltaDelante = false;
154         vueltaDerecha = false;
155         Serial1.print("5");
156     }
157 }
158 }
159 else if(!fin){ //Going back to the starting cell
160
161     Serial1.print("t");
162     alcanzada = true;
163     girarDerecha(R); //180 degree turn
164     girarDerecha(R);
165
166
167 while(!pila.empty()){
168     char mov = pila.top();
169     pila.pop();
170
171     switch(mov) {
172
173         case 'R': //If it went to the right, goes forward then turns left
174             moverDelante(R);
175             girarIzquierda(R);
176             break;
177         case 'F': //If it went forwards, simply goes forward as well
178             moverDelante(R);
179             break;
180         case 'L': //If it went to the left, goes forward then turns right
181             moverDelante(R);
182             girarDerecha(R);
183             break;
184     }
185     fin = true;
186 }
187 }
188 while(fin)
189     delay(100000000);
190
191
192 }
193
194 //Functions and procedures implementations

```

```

195 //-----
196
197 bool meta(const Robot& R) {
198
199     int escaneos = 3;
200     bool b = true;
201
202     for(int i=0; i< escaneos; ++i) {
203         b &= (R.negroAtras() && R.negroIzq() && R.negroDcha());
204         delay(100);
205     }
206
207     return b;
208 }
209
210
211 celda escanear(const Robot& R, bool& ultimoMuroDelante) {
212
213     float delante=0.0f, izquierda=0.0f, derecha=0.0f;
214     int escaneos = 5;
215
216
217     for(int i=0; i<escaneos; ++i){
218         delante += R.distanciaDelante();
219         izquierda += R.distanciaIzq();
220         derecha += R.distanciaDcha();
221         delay(100);
222     }
223     //Cell width and length is 20 cm
224     return celda( (delante/float(escaneos)) <= 20.0f,      //front
225                 ultimoMuroDelante,                      //behind, equal to last
226                                         //front read
227                 (izquierda/float(escaneos)) <= 20.0f, //left
228                 (derecha/float(escaneos)) <= 20.0f ); //right
229 }
230
231
232 void moverDelante(Robot& R) {
233
234     unsigned v = 255; //speed
235
236     R.motorIzqDelante(v); //Forward both ...
237     R.motorDchaDelante(v);
238
239     while(!R.negroIzq() && !R.negroDcha()); // ... while no CNY detects black
240
241     R.motorIzqParar(); //Stops both once one of them does
242     R.motorDchaParar();
243
244     delay(200);
245
246     //Moves forward the wheel which does not have black under it, so both end
247     //lined up. The robot is just about to enter the new cell now
248     if (!R.negroIzq()) {

```

```

249     R.motorIzqDelante(v);
250     while(!R.negroIzq());
251     delay(200);
252     R.motorIzqParar();
253 }
254 else if(!R.negroDcha()) {
255     R.motorDchaDelante(v);
256     while(!R.negroDcha());
257     delay(200);
258     R.motorDchaParar();
259 }
260
261 R.motorIzqDelante(v); //Forward for a while, then it stops around the middle
262 R.motorDchaDelante(v);
263
264 delay(1200);
265
266 R.motorIzqParar();
267 R.motorDchaParar();
268 }
269
270 void moverDetras(Robot& R) { //The same, but backwards
271
272     unsigned v = 255;
273
274     R.motorIzqDetras(v);
275     R.motorDchaDetras(v);
276
277     while(!R.negroIzq() && !R.negroDcha());
278
279     R.motorIzqParar();
280     R.motorDchaParar();
281
282     delay(200);
283
284     if(!R.negroIzq()) {
285         R.motorIzqDetras(v);
286         while(!R.negroIzq());
287         delay(200);
288         R.motorIzqParar();
289     }
290     else if(!R.negroDcha()) {
291         R.motorDchaDetras(v);
292         while(!R.negroDcha());
293         delay(200);
294         R.motorDchaParar();
295     }
296
297     R.motorIzqDetras(v);
298     R.motorDchaDetras(v);
299
300     delay(600);
301
302     R.motorIzqParar();

```

```

303     R.motorDchaParar();
304 }
305
306 void girarDerecha(Robot& R) {
307
308     unsigned v = 255;
309
310     R.motorIzqDelante(v); //Left wheel forward, right one backwards, so it turns
311     //in place to the right.
312     R.motorDchaDetras(v);
313
314     delay(800); //Stops once it has completed around a 90 degrees turn
315
316     R.motorIzqParar();
317     R.motorDchaParar();
318 }
319
320 void girarIzquierda(Robot& R) { //The same, but to the left
321
322     unsigned v = 255;
323
324     R.motorDchaDelante(v);
325     R.motorIzqDetras(v);
326
327     delay(800);
328
329     R.motorDchaParar();
330     R.motorIzqParar();
331 }
```

11.2. Pruebas

Los códigos usados para las pruebas fueron los siguientes:

11.2.1. Primeras pruebas

Ultrasonidos

```

1 float distance;
2 unsigned long time_bounce;
3 const int pin = A3;
4
5 void setup() {
6     Serial.begin(9600); // Setup the serial port
7 }
8
9 void loop() {
10    digitalWrite(pin, LOW);
11    delayMicroseconds(5);
12
13    /* Send a pulse HIGH for 10 us (According to datasheet) */
```

```

14     digitalWrite(pin, HIGH);
15     delayMicroseconds(10);
16     digitalWrite(pin, LOW);
17
18     pinMode(pin, INPUT);
19
20     time_bounce = pulseIn(pin, HIGH);
21
22
23     /* Formula to figure out the distance
24     * speed = space /time --> space = speed * time
25     * sound speed = 340 m/s --> 0.034 cm/us
26     * space --> to travel from the sensor to the object and from de object to
27     * sensor distance to object = space / 2
28     */
29
30     distance = 0.017 * time_bounce; //Formula para calcular la distancia
31
32     /*Show in cm the distance measured, by serial monitor */
33
34     Serial.println("Distancia:");
35     Serial.print(distance);
36     Serial.println("cm");
37
38     delay(1000);
39 }
```

Infrarrojos

```

1 const float ResolutionADC=0.00488; //4.88mV
2 const int Sharp_Pin=A3;
3 int Value_Sharp_Pin=0;
4 float Voltage;
5
6 void setup(){
7     Serial.begin(9600); //Enable the serial port
8 }
9
10 void loop(){
11     // Reads the sensor and return a value between 0-1023
12     Value_Sharp_Pin=analogRead(Sharp_Pin);
13
14     // Calculates the equivalent voltage
15     Voltage=Value_Sharp_Pin*ResolutionADC;
16
17     Serial.println (Value_Sharp_Pin);
18     Serial.print ("Voltage: ");
19     Serial.print (Voltage);
20     Serial.println ("V");
21     Serial.print ("Distancia: ");
22     Serial.print(distancia(Voltage));
23     Serial.println("cm");
24     delay(1000);
25 }
```

```

26
27     float distancia(float Voltage)
28     {
29         if ((Voltage > 0.5 && Voltage < 2.7))
30         {
31             return (12/(Voltage - 0.2)) + 0.42;
32         } else {
33             return 0;
34         }
35     }

```

CNY70

```

1  const float ResolutionADC=0.00488; //4.88mV
2  const int delante=A5;
3  const int izquierdo=A0;
4  const int derecho=A1;
5  const int OUT_LED=8;
6  int Value_CNY_Pin=0;
7  float Voltage;
8
9  void setup(){
10     Serial1.begin(9600); //Enable the serial port
11     pinMode(OUT_LED,OUTPUT);
12 }
13
14 void loop(){
15 // Reads the sensor and return a value between 0-1023
16     Serial1.println("DELANTE:");
17     Value_CNY_Pin=analogRead(delante);
18
19 // Calculates the equivalent voltage
20     Voltage=Value_CNY_Pin*ResolutionADC;
21
22     Serial1.println (Value_CNY_Pin);
23     Serial1.print ("Voltage:");
24     Serial1.print (Voltage);
25     Serial1.println ("V");
26
27     if (Value_CNY_Pin <= 600){
28         digitalWrite(OUT_LED,HIGH);
29         Serial1.println ("NEGRO");
30     } else {
31         digitalWrite(OUT_LED,LOW);
32         Serial1.println ("BLANCO");
33     }
34
35     Serial1.println();
36     Serial1.println("DERECHO");
37
38     Value_CNY_Pin=analogRead(derecho);
39
40 // Calculates the equivalent voltage
41     Voltage=Value_CNY_Pin*ResolutionADC;

```

```

42
43     Serial1.println (Value_CNY_Pin);
44     Serial1.print ("_Voltage:_");
45     Serial1.print (Voltage);
46     Serial1.println (_V);
47
48     if (Value_CNY_Pin <= 600){
49         digitalWrite(OUT_LED,HIGH);
50         Serial1.println ("NEGRO");
51     } else {
52         digitalWrite(OUT_LED,LOW);
53         Serial1.println ("BLANCO");
54     }
55     Serial1.println ();
56     Serial1.println ("IZQUIERDO:");
57     Value_CNY_Pin=analogRead(izquierdo);
58
59 // Calculates the equivalent voltage
60     Voltage=Value_CNY_Pin*ResolutionADC;
61
62     Serial1.println (Value_CNY_Pin);
63     Serial1.print ("_Voltage:_");
64     Serial1.print (Voltage);
65     Serial1.println (_V);
66
67     if (Value_CNY_Pin <= 600){
68         digitalWrite(OUT_LED,HIGH);
69         Serial1.println ("NEGRO");
70     } else {
71         digitalWrite(OUT_LED,LOW);
72         Serial1.println ("BLANCO");
73     }
74     Serial1.println ();
75     Serial1.println ();
76     delay(2000);
77 }
```

Motores

```

1 const int in1Pin=10;
2 const int in2Pin=11;
3
4 int PushButton = 12; //Alias for Pushbutton pin
5 int First_Value_PushButton = 1; //It stores the FIRST reading
6 int Second_Value_PushButton = 0; //It stores the SECOND reading
7 boolean SpinMotor=false; //Variable to store the state of the LED
8 int SpeedMotor;
9
10 void setup() {
11     // put your setup code here, to run once:
12     pinMode(in1Pin, OUTPUT);
13     pinMode(in2Pin, OUTPUT);
14     pinMode(PushButton, INPUT_PULLUP);
15 }
```

```

16
17 void loop() {
18
19 //First time, "First_ValPush" is the initialized value
20 //Microcontroller reads a second value from the button
21 Second_Value_PushButton = digitalRead(PushButton);
22
23 //Check if the input is LOW (button is pressed) and change the "state"
24
25 if((First_Value_PushButton == HIGH) && (Second_Value_PushButton == LOW))
26 {
27     SpinMotor = ! SpinMotor; //Toggle the value of state
28 }
29
30 First_Value_PushButton = Second_Value_PushButton;
31
32 SpeedMotor=analogRead(A0);
33 setMotor(SpeedMotor, SpinMotor);
34 }
35
36 void setMotor(int SpeedMotor, boolean SpinMotor){
37 //Gira a la izquierda
38 SpeedMotor=map(SpeedMotor, 0, 1023, 0, 255);
39 if(SpinMotor==true){
40     //Gira a la izquierda
41     analogWrite(in1Pin, SpeedMotor);
42     analogWrite(in2Pin, 0);
43 } else{
44     //Gira a la derecha
45     analogWrite(in1Pin, 0);
46     analogWrite(in2Pin, SpeedMotor);
47 }
48 }
```

Conección bluetooth

```

1 void setup() {
2     Serial1.begin(9600);
3 }
4
5 void loop(){
6     Serial1.println("Hola_bluetooth.");
7     delay(200);
8 }
```

11.2.2. Pruebas finales

```

1 // Testing code
2 /*
3 SENSORES
4
5 Serial1.print("\nDistancia delante: ");
6 Serial1.println(R.distanciaDelante());
```

```

8 Serial1.print("Distancia izquierda: ");
9 Serial1.println(R.distanciaIzq());
10
11 Serial1.print("Distancia derecha: ");
12 Serial1.println(R.distanciaDcha());
13
14 Serial1.print("CNY detras: ");
15 Serial1.print(R.numeroAtras());
16 if(R.negroAtras())
17     Serial1.println(", negro ");
18 else
19     Serial1.println(", blanco ");
20
21 Serial1.print("CNY izquierdo: ");
22 Serial1.print(R.numeroIzq());
23 if(R.negroIzq())
24     Serial1.println(", negro ");
25 else
26     Serial1.println(", blanco ");
27
28 Serial1.print("CNY derecho: ");
29 Serial1.print(R.numeroDcha());
30 if(R.negroDcha())
31     Serial1.println(", negro ");
32 else
33     Serial1.println(", blanco ");
34
35 delay(3000);
36
37
38 PAREDES
39
40 celda c = escanear(R, ultimoMuroDelante);
41
42 Serial1.print("\nDistancia delante: ");
43 Serial1.println(R.distanciaDelante());
44
45 if(c.delante)
46     Serial1.println("Hay muro delante");
47 else
48     Serial1.println("No hay muro delante");
49
50 Serial1.print("Distancia izquierda: ");
51 Serial1.println(R.distanciaIzq());
52
53 if(c.izquierda)
54     Serial1.println("Hay muro izquierda ");
55 else
56     Serial1.println("No hay muro izquierda ");
57
58 Serial1.print("Distancia derecha: ");
59 Serial1.println(R.distanciaDcha());
60
61 if(c.derecha)

```

```

62     Serial1.println("Hay muro derecha");
63 else
64     Serial1.println("No hay muro derecha");
65
66
67 Serial1.println("");
68 delay(2000);
69
70 BLUETOOTH
71
72 celda c = escanear(R, ultimoMuroDelante);
73
74 if(!c.delante)
75     Serial1.print("O");
76 else
77     Serial1.print("X");
78
79 if(!c.izquierda)
80     Serial1.print("O");
81 else
82     Serial1.print("X");
83
84 if(!c.derecha)
85     Serial1.print("O");
86 else
87     Serial1.print("X");
88
89 if(!c.detras)
90     Serial1.print("O");
91 else
92     Serial1.print("X");
93
94 if(!c.derecha)
95     Serial1.print("2");
96
97 else if(!c.delante)
98     Serial1.print("0");
99
100 else if(!c.izquierda)
101     Serial1.print("1");
102
103 else if(!c.detras)
104     Serial1.print("3");
105 */

```

11.3. Monitorización

El código para la monitorización mediante bluetooth es el siguiente:

```

1 import Tkinter, serial, time
2
3 root = Tkinter.Tk()
4 orientacion = 'N'; #N norte S sur E este W oeste

```

```

5 rx=15 #Global Variables initialized at 6 because its the center of the Matrix
6 ry=15 #
7
8 for x in range(0,31):
9     for y in range (0,31):
10        if x==0 or y==0 or x==30 or y==30: # Loops that draw de border of the labyrinth
11            label12 = Tkinter.Label(root, text=" „#„").grid(row=x, column=y)
12        else:
13            label12 = Tkinter.Label(root, text=" „„").grid(row=x, column=y)
14
15
16
17 def Casilla(x, y): # Draws a O if theres no Wall in the square
18     Tkinter.Label(root, text=" „O„").grid(row=x, column=y)
19
20 def Muro(x, y): # Draws a O if theres a Wall in the square
21     Tkinter.Label(root, text=" „X„").grid(row=x, column=y)
22
23 def Robot(x, y): # Draws a R where the Robot is
24     Tkinter.Label(root, text=" „R„").grid(row=x, column=y)
25
26 def Inicial(x, y): # Draws a R where the Robot is
27     Tkinter.Label(root, text=" „I„").grid(row=x, column=y)
28
29 def Orienta(Cadena):
30     global orientacion
31     global rx ,ry
32     Casilla(rx ,ry )
33     Delante = Cadena[0]
34     Izq = Cadena[1]
35     Dcha = Cadena[2]
36     Detras = Cadena[3]
37     Mov = Cadena[4]
38     if(orientacion == 'N'):
39         Delante = Cadena[0]
40         Izq = Cadena[1]
41         Dcha = Cadena[2]
42         Detras = Cadena[3]
43         Mov = Cadena[4]
44
45     if(orientacion == 'S'):
46         Delante = Cadena[3]
47         Izq = Cadena[2]
48         Dcha = Cadena[1]
49         Detras = Cadena[0]
50         Mov = Cadena[4]
51
52     if(orientacion == 'E'):
53         Delante = Cadena[1]
54         Izq = Cadena[3]
55         Dcha = Cadena[0]
56         Detras = Cadena[2]
57         Mov = Cadena[4]
58

```

```

59     if(orientacion == 'W'):
60         Delante = Cadena[2]
61         Izq = Cadena[0]
62         Dcha = Cadena[3]
63         Detras = Cadena[1]
64         Mov = Cadena[4]
65
66     if(Delante == 'X'):
67         Muro(rx-1,ry)
68     else:
69         Casilla(rx-1,ry)
70
71     if(Izq == 'X'):
72         Muro(rx,ry-1)
73     else:
74         Casilla(rx,ry-1)
75
76     if(Dcha == 'X'):
77         Muro(rx,ry+1)
78     else:
79         Casilla(rx,ry+1)
80
81     if(Detras == 'X'):
82         Muro(rx+1,ry)
83     else:
84         Casilla(rx+1,ry)
85
86     newori = orientacion;
87     if(Mov == '0'):
88         if(orientacion == 'N'):
89             rx=rx-2
90         if(orientacion == 'S'):
91             rx=rx+2
92         if(orientacion == 'E'):
93             ry=ry+2
94         if(orientacion == 'W'):
95             ry=ry-2
96
97     if(Mov == '1'):
98         if(orientacion == 'N'):
99             newori = 'W'
100            ry=ry-2
101        if(orientacion == 'W'):
102            newori = 'S'
103            rx=rx+2
104        if(orientacion == 'E'):
105            newori = 'N'
106            rx=rx-2
107        if(orientacion == 'S'):
108            newori = 'E'
109            ry=ry+2
110
111    if(Mov == '2'):
112        if(orientacion == 'N'):

```

```

113         newori = 'E'
114         ry=ry+2
115     if(orientacion == 'W'):
116         newori = 'N'
117         rx=rx-2
118     if(orientacion == 'E'):
119         newori = 'S'
120         rx=rx+2
121     if(orientacion == 'S'):
122         newori = 'W'
123         ry=ry-2
124
125 if(Mov == '3'):
126     if(orientacion == 'N'):
127         newori = 'N'
128         rx=rx+2
129     if(orientacion == 'W'):
130         newori = 'W'
131         ry=ry+2
132     if(orientacion == 'E'):
133         ry=ry-2
134         newori = 'E'
135     if(orientacion == 'S'):
136         rx=rx-2
137         newori = 'S'
138
139 if(Mov == '5'):
140     if(orientacion == 'N'):
141         newori = 'E'
142         rx=rx+2
143     if(orientacion == 'W'):
144         newori = 'N'
145         ry=ry+2
146     if(orientacion == 'E'):
147         ry=ry-2
148         newori = 'S'
149     if(orientacion == 'S'):
150         rx=rx-2
151         newori = 'W'
152
153 if(Mov == '4'):
154     if(orientacion == 'N'):
155         newori = 'W'
156         rx=rx+2
157     if(orientacion == 'W'):
158         newori = 'S'
159         ry=ry+2
160     if(orientacion == 'E'):
161         ry=ry-2
162         newori = 'N'
163     if(orientacion == 'S'):
164         rx=rx-2
165         newori = 'E'
166 orientacion = newori

```

```

167
168
169 def RecibeCelda(Cadena): #Receives an String that describes the actual situation
170     global rx ,ry # Uses the global Variables
171     Orienta(Cadena)
172     Robot(rx ,ry) #Allocates the Robot at the new Position
173
174 ##### MAIN CODE
175 try:
176     arduino=serial.Serial("COM4" ,9600)
177     arduino . write('e')
178     while arduino . read() != 'e':
179         root . update()
180     while True:
181         Inicial(15 ,15)
182         c0 = arduino . read()
183         c1 = arduino . read()
184         c2 = arduino . read()
185         c3 = arduino . read()
186         c4 = arduino . read()
187         cadena = c0+c1+c2+c3+c4
188         print cadena
189         Robot(rx ,ry)
190         print orientacion
191         RecibeCelda(cadena)
192         time . sleep(1)
193         root . update()
194     root . mainloop()
195     Robot(rx ,ry)
196     arduino . close()
197     Robot(rx ,ry)
198
199 except:
200     print "Fallo"
201     exit()
202
203
204
205
206 ##### END OF MAIN CODE

```

La interfaz gráfica quedaría de así:

