



# DESARROLLO DE UNA LIBRERÍA PARA EL CONTROL DE UNA COLONIA DE MINI- ROBOTS

*TRABAJO DE FIN DE GRADO  
GRADO EN INGENIERÍA ELECTRÓNICA INDUSTRIAL Y  
AUTOMÁTICA*

*UNIVERSIDAD CARLOS III DE MADRID*

Autor: María Blázquez Partido

Director: Raúl Pérula Martínez

Co-director: Juan Miguel García Haro

Leganés, Septiembre 2014



# AGRADECIMIENTOS

En primer lugar me gustaría agradecer a mi tutor del TFG el haberme permitido trabajar en este proyecto, valorando además su gran ayuda y paciencia durante todo el proceso.

También quisiera expresar mi agradecimiento a los profesores que he tenido durante estos 4 años, que han ayudado proporcionándome el conocimiento necesario y la motivación para convertirme en una Ingeniera que disfruta desempeñando su profesión.

Sin duda este duro camino de estudios no lo he recorrido sola y por eso también merecen una mención todos los compañeros que han estado conmigo durante la carrera, por todas las anécdotas y buenos momentos que he pasado con ellos, llevándome buenos amigos y compañeros que espero que también estén en mi futuro.

De manera más cercana también he recibido el apoyo de mi novio Daniel, que siempre ha estado conmigo en los buenos y malos momentos, tanto durante el curso como realizando este proyecto, siempre creyendo en mí y confiando en que puedo sacar adelante todo lo que me proponga.

Y por último, pero no menos importante, a mis padres, pues siempre han estado a mi lado proporcionándome todo lo necesario para seguir estudiando y poder llegar a tener un gran futuro.

## RESUMEN

Con la gran cantidad de robots de bajo coste que utilizan microcontroladores del tipo Arduino surge la necesidad de crear un sistema para controlarlos y adaptarlos a las funcionalidades requeridas con el fin de poder crear funciones y tareas complejas partiendo de un manejo sencillo e intuitivo.

Para el control de una colonia de mini-robots de bajo coste, en este proyecto se ha creado una librería *Open Source* compatible con Arduino, que permite a cualquier usuario poder adaptarla a las aplicaciones deseadas y los elementos de sus robots.

Como complemento y testeo de esta librería se ha creado una interfaz gráfica de usuario tanto para PC como para Smartphone Android.

## ABSTRACT

With the large number of low-cost robots using Arduino microcontrollers, arises the need to create a system to control and adapt them to the required functionalities in order to create complex functions and tasks based on a simple, intuitive operation.

To control a low cost mini-robots colony, in this project has been created an *Open Source* library that supports Arduino, which allows any user to adapt it to the desired applications and elements of their robots.

As complement and testing of this library, it has been created a graphical user interface for PC and a Smartphone app for Android.

# ÍNDICE

Agradecimientos .....	3
Resumen .....	4
Abstract .....	5
Índice de Figuras .....	9
Índice de tablas .....	12
Índice de diagramas .....	13
1 Introducción .....	14
1.1 Definición del problema real.....	14
1.2 Definición del problema técnico.....	14
2 Objetivos .....	18
3 Estado del arte .....	20
3.1 Robótica de bajo coste.....	20
3.2 Robótica colaborativa.....	24
4 Restricciones .....	32
4.1 Factores dato.....	32
4.2 Factores estratégicos.....	33
5 Recursos .....	35
5.1 Recursos humanos.....	35
5.1.1 Directores .....	35
5.1.2 Autor .....	35
5.2 Recursos materiales.....	36
5.2.1 Recursos hardware .....	36
5.2.2 Recursos software .....	52
6 Presupuesto .....	55
6.1 Costes Materiales.....	55
6.2 Costes de Personal.....	57
6.3 Resumen del presupuesto.....	58
7 Diseño e Implementación del sistema .....	59
7.1 Librería de control.....	59

7.1.1	Diseño del sistema.....	59
7.1.2	Implementación.....	69
7.2	Aplicación para Smartphone.....	73
7.2.1	Diseño del sistema.....	73
7.2.2	Implementación.....	78
7.3	Interfaz gráfica de usuario para PC.....	81
7.3.1	Diseño del sistema.....	81
7.3.2	Implementación.....	82
8	Pruebas y resultados .....	86
8.1	Pruebas de calibración.....	86
8.1.1	Desarrollo .....	86
8.1.2	Resultados .....	86
8.2	Pruebas orientadas a comunicación punto a punto.....	87
8.2.1	Descripción del problema .....	87
8.2.2	Desarrollo .....	88
8.2.3	Resultados .....	97
8.3	Pruebas orientadas a comunicación multidispositivo.....	98
8.3.1	Descripción del problema .....	98
8.3.2	Desarrollo .....	99
8.3.3	Resultados .....	100
9	Conclusiones y futuras mejoras .....	102
9.1	Conclusiones.....	102
9.2	Futuras mejoras.....	103
Bibliografía.....		104
Apéndice A. Manual de usuario.....		110
A.1	Instalación.....	110
A.1.1	Arduino IDE .....	110
A.1.2	Aplicación Android .....	110
A.1.3	QT Creator .....	112
A.2	Desinstalación.....	112
A.2.1	Arduino IDE .....	112
A.2.2	Aplicación Android .....	113

A.2.3	QT Creator .....	114
A.3	Modo de ejecución.....	115
A.3.1	Librería.....	115
A.3.2	Interfaz Gráfica de Usuario para PC.....	122
A.3.3	Aplicación para Smartphone .....	123

# ÍNDICE DE FIGURAS

Figura 1. Diagrama de Gantt de realización del TFG.....	17
Figura 2. Robot de bajo costo QuickBot.....	21
Figura 3. Impresoras del parque de impresoras de la UC3M.....	22
Figura 4. Diseño 3D por ordenador. ....	22
Figura 5. Diseños para impresoras 3D compartidos en Thingiverse. ....	23
Figura 6. Miniskybot 1.0, Scout Robot y Protobot.....	24
Figura 7. Foot-boots y Eye-bots.....	25
Figura 8. Agrupación de Centibots.....	25
Figura 9. Robots Termes realizando tareas de construcción. ....	26
Figura 10. Robots industriales manipuladores en la fábrica de Hyundai en Ulsan. ....	27
Figura 11. Mapa realizado por los robots Centibots. ....	28
Figura 12. UAVs Elbit System Hermes.....	29
Figura 13. Uno de los robots rover enviados a Marte. ....	29
Figura 14. Robots escolta Legged Squad Support System de DARPA.....	30
Figura 15. Robots de Kiva Systems realizando tareas en de almacenamiento. ....	31
Figura 16. Esquema eléctrico microcontrolador ATmega 320. ....	36
Figura 17. Placa Freaduino Uno. ....	37
Figura 18. Esquema eléctrico I/O ATmega 328 .....	38
Figura 19. Ciclo de trabajo.....	39
Figura 20. Señal modulada. ....	40
Figura 21. Servomotor de rotación continua. ....	42
Figura 22. Conexionado servomotor con placa Arduino. ....	43
Figura 23. Sensor de Ultrasonidos HC-SR04.....	43
Figura 24 Diagrama de temporización. ....	44
Figura 25. Módulo Bluetooth HC-05/HC-06. ....	45
Figura 26. Conexionado Módulo Bluetooth con la placa Arduino. ....	46
Figura 27. Alimentación externa 9V.....	49
Figura 28. Conector DC Alimentación externa a Freaduino. ....	49
Figura 29. Alimentación externa de 6V con soporte. ....	50

Figura 30. Chasis de los mini-robots.....	51
Figura 31. Ruedas de los mini-robots. ....	51
Figura 32. Soporte para los sensores de Ultrasonido.....	51
Figura 33. Arduino IDE. ....	53
Figura 34. Diagrama de clases de la librería.....	63
Figura 35. Conexionado del módulo Bluetooth con el mini-robot.....	66
Figura 36. Ciclo de vida de las actividades en Android.....	74
Figura 37. Interfaz para Smartphone – Ventana principal.....	79
Figura 38. Interfaz para Smartphone – Guiar al robot.....	79
Figura 39. Interfaz para Smartphone – Resolución libre. ....	80
Figura 40. Interfaz para PC – Vista general. ....	83
Figura 41. Editar diseño de los elementos de la interfaz.....	83
Figura 42. Interfaz para PC – Conexión con los robots. ....	84
Figura 43. Tele-operación.....	85
Figura 44. Disposición de los robots en el laberinto.....	88
Figura 45. Buffer de resolución del laberinto.....	91
Figura 46. Smartphone con la aplicación Android creada. ....	96
Figura 47. Escenario creado para pruebas. ....	97
Figura 48. Tele-operación simultánea de los mini-robots. ....	99
Figura 49. Proceso de instalación del .apk. ....	111
Figura 50. Bloqueo del .apk por origen desconocido. ....	111
Figura 51. Permiso de aplicaciones de origen desconocido. ....	112
Figura 52. Desinstalar Arduino en Windows. ....	113
Figura 53. Menú de desinstalación. ....	113
Figura 54. Opción de Aplicaciones en el menú ajustes o configuración. ....	114
Figura 55. Proceso de desinstalación de la aplicación. ....	114
Figura 56. Desinstalar QT Creator en Windows. ....	115
Figura 57. Menú de desinstalación. ....	115
Figura 58. Localización del sketchbook en Arduino.....	116
Figura 59. Importación de librerías.....	116
Figura 60. Utilizar programa de ejemplo para el uso de la librería. ....	121
Figura 61. Modificar dirección Bluetooth a emparejar.....	121

Figura 62. Propiedades del dispositivo Bluetooth.....	122
Figura 63. Iniciar compilación.....	123
Figura 64. Conexión con el dispositivo Bluetooth. ....	124
Figura 65. Búsqueda de dispositivos en el modo guiado.....	124
Figura 66. Selección de dispositivo a conectar en el modo guiado.....	125
Figura 67. Conectado con dispositivo. .....	125

# ÍNDICE DE TABLAS

Tabla 1. Comandos AT comunes HC-05 y HC-06. ....	47
Tabla 2. Comandos AT específicos HC-05. ....	48
Tabla 3. Presupuesto total. ....	58
Tabla 4. Calibración de los Servomotores. ....	62
Tabla 5. Correspondencia módulos Bluetooth con robots. ....	65
Tabla 6. Tele-operación mediante el teclado del PC. ....	82

# ÍNDICE DE DIAGRAMAS

Diagrama 1. Funciones de movimiento. ....	64
Diagrama 2. Función de modo. ....	68
Diagrama 4. Función de escucha. ....	68
Diagrama 3. Función emparejar y conectar. ....	69
Diagrama 5. MainActivity – Método onDestroy(). ....	75
Diagrama 7. GuidedLabyrinth – Método onDestroy(). ....	75
Diagrama 6. MainActivity – Método onCreate(). ....	76
Diagrama 8. Dispositivo encontrado y Dispositivo seleccionado para conectar. ....	76
Diagrama 9. GuidedLabyrinth – Método onCreate(). ....	77
Diagrama 10. Free labyrinth – Método onCreate(). ....	78
Diagrama 11. Resolución del laberinto. ....	89
Diagrama 12. Moverse esquivando obstáculos.....	90
Diagrama 13. Moverse según orden.....	91
Diagrama 14. Rellenar Buffer de resolución. ....	92
Diagrama 15. Enviar el Buffer.....	93
Diagrama 16. Colocar robot en posición inicio de movimiento. ....	94
Diagrama 17. Colocar robot posición de fin de movimiento. ....	95
Diagrama 18. Tele-operación. ....	100

# 1 INTRODUCCIÓN

En este capítulo se identificarán las necesidades por las que se requiere la realización del proyecto como la definición del problema real, y los componentes necesarios para satisfacer esas necesidades como definición del problema técnico.

## 1.1 DEFINICIÓN DEL PROBLEMA REAL

El abaratamiento de los componentes electrónicos y el hardware y software *Open Source*<sup>1</sup> hacen que la fabricación e implementación de robots y dispositivos automáticos se encuentre en auge. Gracias a estas herramientas y a esta filosofía *Open Source*, la industria tecnológica y el campo de la robótica de bajo presupuesto avanza notablemente encontrando un nexo entre la robótica de investigación y los nuevos avances en robótica.

En la Asociación de Robótica de la Universidad Carlos III de Madrid [1], se desarrollan varios proyectos de investigación entre los que se encuentran los Robots Personales de Competición (RPC) y más concretamente las colonias de robots.

Se requiere implementar una librería *Open Source* en Arduino para una colonia de mini-robots compuestos por un microcontrolador del tipo Arduino, dos servomotores, dos sensores de Ultrasonido y un módulo Bluetooth.

Esta librería debe permitir el control de todos los elementos que forman parte de los mini-robots, para que éstos puedan realizar tareas de manera conjunta y coordinada.

## 1.2 DEFINICIÓN DEL PROBLEMA TÉCNICO

A continuación se definirán los condicionantes del problema y las especificaciones a llevar a cabo durante el proceso de desarrollo.

- Se utilizarán robots de bajo coste.
  - Los componentes externos del diseño serán creados mediante impresión 3D.
  - Se utilizará un microcontrolador integrado en una placa Arduino.
- Se realizará una librería *Open Source* para el manejo de los sensores, actuadores y comunicación entre robots de manera genérica, de forma que el

---

<sup>1</sup> Software distribuido y desarrollado libremente también llamado Software libre.

código sea reutilizable y adaptable para cualquier tipo de robot y aplicación que se quiera realizar.

- La librería diseñada estará orientada a su aplicación en microcontroladores Arduino por lo que el código de programación utilizado será compatible con su entorno y microcontroladores.
- Se realizarán pruebas en el hardware y en el software realizado para testear el correcto funcionamiento de todos los elementos y determinar su vida útil. Se implementará una aplicación *Open Source* de interfaz de usuario para PC y para Smartphone Android para facilitar el uso de la librería creada y facilitar al usuario la interactuación directa con los mini-robot, así como para realizar funciones de testeado.
- Se cumplirá el plazo acordado para la realización del proyecto de acuerdo a la distribución de tareas indicada en el diagrama de Gantt de la Figura 1.



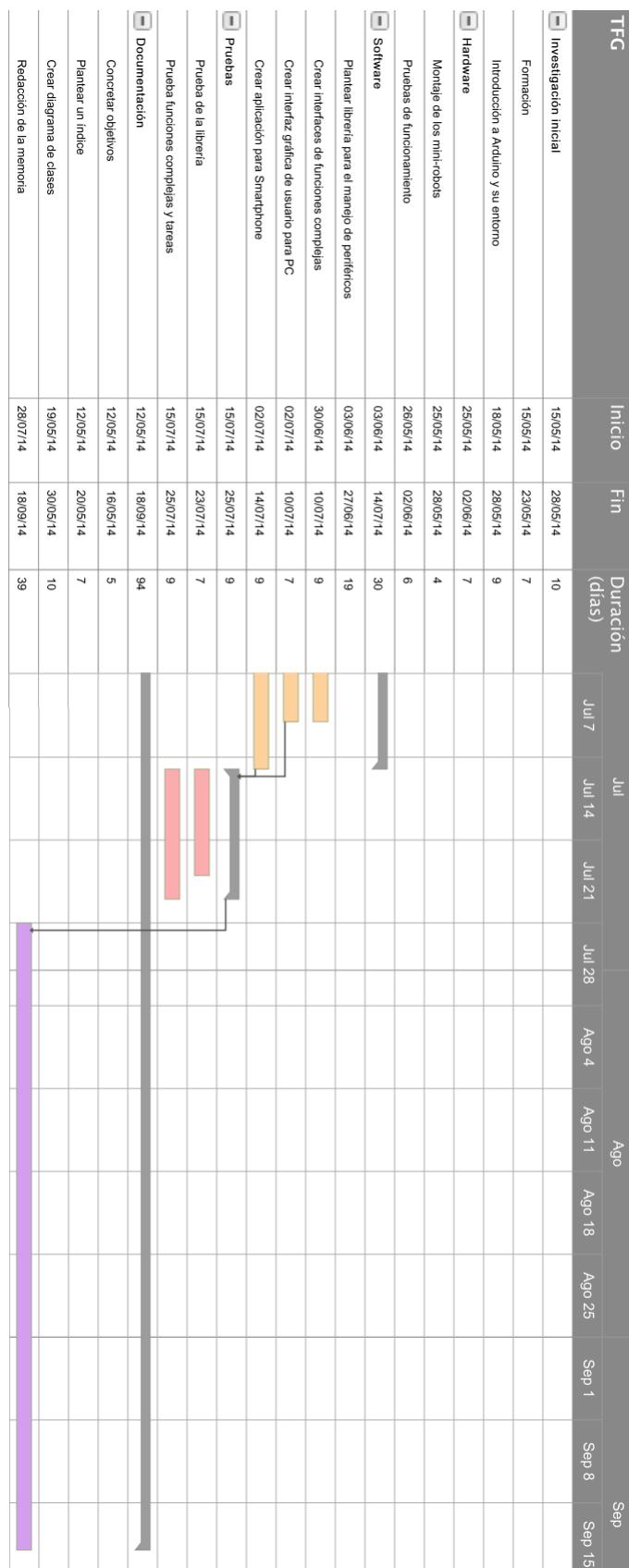


Figura 1. Diagrama de Gantt de realización del TFG.

## 2 OBJETIVOS

En el presente capítulo se expondrán todos los objetivos funcionales que se pretenden alcanzar en el desarrollo de este proyecto. Éstos se dividirán en objetivos principales orientados a la construcción, el estudio y el manejo de una colonia de mini-robot, y objetivos secundarios para el ajuste y mejora de los componentes, así como la creación de sistemas de testeado para el usuario.

En primer lugar se desarrollarán los objetivos básicos a desempeñar durante la realización del proyecto; el estudio del estado del arte de los sistemas multi-robot y el montaje de la colonia de mini-robots.

### ***Estudio del estado del arte de los sistemas multi-robot***

Comprensión del funcionamiento de las colonias de robots e interpretación y análisis de los avances realizados en el campo.

Estudio previo de los componentes necesarios para la realización de un mini-robot modular, teniendo en cuenta las especificaciones técnicas de cada uno de los elementos y los requerimientos necesarios para cada mini-robot.

Pruebas de compatibilidad de periféricos con el microcontrolador y unificación de todos los componentes utilizándose componentes auxiliares si fuese necesario.

### ***Montaje de la colonia de mini-robots***

- **Diseño:** Creación de los componentes que forman la estructura física de los mini-robots e impresión de los mismos utilizando una impresora 3D.
- **Hardware:** Montaje completo e interconexión de todos los componentes tras realizar el estudio de viabilidad y las pruebas iniciales.
- **Software:**
  - Implementación de una librería *Open Source* para el manejo de sensores, actuadores y comunicación compatible con Arduino.
  - Realización de programas de pruebas individuales para cada componente.
  - Realización de movimientos coordinados entre robots (estructura cliente-servidor)

A continuación se indicarán los objetivos derivados de los principales como mejora y ampliación de los sistemas creados.

- **Calibración de los servomotores:** Realización de ensayos para estandarizar los movimientos realizados por los mini-robots.
- **Impresión 3D:** Realización de posibles mejoras en el diseño de los mini-robot.
- **Implementación de algún modulo electrónico,** para añadir funcionalidad a los mini-robot, o mejorar las ya existentes.
- **Implementación de una aplicación para PC,** como interfaz gráfica de usuario para la tele-operación de los mini-robots de manera remota, con el objeto de testear la comunicación con todos los mini-robots y su ejecución de movimientos de manera coordinada.
- **Implementación de una aplicación para Smartphone Android,** para el testeo de la librería creada y de la comunicación punto a punto entre los mini-robots.

# 3 ESTADO DEL ARTE

En este capítulo se tratarán principalmente los elementos de bajo costo orientados a la fabricación de robots imprimibles, libres y replicables y Las agrupaciones de robots y la aplicación de la coordinación de las mismas al mundo real.

## 3.1 ROBÓTICA DE BAJO COSTE

La robótica es un campo relativamente joven que cruza las fronteras tradicionales de la ingeniería. La comprensión de la complejidad de los robots y su aplicación requiere el conocimiento multidisciplinar de electrónica, mecánica, informática e incluso economía. Debido a esto, la educación robótica es más eficaz cuando conceptos teóricos se complementan con experimentos tangibles [2].

Cada vez es más común en las instituciones de investigación e universidades la utilización de robots de manera experimental, ya sea mediante simuladores o mediante robots reales. Hace algunos años el uso y la construcción de robots físicos era únicamente accesible para algunas instituciones con grandes recursos económicos y para proyectos de investigación con un alto presupuesto.

La aparición de nuevos microprocesadores de bajo costo y de impresoras 3D como medios para la creación de robots ha revolucionado el mundo de la robótica.

Un ejemplo de ello es *Arduino*, que utiliza **microprocesadores de bajo costo** integrados en unas placas de uso sencillo, útiles tanto para usuarios de todos los niveles formativos. Además tanto su hardware como su software son abiertos y multiplataforma, incluyendo documentación con gran variedad de ejemplos y funciones ya desarrolladas.

Existe gran variedad de placas en el mercado para las necesidades de cada usuario, sin embargo, al tratarse de hardware libre cualquiera puede replicarlo o modificarlo creando su propio modelo compatible con software Arduino. Además la plataforma Arduino dispone de funciones ya implementadas para la lectura de periféricos y el uso de actuadores, de manera que no hay límite en la creación de los diseños electrónicos para los robots.

Por ello, los proveedores de artículos de electrónica, están integrando cada vez más entre sus productos kits para la fabricación de robots de manera económica en los que se incluyen microprocesadores de éste tipo.

Un ejemplo de robot de bajo coste es el *QuickBot* en la Figura 2, que propone la universidad *Georgia Tech*, para poder realizar de manera sencilla un robot modular programable por menos de 120€, con un potente microprocesador, Wi-Fi, encoders para las ruedas, sensores de infrarrojos, etc. [2]



Figura 2. Robot de bajo costo QuickBot.

## Robots imprimibles

El hardware de los robots, es decir, su soporte físico, es fácilmente replicable para cualquier usuario que tenga acceso a una impresora 3D.

El origen de las **impresoras 3D** viene del proyecto *RepRap* cuyo objetivo era desarrollar una máquina autoreplicable de código abierto. *RepRap* es una máquina de prototipado rápido libre que es capaz de replicarse a sí misma, es decir, que puede generar las partes necesarias para construir otra máquina igual a ella [3].

Una impresora *Open Source* según el modelo puede llegar a ser hasta 10 veces más barata frente a una impresora industrial cuyo precio oscila entre 12.000€ y 60.000€.

En la Figura 3 se muestran las impresoras 3D *Open Source* que forman parte del parque de impresoras de la Universidad Carlos III de Madrid. [4]

“PADRE” (izquierda de la Figura 3), es una impresora *Marketbot*, que como parte de la familia *RepStrap* ha sido construida en la universidad a partir de distintos componentes con un coste total aproximado de 920€. A partir de esta impresora, se han creado numerosas impresoras del tipo *RepRap*, como parte del proyecto denominado “*Clone wars*”, cuyo propósito es la construcción de impresoras 3D de bajo costo utilizando las ya existentes.

“HIJA” (derecha de la Figura 3) forma parte de la familia *Prusa Air*, y fue creada a partir de otras impresoras 3D, destacando por su gran robustez y modernidad.



Figura 3. Impresoras del parque de impresoras de la UC3M.

Mediante la impresión 3D se pueden crear objetos físicos a partir de diseños tridimensionales como los mostrados en la Figura 4, generados por ordenador mediante cualquier software de diseño CAD<sup>2</sup> como *OpenScad*, *SolidWorks*, etc.

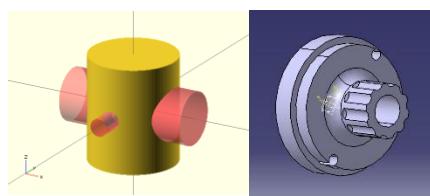


Figura 4. Diseño 3D por ordenador.

Existen webs dedicadas a la distribución de archivos de diseño digital creados por los usuarios, un ejemplo de ello es **Thingiverse** que proporciona diseños hardware de código abierto bajo la licencia Creative Commons<sup>3</sup>.

*Thingiverse* es ampliamente utilizado en las comunidades tecnológicas, por el Proyecto *RepRap*, y por los operadores de *MakerBot* y demás usuarios de impresoras 3D. Numerosos proyectos técnicos utilizan *Thingiverse* como un repositorio para la innovación y la difusión de los materiales de base para el público común, es decir, *Open Source* [5]. En la Figura 5, se muestran algunos ejemplos de diseños compartidos por los usuarios en *Thingiverse*.

<sup>2</sup> Del inglés Computer-Aided Design; Diseño asistido por ordenador.

<sup>3</sup> Bienes Comunes Creativos



Figura 5. Diseños para impresoras 3D compartidos en Thingiverse.

El modelo ***Open Source*** ofrece a los usuarios versatilidad, y acceso a todo tipo de recursos para poder expandir sus conocimientos y creaciones. Miles de personas forman parte de una misma comunidad colaborativa en la que aparecen repositorios, wikis, etc en búsqueda del avance tecnológico.

El éxito de la metodología ***Open Source*** recae en la comunicación, la colaboración y la mejora continua desde la distancia a través de internet [6]. Además, en contraste con el modelo clásico de empresa, la motivación no es el dinero, sino el entretenimiento y la pasión que surge del proyecto dónde el desarrollador está participando y se siente involucrado [7].

El modelo ***Open Source*** es de gran utilidad aplicado al mundo de la robótica permitiendo que los robots creados por la comunidad sean compartidos por internet y puedan evolucionar de manera colaborativa.

De la distribución colaborativa de diseños de hardware y software siempre bajo la filosofía ***Open Source*** surgen los robots imprimibles denominados “**Printbots**” (PRINTable roBOTS).

Una de las ventajas de los robots imprimibles, es la posibilidad de desarrollar robots bajo demanda, sin necesidad de recurrir a caros equipos comerciales. Además los robots libres e imprimibles suponen una revolución en el paradigma de la robótica, comienzan a formarse proyectos de hardware abierto que evoluciona gracias a la comunidad [8].

Un ejemplo de la evolución y mejora de los diseños libres compartidos por los usuarios puede apreciarse en el robot *Probot*, inspirado en los robots *Miniskybot* y *Scout Robot* en la Figura 6.

El robot *Miniskybot* creado en 2011 como colaboración de la Universidad Carlos III con la Universidad Autónoma de Madrid, fue diseñado para fines educativos utilizando únicamente software ***Open Source***, para garantizar la realización de diseños derivados de manera libre. [9]

El *Scout robot*, una “*tele-copia*” creada en Missouri (EE.UU.), incorporó una canica como rueda loca y un chasis mejorado adaptado a placas de tipo Arduino. [10]

El *Probot* desarrollado en la Universidad Carlos III de Madrid incorporó un soporte para protoboard, y un compartimento adaptado para una alimentación de 9V, además de realizarse mejoras en el diseño. [11]

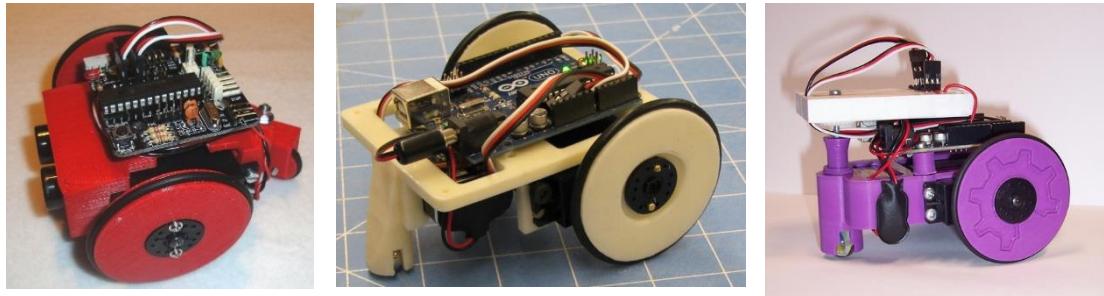


Figura 6. Miniskybot 1.0, Scout Robot y Probot.

Estos robots son fácilmente adaptables a cualquier periférico nuevo a incluir, siendo personalizables para cualquier aplicación deseada, y sin límites en su evolución.

## 3.2 ROBÓTICA COLABORATIVA

Las colonias de robots, o agrupaciones de robots, surgen de la necesidad de gestionar las tareas individuales desempeñadas por cada unidad para realizar una tarea final. Un ejemplo de esto son las cadenas de montaje de coches, cuyo punto fuerte radica en la unión de varios robots trabajando de forma conjunta

Cada unidad robótica individual en colaboración de más robots necesita ser coordinada y adaptada para evitar interferencias y mejorar el desempeño de las tareas e incluso adaptarse a tareas en inminente aparición.

Muchos trabajos de investigación como el realizado por *Ronald Arkin* [12], utilizan la naturaleza como guía para implementar comportamientos coordinados entre agrupaciones de robots.

De la denominación de grandes comunidades de robots unidas en grupos o enjambres actuando en colaboración surge el término *Swarm robotics*<sup>4</sup>. El objetivo es estudiar las conductas presentes en agrupaciones de seres vivos presentes en la naturaleza e implantarlas en el comportamiento de las colonias de robots.

---

<sup>4</sup> Robótica de enjambres

En la Figura 7 se muestran los denominados *Swarm-bots* y *Swarmanoid* como ejemplo de algunos proyectos desarrollados en este campo.

El proyecto *swarm-bots* está centrado en el diseño, implementación y control de los denominados *s-bots*; un enjambre de pequeños robots que son capaces de organizarse y reagruparse.

Los denominados *Swarmanoid* consisten en tres tipos diferentes de robots: *foot-bots*, que se mueven por el suelo con capacidades similares a los *s-bots*, *eye-bots* que vuelan y pueden adherirse al techo, y *hand-boots* que pueden manipular objetos y trepar en vertical usando una cuerda [13].

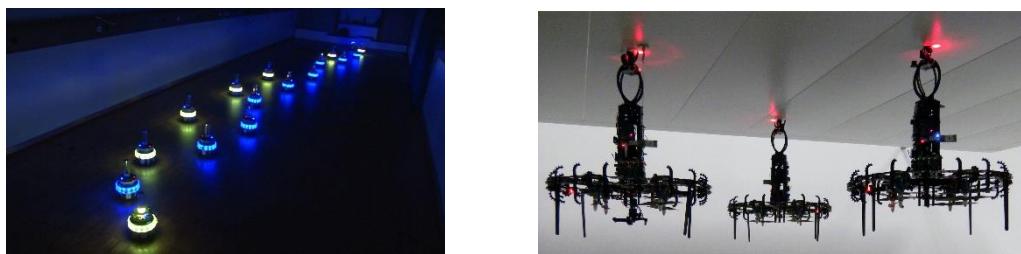


Figura 7. Foot-boots y Eye-bots.

Las agrupaciones de robots y su relación cooperativa aumentan la eficiencia y la productividad, tolerando errores en alguno de sus miembros. De esta forma, se pueden realizar tareas como navegación, recolección, etc. que realizadas por un solo robot resultarían largas y tediosas.

En la Figura 8, el proyecto *Centibots* [14] como ejemplo de aplicación de mapeado y de tareas de vigilancia realizadas por una colonia de 100 robots.

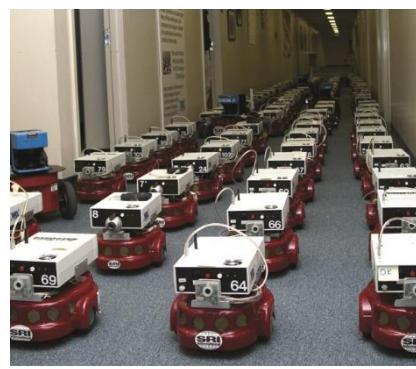


Figura 8. Agrupación de Centibots.

La Universidad de Harvard (EE.UU.), ha desarrollado unos robots termita denominados *Termes*, diseñados para levantar estructuras de diversa complejidad, y en el futuro poder edificar colonias en otros planetas si necesidad de planos o capataces, según

afirman sus creadores. Estos robots están equipados con sensores que les permiten orientarse en el entorno, además cada unidad independiente sabe cómo debe quedar la construcción final. Gracias a algoritmos especiales, estos robots son capaces de coger un bloque y colocarlo sin causar colisiones ni atascos [15], ver Figura 9.

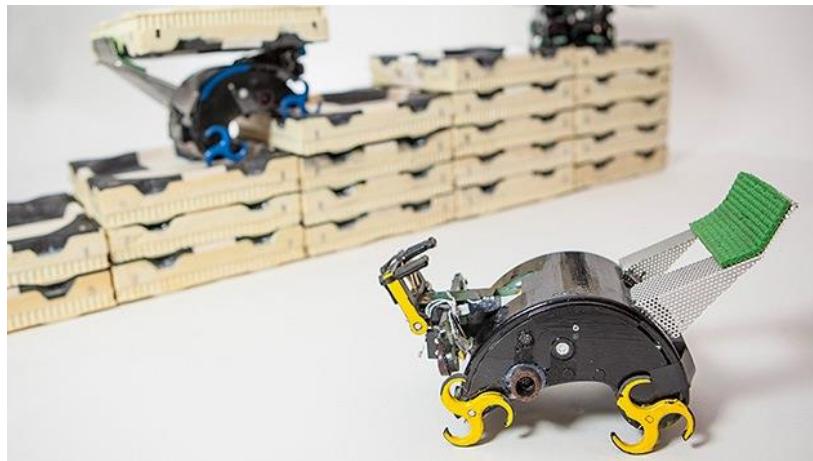


Figura 9. Robots Termes realizando tareas de construcción.

Según las aplicaciones a realizar por parte de la colonia ésta puede ser homogénea (todos sus miembros tienen el mismo hardware y software) o heterogénea (uno o más robots difieren del resto). A menudo en las agrupaciones de robots se establecen roles distintivos para uno o más participantes de la colonia por ejemplo asignándose un líder, de forma que se crea una agrupación heterogénea, ya que aunque los robots en apariencia puedan ser iguales, el comportamiento del líder es distinto del resto.

La robótica colaborativa, si se realiza de manera coordinada puede ampliar de manera significativa los horizontes de la robótica. Para que exista una colaboración inteligente entre los miembros de la colonia es importante que se realice una correcta comunicación, en caso contrario, la tarea puede ser contraproducente, pudiendo interferir unos robots en las tareas de otros y dificultando su correcta implementación.

Un método simple de auto-detección de los demás robots de la colonia es la implantación de elementos de repulsión, como por ejemplo imanes. También es común el uso de sensorización de manera que se pueda esquivar a los demás participantes de la colonia al igual que a los obstáculos. El método con mejores resultados para la mayor parte de aplicaciones es la comunicación explícita, por ejemplo a través de dispositivos inalámbricos tales como Bluetooth, ZigBee, etc.

## Aplicaciones de la coordinación de robots

La coordinación de robots es un relativamente reciente campo de estudio y de continua evolución. Esto se debe a su gran variedad de aplicaciones a desempeñar y a la ausencia de límites en cuanto a innovación y mejora de la vida de las personas.

Algunas de las aplicaciones más comunes de la coordinación de robots son:

### Manipulación

Es muy común en entornos industriales el uso de robots manipuladores que actúan de manera coordinada manipulando objetos. En muchas ocasiones este tipo de colaboración entre robots industriales se utiliza para interactuar con objetos muy pesados, que un solo robot no podría manipular.

La mayoría de estos robots actúan según trayectorias previamente programadas para evitar la colisión entre los demás miembros de la colonia.

En la Figura 10, pueden observarse robots industriales trabajando en colaboración en la fábrica de coches más grande del mundo situada en Ulsan (Corea del sur), dónde se produce un coche cada 13 segundos.



Figura 10. Robots industriales manipuladores en la fábrica de Hyundai en Ulsan.

### Mapeado

Esta aplicación es un claro ejemplo de cómo el uso de varios robots en lugar de uno puede simplificar en gran medida una tarea que de otra forma sería muy larga e inexacta.

El uso de la coordinación de robots para la exploración y el reconocimiento de entornos desconocidos, requiere la implementación de algoritmos de localización y otros tipos de sistemas para poder obtener y recopilar con exactitud los datos.

Los errores en la información recopilada decrementan considerablemente al usar grandes cantidades de robots en lugar de uno aislado, debido a que cada robot puede tomar como referencia a los demás y así sucesivamente, de manera que el error cometido global puede corregirse y eliminarse.

En la Figura 11, puede observarse un mapa del entorno realizado por los robots *Centibots* en color gris en contraste con el entorno real en rojo, durante una evaluación del proyecto final en Enero del 2014. [16]



Figura 11. Mapa realizado por los robots Centibots.

#### Percepción distribuida

El uso de varios robots para percibir objetos tiene su campo de aplicación en tareas de vigilancia o seguridad.

Una de las principales ventajas del uso de un conjunto de robots, es la mayor cobertura en la sensorización y por tanto una mayor robustez en las tareas.

Los UAVs<sup>5</sup> son un ejemplo de este tipo de agrupaciones de robots, muy utilizadas debido a su posibilidad de uso en zonas de difícil acceso o riesgo. Generalmente son utilizados para tareas de espionaje en entornos militares. En la Figura 12, flota de 6 UAVs operados por la armada Inglesa con fines militares. [17]

---

<sup>5</sup> Del inglés Unmanned Aerial Vehicle; Vehículo aéreo no tripulado



Figura 12. UAVs Elbit System Hermes.

### Recolección

La colonia de robots debe encontrar y recoger una serie de objetos distribuidos por un entorno generalmente tóxico o de difícil accesibilidad para las personas.

Resulta de gran utilidad para la recolección de sustancias tóxicas, reconocimiento de entornos peligrosos (aplicaciones militares), y tareas de rescate.

Este tipo de robots suelen ser homogéneos, utilizando la sensorización como método de comunicación para evitar posibles colisiones entre los miembros.

Un ejemplo de ello son los robots enviados a marte por la NASA, los *rovers* que realizan tareas de recolección entre otras. En la Figura 13 el robot *Curiosity rover* cuyo brazo articulado de dos metros dispone de un recolector de tierra que permite la toma de muestras para su posterior análisis. [18]



Figura 13. Uno de los robots rover enviados a Marte.

## Formación

Se utilizan principalmente en el ámbito militar para tareas de exploración y escolta.

Los robots se mantienen a una distancia determinada los unos de los otros y se desplazan en una orientación determinada de acuerdo a su planificación.

Las colonias de robots coordinadas en formación deben mantener un orden en formación y adaptarse a nuevas agrupaciones cuando se presente un obstáculo, siempre manteniendo las características de distancias e orientación establecidas.

Un ejemplo de formaciones de robots realizando tareas de escolta son los *Legged Squad Support System (LS3)* propiedad de DARPA que son capaces de preceder de forma autónoma a los escuadrones militares por terrenos accidentados y de interpretar órdenes verbales y visuales, ver Figura 14. [19]



Figura 14. Robots escolta Legged Squad Support System de DARPA.

## Almacenamiento

En la industria son muy conocidos e utilizados los AGVs<sup>6</sup>, robots que actúan de manera colaborativa y coordinada y que se encargan de transportar elementos a través de unas trayectorias programadas. Uno de los mayores avances en sistemas de coordinación de colonias de robots aplicados en la industria es el sistema logístico robotizado de *Kiva Systems* en la Figura 15, que ya se está utilizando en grandes compañías como *Amazon* para la manipulación de inventario en sus almacenes. [20]

<sup>6</sup> Del inglés Automatic Guided Vehicle; vehículos de guiado automático



Figura 15. Robots de Kiva Systems realizando tareas en de almacenamiento.

# 4 RESTRICCIONES

En este capítulo se expondrán todas las restricciones existentes en la realización de este proyecto, junto con las principales decisiones de diseño adoptadas durante la realización del mismo.

## 4.1 FACTORES DATO

Se definirán aquellos factores aportados de forma externa en cuanto a limitaciones en la realización del proyecto.

### ***Restricciones de hardware***

- *Microcontrolador*
  - Al estar la placa Freaduino UNO ya ensamblada quita flexibilidad a la hora de adaptarla con algunos periféricos, siendo en muchos casos necesarios elementos auxiliares.
  - El voltaje de salida de la alimentación de la placa sólo llega a los 5V por lo que para utilizar elementos tales como servomotores ha sido necesario el uso de alimentaciones auxiliares.
- *Actuadores*
  - Es necesaria una calibración previa para utilizar los servomotores de rotación continua para fijar la velocidad de movimiento y adaptarla a las necesidades.
- *Sensores*
  - Los sensores de Ultrasonidos utilizados para la detección de obstáculos por parte de los mini-robot, proporcionan un pequeño error en la medida que aparece sobre todo cuando los mini-robot están en movimiento.
- *Módulos de comunicación*
  - Los módulos Bluetooth disponibles para la realización de este proyecto proporcionan una comunicación punto a punto, lo cual en ocasiones necesita largos tiempos de emparejamiento y conexión que dificulta la realización de las tareas de manera continua por parte de los mini-robots.

- *Alimentaciones*
  - La duración de la carga de las alimentaciones es corta, por lo que requieren su cambio constante. Por ello se han utilizado baterías recargables para tener más independencia en los experimentos.

### ***Restricciones de software***

- *IDEs*<sup>7</sup>

Para utilizar los microcontroladores del tipo Arduino es necesario utilizar el entorno de desarrollo del fabricante que tiene algunos inconvenientes con respecto a otros entornos de desarrollo.

  - Los tiempos de compilación y carga de los programa es largo en el sistema operativo Windows.
  - No indica el número de línea con lo que dificulta la depuración del código.
  - No consta de función de autorrellenado.

### ***Restricciones de diseño***

- La orientación del *plugin* para los sensores de Ultrasonidos en ocasiones hace que se detecten obstáculos que no interfieren en la trayectoria de los mini-robots. Por otra parte, debido al tamaño de los mini-robots solo es posible utilizar más de dos sensores de ultrasonidos con lo que no se realizan medidas del entorno con exactitud, de forma que la implementación del sistema se ve condicionada.

## **4.2 FACTORES ESTRATÉGICOS**

Se indicarán aquellos factores que pueden ser modificados durante la realización del proyecto y adaptados a las necesidades propias del mismo.

### ***Metodología de trabajo***

- Se ha utilizado un sistema de control de versiones para la gestión de los archivos generados y del control de cambios utilizando *SVN*.
- Para el seguimiento de las tareas realizadas por parte del tutor del proyecto se ha utilizado la plataforma de trabajo en grupo online *Trello*.

---

<sup>7</sup> Del inglés de Integrated Development Environment; entorno de desarrollo integrado.

***Factores de software***

- Para trabajar con el entorno de desarrollo de Arduino se ha utilizado el sistema operativo Windows, aunque el IDE está disponible además para las plataformas Ubuntu y OS X.
- Se ha utilizado el lenguaje de programación C++ junto a Qt, para la creación de la interfaz gráfica de usuario para PC, y Android (Java) para la creación de la aplicación para Smartphone.
- Se han elegido QT Creator y Eclipse como herramientas de desarrollo para la realización de las interfaces de usuario.

# 5 RECURSOS

En este capítulo se expondrán de forma clara y concisa los recursos humanos y materiales necesarios para este proyecto. Los recursos se definen como aquellos medios de los que se dispone para abordar el proceso de desarrollo del proyecto. El análisis de los recursos existentes se realiza atendiendo a una doble perspectiva:

- Recursos humanos: son aquellos que están constituidos por toda persona que intervenga en el proceso de desarrollo del sistema.
- Recursos materiales: son aquellos que pueden definirse como el conjunto de todas las entidades no animadas que permiten realizar el proceso de desarrollo de la aplicación, así como la generación de la documentación relativa a la misma.

## 5.1 RECURSOS HUMANOS

El conjunto de personas que intervendrán durante el proceso de desarrollo del presente proyecto se muestran a continuación:

### 5.1.1 DIRECTORES

#### Raúl Pérrula Martínez

- Personal Investigador en Formación del Departamento de Ingeniería de Sistemas y Automática y miembro investigador del grupo *Robotics Lab*.
- Coordinador del grupo de Robot Personales de Competición en la Asociación de Robótica [1] de la Universidad Carlos III de Madrid.

#### Juan Miguel García Haro

- Personal Investigador en Formación del Departamento de Ingeniería de Sistemas y Automática y miembro investigador del grupo *Robotics Lab*.

### 5.1.2 AUTOR

María Blázquez Partido, graduada en Grado en Ingeniería Electrónica Industrial y Automática en la Universidad Carlos III de Madrid.

## 5.2 RECURSOS MATERIALES

Para la realización de este proyecto se han utilizado los siguientes recursos de hardware y software.

### 5.2.1 RECURSOS HARDWARE

Todas las partes tangibles del sistema; componentes electrónicos, mecánicos, periféricos, piezas que componen los robots, etc.

#### 5.2.1.1 ARDUINO UNO

Arduino es una plataforma de desarrollo de computación física de código abierto, basada en una placa con un sencillo microcontrolador y un entorno de desarrollo para crear software para la placa [21].

Arduino permite acceder a numerosas librerías y elementos creados específicamente para su placa. De esta forma se puede trabajar de una manera más sencilla e intuitiva con microcontroladores y hardware externo.

Arduino UNO es una placa o circuito impreso que consta de un microcontrolador de 8 bits Atmega 328 cuyo esquema eléctrico se muestra en la Figura 16. [22]

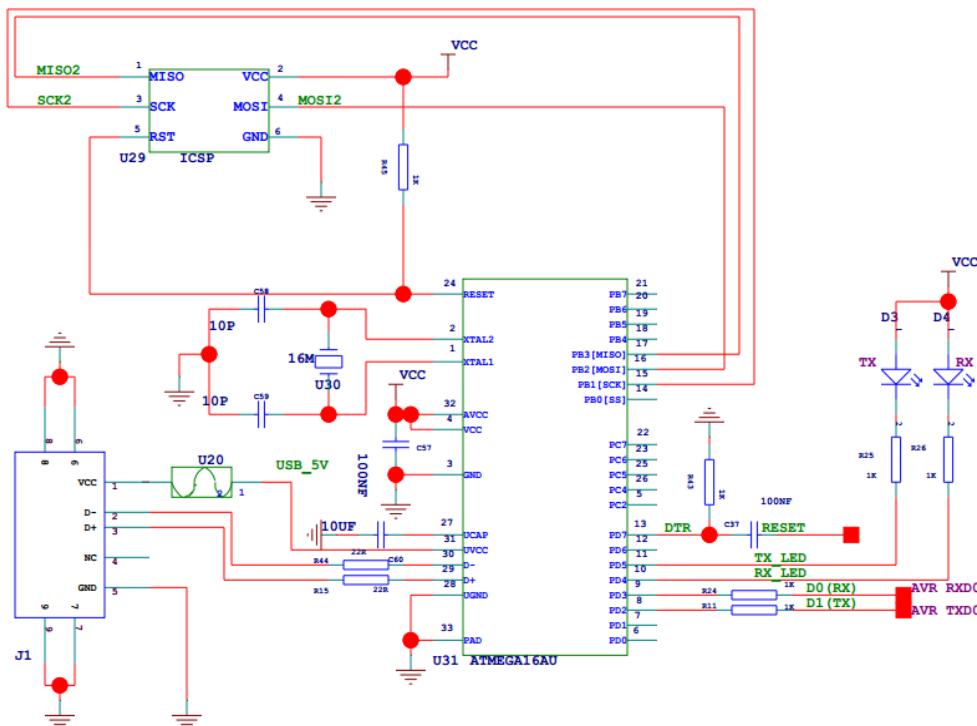


Figura 16. Esquema eléctrico microcontrolador ATmega 320.

Las características técnicas del hardware Arduino son:

- Microcontrolador: ATmega328
- Voltaje de operación: 5V
- Voltaje de entrada recomendado: 7-12V
- Límites superior e inferior de voltaje de entrada: 6-20V
- Pines de entrada/salida programables: 14 (De los cuales 6 proporcionan una salida PWM; 3, 5, 6, 9, 10 y 11)
- Pines de entrada analógica: 6
- Corriente DC por cada pin de entrada/salida: 40 mA
- Corriente DC para el pin de 3,3V: 50 mA
- Memoria Flash: 32KB (Proveniente del microcontrolador), de los cuales 0,5KB se utilizan al arranque
- Memoria EEPROM: 1KB (Proveniente del microcontrolador)
- Memoria SRAM: 2KB (Proveniente del microcontrolador)
- Velocidad máxima del reloj: 16MHz [23]

### 5.2.1.2 FREADUINO UNO

Freaduino UNO, en la Figura 17, es una placa compatible de Arduino basada en el diseño de Arduino UNO Rev3. Freaduino UNO presenta visibles mejoras en el hardware haciendo de ella una placa más flexible y fácil de utilizar. [24]

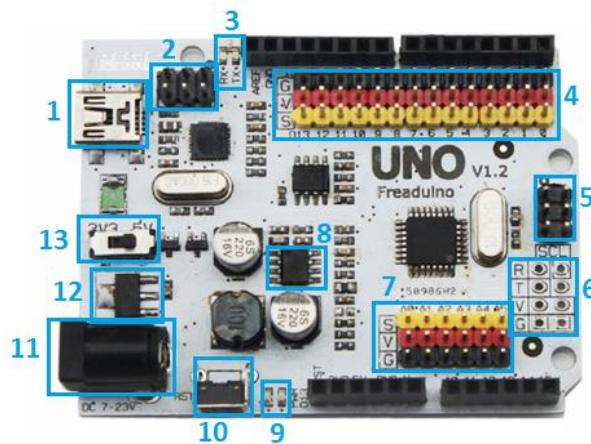


Figura 17. Placa Freaduino Uno.

Las características técnicas del hardware Freaduino UNO son:

1. Mini USB para conectar con el PC
2. ISP para Mega8U2
3. LEDs TX y RX que indican comunicación y recepción de datos
4. Pines digitales de entrada/salida programable

5. ISP para Arduino
6. puerto COM
7. Pines analógicos de entrada/salida programable
8. DCDC MP2307
9. LEDS
10. Botón de Reset
11. Entrada de corriente DC 7-23V
12. LDO 80mA
13. Alimentación Arduino 3V/5V

## Especificaciones

---

### Pines digitales

---

Los pines I/O<sup>8</sup> del microcontrolador del Arduino, cuando son utilizados como puertos de I/O digitales son de lectura y escritura. Pueden configurarse como entradas (habilitando o deshabilitando la resistencia de pull-up, o como salidas realizando cambios en el driver del controlador. En la Figura 18 se puede observar el esquema eléctrico de los pines del controlador ATmega 823, que consta de una resistencia de pull-up, diodos protectores para Vcc y tierra, entre otros elementos. [22]

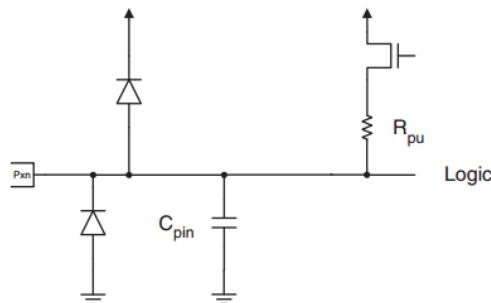


Figura 18. Esquema eléctrico I/O ATmega 328

Se proporcionan librerías específicas para elegir si un pin digital actúa como salida o como entrada, sin necesidad de acceder a los drivers del microcontrolador.

Cuando actúan como entradas, deben conectarse a elementos que envíen una señal lógica, por ejemplo un pulsador. Toman el valor binario 0 o 1 según si obtienen una tensión a la entrada o no.

---

<sup>8</sup> Del inglés Input/Output; Entrada/Salida

Actuando como salidas, los pines digitales que no son PWM toman el valor digital 0 o 1, de forma que cuando el valor es 0, el pin tendrá un voltaje de salida de 0V, y cuando el valor es 1 la salida del pin tendrá el voltaje al que esté operando el Arduino; 3,3V o 5V.

El funcionamiento para los pines digitales PWM es totalmente diferente como se tratará en el siguiente apartado.

- PWM: Señal modulada

Las señales digitales pueden tomar los valores digitales 0 o 1 que corresponden a un valor de voltaje de 0V o (5V-3.3V). Este último valor depende del voltaje al que queremos que opere nuestro Arduino.

Las salidas PWM se diferencian de las digitales en que pueden simular un voltaje analógico a la salida, es decir que modulando la anchura de pulso, la salida tendrá un voltaje variable entre 0V y 5V.

Para entender el funcionamiento de este tipo de señal, necesitamos entender el concepto de **Ciclo de trabajo** (Duty Cycle).

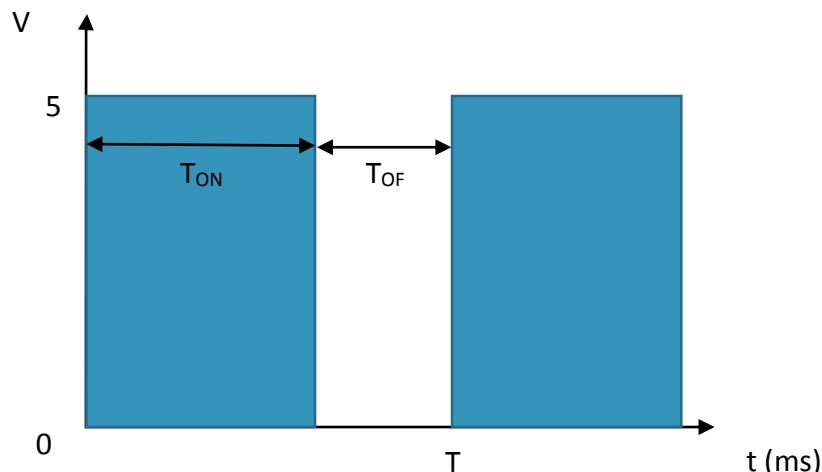


Figura 19. Ciclo de trabajo.

Tal como se puede observar en la Figura 19, durante un periodo ( $T$ ), la señal se mantiene durante un tiempo determinado en su valor máximo ( $T_{ON}$ ) y en su valor mínimo ( $T_{OF}$ ).

En la ecuación (1) se indica la expresión para el ciclo de trabajo  $D$  al que opera la señal.

$$D = \frac{T_{ON}}{T} = \frac{T_{ON}}{T_{ON}+T_{OF}} \quad (1)$$

El valor del ciclo debe estar siempre entre 0 y 1, valiendo 0 cuando la señal es nula, y 1 cuando la señal siempre se encuentra en el valor máximo.

En el caso de las salidas PWM de Arduino, nos interesa poder operar con otros valores del ciclo de trabajo, para poder regular el voltaje a la salida.

La señal que se obtiene en los pines PWM del Arduino, tendrán en función del ciclo de trabajo una tensión variable entre 0V y 5V.

Como sabemos, cuando tenemos una señal modulada, el valor con el que opera nuestro sistema es con el valor medio de la señal, de esta forma, cuando el ciclo de trabajo es 1 o 0, el voltaje a la salida será de 5V o 0V respectivamente.

En (2) se muestra la expresión para calcular el valor medio de una señal.

$$V_m = \frac{1}{T} \int_0^T v(t) \cdot dt \quad (2)$$

De esta forma, si tenemos la señal modulada de la Figura 20, el ciclo de trabajo D será el indicado en la expresión (3).

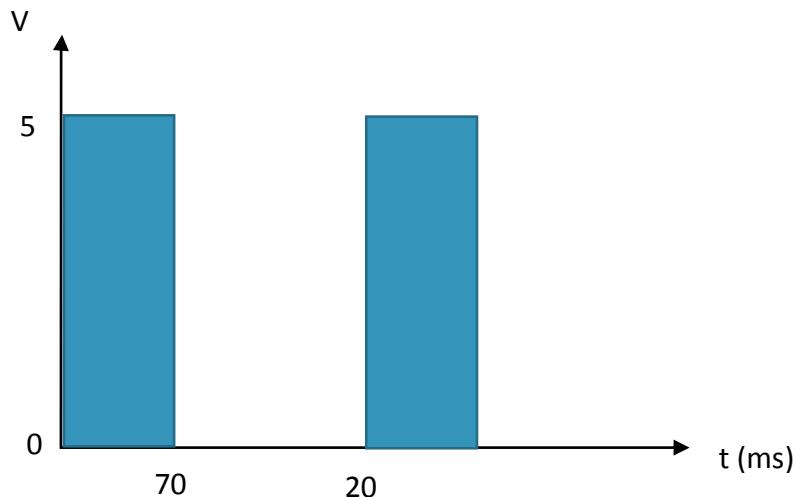


Figura 20. Señal modulada.

$$D = \frac{70ms}{200ms} = 0,35 \quad (3)$$

En (4) se muestra como se calcula el valor medio de la señal, con lo que queda la expresión (5).

$$V_m = \frac{1}{200} \left( \int_0^{70} 5 \cdot dt + \int_{70}^{200} 0 \cdot dt \right) = \frac{1}{200} \cdot 5t \Big|_0^{70} = \frac{350}{200} = 1,75V \quad (4)$$

$$V_m = D \cdot V_p = 0,35 \cdot 5V = 1,75V \quad (5)$$

Por tanto 1,75V será el valor de tensión a la salida del pin PWM.

De esta forma el pin PWM nos resulta útil para algunas aplicaciones, en las que queremos regular el valor de la tensión.

### Pines analógicos

---

Los pines analógicos, son al igual que los pines digitales, programables como entradas o salidas.

Estos pines nos permiten obtener una sensibilidad en la medida del voltaje de entrada, pudiendo diferenciar los Voltios que entran al pin, a diferencia de los pines digitales que solo diferencian tensión y no tensión.

Además con estos pines podemos generar un voltaje variable entre 0V y 5V definido por el usuario. Este mismo resultado puede obtenerse utilizando las salidas PWM de los pines digitales.

### 5.2.1.3 ACTUADORES

Para que los robots puedan interactuar con el entorno necesitan de actuadores. Por tanto, es una parte fundamental en el diseño del hardware, elegir los actuadores más adecuados para cada tarea.

En este proyecto, se requiere que los robots sean capaces de desplazarse por el entorno y de realizar movimientos con suficiente precisión. Para ello se ha decidido utilizar como actuadores dos servomotores de rotación continua, uno para cada una de las ruedas de los robots.

### Servomotores SM-S4303R

---

Los servomotores o “servos” son dispositivos capaces de colocar su eje en cualquier posición dentro de su rango de operación. Son similares a un motor de corriente continua y pueden controlarse tanto en velocidad como en posición.

Los servomotores se conectan a una señal continua de anchura de pulso modulada y según la tensión que llega al servomotor éste gira un número de grados determinado.

Un servomotor de rotación continua permite una rotación de  $0^\circ$  a  $360^\circ$ , de manera que no hay saltos entre movimientos. Por ello, se ha elegido el servomotor SPRINGRC

SM-S4303R que se muestra en la Figura 21 [25], del que a continuación se explicará el funcionamiento.



Figura 21. Servomotor de rotación continua.

Algunas características técnicas del servomotor de rotación continua SPRINGRC SM-S4303R son las siguientes:

- Velocidad de rotación (6V): 0,13 sec / 60°
- Fuerza: 39.2 oz-in

#### Modo de funcionamiento

---

Estos servomotores funcionan mediante una señal PWM. Usando la librería “Servo.h” disponible para Arduino, de la que se hablará más adelante, se debe conectar el pin de señal del servomotor a un pin digital normal, no PWM.

Esto se debe a que la librería “Servo.h” ya incluye la generación de señales PWM, es decir, que es capaz de regular el tiempo en el que la salida digital vale ‘1’ y ‘0’. De esta forma la señal del servomotor ha de conectarse a cualquier pin de señal digital en la placa Arduino.

El esquema del conexionado se observa en la

Figura 22:

- Señal (Amarillo) – Pin de señal digital
- GND (Negro) – GND de la placa Arduino
- Vcc (Rojo) – 5V Placa Arduino

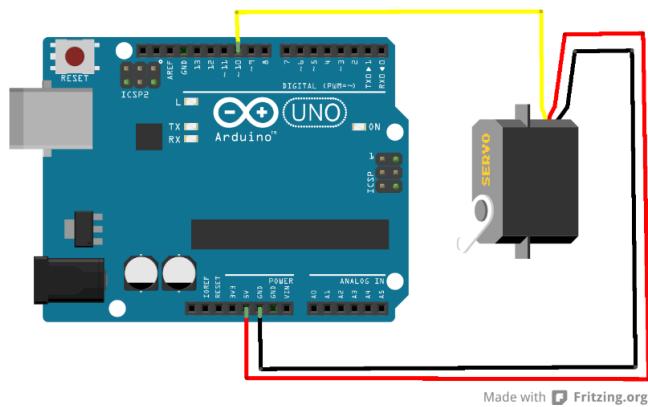


Figura 22. Conexionado servomotor con placa Arduino.

### 5.2.1.4 SENSORES

Para que los robots puedan obtener información sobre el entorno en el que se encuentran se van a utilizar sensores de distancia. De esta forma se puede obtener con una cierta precisión si hay obstáculos frente a los robots, y de ser así, a qué distancia se encuentran.

#### Sensor de Ultrasonido HC-SR04

Su modo de funcionamiento es la emisión de ultrasonido, y midiendo el eco del mismo se obtiene la distancia a la que se encuentran los objetos.

Este tipo de sensores son muy utilizados en robótica debido a su precio económico y su fiabilidad.

Se va a utilizar el sensor ultrasónico HC-SR04, ver Figura 23. [26]



Figura 23. Sensor de Ultrasonidos HC-SR04.

El sensor de Ultrasonidos consta de 4 pines de izquierda a derecha en la imagen anterior Vcc, Trigger entrada de pulso, Echo salida de pulso y GND. Estos pines se explicarán de manera más detallada a continuación.

Algunos detalles técnicos importantes del sensor de Ultrasonido HC-SR04, el cual se muestra en la Figura 23, son:

- Alimentación: 5V DC
- Corriente de operación: 15mA
- Frecuencia de operación: 40Hz
- Corriente de inactividad: <2mA
- Rango máximo: 4m
- Rango mínimo: 2cm
- Ángulo eficaz: <15°
- Rango de distancia: 2cm – 500 cm
- Resolución: 0,3cm

### Modo de funcionamiento

---

El conexiónado del sensor de Ultrasonidos con la placa de Arduino es de la siguiente manera:

- Echo (Amarillo) – Pin señal digital
- Trigger (Negro) – Pin señal digital
- Vcc (Rojo) – 5V DC
- GND (Azul) – GND de la placa Arduino

En la Figura 24 se indica el modo de operación del sensor de ultrasonidos. [27]

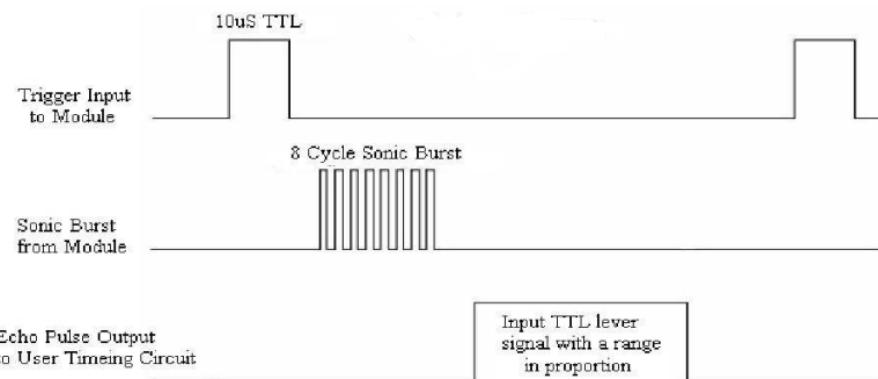


Figura 24 Diagrama de temporización.

- En primer lugar se debe enviar un pulso a su entrada Trigger de al menos 10µs.

- A continuación el módulo envía automáticamente 8 pulsos a 40Hz de frecuencia y espera a recibir una señal.
- Si se detecta un obstáculo se recibirá un eco y por tanto, el pin Echo enviará una señal al Arduino. Mediante la medida del tiempo en el que la señal está en nivel alto, se obtiene la distancia a la que se encuentra el obstáculo mediante la expresión (6).

$$\text{Distancia (m)} = \frac{(Duración \text{ en nivel alto (s)}) \cdot \left( \text{Velocidad del sonido: } 340 \left( \frac{\text{m}}{\text{s}} \right) \right)}{2} \quad (6)$$

### 5.2.1.5 COMUNICACIONES

La comunicación entre los distintos mini-robots se va a realizar mediante protocolo Bluetooth, para ello se utilizarán módulos Bluetooth con las características adecuadas para desempeñar las tareas requeridas.

#### Módulos Bluetooth HC-05/06

Para las metodologías que se llevarán a cabo se van a utilizar 3 módulos Bluetooth Esclavos y dos módulos Bluetooth Maestro-Esclavo. [28]

Los modelos elegidos debido a sus características y a su simplicidad son:

- HC-05 módulo Maestro-Esclavo.
- HC-06 módulo Esclavo.

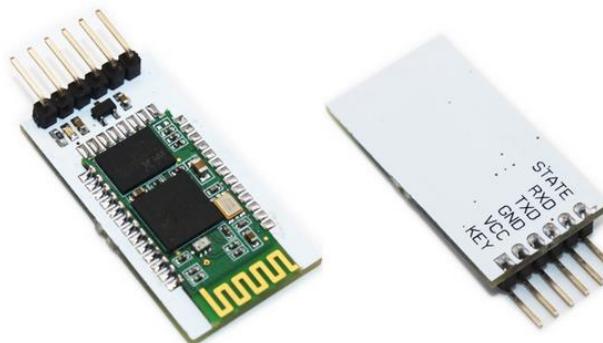


Figura 25. Módulo Bluetooth HC-05/HC-06.

## Modo de funcionamiento

Los módulos Bluetooth elegidos tienen dos tipos de funcionamiento de acuerdo a los valores lógicos que tomen cada uno de sus pines.

Para poder variar entre los dos modos se ha realizado el siguiente conexionado con la placa Arduino, ver Figura 26:

- KEY (Morado)- Pin digital de señal
- Vcc (Rojo)- 5V DC
- GND (Negro)- GND placa Arduino
- RXD (Amarillo)- TXD placa Arduino
- TXD (Verde)- RXD placa Arduino
- STATE (Naranja)- Pin digital de señal

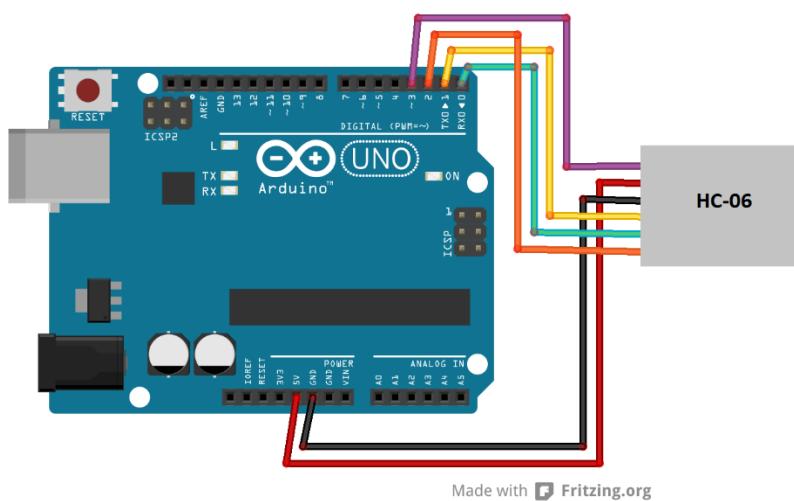


Figura 26. Conexionado Módulo Bluetooth con la placa Arduino.

## Modo de funcionamiento AT

En este modo es en el que se cambian los parámetros del Bluetooth tales como nombre, tasa de baudios de la comunicación y contraseña.

Además se pueden elegir los dispositivos con los que se va a emparejar el módulo y realizar la conexión.

Para que el Bluetooth funcione en modo AT el pin KEY debe de estar conectado a 5V. Para ello mediante software se indicará al pin de señal digital correspondiente a KEY el valor digital '1'.

En la ver Tabla 1, se muestran los comandos comunes para ambos módulos; Maestro-Eslavo.

El módulo Maestro-Esclavo utiliza otros muchos comandos entre los que destacan los indicados en la Tabla 2.

### Modo de funcionamiento de comunicación

Para que el módulo Bluetooth sea capaz de ser encontrado por otros dispositivos debe encontrarse en modo comunicación.

Para que el Bluetooth funcione en modo comunicación el pin KEY debe de estar conectado a tierra. Para ello mediante software se indicará al pin de señal digital correspondiente a KEY el valor digital ‘0’.

	HC-05		HC-06	
	Comando	Respuesta	Comando	Respuesta
Verificar comunicación con el módulo	AT	OK	AT	
Cambiar el nombre del módulo	AT+NAME= <i>nombre</i>	OK	AT+NAMEnombre	OKsetname
Cambiar la velocidad de comunicación	AT+UART=<baudios>, <bits de stop>, <bits de paridad>		AT+BAUDx Siendo x: 1-----1200 2-----2400 3-----4800 4-----9600 5-----19200 6-----38400 7-----57600 8-----115200	OKx Siendo x: 1200 2400 4800 9600 19200 38400 57600 115200

Tabla 1. Comandos AT comunes HC-05 y HC-06.

Descripción	Comando	Respuesta
Obtener la dirección del dispositivo del tipo NAP:UAP:LAP (Hexadecimal)	AT+ADDR?	+ADDR: <dirección> OK
Elegir el modo de funcionamiento del módulo	AT+ROLE=x Siendo x: 1-----Maestro 0-----Esclavo	OK
Emparejarse con un dispositivo	AT+PAIR=<Dirección del BT a	OK

	conectar>, <tiempo de conexión máximo en segundos>	FAIL
Conectarse con un dispositivo	AT+LINK= <Dirección del BT a conectar>	OK FAIL

Tabla 2. Comandos AT específicos HC-05.

Por tanto la comunicación funcionará de la siguiente manera:

- El dispositivo Maestro busca dispositivos en su entorno
- Entre los dispositivos encontrados envía una petición de emparejado al dispositivo elegido
- Si las contraseñas son iguales el emparejamiento se realiza correctamente y el módulo Bluetooth ya está listo para conectarse
- El maestro realiza una petición de conexión y se conecta con el dispositivo ya emparejado
- Los dispositivos están conectados y el esclavo puede leer la información que obtiene del máster correctamente

Hay que tener en cuenta que la tasa de baudios en ambos módulos maestro y esclavo ha de ser la misma para que los datos enviados por el maestro sean leídos e interpretados correctamente.

### 5.2.1.6 ALIMENTACIÓN

Teniendo en cuenta las características de los elementos a utilizar y de los umbrales de alimentación y de corriente soportados por las placas Freaduino se han utilizado dos tipos de alimentaciones externas para alimentar a algunos de sus componentes y a la misma placa para que funcione de manera independiente.

Asimismo de acuerdo a los ensayos realizados, y a los problemas de autosuficiencia y funcionamiento que suponía utilizar una única alimentación para todos los sensores y actuadores del robot se utilizarán alimentaciones auxiliares para los servomotores como se explicará a continuación.

#### Alimentación externa para el microcontrolador

Para evitar la conexión USB al PC y para dar autonomía a los mini-robots se van a utilizar pilas recargables de 9V DC y 150 MAH como las que se muestran en la Figura 27, suficientes para alimentar la placa Freaduino, los dos sensores de Ultrasonido y el módulo Bluetooth en cada uno de los mini-robots.

Para el conexionado con la placa Freaduino se ha añadido a la pila el conector DC 5.5x2.1 mm Macho de la Figura 28 para la entrada de alimentación general a la placa Freaduino.



Figura 27. Alimentación externa 9V.



Figura 28. Conector DC Alimentación externa a Freaduino.

## Alimentación actuadores

De acuerdo a las características eléctricas de los Servomotores, necesitan de 6V de entrada de alimentación en su pin de Vcc para ser alimentados. Debido a que la placa Freaduino únicamente puede aportar 5V de tensión máxima de salida, es necesario utilizar una alimentación externa.

Si no se utiliza una alimentación auxiliar para los mismos, se pueden dar problemas al cargar el software en la placa, así como en la comunicación serie con el ordenador mediante USB de manera que en múltiples ocasiones el puerto quedaba bloqueado o es indetectable. Para solucionar todos estos problemas es recomendable utilizar una alimentación de 6V DC para todos los robots, de forma que cada uno alimenta a dos servomotores.

La Figura 29 muestra el portapilas para las 4 pilas AA recargables de 1,3V DC cada una utilizadas como alimentación para los servomotores. Esta alimentación hace un total de 5,2V DC que de acuerdo a los ensayos realizados es suficiente para alimentar a los servomotores.



Figura 29. Alimentación externa de 6V con soporte.

### 5.2.1.7 ESTRUCTURA

La mayor parte de las piezas y componentes de los mini-robots han sido creadas mediante impresión 3D a través de diseños *Open Source*.

La estructura de estos robots es imprimible al 100% de manera que son fácilmente replicables y adaptables.

Las piezas que componen la estructura física de cada mini-robot son las que detallan a continuación.

Figura 30 se muestra el diseño 3D realizado en OpenSCAD del **chasis de los mini-robots**, soporte con espacio reservado para los dos servomotores y una alimentación auxiliar de 9V. [29]

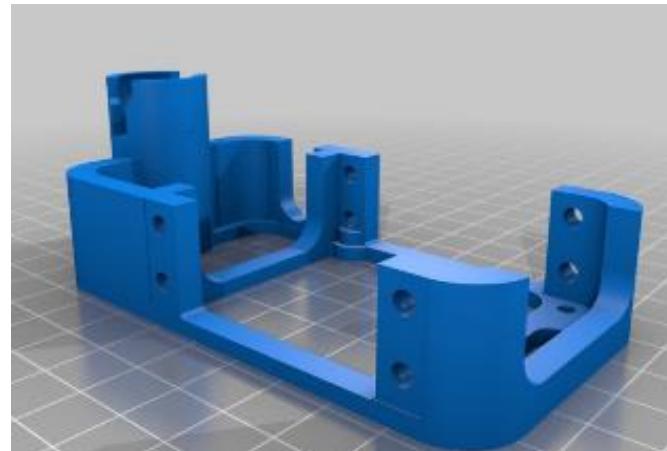


Figura 30. Chasis de los mini-robots.

En la Figura 31 puede observarse el diseño 3D de las **ruedas**, específicamente diseñadas para su uso con servomotores y con tamaño adaptado al Chasis.



Figura 31. Ruedas de los mini-robots.

En la Figura 32 se muestra el **soporte para los sensores de Ultrasonido**, pieza fácilmente adherible al chasis creada para la utilización con sensores de Ultrasonidos HC-SR04. [30]

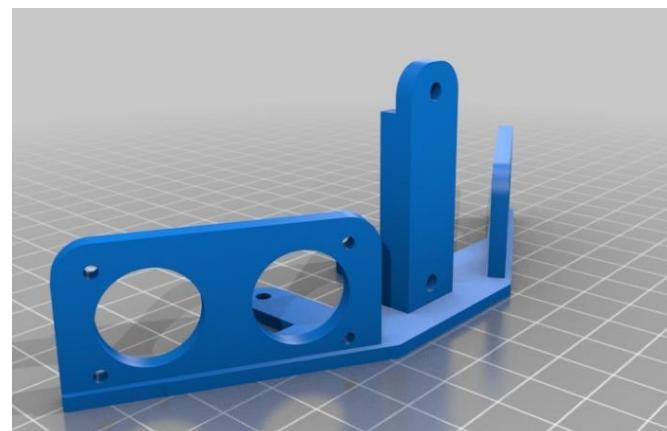


Figura 32. Soporte para los sensores de Ultrasonido.

### 5.2.1.8 ORDENADOR PORTATIL SONY VAIO

Para la realización de este proyecto se ha utilizado un portátil Sony VAIO modelo 3131.

Sus especificaciones técnicas son las siguientes:

- Pantalla de 12"
- Procesador AMD VISION E2
- 1,6 GHZ
- 4 GB RAM
- Disco duro 300 GB
- Windows 7 Home Premium preinstalado.

### 5.2.1.9 SMARTPHONE ANDROID

Durante la implementación de la aplicación para Smartphone Android se ha utilizado un Smartphone THL W100.

Sus características técnicas son las siguientes:

- Sistema operativo Android 4.3.1
- CPU MTK6589, Cortex A7 quad core, 1.2GHz; GPU: PowerVR SGX 544
- 4GB de memoria ROM
- 1GB de memoria RAM
- Pantalla de 4,5" con resolución de 960 x 540 pixeles

## 5.2.2 RECURSOS SOFTWARE

En este apartado se describirán los recursos de software involucrados en la realización de este proyecto, indicando los entornos de desarrollo en los que se ha llevado a cabo la implementación y las librerías utilizadas.

### 5.2.2.1 PROGRAMACIÓN E IDEs

#### Entorno de Desarrollo Arduino

Para la realización del software para el funcionamiento de los mini-robots se ha utilizado el entorno de desarrollo de Arduino así como librerías e información en código abierto disponibles. Para trabajar con un microcontrolador del estilo a Arduino y poder programarla se va a utilizar el entorno de desarrollo de Arduino, que permite trabajar de una manera más cómoda el software.

La versión utilizada para este proyecto es la 1.0 para Windows, descargable a través de la página web de Arduino.

Como todos los entornos de desarrollo dispone de un compilador y de un verificador de errores, y de permite añadir librerías e indicar el tipo de modelo de Arduino que se está utilizando. En la Figura 33 se puede observar el entorno de desarrollo.

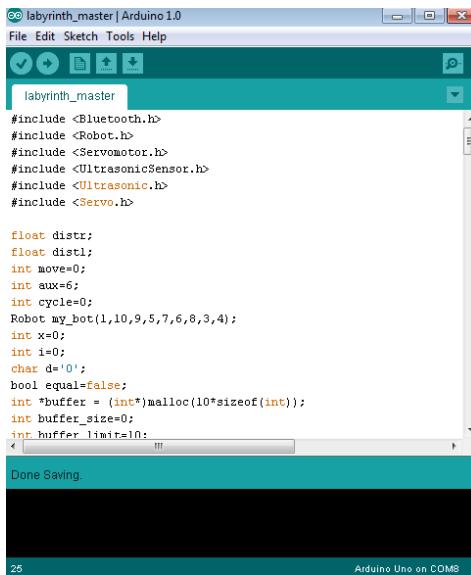


Figura 33. Arduino IDE.

## Librerías de Arduino

La página oficial de Arduino [23] proporciona librerías para utilizar componentes específicos tales como Servomotores o sensores de Ultrasonido.

- La librería **Servo.h** se puede importar al programa a través del IDE y se utilizará durante el proyecto para poder crear objetos en C++ de tipo Servo, y para asignar los valores de 0 a 180 de giro del servomotor.  
De esta manera a través del pin de señal digital cableado a los servos se enviará una señal PWM para asignar la velocidad y el sentido del movimiento.
- La librería **HCSR04Ultrasonic.h** es una librería *Open Source* que se puede obtener en la web de Arduino.  
Es específica para el modelo de sensores utilizado en este proyecto. Gracias a esta librería se puede obtener la distancia del obstáculo medida por los sensores en centímetros.

### 5.2.2.2 ENTORNO DE DESARROLLO ECLIPSE

Para elegir la metodología a llevar a cabo; resolución guiada o resolución libre, se ha creado una interfaz usuario para Smartphone Android mediante el entorno de programación Eclipse utilizando la herramienta Android SDK.

**Eclipse** es un programa de desarrollo de software compuesto por un conjunto de herramientas de programación de código abierto multiplataforma.

**Android SDK** proporciona librerías API<sup>9</sup> y las herramientas para desarrollar, compilar, testear, y depurar aplicaciones para Android [31].

### 5.2.2.3 ENTORNO DE DESARROLLO QT CREATOR

La interfaz para la tele-operación de los robots mediante el PC ha sido desarrollada mediante el software libre **QT Creator**.

**Qt Creator** es un entorno de programación integrado multiplataforma C++, JavaScript y QML, parte de SDK para el marco de aplicación QT GUI<sup>10</sup>.

Incluye un depurador visual, un diseño de interfaz gráfica de usuario y formas de diseño integrado. Qt Creator utiliza el compilador de C ++ de la colección de compiladores de GNU en Linux y FreeBSD. En Windows se puede utilizar MinGW o MSVC con la instalación por defecto y también puede utilizar cdb cuando se compila desde el código fuente. [32]

---

<sup>9</sup> Del inglés Application Programming Interface; Interfaz de programación de aplicaciones (IPA)

<sup>10</sup> Del inglés Graphical User Interface; Interfaz gráfica de usuario

# 6 PRESUPUESTO

Para el cálculo del presupuesto se han tenido en cuenta todos los medios materiales empleados en la realización de este proyecto y sus precios por unidad. Además se han estimado los costes por hora del personal involucrado en la realización del proyecto, para finalmente indicar su coste total.

## 6.1 COSTES MATERIALES

En este apartado se realizará el presupuesto de todos los materiales utilizados en la fabricación de los mini-robots, indicándose el coste de fabricación de un mini-robot individualmente y de toda la colonia.

Descripción	Cantidad	Precio Ud.	Precio unitario	Precio total
<b>Materiales</b>				
<b>Freaduino UNO</b> Suministro de la placa Freaduino UNO, con microcontrolador ATmega328 con 14 pines digitales de Entrada/Salida y 6 entradas analógicas entre otras características.	5	18,67€	93,35€	
<b>Servomotor de rotación continua SM-S4303R</b> Suministro del servomotor de rotación Continua SM-S4303R con velocidad de rotación a 6V de 0,13 sec/60°.	10	13,19€	131,90€	
<b>Módulo Bluetooth HC-05</b> Suministro del módulo Bluetooth HC-05 Maestro/Esclavo.	2	12,45€	24,90€	
<b>Módulo Bluetooth HC-06</b> Suministro del módulo Bluetooth HC-06 Esclavo.	3	10,90€	32,70€	
<b>Sensor de Ultrasonidos HC-SR04</b> Suministro del sensor de Ultrasonidos HC-SR04 capaz de detectar obstáculos a distancias entre 2-400cm.	10	2,72€	27,20€	

<b>Batería de 9V</b>	Suministro de batería de 9V recargable.	5	6,00€	30,00€
<b>Batería AA de 6V</b>	Suministro de batería AA recargable de 6V.	20	4,30€	86,00€
<b>Portapilas 9V</b>	Suministro portapilas para una pila de 9V.	5	0,35€	1,75€
<b>Portapilas 4xAA</b>	Suministro de portapilas con capacidad para 4 pilas AA.	5	1,10€	5,50€
<b>Conector Jack</b>	Suministro conector Jack compatible con entrada de alimentación de Arduino.	5	0,70€	3,50€
<b>Conector hembra</b>	Conector para conexión de alimentaciones externas	2	0,02€	0,04€
<b>Pin</b>	Pin de conexión.	60	0,05€	3,00€
<b>Chasis mini-robot</b>	Fabricación de un chasis de mini-robot por impresora 3D utilizando tarifa estimada de 0,1€/hora.	5	0,15€	0,75€
<b>Rueda</b>	Fabricación de una rueda de mini-robot por impresora 3D utilizando tarifa estimada de 0,1€/hora.	10	0,025€	0,25€
<b>Soporte para sensor de Ultrasonidos</b>	Fabricación de un soporte para sensor de Ultrasonidos modelo HC-SR04 por impresora 3D utilizando tarifa estimada de 0,1€/hora.	5	0,125€	0,62€
<b>COSTE POR MINI-ROBOT</b>			<b>88,30€</b>	
<b>TOTAL MATERIALES</b>			<b>441,46€</b>	

El coste total de los materiales utilizados en este proyecto es de **441,46€, Cuatrocientos cuarenta y un euros con 46 céntimos.**

## 6.2 COSTES DE PERSONAL

En este apartado se indicará el presupuesto del proyectante, director y codirector en la realización de este proyecto.

Descripción	Cantidad h.	Precio unitario	Precio total
<b>Personal</b>			
<i>Proyectante</i> María Blázquez Partido. Graduada en Grado en Ingeniería Electrónica Industrial y Automática.	400	25€	10.000€
<i>Director</i> Raúl Péruela Martínez. Personal Investigador en Formación del Departamento de Ingeniería de Sistemas y Automática y miembro investigador del grupo <i>Robotics Lab.</i>			
	100	60€	6.000€
<i>Co-director</i> Juan Miguel García Haro. Personal Investigador en Formación del Departamento de Ingeniería de Sistemas y Automática y miembro investigador del grupo <i>Robotics Lab.</i>			
	10	60€	600€
<b>TOTAL PERSONAL</b>			<b>16.600€</b>

El coste total de personal es de **16.600€, dieciséis mil seiscientos euros.**

## 6.3 RESUMEN DEL PRESUPUESTO

En la Tabla 3 se resume el presupuesto de Materiales y Personal indicando el presupuesto total.

Como se puede observar los robots utilizados en este proyecto son de bajo costo, siendo el precio por robot de **88,30€, ochenta y ocho euros con treinta céntimos**.

Descripción	Cantidad Ud.	Precio unitario	Precio total
Mini-robot modular	5	88,30€	441,46€
<b>TOTAL MATERIALES</b>			<b>441,46€</b>
<b>TOTAL PERSONAL</b>			<b>16.600€</b>
<b>TOTAL</b>			<b>17.041,46€</b>

Tabla 3. Presupuesto total.

El presupuesto total para la realización de este proyecto asciende a la cantidad de **17.041,46€, diecisiete mil cuarenta y un euros con cuarenta y seis céntimos**.

# 7 DISEÑO E IMPLEMENTACIÓN DEL SISTEMA

En este capítulo se hablará de las decisiones de diseño tomadas durante el proceso de desarrollo, indicando las funcionalidades y características del software creado y desarrollando su implementación.

Para el manejo de los sensores y actuadores que componen un robot se ha creado una librería compatible con Arduino como parte de los objetivos principales de este proyecto. De la misma forma, se ha creado una interfaz gráfica de usuario para PC y una aplicación para Smartphone Android, para complementar la librería creada y poder testearla en situaciones específicas.

## 7.1 LIBRERÍA DE CONTROL

Para la realización de esta librería se han creado las clases necesarias para su funcionamiento en un robot modular que puede contener servomotores, sensores de Ultrasonidos y un módulo Bluetooth. Como parte del proceso de diseño se han identificado e implementado las clases adecuadas para la adaptación de la librería a cualquier tipo de robot, buscando su integración en gran variedad de sistemas.

### 7.1.1 DISEÑO DEL SISTEMA

Se detallaran las clases que será necesario utilizar e implementar. Para cada clase se especificará el nombre de la clase, una descripción general de la clase, las variables que usa y los métodos que utiliza.

Para cada una de las clases que se describirán a continuación se distinguirán principalmente los siguientes apartados:

- **Nombre de la clase:** se especificará el nombre de la clase que se vaya a explicar. Este nombre será identificativo y se corresponderá con el estilo que tienen las librerías en Arduino para que la integración sea lo más correcta posible.
- **Descripción general de la clase:** se hará una descripción general de la funcionalidad y el manejo que tendrá la clase.
- **Métodos de la clase:** se detallaran las funciones que se usan para el correcto funcionamiento de dicha clase.

Para controlar los mini-robots, coordinarlos entre ellos y obtener información de su entorno se ha creado una librería de acuerdo al diagrama de clases que se muestra en la Figura 34.

El número de objetos de cada clase viene determinado por el tipo de robot a utilizar, en el diagrama de clases pueden observarse los elementos para los mini-robots del proyecto.

Las clases que componen este diagrama son las siguientes:

### 7.1.1.1 CLASE ROBOT

Clase principal que consta de los elementos propios de un robot tales como servomotores además de uno a varios sensores de ultrasonidos y un módulo Bluetooth.

Esta librería ha sido implementada para poder ser modificada fácilmente y adaptada a cualquier tipo de robot, pudiéndose modificar el número de servomotores a utilizar, por ejemplo; 4 en el caso de un robot cuadrúpedo.

#### Constructores

---

Para adaptarse a los elementos de cada uno de los robots se ha creado un constructor de la clase con gran parte de los parámetros por defecto, de manera que se pueda utilizar el mismo constructor independientemente de los elementos del robot.

Además se ha creado un constructor de copia para poder crear objetos iguales del tipo robot de manera rápida y sencilla.

#### Destructor

---

Con el fin de evitar carga innecesaria en el programa, se ha creado un destructor de la clase Robot para deshabilitar los servomotores y desvincularlos a los pines I/O específicos.

#### Funciones

---

A parte de las funciones específicas de obtención y modificación del valor de las variables privadas (sets() y gets()), la clase robot dispone de funciones para realizar los 5 movimientos básicos; ir recto, retroceder, girar a la derecha y girar a la izquierda.

## Funciones de movimiento

Con el objetivo de que los robots puedan realizar los movimientos los más similares y precisos posible se han calibrado los servomotores de cada uno de ellos.

Cada uno de los movimientos además tiene 4 velocidades de actuación de manera que el usuario puede indicar la velocidad más adecuada a sus necesidades.

Mediante ensayos experimentales se han obtenido los valores numéricos de realización de cada movimiento conforme a los robots descritos en la Tabla 4.

ROBOT 1			
	AVANZAR		RETROCEDER
	Izquierda	Derecha	Izquierda
Muy lento	99	87	87
Lento	104	82	81
Rápido	111	77	76
Turbo	180	0	0
			180

ROBOT 2			
	AVANZAR		RETROCEDER
	Izquierda	Derecha	Izquierda
Muy lento	100	87	88
Lento	105	82	82
Rápido	112	77	77
Turbo	180	0	0
			180

ROBOT 3			
	AVANZAR		RETROCEDER
	Izquierda	Derecha	Izquierda
Muy lento	105	87	91
Lento	108	82	86
Rápido	111	77	81
Turbo	180	0	0
			180

ROBOT 4				
	AVANZAR		RETROCEDER	
	Izquierda	Derecha	Izquierda	Derecha
Muy lento	98	90	88	101
Lento	103	85	82	107
Rápido	110	80	77	112
Turbo	180	0	0	180

ROBOT 5				
	AVANZAR		RETROCEDER	
	Izquierda	Derecha	Izquierda	Derecha
Muy lento	100	87	88	98
Lento	105	82	82	104
Rápido	111	77	77	109
Turbo	180	0	0	180

TODOS LOS ROBOTS	
PARAR	
Izquierda	Derecha
95	0

Tabla 4. Calibración de los Servomotores.

En el Diagrama 1 se indica el funcionamiento de las funciones para la realización de movimientos. En función del número de identificación del robot, “Robot ID”, se utilizarán los valores de calibración del servomotor asociado a cada robot a la velocidad elegida por el usuario.

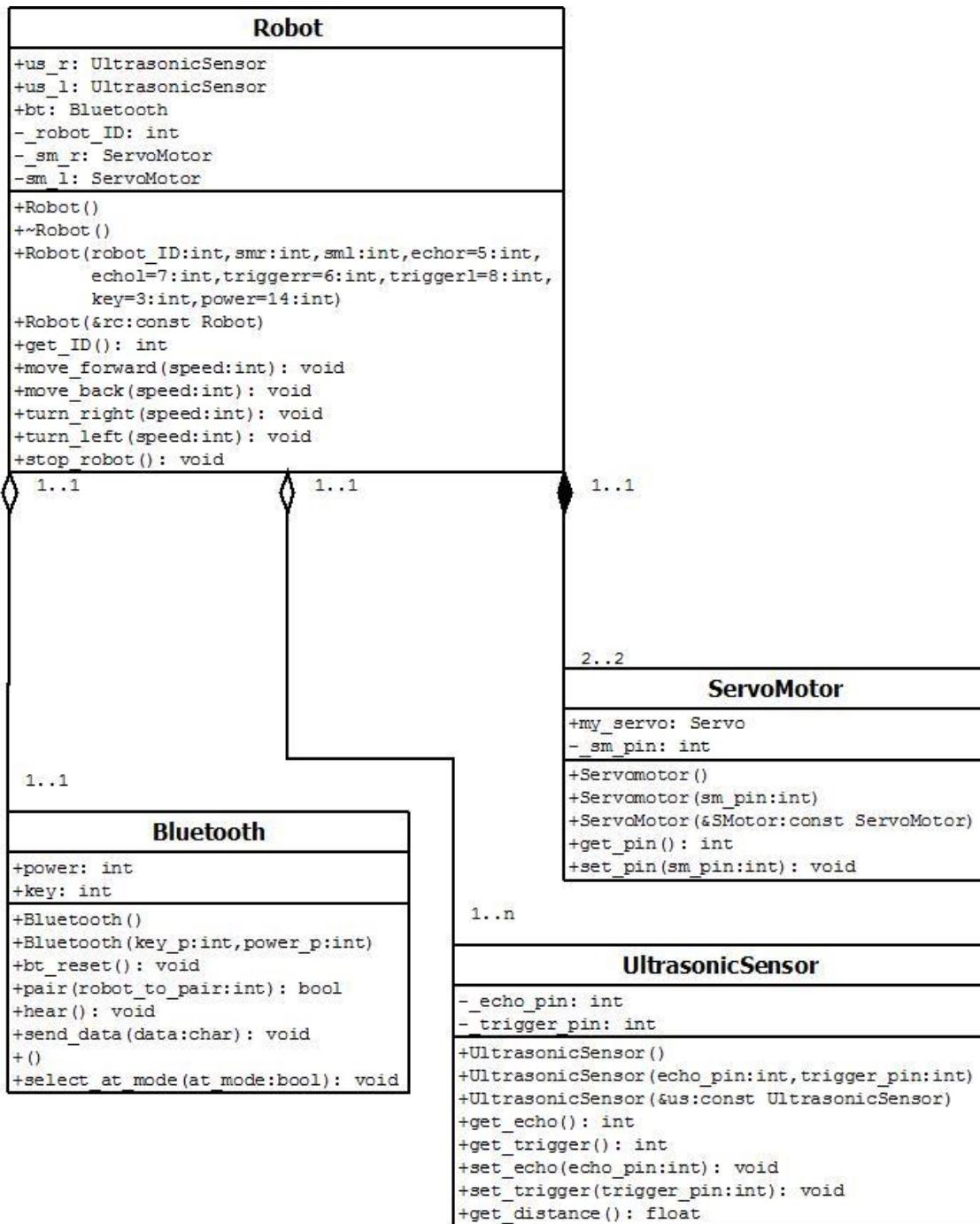


Figura 34. Diagrama de clases de la librería.

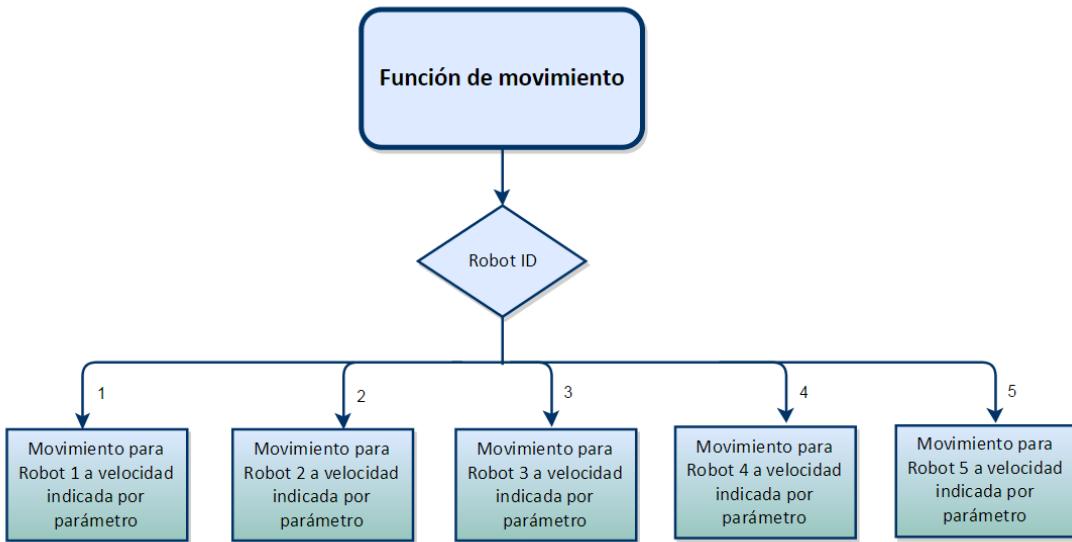


Diagrama 1. Funciones de movimiento.

### 7.1.1.2 CLASE SERVOMOTOR

La clase Servomotor representa a cada uno de los servomotores de rotación continua utilizados por los robots. Por cada uno de los mini-robot utilizados en este proyecto, se crearán 2 objetos de tipo Servomotor, cada uno de ellos asociado a una rueda; derecha e izquierda. Para otro tipo de robots con distinto número de servomotores, se creara un objeto Servomotor por cada uno.

### 7.1.1.3 CLASE ULTRASONICSENSOR

La clase UltrasonicSensor representa cada uno de los sensores de Ultrasonido que utiliza cada robot. Se crearán 2 objetos de tipo UltrasonicSensor por cada uno de los mini-robots utilizados en este proyecto, cada uno de ellos asociado a un sensor derecho e izquierdo. Para otro tipo de robots se creará un objeto del tipo UltrasonicSensor por cada sensor de Ultrasonidos de los que disponga.

## Funciones

A parte de las funciones específicas para obtener y modificar el valor de las variables privadas (`sets()` y `gets()`), la clase UltrasonicSesnsor consta de una función para obtener la distancia a la que se encuentra el obstáculo.

Para ello se utilizan algunas funcionalidades de la librería HCSR04Ultrasonic (ver Capítulo 5.2.2.1, Librerías de Arduino).

### 7.1.1.4 CLASE BLUETOOTH

La clase Bluetooth representa a cada módulo Bluetooth utilizado por los robots. Por cada robot se creará 1 objeto de tipo Bluetooth que en función del ID del robot hará referencia a un módulo u otro, como figura en la Tabla 5.

Robot ID	Módulo Bluetooth
1	M-1 (Maestro-Esclavo)
2	M-2 (Maestro-Esclavo)
3	S-1 (Esclavo)
4	S-2 (Esclavo)
5	S-3 (Esclavo)

Tabla 5. Correspondencia módulos Bluetooth con robots.

#### **Comunicación**

A través del puerto serial se puede realizar la conexión y comunicación bidireccional con la placa Freaduino.

Esto resulta interesante con el fin de poner manejar los valores obtenidos por los sensores y poder elegir un modo de actuación adecuado. Además de manejar esta información, se puede enviar información a los actuadores para que el robot realice las acciones adecuadas en cada situación.

En la comunicación serial bidireccional es necesario que la transmisión y recepción del PC se conecten a los pines de recepción y transmisión de la placa Freaduino, respectivamente.

Para cargar el software en la placa Freaduino se utiliza una comunicación serial a través de USB de manera que la información se transmite desde el PC a la placa.

En este proyecto se utilizará además la comunicación serial de la placa con el objetivo de poder manejar al robot de manera inalámbrica a través del Bluetooth. Los pines de transmisión TXD y recepción RXD de la placa Freaduino deben conectarse a los pines RXD y TXD respectivamente en el módulo Bluetooth. En la Figura 35 se muestra el conexionado que se indica a continuación:

- TXD - RXD placa Arduino (D0)
- RXD - TXD placa Freaduino (D1)
- STATE - Pin digital de señal (D2)
- KEY - Pin digital de señal (D3)
- Vcc (Rojo)- 5V DC
- GND (Negro)- GND placa Freaduino

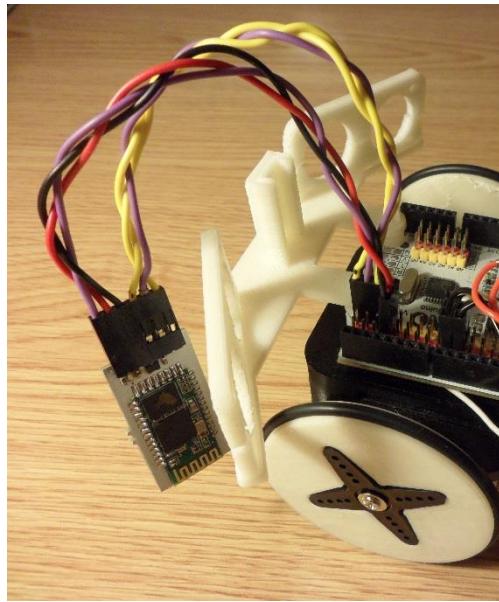


Figura 35. Conexionado del módulo Bluetooth con el mini-robot.

De esta forma el módulo Bluetooth podrá obtener y enviar la información de manera inalámbrica, mientras que actuará como conector serie de la placa Arduino.

Es importante tener en cuenta los tiempos de procesamiento de los módulos, de manera que cada vez que se recibe o envía un dato es aconsejable esperar una serie de milisegundos para evitar la confluencia de mensajes y por tanto la interpretación errónea.

Además para que todos los datos sean interpretados de la misma forma por todos los módulos Bluetooth deben funcionar a la misma velocidad de comunicación, en este caso se ha elegido 38400 baudios por segundo.

## Funciones

La clase Bluetooth consta de las funciones propias para el manejo del Bluetooth tanto si el módulo es maestro-esclavo como esclavo. Todas las funciones han sido creadas teniendo en cuenta el conexionado específico de los módulos Bluetooth con la placa Arduino y sus características.

### Función de reset

Para que el módulo Bluetooth cambie de modo AT a modo comunicación es necesario resetearlo, para ello se ha creado una función que se encarga de ello.

### Función de modo

Como ya se explicó anteriormente los módulos Bluetooth disponen de dos modos de funcionamiento, modo AT y modo conectividad. En el Diagrama 2 se muestra la función creada para utilizar ambos, que según el valor de la variable booleana recibida por parámetro selecciona uno u otro.

En adelante, al explicar las demás funciones se entenderá que al indicar “Seleccionar modo AT” o “Seleccionar modo comunicación”, se llama a esta función indicando por parámetros ‘0’ o ‘1’ según corresponda.

### **Función de emparejado**

Esta función es utilizada por el mini-robot líder para emparejarse con los demás mini-robots con módulos Bluetooth esclavos.

Esta función realizará el emparejamiento con el robot indicado por parámetro, y devolverá una variable booleana con valor true una vez que éste se haya realizado.

En el Diagrama 4 se muestra la función emparejar que se encarga de emparejar y conectar al robot Maestro con el robot Esclavo indicado.

### **Función de escucha**

El Diagrama 3 muestra la función de escucha que activa el módulo con rol Esclavo y en modo conectividad para poder ser encontrado por un dispositivo Maestro y recibir datos.

### **Función de envío**

Se encarga de enviar el dato de tipo carácter recibido por parámetro escribiéndolo en el puerto serie del Arduino, de manera que se envía a través del módulo Bluetooth.

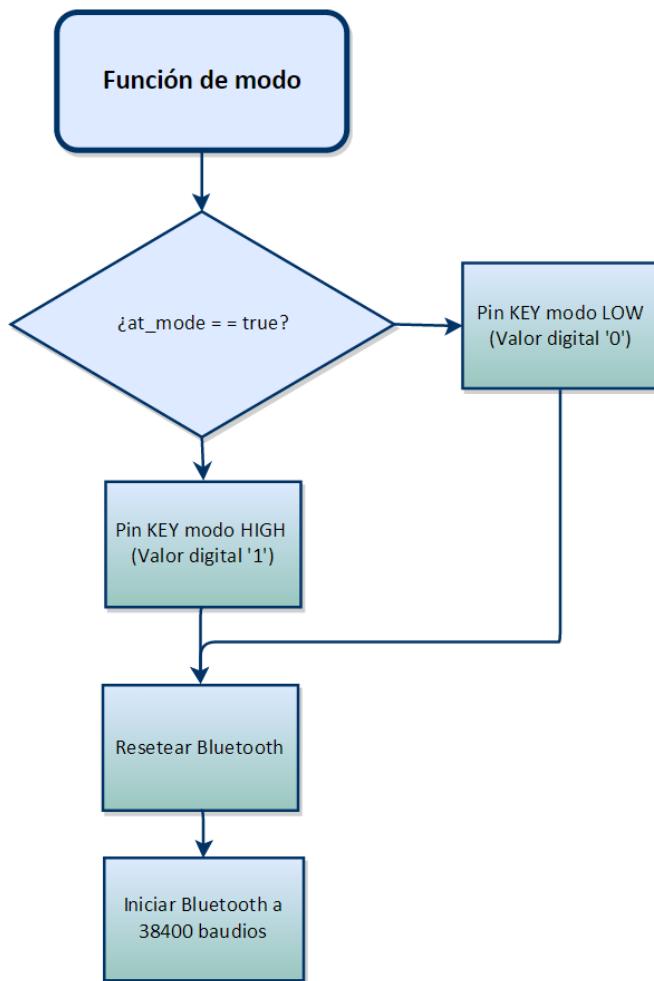


Diagrama 2. Función de modo.

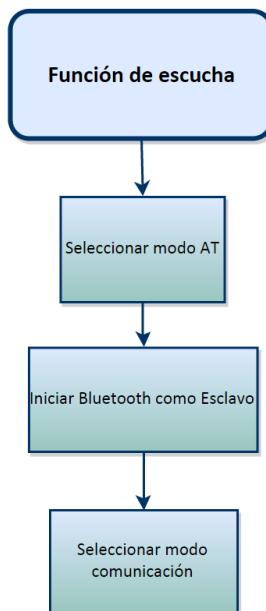


Diagrama 3. Función de escucha.

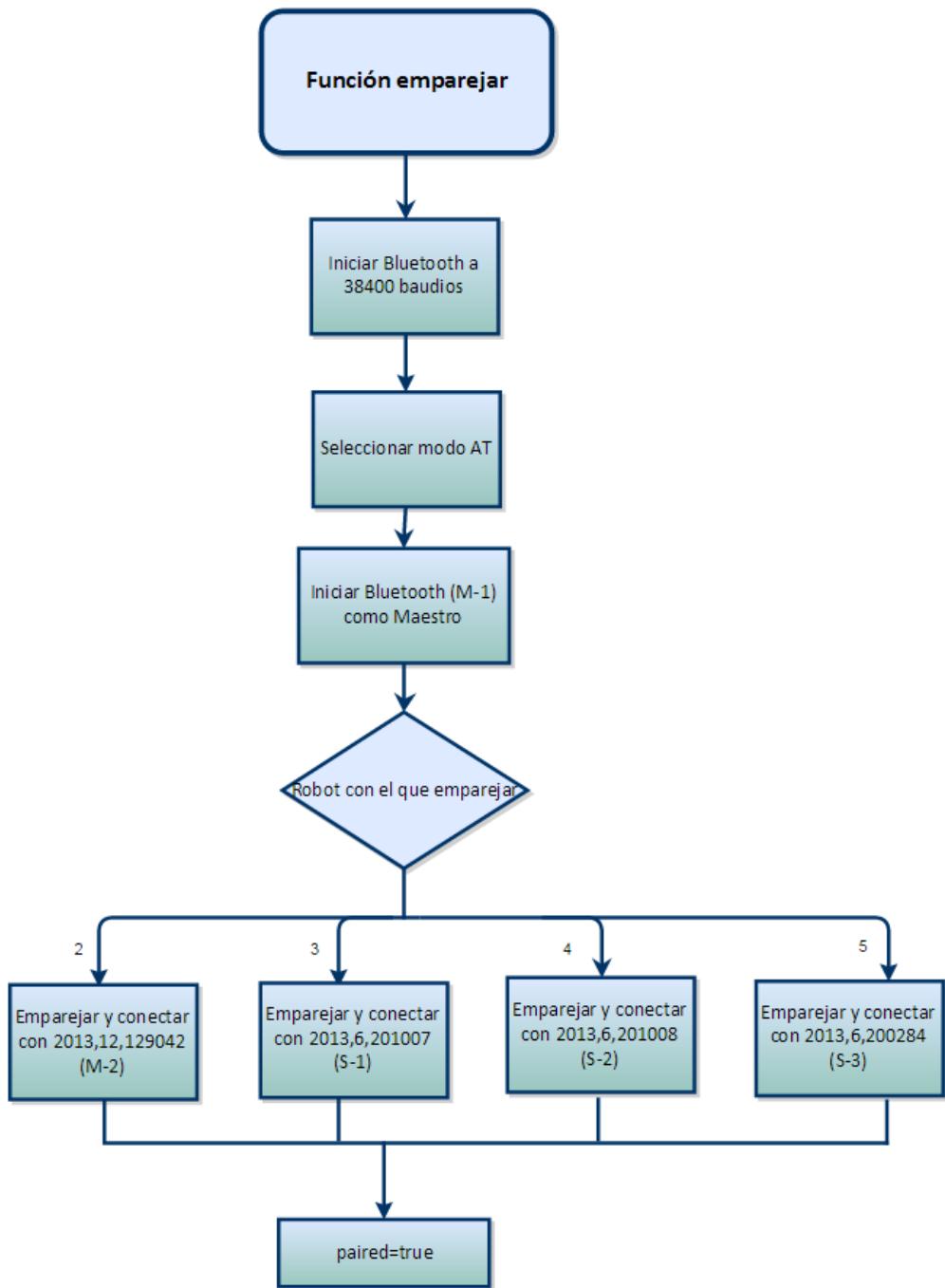


Diagrama 4. Función emparejar y conectar.

## 7.1.2 IMPLEMENTACIÓN

La implementación de la librería se ha realizado en C++ utilizando los elementos de Arduino, abarcando todas las funcionalidades de los periféricos utilizados en gran parte de los robots modulares.

### 7.1.2.1 CLASES

Las clases se han creado de lo particular a lo general, es decir, UltrasonicSensor, Bluetooth y ServoMotor son independientes del resto ya que no utilizan ninguna funcionalidad de las demás, mientras que la clase Robot es la que se utiliza como nexo de unión entre las demás, integrándolas.

### 7.1.2.2 CONSTRUCTORES

Los constructores utilizados en las clases UltrasonicSensor, ServoMotor y Bluetooth nunca son llamados directamente desde el programa principal, sino que son utilizados por la clase Robot como medio de creación de objetos de cada tipo.

**La clase Robot** consta de un constructor parametrizado con gran parte de sus parámetros por defecto como se muestra a continuación. En la implementación de este constructor se llama a los constructores del resto de clases.

Robot.h

```
/**  
 * @brief Robot constructor with default parameters  
 * @param smr: pin number where right servomotor signal is wired  
 * @param sml: pin number where left servomotor signal is wired  
 * @param echor: pin number where right ultrasonic sensor echo is  
 *               wired  
 * @param echol: pin number where left ultrasonic sensor echo is wired  
 * @param triggerr: pin number where right ultrasonic sensor trigger  
 *                  is wired  
 * @param triggerl: pin number where left ultrasonic sensor trigger  
 *                  is wired  
 * @param key: pin number where bluetooth module key pin is wired  
 * @param power: pin number where bluetooth module power pin is wired  
 * @return  
 */  
Robot(int robot_ID,int smr,int sml,int echor=5,int echol=7,int  
triggerr=6,int triggerl=8, int key=3, int power=14);
```

Las clases Robot, UltrasonicSensor y ServoMotor además constan de constructores copia para poder replicar objetos de la clase de manera rápida y sencilla tal y como se muestra a continuación:

Robot.cpp

```
Robot::Robot(const Robot& rc): _sm_r(rc._sm_r), _sm_l(rc._sm_l)  
{  
    _robot_ID=rc._robot_ID;  
}
```

## ServoMotor.cpp

```
ServoMotor:::ServoMotor(const ServoMotor & SMotor)
{
    _sm_pin=SMotor._sm_pin;
    my_servo=SMotor.my_servo;
}
```

## UltrasonicSensor.cpp

```
UltrasonicSensor:::UltrasonicSensor(const UltrasonicSensor & us)
{
    _echo_pin=us._echo_pin;
    _trigger_pin=us._trigger_pin;
}
```

## 7.1.2.3 DESTRUCTORES

Para desvincular los servomotores de sus pines correspondientes una vez finalizado el programa, se ha implementado un destructor de la clase Robot, como se muestra a continuación.

## Robot.cpp

```
Robot::~Robot()
{
    _sm_r.my_servo.detach();
    _sm_l.my_servo.detach();
}
```

## 7.1.2.4 ATRIBUTOS

Las clases disponen de unos atributos públicos y privados según el uso que se les vaya a dar, y según se requiera que sean accesibles o no dentro de la misma.

La clase Robot consta de atributos públicos y privados, de los cuales la mayor parte forman parte del resto de clases. Tal como se indica a continuación, la clase Robot consta de los siguientes atributos:

**Públicos**

- 2 sensores de ultrasonido Derecho e Izquierdo
- 1 Módulo Bluetooth

## Robot.h

```
UltrasonicSensor us_r;
UltrasonicSensor us_l;
Bluetooth bt;
```

**Privados**

- ID del Robot
- 2 Servomotores Derecho e Izquierdo

## Robot.h

```
int _robot_ID;
ServoMotor _sm_r;
ServoMotor _sm_l;
```

La clase ServoMotor consta de un atributo del tipo “Servo” como se muestra a continuación, proveniente de la librería “Servo.h” explicada en el apartado Librerías de Arduino.

## Servomotor.h

```
Servo my_servo; //! Arduino Servo.h library
```

### 7.1.2.5 FUNCIONES

La mayor parte de las funciones implementadas utilizan los pines I/O de Arduino. Un ejemplo de ello se muestra a continuación con la función de reseteo del Bluetooth dentro de la clase Bluetooth, que utiliza elementos propios de Arduino en la implementación de un método en C++.

## Bluetooth.cpp

```
void Bluetooth::bt_reset()
{
    Serial.flush();
    delay(500);

    /*!
     * Reset the bluetooth device using the power
     */

    digitalWrite(power, LOW);
    delay(1000);
```

```
    digitalWrite(power, HIGH);
    delay(500);
}
```

## 7.2 APLICACIÓN PARA SMARTPHONE

Se ha creado una Aplicación para Smartphone con sistema operativo Android para utilizar con los programas en Arduino creados utilizando la librería previamente descrita. En este apartado se indicarán aquellos aspectos relevantes en el diseño de la aplicación y en su implementación utilizando el entorno de desarrollo de Eclipse.

### 7.2.1 DISEÑO DEL SISTEMA

Para utilizar las funciones complejas creadas para la realización de las pruebas, se ha diseñado una aplicación para Smartphone con sistema operativo Android.

Ha sido testeada en un Smartphone con sistema operativo 4.1.1. y con una pantalla de 3,7" (480x800: hdpi).

A continuación se definirá el funcionamiento de la Clase Activity de Android y de las distintas actividades creadas en esta aplicación.

#### Clase Activity

Es la clase principal para interactuar con el usuario. Está diseñada para proporcionar una interfaz visual a través de la cual el usuario puede interactuar con la aplicación.

El usuario navegará por las distintas actividades de la aplicación, y Android le ayudará en esta navegación. El orden en el que los métodos del ciclo de vida de una actividad pueden ser invocados se muestra en Figura 36. [33]

Cuando lanzamos la aplicación se invocarán de manera sucesiva los métodos `onCreate()`, `onStart()` y `onResume()`. A continuación, la interfaz de usuario de la actividad aparecerá en la pantalla del dispositivo, lista para que el usuario pueda interactuar con ella. Existen los métodos `onPause()` y `onStop()` que se activan en circunstancias como por ejemplo cuando el Smartphone lleva mucho tiempo inactivo, y también el método `onDestroy()` que elimina la actividad por completo.

En la aplicación realizada en este proyecto, las principales tareas de la aplicación se han desarrollado dentro de los métodos `onCreate()` y `onDestroy()`.

Se han creado tres actividades distintas llamadas:

- MainActivity (Pantalla principal de selección de modo)
  - GuidedLabyrinthScreen (Pantalla para el laberinto guiado)
  - FreeLabyrinthScreen (Pantalla para el laberinto libre)

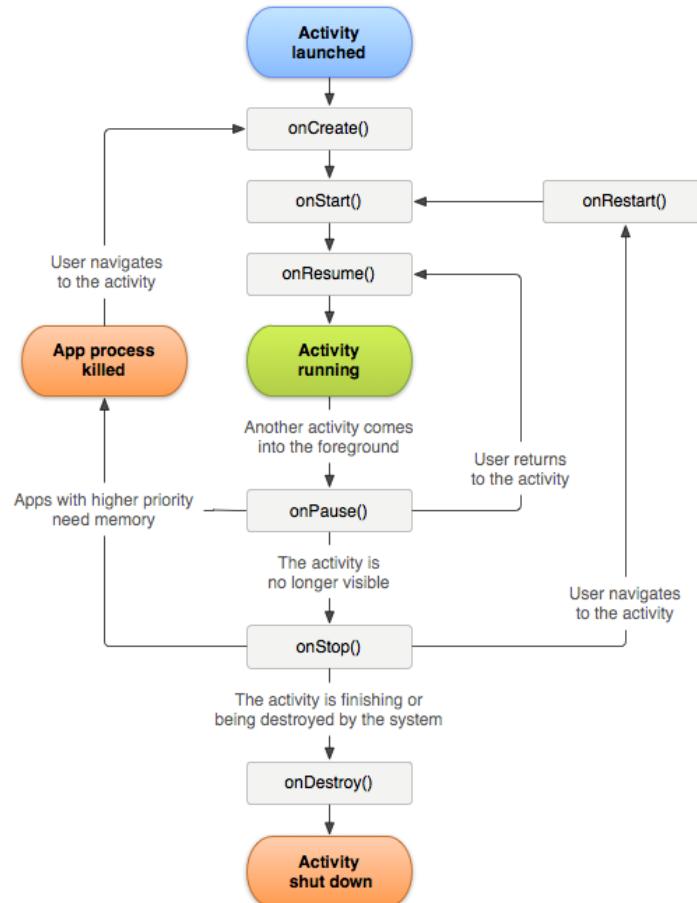


Figura 36. Ciclo de vida de las actividades en Android.

### Main Activity (Actividad principal)

En esta actividad, se podrán seleccionar las distintas metodologías de resolución del laberinto pulsando los respectivos botones. En el Diagrama 7 se muestra el funcionamiento del método `onCreate()` dónde se comprueba si el dispositivo dispone de Bluetooth y si es así se activa. Al pulsarse algunos de los botones disponibles en la ventana principal, en la Figura 37, se inicia la actividad correspondiente o se apaga el Bluetooth y se finaliza la actividad. En el método `onDestroy()` representado en el

Diagrama 5 , se desactiva el Bluetooth y se destruye la actividad existente.

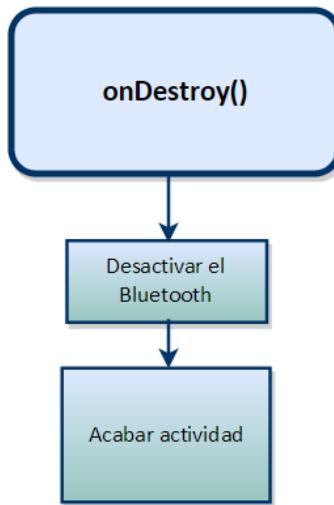


Diagrama 5. MainActivity – Método onDestroy().

### Guided labyrinth (Laberinto guiado)

---

En esta actividad el método onCreate() busca dispositivos en el entorno y espera a una conexión. Esta conexión se realiza paralelamente como interrupción, ver Diagrama 8. Una vez se ha realizado la conexión se envían los datos pertinentes a través del Bluetooth según el botón pulsado, tal como se indica en el Diagrama 9.

En el Diagrama 6 se muestra el método onDestroy() que finaliza la búsqueda de dispositivos Bluetooth y destruye la actividad existente.



Diagrama 6. GuidedLabyrinth – Método onDestroy().

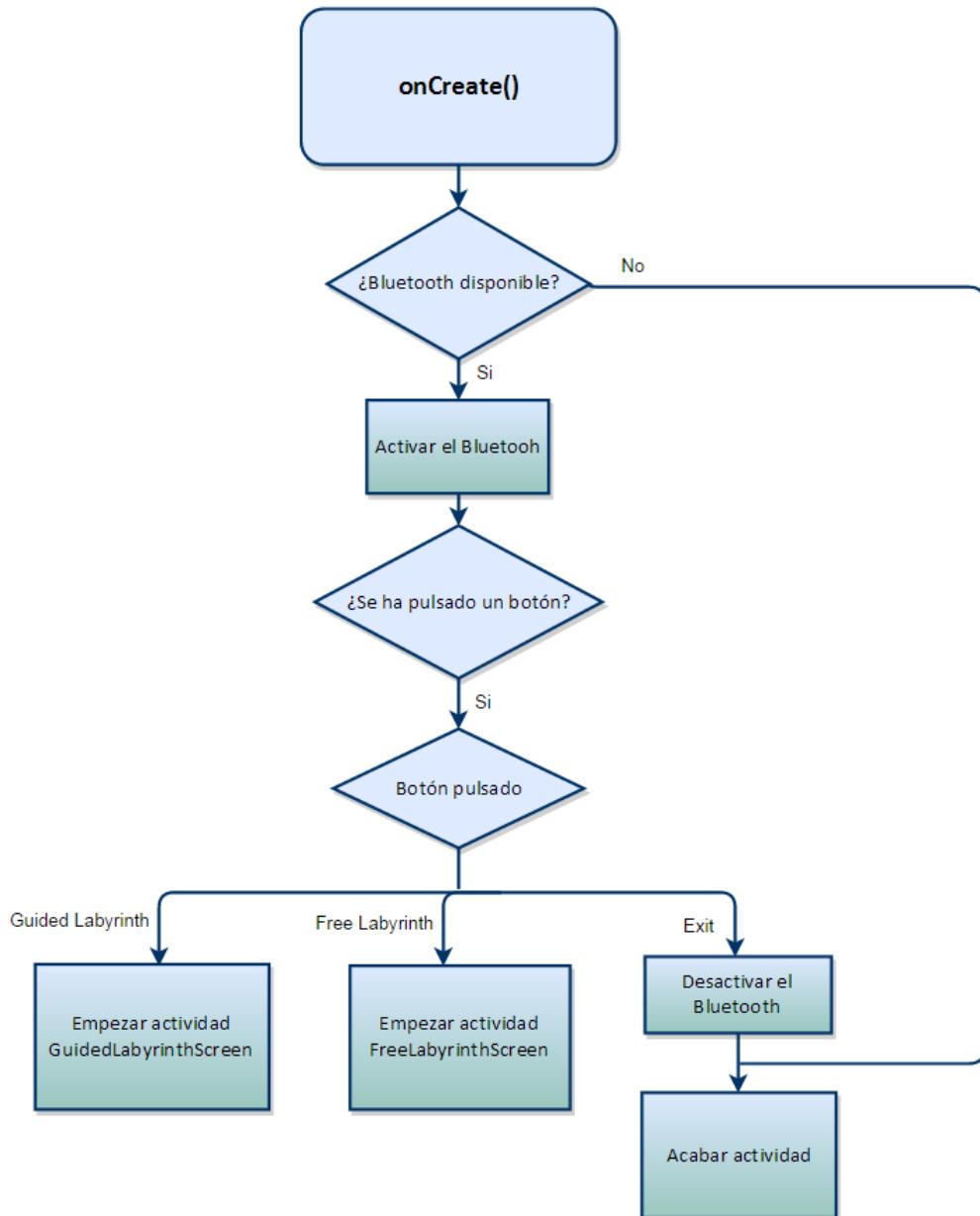
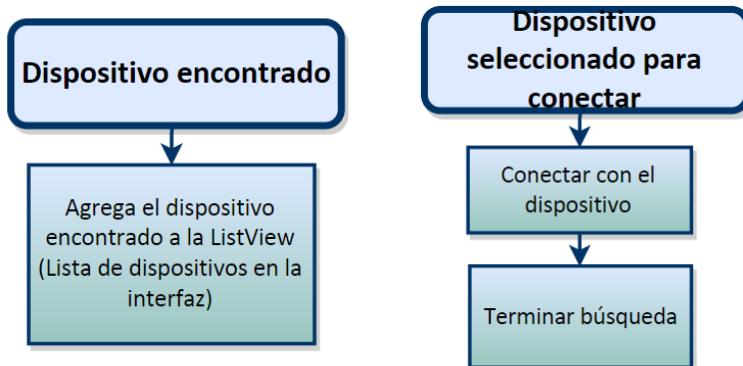
Diagrama 7. MainActivity – Método `onCreate()`.

Diagrama 8. Dispositivo encontrado y Dispositivo seleccionado para conectar.

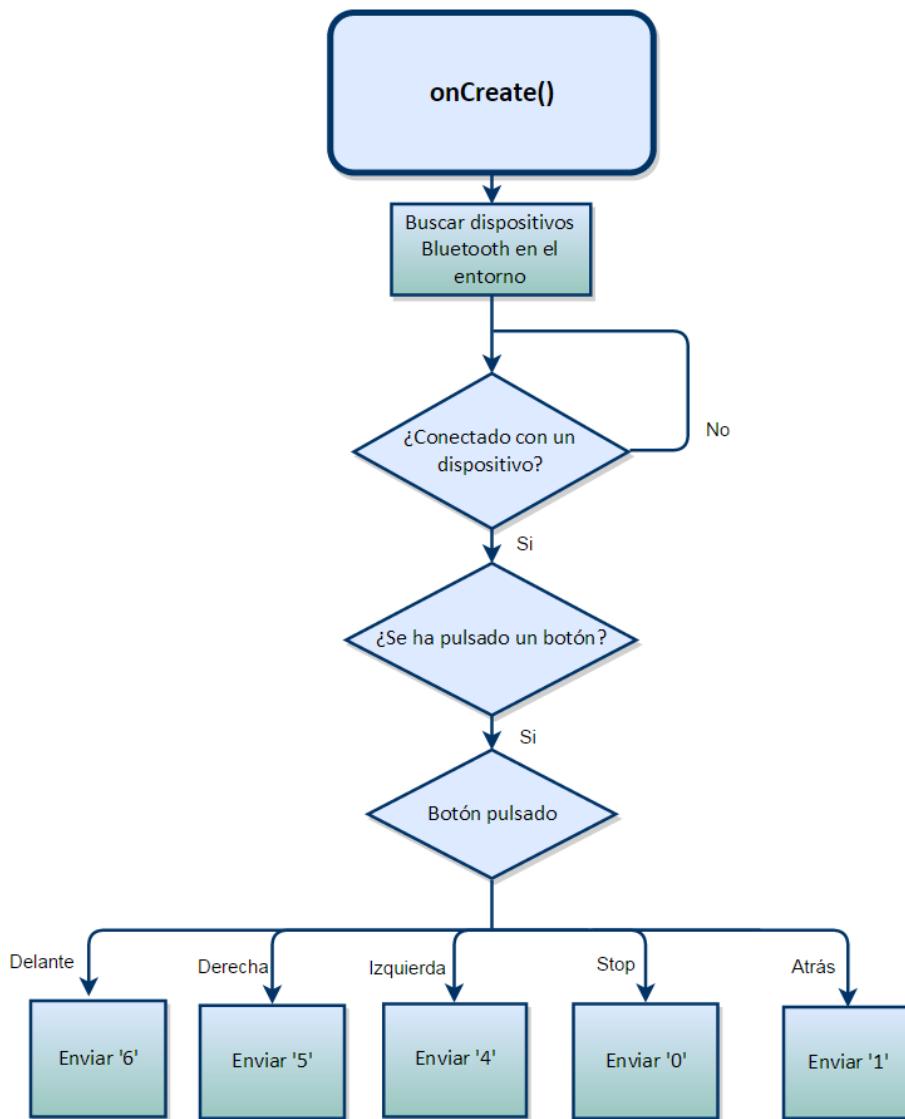
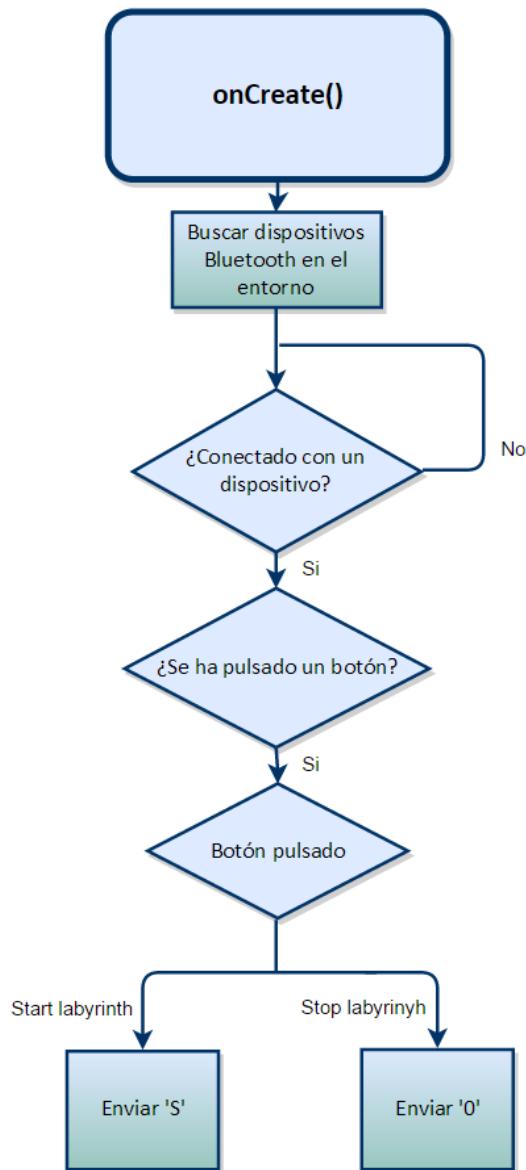


Diagrama 9. GuidedLabyrinth – Método onCreate().

## Free labyrinth (Laberinto libre)

En esta actividad, el método onDestroy() y las interrupciones son iguales que los de la actividad anterior, mostrados en el Diagrama 8 y Diagrama 6.

El método onCreate() para esta actividad, que se muestra en el Diagrama 10, opera de la misma forma que en el Diagrama 9, variando las acciones a realizar y los datos enviados en consecuencia.

Diagrama 10. Free labyrinth – Método `onCreate()`.

## 7.2.2 IMPLEMENTACIÓN

Se indicarán las interfaces gráficas de usuario diseñadas para el funcionamiento de la aplicación, bajo las que se encuentran los métodos `OnCreate()` y `OnDestroy()` descritos anteriormente.

### Main Activity (Actividad principal)

En la Figura 37 se muestra la interfaz gráfica principal que se encuentra el usuario al lanzar la aplicación. Esta interfaz dispone de tres botones para acceder a las dos actividades restantes, o salir de la aplicación.



Figura 37. Interfaz para Smartphone – Ventana principal.

#### Guided labyrinth (Laberinto guiado)

Para el laberinto guiado por el usuario se mostrará la interfaz gráfica de la actividad, mostrada en la Figura 38, en posición horizontal para un mejor manejo de los controles. El software ha sido adaptado al programa en Arduino utilizado por los mini-robots, enviando los datos pertinentes a cada movimiento.

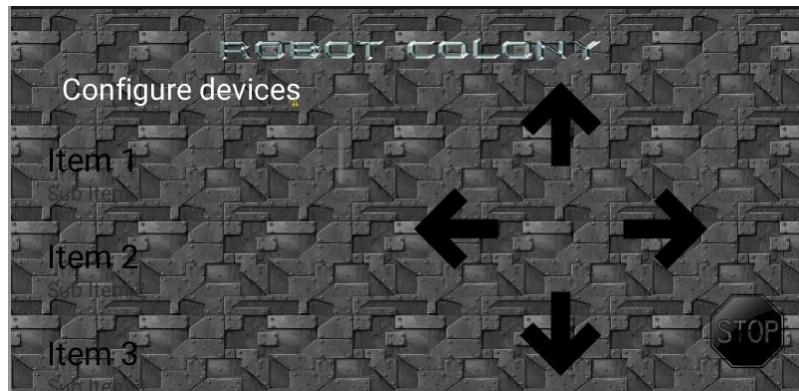


Figura 38. Interfaz para Smartphone – Guiar al robot.

#### Free labyrinth (Laberinto libre)

Para el laberinto libre se mostrará una interfaz gráfica como la de la Figura 39, en la que el usuario puede indicar el inicio y fin del laberinto. De la misma forma que en la actividad anterior, los datos serán enviados conforme a los requerimientos del programa en Arduino.

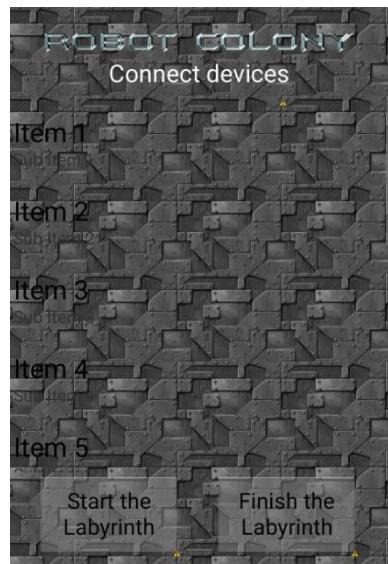


Figura 39. Interfaz para Smartphone – Resolución libre.

## AndroidManifest

---

En este archivo con extensión .xml se indicarán todas las especificaciones de las actividades en relación al dispositivo. A continuación se indicarán dos muy importantes. Para que la propia aplicación pueda gestionar el Bluetooth del dispositivo se necesitarán los permisos del mismo, para ello se debe incluir en AndroidManifest.xml el siguiente código:

### AndroidManifest.xml

```
<uses-permission android:name="android.permission.BLUETOOTH" />
<uses-permission android:name="android.permission.BLUETOOTH_ADMIN" />
```

Para indicar algunas especificaciones importantes se ha implementado el siguiente código que define la actividad principal, el nombre de todas las actividades, y la orientación en la que se mostrarán en el Smartphone, en este caso MainActivity y FreeLabyrinthScreen en vertical y GuidedLabyrinthScreen en horizontal.

## AndroidManifest.xml

```
<application
    android:allowBackup="true"
    android:icon="@drawable/ic_launcher"
    android:label="@string/app_name"
    android:theme="@style/AppTheme" >
    <activity
        android:name=".MainActivity"
        android:label="@string/app_name"
        android:screenOrientation="portrait" >
        <intent-filter>
            <action android:name="android.intent.action.MAIN" />
            <category android:name="android.intent.category.LAUNCHER" />
        </intent-filter>
    </activity>
    <activity
        android:name=".GuidedLabyrinthScreen"
        android:screenOrientation="landscape" >
    </activity>
    <activity
        android:name=".FreeLabyrinthScreen"
        android:screenOrientation="portrait" >
    </activity>
</application>
```

## 7.3 INTERFAZ GRÁFICA DE USUARIO PARA PC

En este apartado se describirán las especificaciones de diseño y la implementación llevada a cabo en la creación de una interfaz gráfica de usuario para PC utilizando la herramienta QT Creator.

### 7.3.1 DISEÑO DEL SISTEMA

Para la realización de la aplicación para PC se ha creado una *QT Widgets application* en C++. El modo de funcionamiento de la aplicación diseñada se explica detalladamente a continuación.

En primer lugar se han emparejado los módulos Bluetooth al PC, de forma que a cada uno de éstos les ha quedado asignado un puerto Serie en el ordenador. El puerto

COM<sup>11</sup> asignado a cada módulo puede observarse en las propiedades de cada dispositivo Bluetooth.

El usuario puede elegir con cuantos robots va a conectarse dentro del rango especificado de 1 a 5 robots.

Una vez seleccionados los robots se lleva a cabo la conexión con los mismos y el usuario ya está listo para tele-operar.

La tele-operación se puede llevar a cabo de dos maneras:

- Utilizando las teclas del teclado del PC que muestran en la Tabla 6.

<b>W</b>	Ir recto
<b>S</b>	Retroceder
<b>A</b>	Girar a la izquierda
<b>D</b>	Girar a la derecha
<b>X</b>	Parar

Tabla 6. Tele-operación mediante el teclado del PC.

- Utilizando los botones de dirección disponibles en la interfaz.

### 7.3.2 IMPLEMENTACIÓN

En la ver Figura 40 se muestra la vista general de la aplicación para PC. Se compone de una interfaz en la que se puede llevar a cabo la conexión con los dispositivos Bluetooth indicados y realizar la tele-operación.

Para el diseño gráfico se han utilizado los elementos disponibles en el cuadro de diseño como botones etiquetas, cuadros de texto etc. En la Figura 41 se muestra un ejemplo de personalización de la vista de la aplicación modificando los elementos individualmente utilizando código CSS<sup>12</sup>.

---

<sup>11</sup> Puerto Serie

<sup>12</sup> Del inglés Cascading Style Sheets; Hoja de estilos en cascada

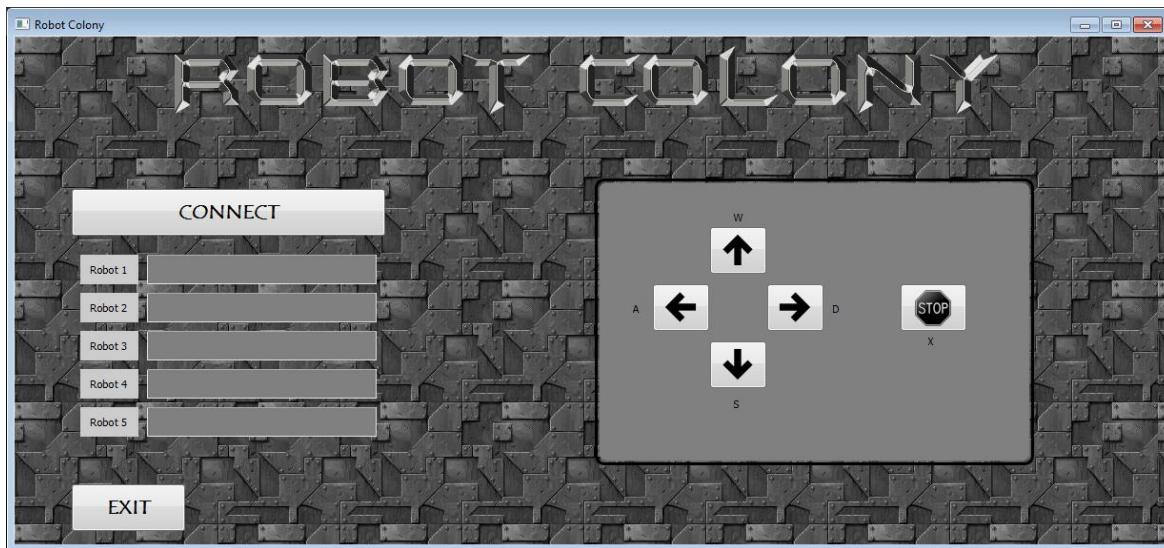


Figura 40. Interfaz para PC – Vista general.

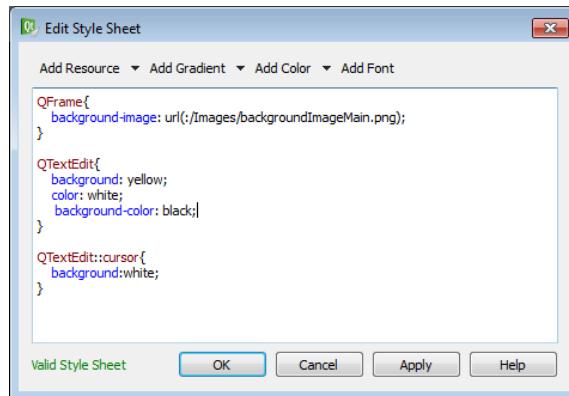


Figura 41. Editar diseño de los elementos de la interfaz.

En la interfaz de usuario, se han creado 5 cuadros de texto, que pueden apreciarse en la Figura 42, en los que se puede establecer el puerto COM de cada uno de los robots identificándolo como COMx, siendo x el número de puerto asignado.

Por cada cuadro de texto rellenado por el usuario se supone un robot, por lo que se pueden tele-operar de 1 a 5 robots a elección del usuario (Si no se rellena ningún cuadro de texto se supondrá que no hay robots a tele-operar).

Se creará por cada robot un objeto de tipo puerto Serie a partir de la librería “QSerialPort” disponible en el entorno. Cada objeto del tipo puerto serie creado tendrá asignadas las siguientes características:

- Velocidad: 38400 baudios por segundo
- 8 bits de datos
- No paridad
- 1 bit de stop

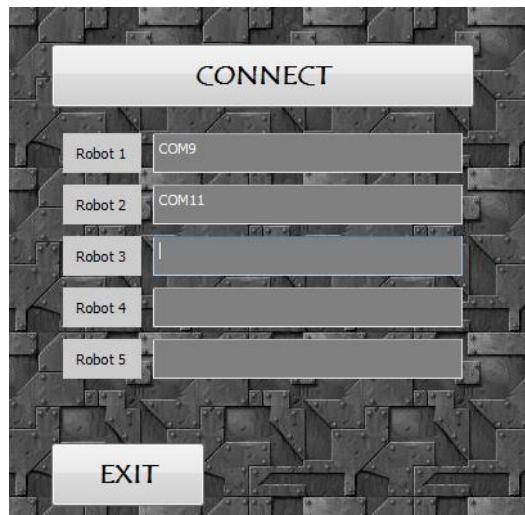


Figura 42. Interfaz para PC – Conexión con los robots.

La implementación en QT Creator para obtener la información de los cuadros de texto de la interfaz, y para crear objetos del tipo puerto Serie se muestra a continuación.

mainwindow.cpp

```
void MainWindow::on_pushButton_clicked()
{
    num_robots=0;
    arrayTextEdit[0]= ui->textEdit_1->toPlainText();
    arrayTextEdit[1]= ui->textEdit_2->toPlainText();
    arrayTextEdit[2]= ui->textEdit_3->toPlainText();
    arrayTextEdit[3]= ui->textEdit_4->toPlainText();
    arrayTextEdit[4]= ui->textEdit_5->toPlainText();

    for(int i=0;i<5;i++){
        if(arrayTextEdit[i]!=NULL){
            qDebug() << arrayTextEdit[i];
            arrayserial[i].setPortName(arrayTextEdit[i]);
            arrayserial[i].setBaudRate(QSerialPort::Baud38400);
            arrayserial[i].setDataBits(QSerialPort::Data8);
            arrayserial[i].setParity(QSerialPort::NoParity);
            arrayserial[i].setStopBits(QSerialPort::OneStop);
            arrayserial[i].setFlowControl(QSerialPort::NoFlowControl);
            arrayserial[i].open(QIODevice::WriteOnly);
            num_robots++;
        }
    }
}
```

Una vez que se han conectado uno o más robots con el PC, se pueden tele-operar mediante el teclado, o mediante los botones habilitados para la tarea en la interfaz.

### **Tele-operación mediante teclado**

Cuando se pulsa una tecla se activa el evento keyPressEvent(QKeyEvent \*event) y a continuación se comprueba si se ha pulsado alguna de las teclas correspondientes a la tele-operación en cuyo caso se enviará la correspondiente información a los robots.

mainwindow.cpp

```
void MainWindow::keyPressEvent(QKeyEvent *event)
{
    // Left -> A
    if (event->key() == Qt::Key_A)
    {
        for(int i=0;i<num_robots;i++) {
            arrayserial[i].write("4");
        }
    }
}
```

### ***Tele-operación mediante los botones***

Se ha creado un cuadro de control con 5 botones para cada uno de los movimientos a realizar por los robots.

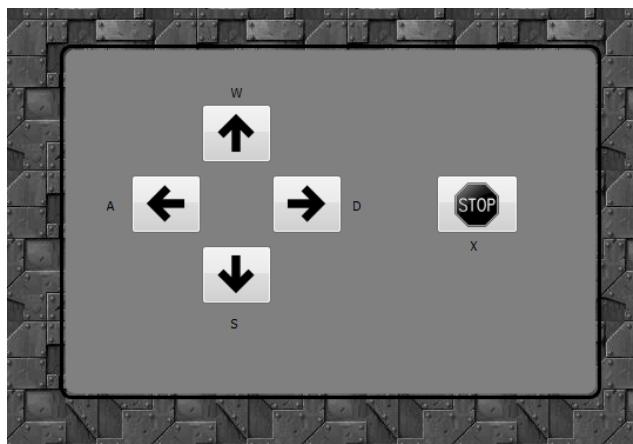


Figura 43. Tele-operación.

Cuando se pulsa alguno de los botones de dirección o de parada se activa el evento on\_pushButton\_clicked().

mainwindow.cpp

```
void MainWindow::on_pushButton_left_clicked()
{
    for(int i=0;i<num_robots;i++) {
        arrayserial[i].write("4");
    }
}
```

# 8 PRUEBAS Y RESULTADOS

En este capítulo se indicarán las pruebas realizadas para el testeo de todas las funcionalidades de la librería creada y del correcto funcionamiento de todos los elementos de los mini-robots.

## 8.1 PRUEBAS DE CALIBRACIÓN

Para comprobar que todos los elementos de los mini-robots funcionan de acuerdo a la librería creada, se han realizado pruebas en todos los mini-robots de manera individual.

### 8.1.1 DESARROLLO

Para las pruebas de calibración se han realizado las siguientes comprobaciones en cada mini-robot:

- Todos los constructores de la librería obtienen correctamente los valores obtenidos por parámetros y en consecuencia los pines de I/O en la placa Arduino están correctamente asignados.
- Se responde a las ordenes especificadas en el software por medio de la librería:
  - Los movimientos se realizan correctamente y de forma consecuente a las indicaciones de velocidad y tipo de movimiento.
- La información obtenida desde el robot es debidamente interpretada:
  - Se detectan obstáculos en el entorno y se obtiene con exactitud la distancia a la que se encuentran.
- El manejo de los módulos Bluetooth es correcto independientemente de si se trata de un módulo Maestro-Esclavo o Esclavo.

### 8.1.2 RESULTADOS

Los resultados obtenidos en esta prueba, tanto para la funcionalidad de los componentes como de la librería, han sido muy satisfactorios.

Se ha comprobado el correcto funcionamiento de todas las especificaciones de la librería creada:

- Constructores parametrizados y copia
- Funciones de movimiento (actuación en el entorno)
- Funciones de sensorización (obtener información del entorno)
- Funciones de comunicación (Comunicación Serie)

## 8.2 PRUEBAS ORIENTADAS A COMUNICACIÓN PUNTO A PUNTO

Como aplicación de las librerías creadas se implementado un pequeño laberinto para testear la precisión de detección de obstáculos, de actuación y de comunicación.

### 8.2.1 DESCRIPCIÓN DEL PROBLEMA

Una de las aplicaciones más importantes de la coordinación de robots es la exploración de entornos desconocidos y la capacidad de desenvolverse por los mismos con soltura.

Es importante en una colonia que si a alguno de los componentes de uno de los robots está dañado, éste pueda poder manejarse de manera autónoma recibiendo la información de otro miembro de la colonia.

Para simular un problema de estas características se planteará la siguiente situación:

- Se creará un escenario de un laberinto simple con varios cambios de dirección.  
Para la resolución del laberinto se proponen dos situaciones:

#### **Resolución libre**

- El robot líder de la colonia dispondrá de sensores para poder detectar los obstáculos.
- El resto de robots de la colonia no dispondrá de ningún sensor de manera que toda la información que obtendrán del entorno será a través del master. Deberán replicar los movimientos del líder y resolver el laberinto. La disposición en el laberinto será como se muestra en la Figura 44.

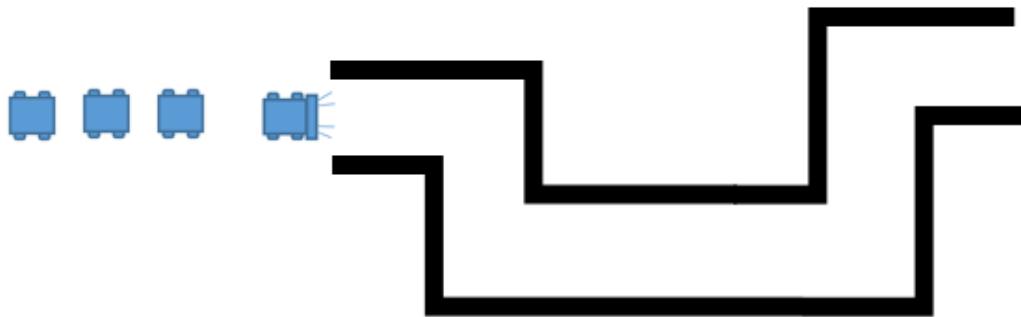


Figura 44. Disposición de los robots en el laberinto.

#### Resolución guiada

- El que el usuario mediante una interfaz dirigirá al robot líder para que solucione el laberinto, es decir, gravará el movimiento en el robot y una vez terminado, el robot líder se encargará de enviar la información a los demás robots.

### 8.2.2 DESARROLLO

Para la resolución se ha creado un programa en Arduino diferente para el mini-robot líder y para los demás mini-robots.

Además se ha implementado una aplicación para Smartphone Android, para guiar a los robots en la resolución del laberinto.

#### 8.2.2.1 COMPORTAMIENTO

La resolución del laberinto por el líder de los mini-robots se ha implementado según Diagrama 11.

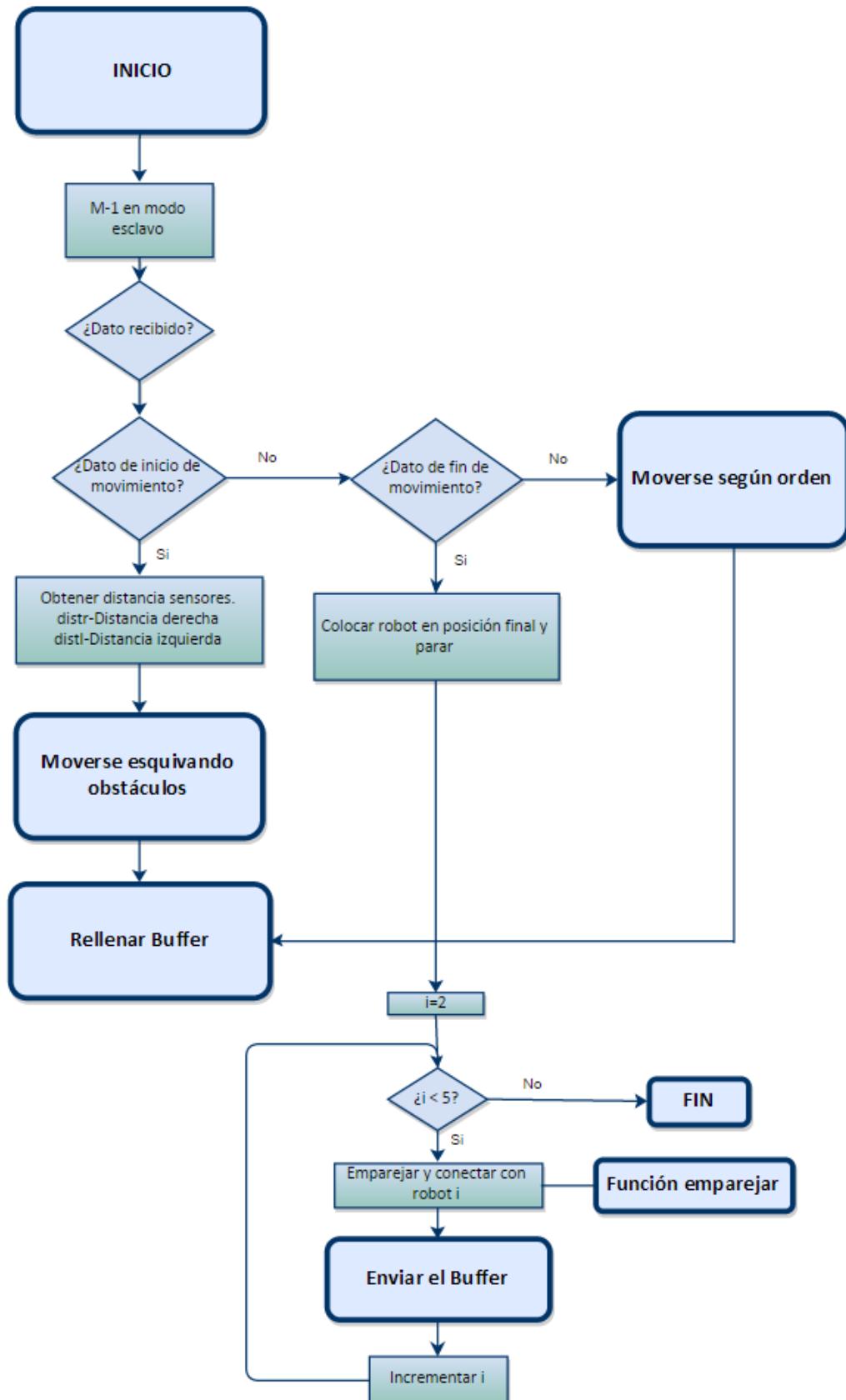


Diagrama 11. Resolución del laberinto.

En el Diagrama 12 se muestra la implementación en la situación en la que el laberinto es libre el robot debe utilizar los sensores de Ultrasonidos para esquivar los obstáculos.

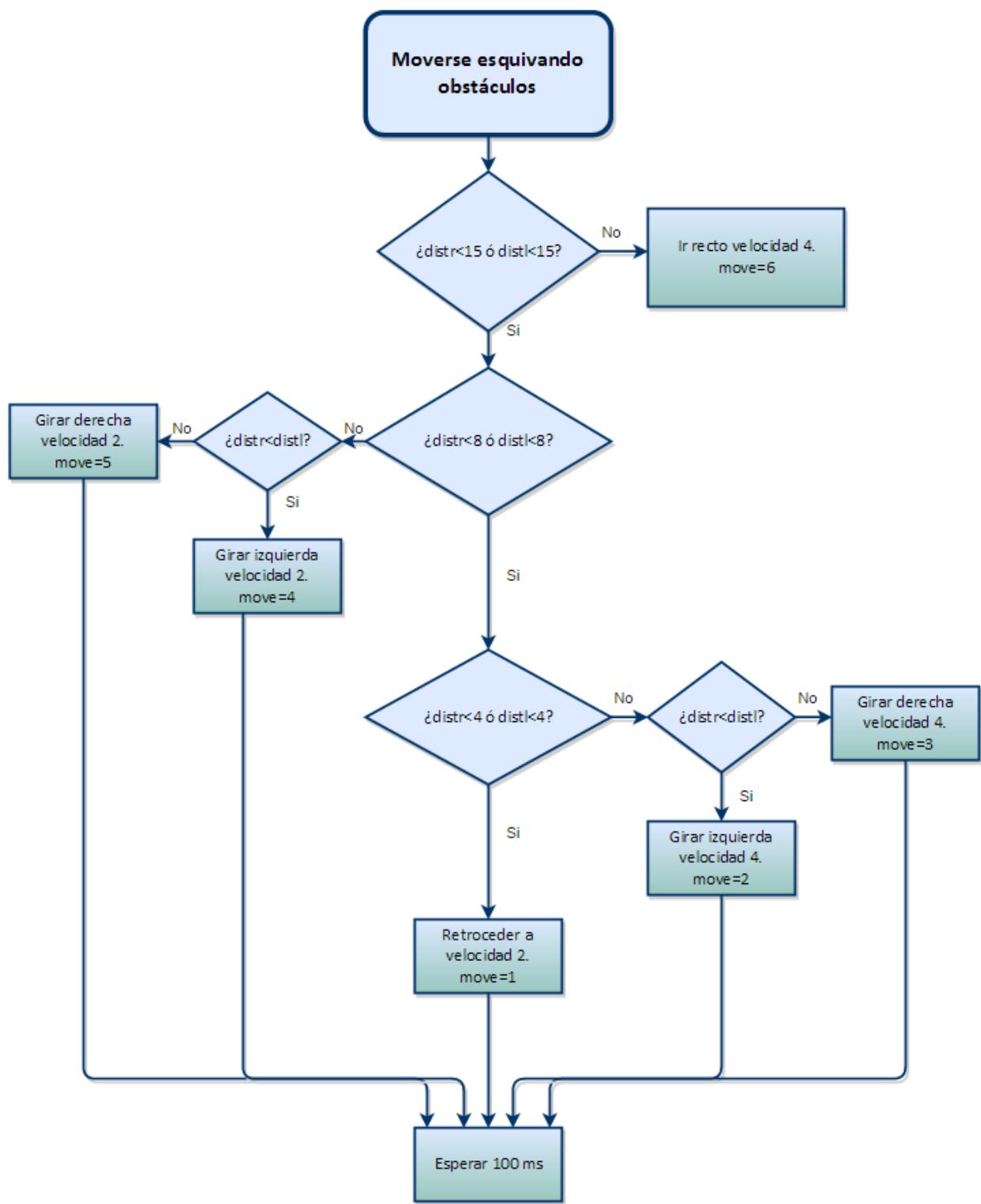


Diagrama 12. Moverse esquivando obstáculos.

Cuando el laberinto es guiado a través del usuario el robot se mueve por el entorno en función de las órdenes recibidas de acuerdo al Diagrama 13.

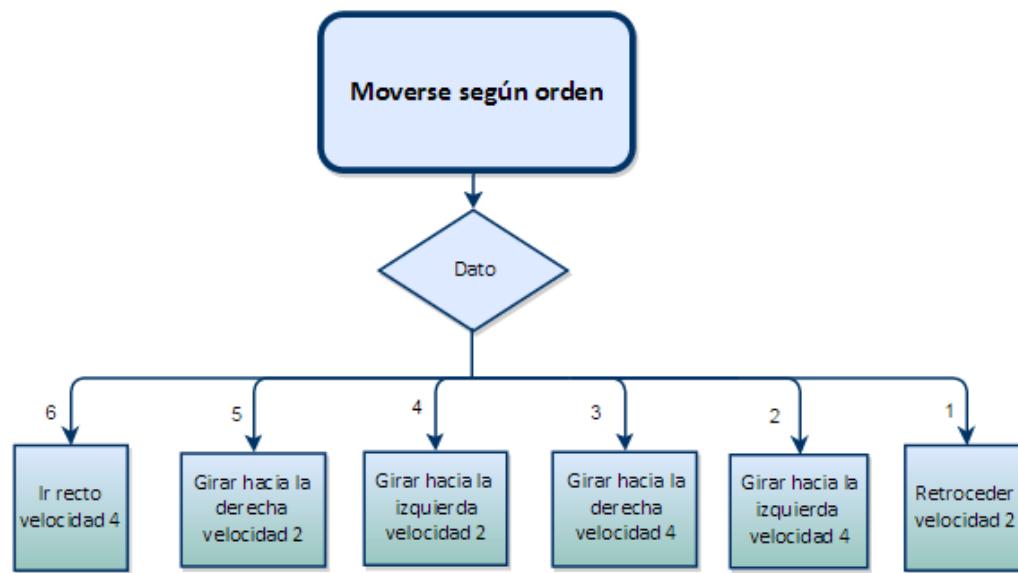


Diagrama 13. Moverse según orden.

A medida que se resuelve el laberinto en el modo libre o en el guiado se almacenan los datos en un Buffer de acuerdo al Diagrama 14.

El Buffer es un array dinámico de forma que empieza con 10 posiciones y se puede ir haciendo más grande a medida que es necesario.

En la Figura 45 se muestra una representación gráfica del buffer de resolución del laberinto.

Movimiento	Número de ciclos	Movimiento	Número de ciclos
Posición	0	1	2

Figura 45. Buffer de resolución del laberinto.

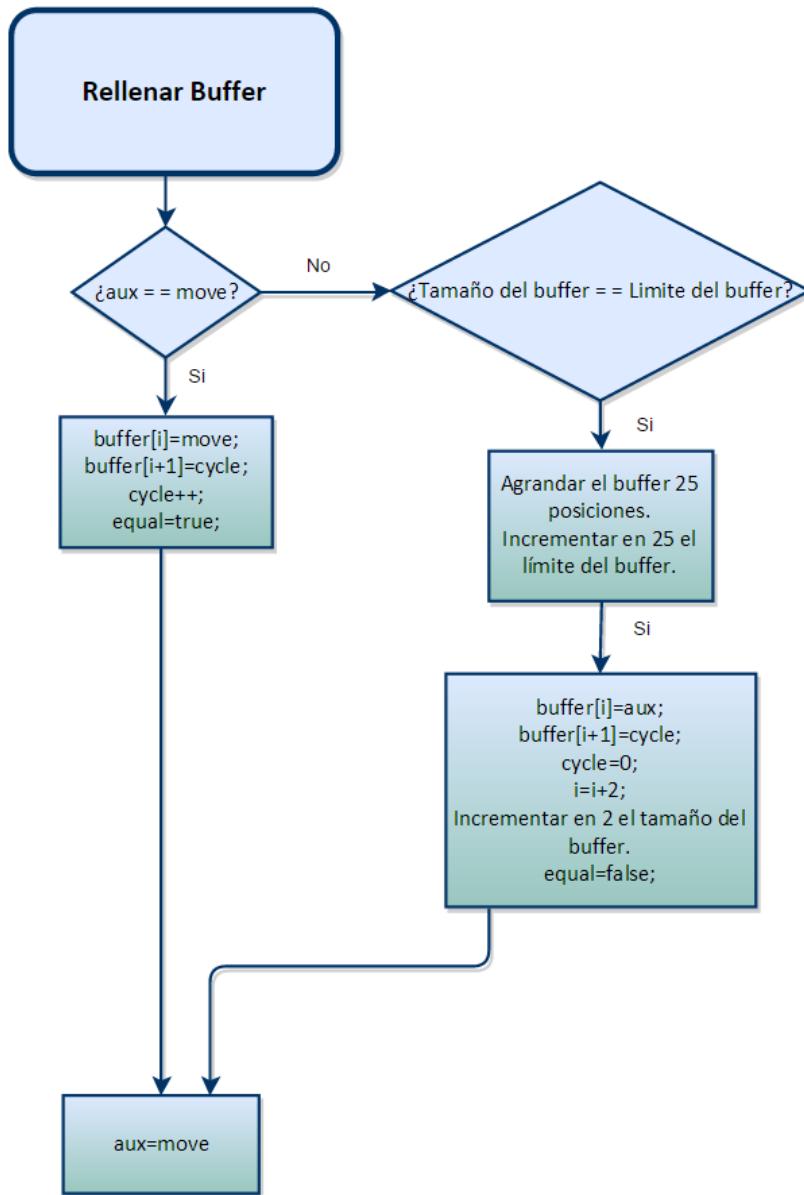


Diagrama 14. Rellenar Buffer de resolución.

En las posiciones pares del Buffer se almacenará el dato con el movimiento que se está realizando. En las posiciones impares se almacenará el valor del número de ciclos que se ha repetido el movimiento de la posición anterior, por ejemplo si al empezar a resolver el laberinto el robot avanza hacia delante durante 45 ciclos de ejecución del programa las posiciones 0 y 1 del buffer serán 6 y 45 respectivamente.

Tal como se muestra en el Diagrama 15, una vez que se recibe la orden del usuario indicando que ya se ha resuelto el laberinto, el robot líder se empareja con los robots de la colonia uno a uno y envía la información del Buffer.

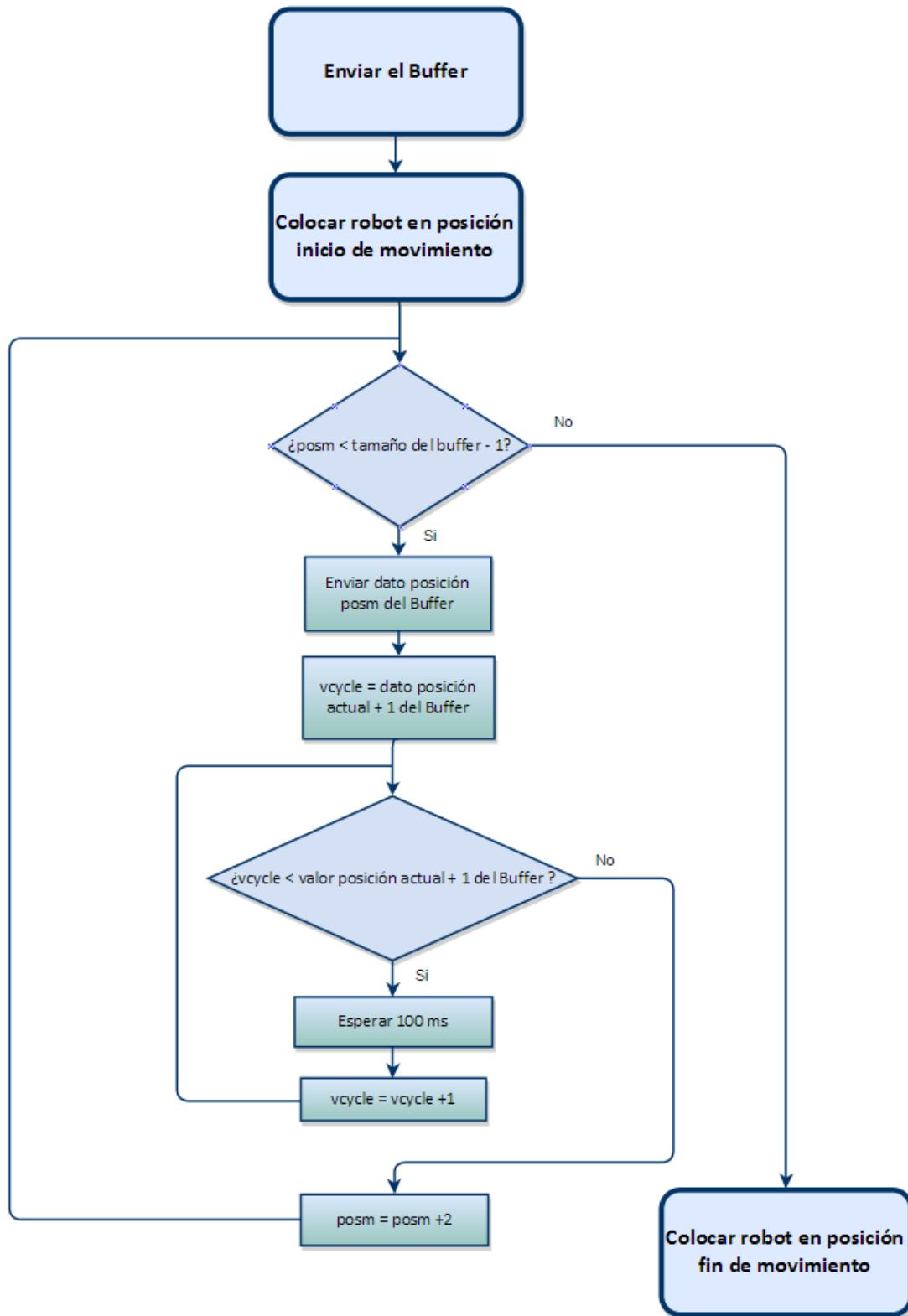


Diagrama 15. Enviar el Buffer.

Asimismo antes de enviar el buffer con los datos de la resolución se colocará el robot conectado en la posición de inicio. A continuación de enviará el buffer y por último se colocará el robot en su posición final según el Diagrama 16 y Diagrama 17.

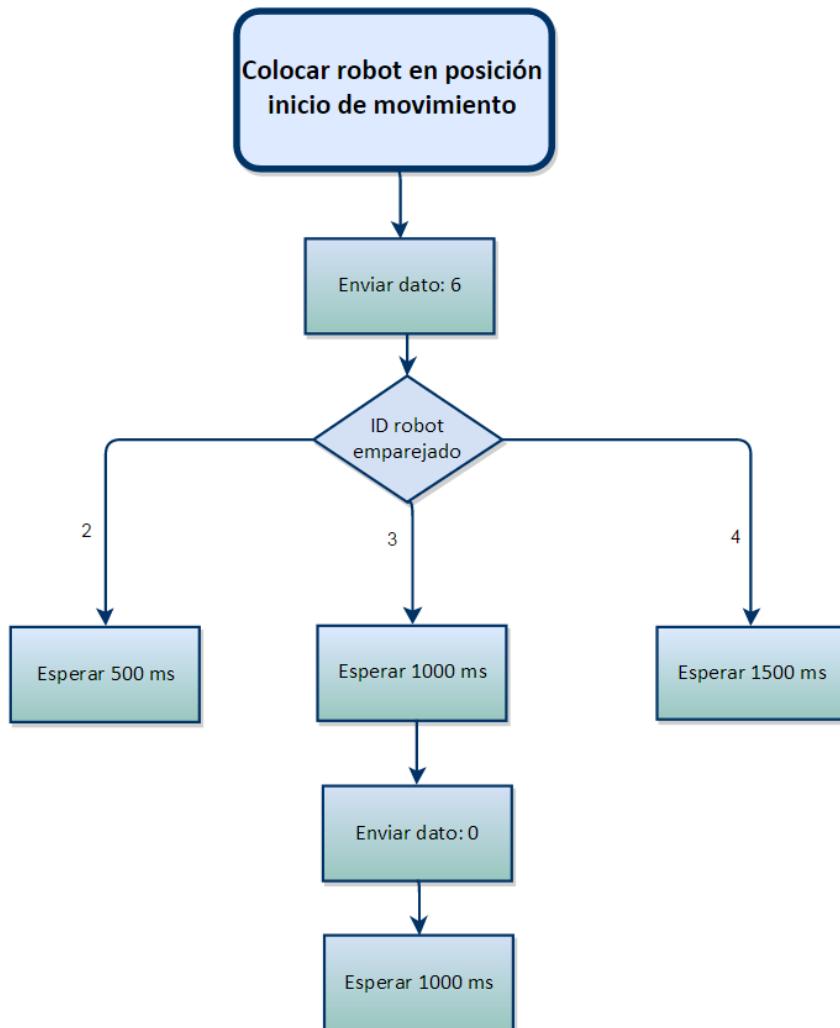


Diagrama 16. Colocar robot en posición inicio de movimiento.

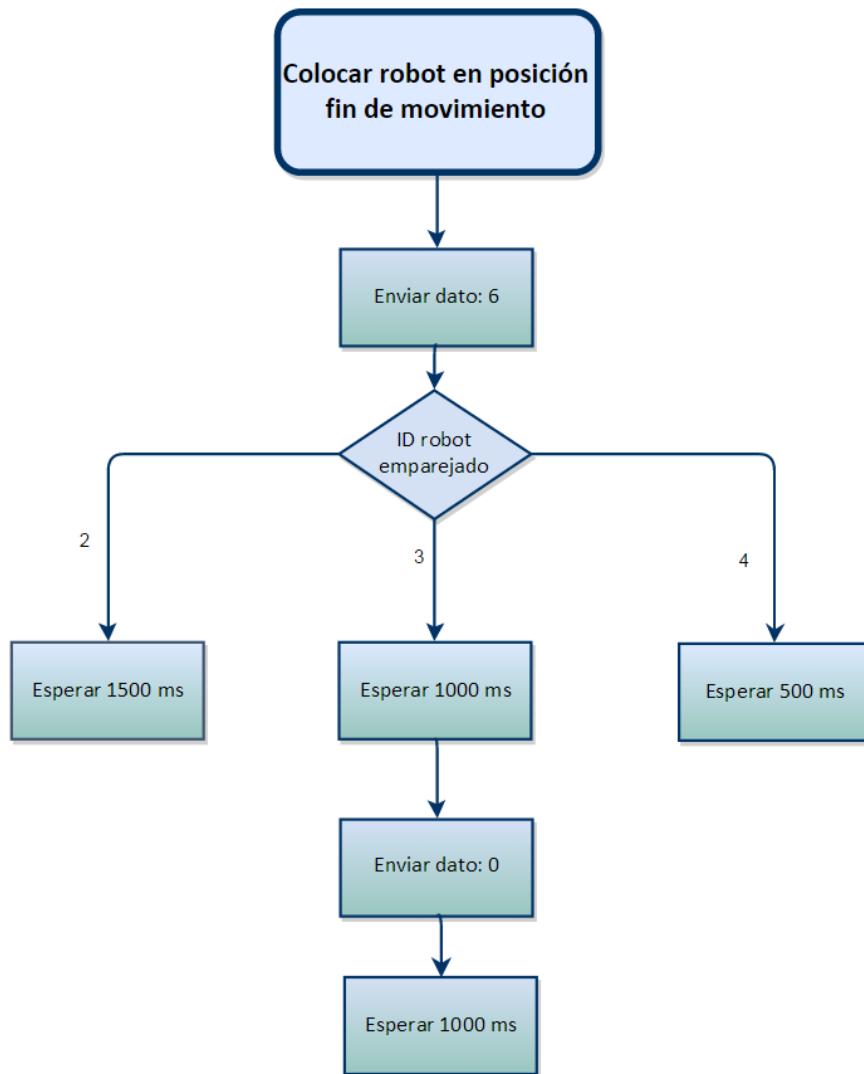


Diagrama 17. Colocar robot posición de fin de movimiento.

### 8.2.2.2 ENSAYOS

Haciendo uso de la aplicación creada para Smartphone Android se ha probado la comunicación Bluetooth punto a punto y las funciones complejas para evitar obstáculos y resolver el laberinto.

En primer lugar se ha comprobado el correcto funcionamiento de la aplicación Android creada, utilizándola en varios dispositivos móviles diferentes y realizando pruebas de búsqueda de dispositivos, conexión, y envío de datos, además de la funcionalidad de todos los elementos gráficos de interacción con el usuario.

En la Figura 46, se muestra la aplicación para Smartphone Android en uno de los dispositivos utilizados para las pruebas.

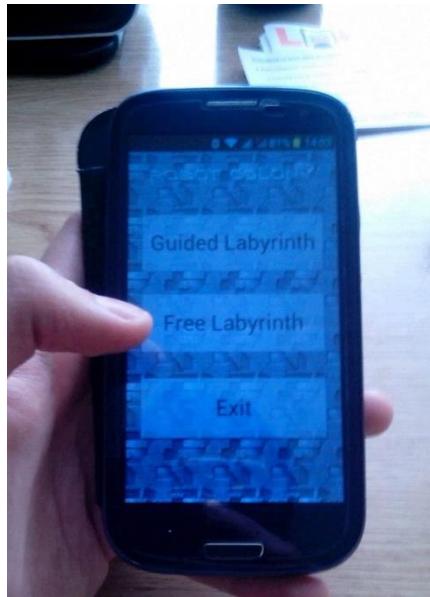


Figura 46. Smartphone con la aplicación Android creada.

Para realizar las pruebas se ha construido un pequeño laberinto como el que se muestra en la Figura 47, en el que el robot líder utilizando sus dos sensores de Ultrasonidos colocados en la parte frontal, tendrá que detectar los obstáculos y ser capaz de resolverlo.

El robot líder utiliza un microcontrolador en el que se ha cargado un programa que utiliza las funciones de la librería:

- Obtención de información de los sensores
- Movimiento
- Cambio de rol y de modo de funcionamiento del módulo Bluetooth
- Obtención de información por Bluetooth e interpretación de la misma
- Conexión con otros dispositivos Bluetooth a elección
- Envío de información a los demás dispositivos Bluetooth

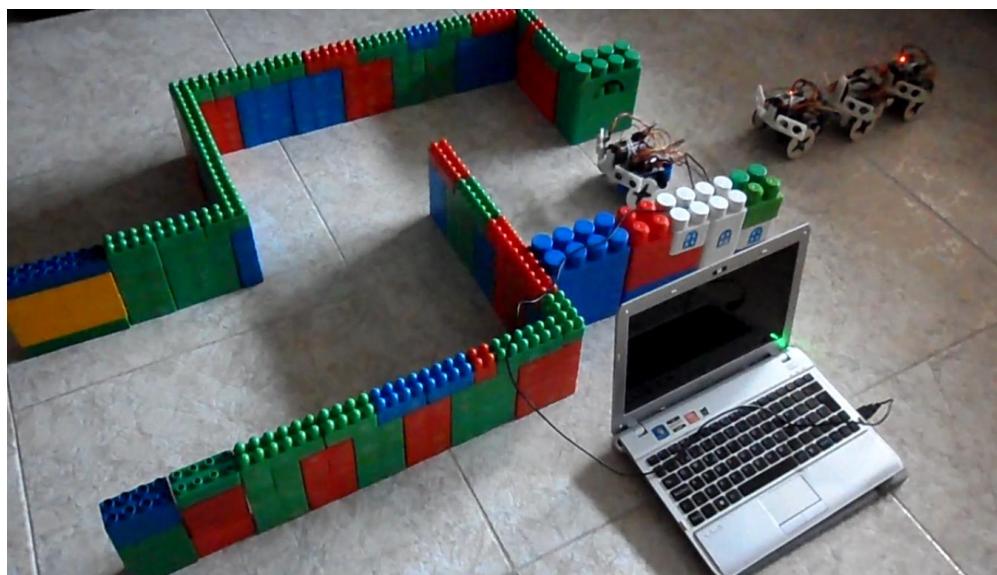


Figura 47. Escenario creado para pruebas.

### 8.2.3 RESULTADOS

Durante esta prueba se han obtenido los siguientes resultados, en cuanto al funcionamiento de los distintos componentes de los mini-robots:

- Los **servomotores** actúan de manera similar en todos los mini-robots, existiendo un pequeño error de variación de velocidad entre unos y otros en ocasiones notable durante la resolución del laberinto pero sin restar funcionalidad a la tarea.
- Los **sensores de Ultrasonidos** obtienen la información del entorno de manera suficientemente precisa para obstáculos que se encuentran inmediatamente delante de los mini-robots, pero hay varios ángulos muertos en la parte frontal de los mini-robots cuando aparecen en el entorno pequeños obstáculos. Por ello los laberintos creados han sido relativamente simples, de acuerdo a las limitaciones de los sensores de Ultrasonidos.

Para el caso concreto de los laberintos utilizados en las pruebas, los sensores de Ultrasonidos han sido capaces de obtener en la mayor parte de los casos una rápida y correcta interpretación de los datos, que han llevado al mini-robot líder a la resolución del laberinto sin ningún tipo de problema.

- La **comunicación** entre los distintos módulos lleva algunos segundos en realizarse por lo que se ha tenido en cuenta en el software utilizado en las pruebas para evitar fallos en la conexión.

En la resolución del laberinto por parte de los demás mini-robots según la información recibida se ha detectado un pequeño intervalo de tiempo entre la

recepción de datos y la realización de los movimientos. Además, el robot líder se ha basado en el número de ciclos de ejecución de su software para almacenar del tiempo en el que ha llevado a cabo cada movimiento, incrementando un contador de ciclos. Hay que tener en cuenta que este tiempo que enviará el robot líder no es exactamente el tiempo durante el que ha estado realizando cada movimiento. Teniendo en cuenta estos dos últimos aspectos existe un pequeño error acumulativo en la resolución del laberinto por parte de los mini-robots, que en la mayor parte de los casos no ha supuesto ningún problema en la realización del laberinto correctamente.

## 8.3 PRUEBAS ORIENTADAS A COMUNICACIÓN MULTIDIPOSITIVO

Como aplicación de las librerías creadas y para testear la velocidad de comunicación y reacción de los Bluetooth se ha creado un sistema de tele-operación controlado a través de una interfaz gráfica de Usuario para PC.

### 8.3.1 DESCRIPCIÓN DEL PROBLEMA

La realización de movimientos coordinados es una de las funciones más importantes de una colonia de robots.

Para aplicaciones en las que se necesita que todos los robots realicen los mismos movimientos de manera simultánea es interesante utilizar la tele-operación masiva.

De esta manera en vez de tener que dirigir a los robots de manera individual, se puede dirigir a todos al mismo tiempo para que realicen las tareas de manera paralela.

Un ejemplo de ello es la exploración de entornos desconocidos en los que los robots se pueden colocar por ejemplo de manera conjunta en el centro del entorno y el usuario es capaz de tele-operarlos a la vez y decidir el área del entorno que tienen que explorar. Así se evitan colisiones entre ellos y se elimina el tiempo de recepción de órdenes entre un robot y otro. En la Figura 48 se muestra un ejemplo de ello.

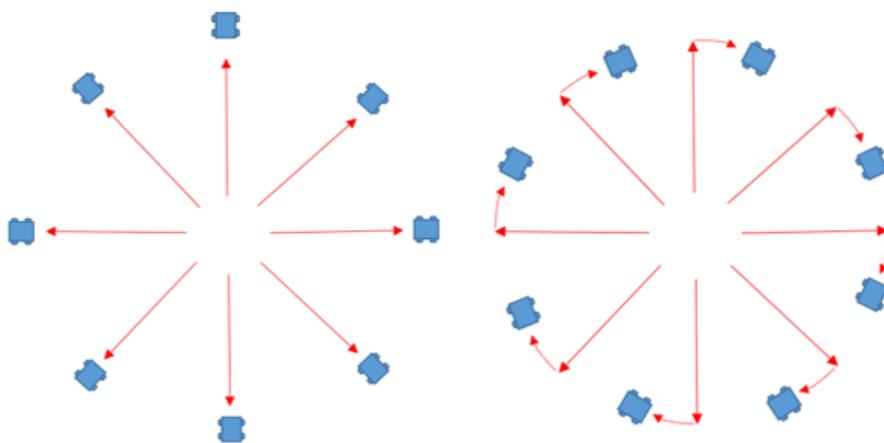


Figura 48. Tele-operación simultánea de los mini-robots.

En este proyecto se ha realizado una interfaz de usuario que es capaz de comunicarse con todos los robots y teledirigirlos simultáneamente. Asimismo puede elegir con cuales de los robots conectarse para tele-operar y con cuales no, de manera que no está condicionada a un número de robots determinado, sino que todo queda a elección del usuario.

### 8.3.2 DESARROLLO

Para la tele-operación se ha creado un programa en Arduino igual para todos los mini-robots, de manera que realicen movimientos de manera simultánea según las indicaciones del usuario a través de la interfaz gráfica para PC.

#### 8.3.2.1 COMPORTAMIENTO

En el programa en Arduino creado para la tele-operación de los robots se utilizará la comunicación Serie de la placa a la que el Bluetooth conectado enviará la información.

En el Diagrama 18 se muestra como al comienzo del programa se configurarán todos los robots; para ello se establecerán todos los módulos de Bluetooth con rol esclavo. A continuación los robots esperarán a recibir los datos para realizar un movimiento de forma simultánea.

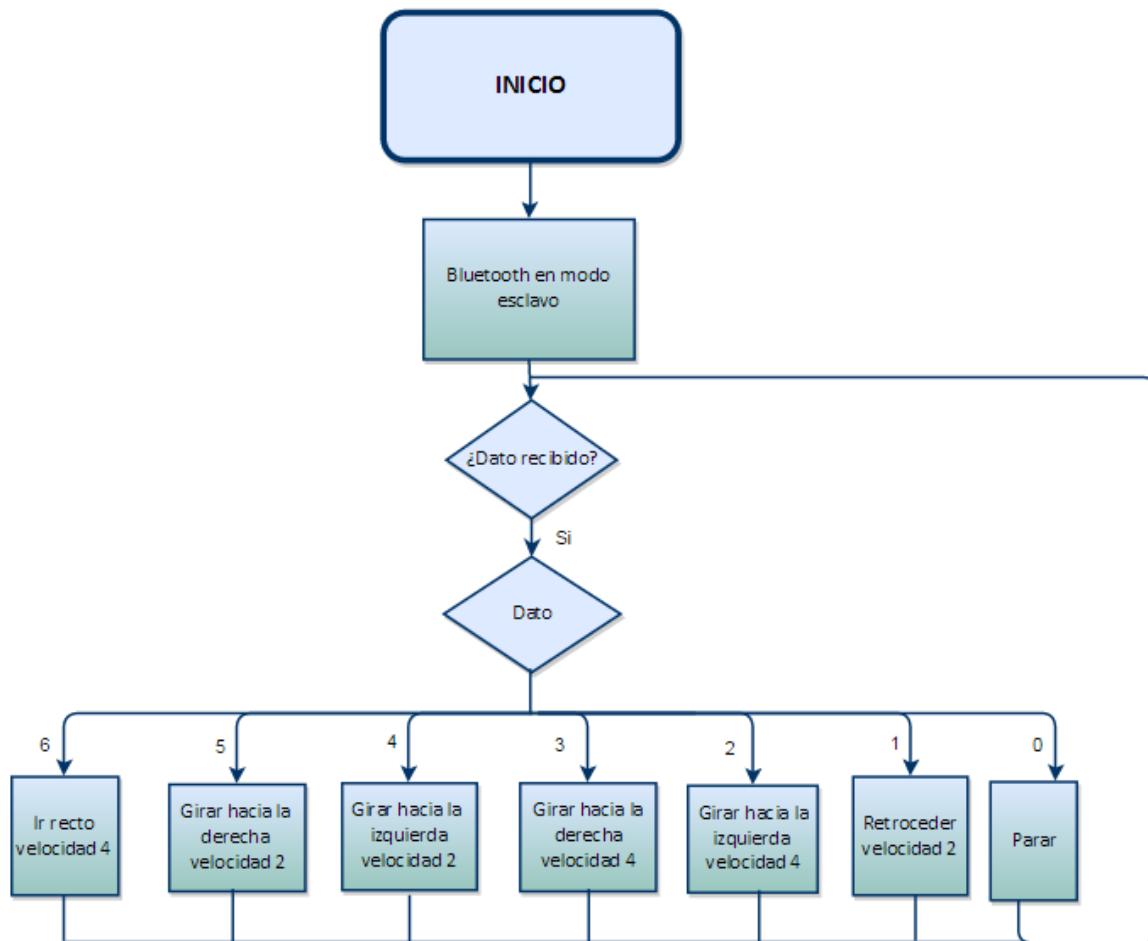


Diagrama 18. Tele-operación.

### 8.3.2.2 ENSAYOS

Gracias a los ensayos de tele-operación realizados se ha podido llevar un seguimiento de:

- Funcionamiento de los módulos y correcta recepción e interpretación de los datos
- Tiempo de espera entre la recepción de los datos y la realización del movimiento
- Errores de calibración en los servomotores
- Saltos en la continuidad del movimiento
- Tiempo de autonomía de las baterías

### 8.3.3 RESULTADOS

Como resultados obtenidos de los ensayos de tele-operación cabe destacar los siguientes aspectos:

- El tiempo de conexión con todos los dispositivos Bluetooth es relativamente corto, pero es recomendable esperar unos segundos antes de empezar la teleoperación debido a desconexiones puntuales.
- Aún actuando todos los servomotores a la máxima velocidad (cuyos valores son comunes sin necesidad de calibración previa), existe una variación de velocidad entre unos y otros debido a factores externos como la carga de las pilas de alimentación de los servomotores.
- Los movimientos son continuos, fluidos y sincronizados en todos los mini-robots, sin existir variación perceptible entre la recepción de datos y actuación de uno u otro.
- El tiempo de vida de las **baterías** no es demasiado largo, tal como se ha comprobado en los ensayos, pero sí suficiente para realizar las tareas encomendadas en las pruebas.

# 9 CONCLUSIONES Y FUTURAS MEJORAS

En este capítulo se realizará un repaso de los objetivos alcanzados durante este proyecto y la satisfacción con los resultados obtenidos así como futuras mejoras y líneas de investigación.

## 9.1 CONCLUSIONES

El objetivo principal de este proyecto, la creación de una librería para la coordinación de una colonia de mini-robots se ha alcanzado llegando a poderse crear programas complejos a través del manejo sencillo y estandarizado de los actuadores, sensores y comunicación Bluetooth desde el entorno Arduino.

Mediante la calibración de los actuadores para aportar precisión a los movimientos de los mini-robot, se ha conseguido que realicen movimientos de manera coordinada.

La estandarización de la librería para su uso con distinto número de sensores y actuadores, ha hecho que se puedan realizar programas reutilizables para distintos tipos de robot.

Con la interfaz gráfica para PC creada en QT Creator, se ha podido testear el correcto funcionamiento del sistema de comunicación implementado en la librería, y la realización de movimientos por parte de los mini-robots de manera simultánea y consecuente a las instrucciones del usuario. De la misma forma ha servido como ayuda para la calibración de los servomotores de rotación continua y de su unificación en todos los mini-robots.

La aplicación para teléfono Android ha servido como principal medio para el testeo de la comunicación Bluetooth punto a punto implementada en la librería, demostrando la realización de tareas por parte de los mini-robots de manera individual siguiendo las órdenes del líder de la colonia.

Cabe destacar que todos los elementos creados en este proyecto han sido desarrollados bajo la filosofía *Open Source*, por lo que podrán ser reutilizados y servir de apoyo e inspiración a gran cantidad de usuarios y estudiantes para continuar innovando y creando dentro del amplio abanico de posibilidades que ofrece la robótica.

## 9.2 FUTURAS MEJORAS

### **Diseño**

Como futuras mejoras, podrían realizarse modificaciones en la estructura de los mini-robots, adaptándose a las alimentaciones auxiliares utilizadas para los servomotores de rotación continua, de manera que quedasen más integradas en el diseño. Además podría modificarse la estructura de los mini-robot para poder añadir nuevos elementos electrónicos como encoders, sensores de infrarrojo, brújula, etc. ampliando de esta forma su campo de actuación.

### **Hardware**

Como futura mejora de hardware, se podría realizar un circuito integrado en una FPGA, en el que incluir componentes tales como LEDs, zumbadores, etc. para indicar al usuario lo que está sucediendo en el software, llevando así un mejor seguimiento del funcionamiento del programa realizado.

### **Software**

Para mejorar el software existente, se podría crear un fichero de configuración para la calibración de los Servomotores y para modificar las direcciones de los módulos Bluetooth a utilizar.

### **Líneas de estudio**

Por otra parte, como línea de estudio e investigación sería de gran utilidad utilizar Teoría de Control para permitir a los mini-robots actuar como un enjambre, reagrupándose y realizando movimientos ajustándose a un fin común como agrupación unificada. Además sería interesante la aplicación de Control Fuzzy o borroso al sistema multi-robot para reducir la incertidumbre e imprecisión en los datos obtenidos por los sensores, y aportar a los mini-robot un aprendizaje en tiempo real y la posibilidad de realizar gran variedad de tareas como mapeado o exploración en entornos desconocidos.

# BIBLIOGRAFÍA

- [1] "Asociación de Robótica UC3M," [En línea]. Enlace:  
<http://asrob.uc3m.es/index.php/RPC>. [Último acceso: 29 08 2014].
- [2] Georgia Tech, "Control of Mobile Robots," Coursera, [En línea]. Enlace:  
<https://class.coursera.org/conrob-002>. [Último acceso: 27 08 2014].
- [3] "Wiki - Reprap," [En línea]. Enlace:  
<http://reprap.org/wiki/RepRap/es>. [Último acceso: 20 08 2014].
- [4] "ASROB-Impresoras 3D," [En línea]. Enlace:  
[http://asrob.uc3m.es/index.php/Impresora-3D\\_Open\\_Source](http://asrob.uc3m.es/index.php/Impresora-3D_Open_Source). [Último acceso: 10 09 2014].
- [5] "Wikipedia - Thingiverse," [En línea]. Enlace:  
<http://en.wikipedia.org/wiki/Thingiverse>. [Último acceso: 20 08 2014].
- [6] D. C. a. K.S.Booth, "Coordinating open-source software development," *Enabling Technologies: Infrastructure for Collaborative Enterprises*, pp. 61-66, 199.
- [7] D. H. Pink, *Drive: The Surprising Trurh About What Motivates Us*, USA, 2011.
- [8] C. G. Saura, "Robots educativos libres, imprimibles y de bajo coste".
- [9] "Wiki Robotics," [En línea]. Enlace:  
<http://www.iearobotics.com/wiki/index.php?title=Archivo:Miniskybot-v1.0-red-r1.jpg>. [Último acceso: 25 08 2014].
- [10] "Thingiverse - Scout robot," [En línea]. Enlace:  
<http://www.thingiverse.com/thing:13042>. [Último acceso: 24 08 2014].

[11] "Thingiverse - Protobot," [En línea]. Enlace:

<http://www.thingiverse.com/thing:18264>. [Último acceso: 25 08 2014].

[12] R. C. Arkin, "Behaviour-Based Robotics (Intelligent Robotics and Autonomous Agents)," *The MIT press*, 1998.

[13] "Swarm robotics IDSIA," [En línea]. Enlace:

[http://www.idsia.ch/~gianni/SwarmRobotics/swarm\\_robotics.html](http://www.idsia.ch/~gianni/SwarmRobotics/swarm_robotics.html). [Último acceso: 26 08 2014].

[14] D. A. R. P. A. (DARPA), "Centibots," [En línea]. Enlace:

<http://www.ai.sri.com/centibots/>. [Último acceso: 26 08 2014].

[15] "Atualidad.rt," [En línea]. Enlace: <http://actualidad.rt.com/ciencias/view/119990-robots-termitas-construir-casas-planetas>. [Último acceso: 29 08 2014].

[16] "Distributed Multi-Robot Exploration and Mapping," *IEEE*, 2006.

[17] "Flight global," [En línea]. Enlace:

<http://www.flightglobal.com/blogs/aircraft-pictures/2010/08/six-uk-heron-uavs/>. [Último acceso: 29 08 2014].

[18] "IEE SPECTRUM," [En línea]. Enlace:

<http://spectrum.ieee.org/automaton/robotics/robotics-hardware/video-friday-3485624>. [Último acceso: 28 08 2014].

[19] "DARPA," [En línea]. Enlace:

[http://www.darpa.mil/Our\\_Work/TTO/Programs/Legged\\_Squad\\_Support\\_System\\_\(LS3\).aspx](http://www.darpa.mil/Our_Work/TTO/Programs/Legged_Squad_Support_System_(LS3).aspx). [Último acceso: 29 08 2014].

[20] "Blog.i-mas- Noticias," [En línea]. Enlace:

<http://blog.i-mas.com/751/sistema-logistico-robotizado-en-los-almacenes-de-amazon/>. [Último acceso: 28 08 2014].

[21] "Introducción a Arduino," [En línea]. Enlace:

- <http://arduino.cc/es/Guide/Introducción>. [Último acceso: 01 05 2014].
- [22] “Datasheet ATmega 328,” [En línea]. [Último acceso: 09 04 2014].
- [23] “Arduino Uno,” [En línea]. Enlace:  
<http://arduino.cc/en/Main/ArduinoBoardUno>. [Último acceso: 06 04 2014].
- [24] “Freaduino Uno producto elecfreaks,” [En línea]. Enlace:  
<http://www.elecfreaks.com/store/freaduino-uno-rev18-p-414.html>. [Último acceso: 09 04 2014].
- [25] “BricoGeek - Servomotor de rotación continua SM-S4303R,” [En línea]. Enlace:  
<http://tienda.bricogeek.com/motores/118-servomotor-de-rotacion-continua-sm-s4303r.html>. [Último acceso: 10 07 2014].
- [26] “Sensor de ultrasonido HC-SR04 producto elecfreaks,” [En línea]. Enlace:  
<http://www.elecfreaks.com/store/hcsr04-ultrasonic-sensor-distance-measuring-module-p-91.html>. [Último acceso: 09 04 2014].
- [27] “Datasheet Sensor de Ultrasonido HC-SR04,” [En línea]. Enlace:  
<http://elecfreaks.com/store/download/HC-SR04.pdf> Consultado 16/02/2014.  
[Último acceso: 01 05 2014].
- [28] “Módulos Bluetooth HC-05/HC-06 producto elecfreaks,” [En línea]. Enlace:  
<http://www.elecfreaks.com/store/bluetooth-modem-minimum-passthrough-module-bth07-p-229.html>. [Último acceso: 17 08 2014].
- [29] Makerbot, “Thingiverse - Pieza del chasis,” [En línea]. Enlace:  
<http://www.thingiverse.com/thing:18264>. [Último acceso: 20 08 2014].
- [30] Makerbot, “Thingiverse - Soporte para los sensores de los mini-robots,” [En línea].  
Enlace:  
<http://www.thingiverse.com/thing:19555>. [Último acceso: 20 08 2014].
- [31] “Android SDK,” [En línea]. Enlace:

- <http://developer.android.com/sdk/index.html>. [Último acceso: 18 08 2014].
- [32] “Wikipedia- QT Creator,” 17 08 2014. [En línea]. Enlace:  
[http://en.wikipedia.org/wiki/Qt\\_Creator](http://en.wikipedia.org/wiki/Qt_Creator).
- [33] [En línea]. Enlace:  
[http://3.bp.blogspot.com/-pNrF8ptUEUw/T23u58aI6gI/AAAAAAA98/J6fnwQhrawI/s1600/activity\\_lifecycle.png](http://3.bp.blogspot.com/-pNrF8ptUEUw/T23u58aI6gI/AAAAAAA98/J6fnwQhrawI/s1600/activity_lifecycle.png). [Último acceso: 24 08 2014].
- [34] “Arduino - Guia Windows,” [En línea]. Enlace:  
<http://arduino.cc/en/pmwiki.php?n=Guide/Windows>. [Último acceso: 2014 09 05].
- [35] “Arduino - Software,” [En línea]. Enlace:  
<http://arduino.cc/en/Main/Software>. [Último acceso: 05 09 2014].
- [36] “QT Creator,” [En línea]. Enlace:  
<https://qt-project.org>. [Último acceso: 10 09 2014].
- [37] S. H. a. M. V. M. W. Spong, “Robot Modeling and Control,” USA, 2005.
- [38] MakerBot, “Thingiverse- Ruedas para servos del robot,” [En línea]. Enlace:  
<http://www.thingiverse.com/thing:19940>. [Último acceso: 20 08 2014].
- [39] C. G. y. B.Arief, “The many meanings of open source..,” *Software, IEE*, vol. 21, pp. 34-40, 2014.
- [40] A. K. y. R. Sulaiman, “The Process of Quality Assurance under Open Source Software Development,” *IEE Symposium on Computers & Informatics*, 2011.
- [41] “MakerBot,” [En línea]. Enlace:  
<http://www.makerbot.com/>. [Último acceso: 27 08 2014].
- [42] “Mundiario,” [En línea]. Enlace:

<http://www.mundiario.com/articulo/economia/impresora-3d-desarrollada-padrón/20140106142735013786.html>. [Último acceso: 27 08 2014].

[43] “THL mobile store,” [En línea]. Enlace:

<http://www.thlmobilestore.com/thl-w100-android-4-2-mtk6589-quad-core-smart-phone-4-5-inch-ips-screen-5-0mp-front-camera-3g-gps-white.html>. [Último acceso: 01 09 2014].

[44] “Redacción TFG. UC3M,” [En línea]. Enlace:

[http://portal.uc3m.es/portal/page/portal/biblioteca/aprende\\_usar/TFG](http://portal.uc3m.es/portal/page/portal/biblioteca/aprende_usar/TFG). [Último acceso: 20 07 2014].

[45] “Tutorial SVN ASROB UC3M,” [En línea]. Enlace:

<http://asrob.uc3m.es/index.php/Tutorial SVN>. [Último acceso: 20 05 2014].

[46] “Wikipedia-Sensores de Ultrasonidos,” [En línea]. Enlace:

[http://es.wikipedia.org/wiki/Sensor\\_ultras%C3%B3nico](http://es.wikipedia.org/wiki/Sensor_ultras%C3%B3nico). [Último acceso: 23 05 2014].

[47] J. M. Carlos E. Agüeño, “Trabajando en equipo: Un repaso a los Robots móviles coordinados,” [En línea]. Enlace:

<http://gsyc.escet.urjc.es/~caguero/pubs/robocity-surveyCoord.pdf>. [Último acceso: 05 08 2014].

[48] C. G. Saura, “Robots educativos libres, imprimibles y de bajo coste,” [En línea]. Enlace:

[http://carlosgs.es/sites/default/files/2012\\_Robots\\_educativos\\_libres\\_imprimibles\\_y\\_de\\_bajo\\_coste.pdf](http://carlosgs.es/sites/default/files/2012_Robots_educativos_libres_imprimibles_y_de_bajo_coste.pdf). [Último acceso: 05 08 2014].

[49] J. González-Gómez, “PRINTBOTS: Robot libres e imprimibles,” [En línea]. Enlace:

[http://es.slideshare.net/obijuan\\_cube/printbots-robots-libres-e-imprimibles-madridbot2012](http://es.slideshare.net/obijuan_cube/printbots-robots-libres-e-imprimibles-madridbot2012). [Último acceso: 06 08 2014].

[50] Z. W. Li Wei, “Path Planning of UAVs Swarm Using Ant Colony System,” *Natural Computation*, pp. 288-292, 2009.

- [51] M. V. Douglas Vail, “Dynamic Multi-Robot coordination,” [En línea]. Enlace:  
<http://www.cs.cmu.edu/~mmv/papers/03NRL.pdf>. [Último acceso: 10 07 2014].
- [52] J. L. Crowley, “The State of the Art in Mobile Robotics,” [En línea]. Enlace:  
[http://www.iaarc.org/publications/fulltext/The\\_state\\_of\\_the\\_art\\_in\\_mobile\\_robotics.PDF](http://www.iaarc.org/publications/fulltext/The_state_of_the_art_in_mobile_robotics.PDF). [Último acceso: 06 07 2014].

# APÉNDICE A. MANUAL DE USUARIO

En este manual se explicará cómo utilizar la librería y las interfaces de usuario creadas en este proyecto además la de la instalación y desinstalación de los medios necesarios para su uso.

## A.1 INSTALACIÓN

Se indicarán los pasos a llevar a cabo en la instalación del entorno de Arduino, de la aplicación para Android y de QT Creator para utilizar la interfaz gráfica de usuario.

### A.1.1 ARDUINO IDE

Para cargar los programas en un microcontrolador del tipo Arduino es necesario disponer del entorno de desarrollo de Arduino.

En la página oficial de Arduino se indican todos los pasos a realizar en Windows para iniciarse con Arduino. [34]

Siguiendo el enlace de la página a las descargas del IDE se debe elegir la versión del entorno deseada, disponible para Windows, Linux o Mac OS X. [35]

Una vez descargado, se instala en el sistema operativo deseado.

Si se está utilizando Windows es necesario instalar los drives cuyos enlaces según la placa a utilizar están disponibles en la web de Arduino [34].

Una vez instalados el entorno y los drivers, la web del fabricante recomienda cargar uno de los programas de ejemplo para verificar el correcto funcionamiento del microcontrolador.

### A.1.2 APLICACIÓN ANDROID

En primer lugar se debe instalar el archivo “*RobotColony.apk*” en el Smartphone Android. Para ello se debe conectar el dispositivo móvil al ordenador mediante el puerto USB y guardar el archivo con extensión “*RobotColony.apk*” en el almacenamiento del teléfono.

A continuación se tiene que buscar el archivo en el Smartphone y ejecutarlo.

En la Figura 49 se muestra el proceso de instalación durante el cual, se debe conceder permiso a la aplicación para tener acceso al Bluetooth del dispositivo, pulsando “Instalar”.

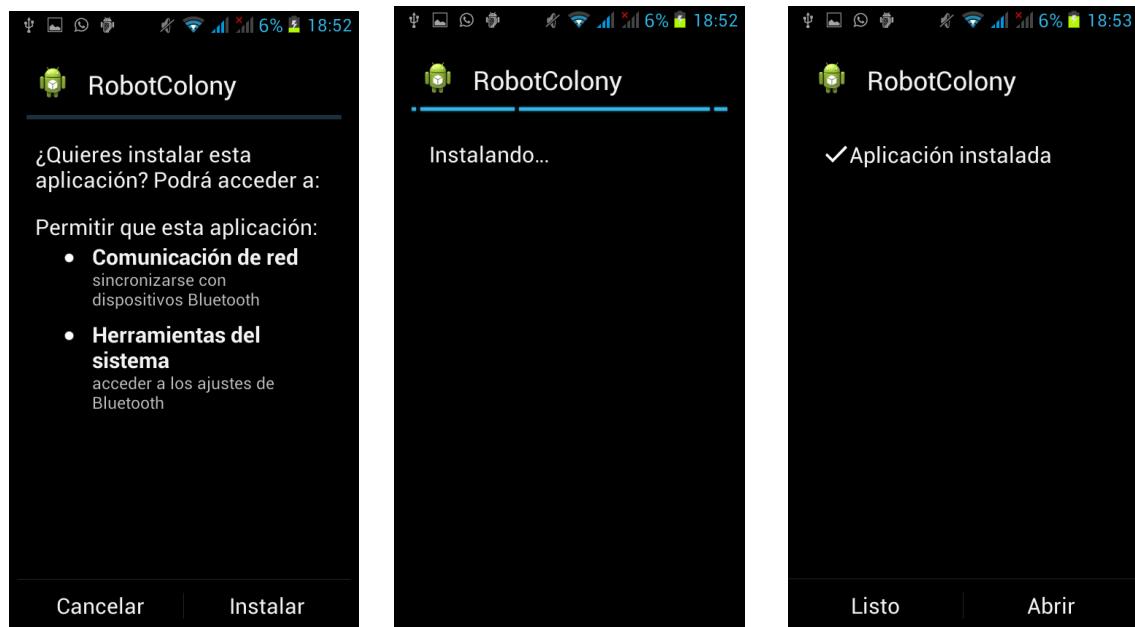


Figura 49. Proceso de instalación del .apk.

**NOTA:**

En la Figura 50 se muestra el caso particular en el que al ejecutar el archivo .apk en el terminal se indique que la aplicación ha sido bloqueada.



Figura 50. Bloqueo del .apk por origen desconocido.

En este caso tal como se muestra en la Figura 51, debemos ir al menú de configuración del dispositivo, en el apartado de seguridad y marcar la opción de “Orígenes desconocidos”.

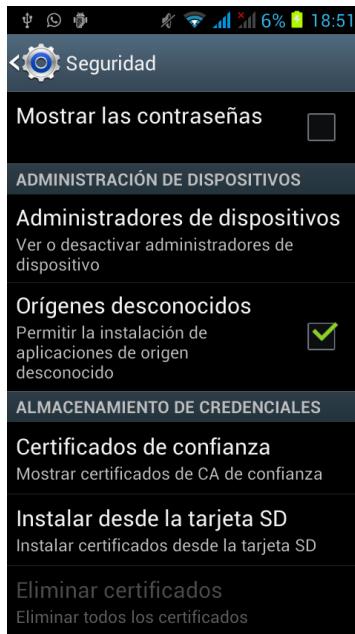


Figura 51. Permiso de aplicaciones de origen desconocido.

### A.1.3 QT CREATOR

Para poder hacer uso de la interfaz gráfica se debe instalar Qt Creator. Podemos descargar la versión gratuita en su página oficial [36], en el apartado “*Descargas/Download*”. Está disponible para los diferentes sistemas operativos, aunque el desarrollo de la interfaz ha sido en Windows utilizando la versión **Qt 5.3.2** de código abierto.

Una vez descargado, ejecutamos el fichero descargado y seguimos los pasos de instalación.

## A.2 DESINSTALACIÓN

En este apartado se indicarán los pasos a llevar a cabo para la desinstalación de los elementos utilizados; entorno de Arduino, aplicación de Android y QT Creator.

### A.2.1 ARDUINO IDE

La desinstalación de Arduino en Windows, dependerá de la manera elegida en el momento de la instalación.

- Si se eligió instalar mediante un instalador “.exe”, debemos seguir los siguientes pasos:

Accedemos a **Inicio>Panel de Control>Programas>Desinstalar un programa** y seleccionamos el programa Arduino como se muestra en la Figura 52

En esta lista buscaremos el programa Arduino y pulsaremos ‘Desinstalar’, tal y como se puede ver en la Figura 53.

#### Desinstalar o cambiar un programa

Para desinstalar un programa, selecciónelo en la lista y después haga clic en Desinstalar, Cambiar o Reparar.

Organizar	Desinstalar	Nombre	Desinstalar este programa.	Editor	Se instaló el	Tamaño	Versión
		Arduino		Arduino LLC	21/09/2014	251 MB	1.0.6

Figura 52. Desinstalar Arduino en Windows.

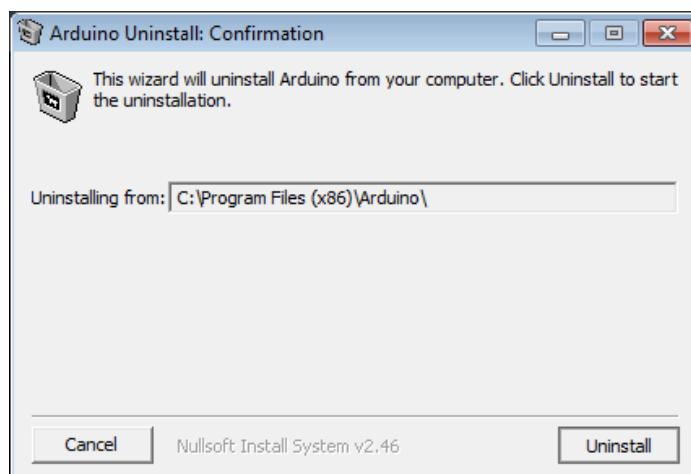


Figura 53. Menú de desinstalación.

- Si por el contrario se eligió la opción de descarga “.ZIP File” simplemente debemos borrar el directorio donde descomprimimos ese archivo descargado.

## A.2.2 APPLICACIÓN ANDROID

Para desinstalar la aplicación en el Smartphone, se deben seguir los siguientes pasos:

Ir a Ajustes o Configuración del dispositivo, y seleccionar la opción “Aplicaciones” como se muestra en la Figura 54.

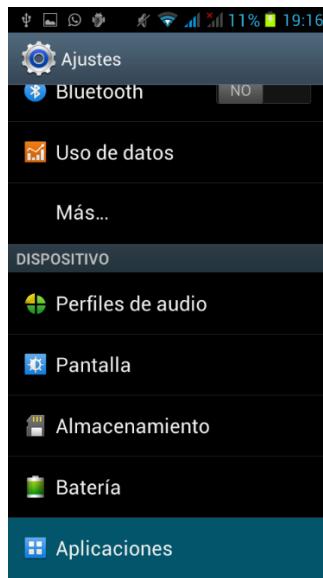


Figura 54. Opción de Aplicaciones en el menú ajustes o configuración.

En la siguiente pantalla buscaremos nuestra aplicación, pulsaremos sobre ella y elegiremos la opción de ‘Desinstalar’. El proceso se muestra en la Figura 55.

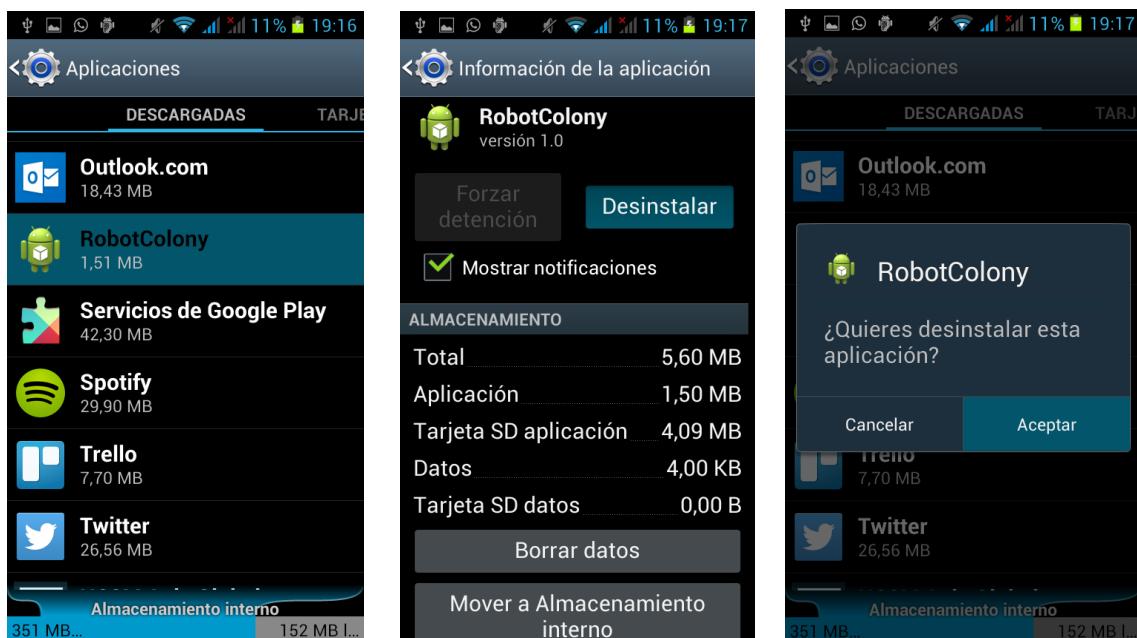


Figura 55. Proceso de desinstalación de la aplicación.

### A.2.3 QT CREATOR

Para desinstalar el programa Qt Creator, se debe seguir el mismo proceso que para el programa Arduino si se instalado mediante intalador “.exe”.

Debemos ir a **Inicio>Panel de Control>Programas>Desinstalar un programa**.

En esta lista buscaremos el programa Qt y pulsaremos ‘Desinstalar’, tal y como se puede ver en la Figura 56.

#### Desinstalar o cambiar un programa

Para desinstalar un programa, selecciónelo en la lista y después haga clic en Desinstalar, Cambiar o Reparar.

Nombre	Desinstalar este programa.	Editor	Se instaló el	Tamaño	Versión
Qt	Digia Plc		31/08/2014		1.0.1

Figura 56. Desinstalar QT Creator en Windows.

A continuación como se muestra en la Figura 57, se deben seguir los pasos de desinstalación seleccionándose la opción “*Remove all components*”.

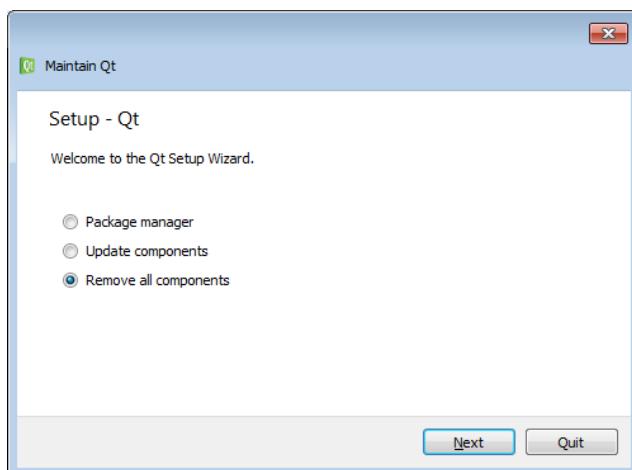


Figura 57. Menú de desinstalación.

## A.3 MODO DE EJECUCIÓN

En este apartado se explicará cómo utilizar las herramientas instaladas con los programas y la librería creados en este proyecto.

### A.3.1 LIBRERÍA

Para utilizar la librería creada en este proyecto es necesario crear una carpeta “*sketchbook*” con una subcarpeta “*libraries*” donde se encuentre la misma. La ruta de las carpetas debe ser de la forma **C:\<ruta>\sketchbook\libraries\<librería a incluir>**.

Además se pueden incluir los programas creados para la resolución de un laberinto (a utilizar junto con las interfaces de usuario) incluyendo la carpeta “*programs*” de la forma **C:\<ruta>\sketchbook\programs\<programas a incluir>**.

Para que el entorno reconozca la librería a utilizar es necesario indicar la localización de la carpeta *sketchbook*, tal como se muestra en la Figura 58, en el menú **File>Preferences**.

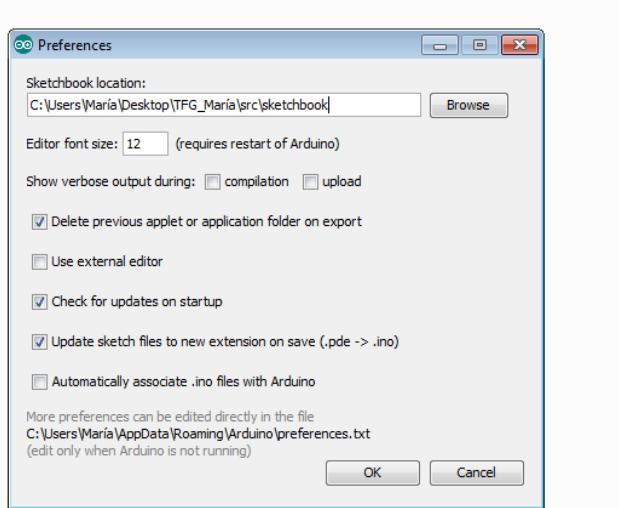


Figura 58. Localización del sketchbook en Arduino.

En la Figura 59 se muestra como importar las librerías del *sketchbook* al entorno de Arduino. Se deberá añadir *HCSR04Ultrasonic*, *Servo*, y la librería creada en el proyecto; *coordination*.

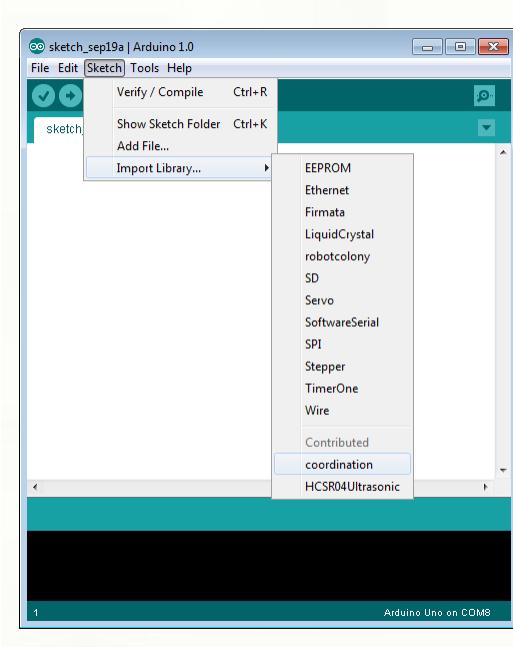


Figura 59. Importación de librerías.

A continuación ya se pueden crear programas usando la librería *coordination*.

### A.3.1.1 EJEMPLO DE USO

En este apartado se van a indicar los pasos necesarios para la creación de un programa en Arduino utilizando la librería *coordination*.

A continuación se muestran los *include* necesarios que deben aparecer en nuestro programa, cuya importación puede consultarse en la Figura 59.

```
#include <Servo.h>
#include <Ultrasonic.h>
#include <Bluetooth.h>
#include <Robot.h>
#include <Servomotor.h>
#include <UltrasonicSensor.h>
```

Una vez importados correctamente los *include* se debe crear un objeto del tipo Robot, llamando al constructor de la clase cuyos parámetros son los siguientes:

```
Robot(int robot_ID,int smr,int sml,int echor=5,int echol=7,int triggerr=6,int triggerl=8, int key=3, int power=14);
```

- **robot\_ID**: Número identificativo del robot. Utilizado en funciones específicas como en el movimiento y la comunicación.
- **Smr**: Pin digital en el que está cableado el pin de señal del servomotor derecho.
- **Sml**: Pin digital en el que está cableado el pin de señal del servomotor izquierdo.
- **Echor**: Pin digital en el que está cableado el pin de señal de Echo del sensor de Ultrasonidos derecho.
- **Echol**: Pin digital en el que está cableado el pin de señal de Echo del sensor de Ultrasonidos izquierdo.
- **triggerr**: Pin digital en el que está cableado el pin de señal de Trigger del sensor de Ultrasonidos derecho.
- **triggerl**: Pin digital en el que está cableado el pin de señal de Trigger del sensor de Ultrasonidos izquierdo.
- **key**: Pin digital en el que está cableado el pin de KEY del módulo Bluetooth.
- **power**: Pin digital en el que está cableado el pin de Vcc del módulo Bluetooth.

Los parámetros de número de identificación del robot y del número de pin de los servomotores derecho e izquierdo son obligatorios. El resto de parámetros que hacen referencia a los sensores de Ultrasonidos y al módulo Bluetooth no son obligatorios, y solo deberán indicarse cuando se disponga de estos elementos.

Por ejemplo, si se dispone de un robot con únicamente dos servomotores ubicados en los pines 9 y 10 se utilizará el constructor de la siguiente forma:

```
Robot my_bot(1,10,9);
```

En cambio si el robot dispone de sensores de Ultrasonidos y opcionalmente de Bluetooth, se puede llamar al constructor en consecuencia:

```
Robot my_bot(1,10,9,X,X,X,X);  
Robot my_bot(1,10,9,X,X,X,X,X,X);
```

Para este ejemplo se va a crear un objeto de tipo Robot con todos los componentes quedando el programa en Arduino de la siguiente forma:

programa\_ejemplo.ino

```
#include <Servo.h>  
#include <Ultrasonic.h>  
#include <Bluetooth.h>  
#include <Robot.h>  
#include <Servomotor.h>  
#include <UltrasonicSensor.h>  
  
Robot my_bot(1,10,9,5,7,6,8,3,4);
```

Si se va a utilizar alguna función que utilice el puerto serie, como la comunicación Bluetooth en la librería se debe indicar en *void setup()* la inicialización del puerto serie:

```
void setup()  
{  
    Serial.begin(38400);  
}
```

El programa a desarrollar utilizando las librerías se realizará en el bucle principal *void loop()*.

Las funciones que se pueden desarrollar con la librería son las siguientes:

### **Movimiento**

Para que el robot realice el movimiento deseado, basta con llamar a la función específica indicando la velocidad por parámetro de 1 a 4 siendo 4 la máxima velocidad.

Las funciones para los movimientos son las siguientes:

- Avanzar: *move\_forward()*

- Retroceder: *move\_back()*
- Girar a la derecha: *turn\_right()*
- Girar a la izquierda: *turn\_left()*

Un ejemplo de uso dentro del programa de ejemplo realizado en el que se desea que el robot avance a velocidad 3 sería de la siguiente forma:

```
void loop()
{
    my_bot.move_forward(3);
}
```

### **Obtención de distancia de los obstáculos al robot**

Para la obtención de la distancia a la que se encuentran los obstáculos que se encuentra el robot, se debe utilizar la función *get\_distance()* en cada uno de los sensores de Ultrasonidos. Esta función devuelve un valor decimal con coma flotante que debe almacenarse en una variable de tipo *float*. La utilización sería de la siguiente forma:

```
float right_distance;
float left_distance;
void setup()
{
}
void loop()
{
    right_distance= my_bot.us_r.get_distance();
    left_distance= my_bot.us_l.get_distance();
}
```

### **Comunicación**

Para utilizar las funciones de comunicación se debe utilizar de un módulo Maestro-Eslavo. Las funciones de comunicaciones de las que dispone la librería son las siguientes:

- Emparejado y conexión: *pair()*

Se indicará por parámetro el número de identificación del robot con el que se quiere conectar nuestro robot. Esta función devuelve un booleano que será true una vez se haya realizado la conexión. La forma de implementación si queremos que nuestro robot se empareje y conecte por ejemplo con el robot 2 es de la siguiente forma:

```
bool paired=0;
void setup()
{
```

```
}

void loop()
{
    if(paired==false)
    {
        paired=my_bot.bt.pair(2);
    }
}
```

- Envío de datos: *send\_data()*

Una vez que se ha realizado la conexión con el módulo Bluetooth del robot requerido se debe indicar por parámetro en la llamada a esta función el dato de tipo carácter a enviar:

```
char d;
void setup()
{
}
void loop()
{
    d='1';
    my_bot.bt.send_data(d);
}
```

- Modo de recepción de datos: *hear()*

Para que el Bluetooth de nuestro robot actúe con rol esclavo y pueda recibir datos se debe utilizar la función de escucha de la siguiente forma:

```
void setup()
{
    my_bot.hear();
}
```

Para ver el funcionamiento de esta librería se pueden utilizar los programas de ejemplo creados para testear su funcionamiento tal como se muestra en la Figura 60.

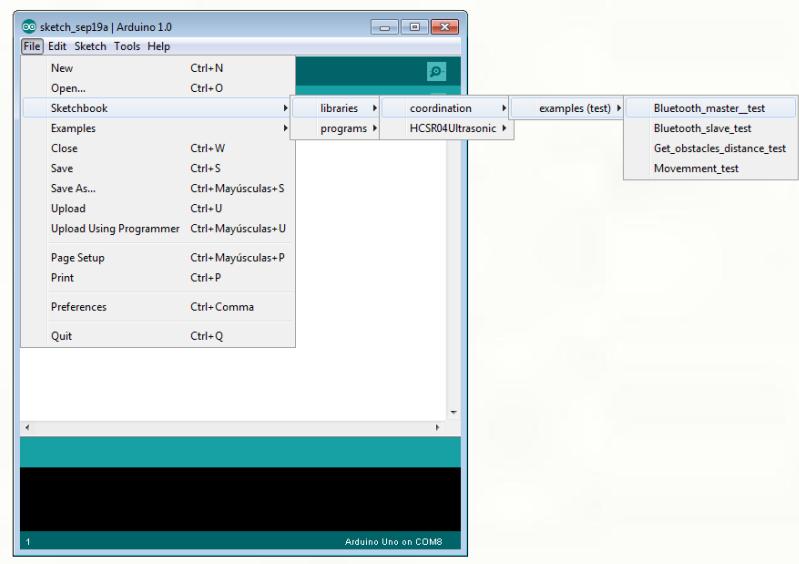


Figura 60. Utilizar programa de ejemplo para el uso de la librería.

De la misma forma se pueden cargar los programas a utilizar con las interfaces desde el menú **File>Sketchbook>programs**.

Para utilizar la aplicación de PC de tele-operación de todos los robots de forma simultánea se cargará el programa *slave.ino* en cada una de las placas.

Si se quiere utilizar la aplicación de Smartphone para la resolución del laberinto se debe cargar el programa *labyrinth\_master.ino* al robot líder (el que disponga de sensores de Ultrasonidos y módulo Bluetooth Maestro-Esclavo), y *slave.ino* al resto de robots.

#### NOTA:

- Hay que tener en cuenta la calibración de los servomotores que puede adaptarse modificando los valores numéricos de las instrucciones "...my\_servo.write(<valor numérico>);" utilizadas en las funciones *move\_forward*, *move\_back*, etc. en el archivo "*Robot.cpp*".
- Es necesario modificar las direcciones de los dispositivos Bluetooth con los que se quiere emparejar el dispositivo Maestro en la función *pair* disponible en el archivo "*Bluetooth.cpp*" tal como se muestra en la Figura 61 .

```
Serial.println("AT+PAIR=[2013,12,129042],40");
delay(4000);
Serial.println("AT+LINK=[2013,12,129042]");
delay(3000);
```

Figura 61. Modificar dirección Bluetooth a emparejar.

## A.3.2 INTERFAZ GRÁFICA DE USUARIO PARA PC

Para utilizar la aplicación de interfaz de usuario para PC, en primer lugar hay que emparejar los dispositivos Bluetooth con el ordenador, para que éstos tengan asignado un puerto COM en el PC.

### A.3.2.1 CONEXIÓN DE LOS DISPOSITIVOS BLUETOOTH

Es necesario activar el Bluetooth del PC a utilizar. A continuación se realizará una búsqueda de dispositivos Bluetooth en el entorno y se emparejarán utilizando un código de emparejamiento que suele ser “1234”.

En la Figura 62 se muestra como visualizar el puerto COM asignado a cada módulo Bluetooth en Windows desde **Panel de control>Hardware y sonido>Dispositivos e impresoras**.

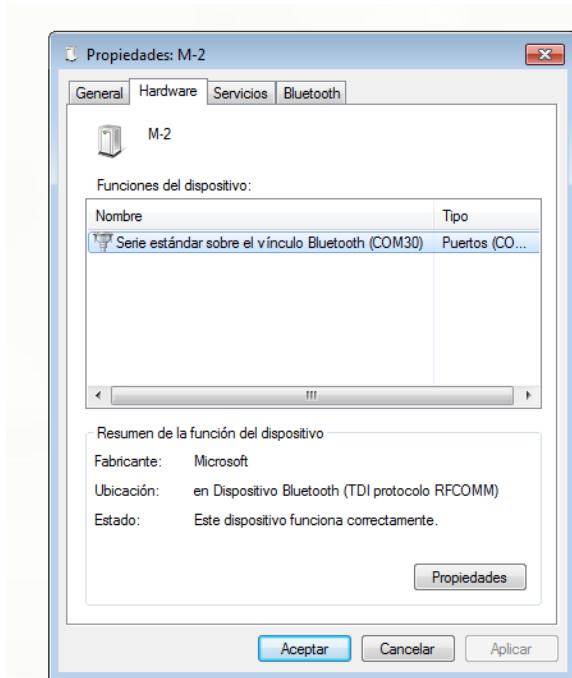


Figura 62. Propiedades del dispositivo Bluetooth.

### A.3.2.2 TELEOPERACIÓN

Después de la instalación, al inicio del programa se debe utilizar el menú **File>Open File or Project** y seleccionar el directorio de la aplicación abriendo el archivo “RobotColony.pro”. Una vez hecho esto para poder ejecutar el programa y visualizar la interfaz gráfica se debe pulsar el botón verde que se muestra en la Figura 63.



Figura 63. Iniciar compilación.

En consecuencia aparecerá la interfaz gráfica de usuario que se muestra en la Figura 40.

A continuación se debe realizar la conexión con todos los mini-robots, indicando en los cuadros de texto el puerto COM, en formato COMx, donde x será el número del COM correspondiente a cada mini-robot, tal como se muestra en la Figura 42 y pulsar el botón “*Connect*”.

Ahora ya se pueden tele-operar los robots mediante los botones de dirección que muestra la interfaz, o los comandos de teclado indicados en la Tabla 6.

### A.3.3 APPLICACIÓN PARA SMARTPHONE

Una vez instalada la aplicación, si se quiere probar la resolución del laberinto por parte de los robots se deben colocar uno detrás de otro según el orden en el que vayan a resolver el laberinto con el líder en la primera posición. A continuación se elige el modo de resolución del laberinto, libre “*Free labyrinth*” o guiada “*Guided labyrinth*”.

#### ***Resolución libre***

En primer lugar se debe conectar el dispositivo móvil con el robot líder, para eso se selecciona en la lista de dispositivos Bluetooth encontrados el módulo en cuestión, y por pantalla se mostrará un mensaje indicando que la conexión se ha realizado correctamente. El proceso se puede ver en Figura 64. Una vez conectados con el robot líder se le da la orden de empezar a resolver el laberinto, utilizando el botón “*Start the labyrinth*”. Cuando el robot ha salido del laberinto se utiliza el botón “*Finish the labyrinth*”, con lo que el robot se colocará en su posición final y parará. A continuación de manera automática, el robot líder enviará la información a los demás robots que resolverán el laberinto uno a uno.

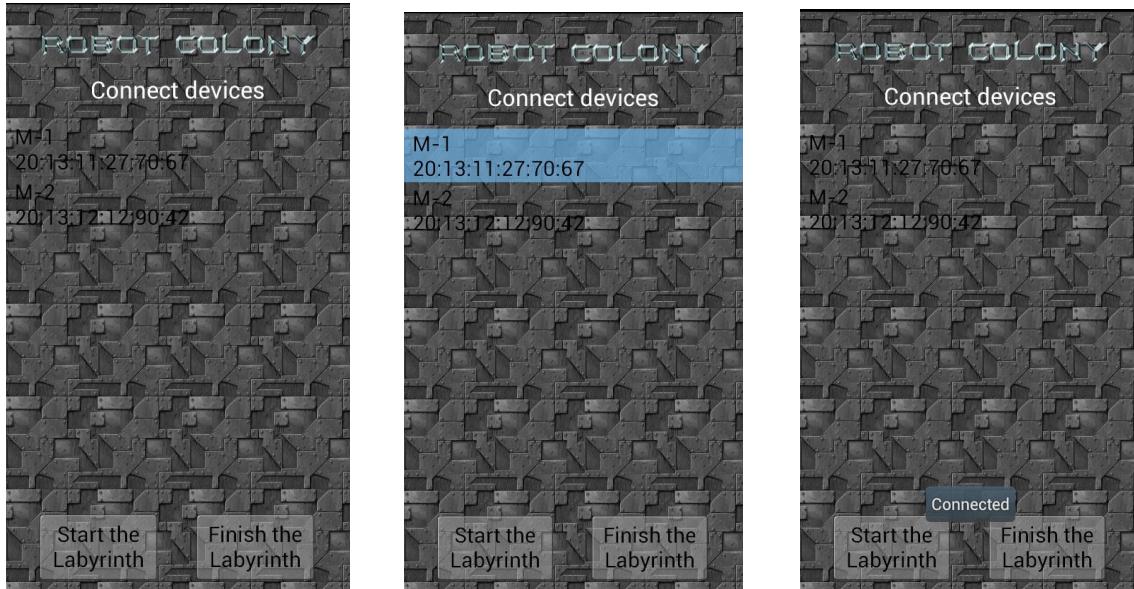


Figura 64. Conexión con el dispositivo Bluetooth.

### **Resolución guiada**

La conexión con el dispositivo se realiza de la misma forma que en la *Resolución libre*. Primero se mostrara la lista de los dispositivos encontrados como se puede ver en la Figura 65, y una vez listado el dispositivo buscado, el Robot líder (M-1), se debe pulsar en él para que se realice la conexión. El proceso de selección y conexión se puede ver en la Figura 66 y Figura 67.

Una vez que el Smartphone está conectado con el robot líder, el usuario puede teleoperarlo libremente utilizando las flechas de dirección y el botón "Stop" una vez se haya finalizado el laberinto. A continuación, el robot líder enviará la información a los demás robots uno a uno de manera que estos replicarán sus movimientos. Este modo es aplicable para la realización de cualquier tipo de trayectorias (sin necesidad de que sea un laberinto), de manera que permite que sean grabadas en el robot líder y repetidas por cada uno de los robots a continuación.

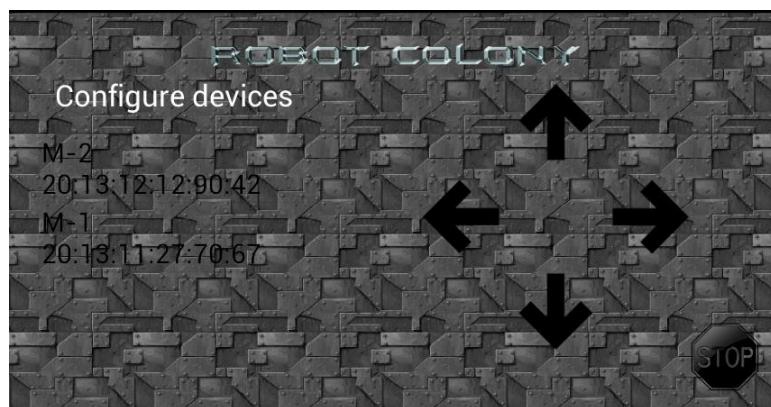


Figura 65. Búsqueda de dispositivos en el modo guiado.

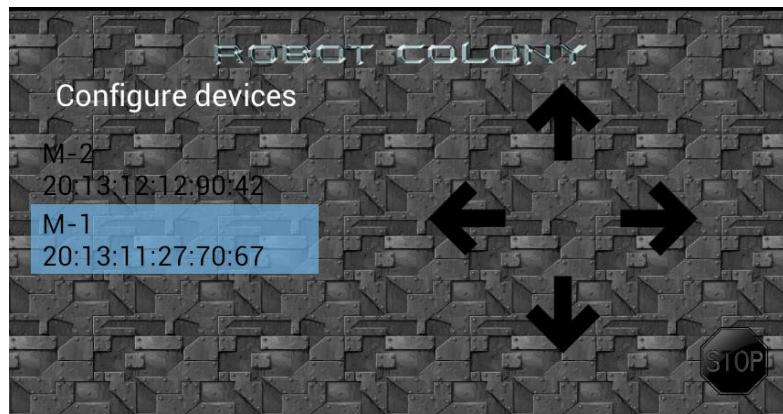


Figura 66. Selección de dispositivo a conectar en el modo guiado.

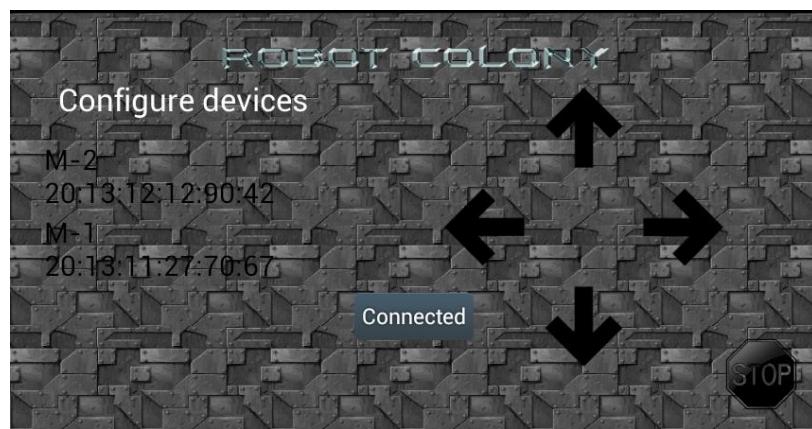


Figura 67. Conectado con dispositivo.