Trabajo de Fin de Grado

Sistema de control remoto para realizar labores agrícolas

Remote control system for agricultural work

Néstor Albelo Jorge

La Laguna, 17 de enero de 2016

D. **Manuel Jesús Rodríguez Valido**, con N.I.F. 52840833-N profesor Titular de Universidad adscrito al Departamento de Ingeniería industrial de la Universidad de La Laguna, como tutor

C E R T I F I C A (N)

Que la presente memoria titulada:

"Sistema de control remoto para realizar labores agrícolas."

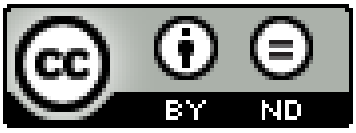
ha sido realizada bajo su dirección por D. **Néstor Albelo Jorge**, con N.I.F. 78646307-S.

Y para que así conste, en cumplimiento de la legislación vigente y a los efectos oportunos firman la presente en La Laguna a 17 de enero de 2016.

Agradecimientos

Agradecer a mi tutor Manuel Jesús el apoyo dado en todo el trayecto de este proyecto, a David Hernández por la ayuda prestada tanto dentro como fuera del laboratorio y por supuesto a la gente de Finca Las Lucanas.

Licencia



Resumen

El proyecto que se ha realizado como TFG, se ha basado en uno previamente iniciado por antiguos alumnos de diversas facultades y que consistía en la creación de un sistema de carga totalmente eléctrico, conocido como PACA. Este sistema se desarrolló junto con una empresa privada llamada *Finca Las Lucanas* y ha ido mejorando a lo largo del tiempo gracias al paso de los diferentes alumnos y profesores que han trabajado en él [1].

La importancia del proyecto PACA viene de poder facilitar las labores agrícolas de los empleados de la planta donde se encuentre el carro, a la hora de transportar carga de manera sencilla en lugares por donde otra maquinaria, debido a sus dimensiones, no pueda o le sea complicado llegar.

En la parte que corresponde al alumno, el principal objetivo del trabajo realizado ha sido, el de mejorar el carro añadiéndole nuevas funcionalidades a nivel tanto de hardware como de software.

Algunas de las principales tareas realizadas han sido la estructuración general del código, implementación del control remoto haciendo uso de un Smartphone, mejoras de la función Joystick añadiéndole un control suavizado, reinserción del mando PS3 como control, creación de una aplicación para el sistema Android, nueva placa de reducción de voltajes, configuración del módulo Wifi entre otras.

Palabras clave: Carro de carga, PACA, control remoto, bluetooth, Wifi, PS3, Smartphone

Abstract

This project has been designed as EOG work, and it is based on a previously version initiated by students of different faculties and the idea was the creation of a charging system known as PACA. This system was developed with the help of a private company called *Finca Las Lucanas*, and has been improving over time thanks to the students and teachers who have worked on it.

The project PACA has its importance in the able to faciltate agricultural work of employees where it is, helping them to transport heavy cargo easily in places where other machinery, due to its size, could not or would be difficult to reach.

The main target of the work developed by the student, was to improve the system adding to it new features at both hardware and software level.

Some of the main tasks performed were the general structure of the code, implementation of remote control using a Smartphone, improvements joystick function by adding a smoothing control reintegration of the PS3 controller as a control, creating an application for the Android system, Wifi module configuration among others.

Keywords: *Carriage, PACA, remote control, bluetooth, Wifi, PS3, Smartphone*

Índice General

Índice de figuras

Índice de tablas

Capítulo 1.
Introducción de PACA

El principal objetivo del trabajo de fin de grado ha sido el de diseñar un sistema de control, tanto a nivel hardware como software, de un prototipo de robot agrícola, PACA, para así poder controlarlo de forma remota y facilitar labores agrícolas como el transporte de materiales o mercancías.

Este proyecto tiene una gran importancia ya que permite agilizar y facilitar el trabajo de los operarios, aumentando así la productividad de la empresa en la que se encuentre el dispositivo. Por otro lado, los trabajos desarrollados aquí permiten seguir avanzando en el desarrollo de nuevas tecnologías para la mejora de tareas agrícolas.

El origen de este proyecto, tal y como ya se ha comentado, viene del convenio entre la empresa *Finca Las Lucanas*, que es donde se encuentra actualmente el prototipo, y la propia Universidad de La Laguna, con el objetivo principal de crear una manera fácil y económica de transportar elementos por la finca, a la vez que sirve de experiencia académica para los alumnos que han ido pasando por el proyecto a lo largo del transcurso del mismo.

1.1 Qué es PACA

El carro PACA (Figura 1) es un sistema de carga totalmente eléctrico, controlado tanto a distancia como de forma manual, formado principalmente, por un chasis, el cual soporta todo el sistema, un cuadro de control y encendido del carro, en donde se encuentran el joystick de control manual y el selector de estado, dos motores eléctricos situados en la parte delantera junto a las baterías de alimentación, y un cajetín en donde se encuentra la circuitería principal de alimentación y procesado de datos. Que resaltar que PACA posee un sistema de Tracción formado por dos motores eléctricos capaz de transportar 200Kg y con una autonomía de 8 horas, una jornada laboral.

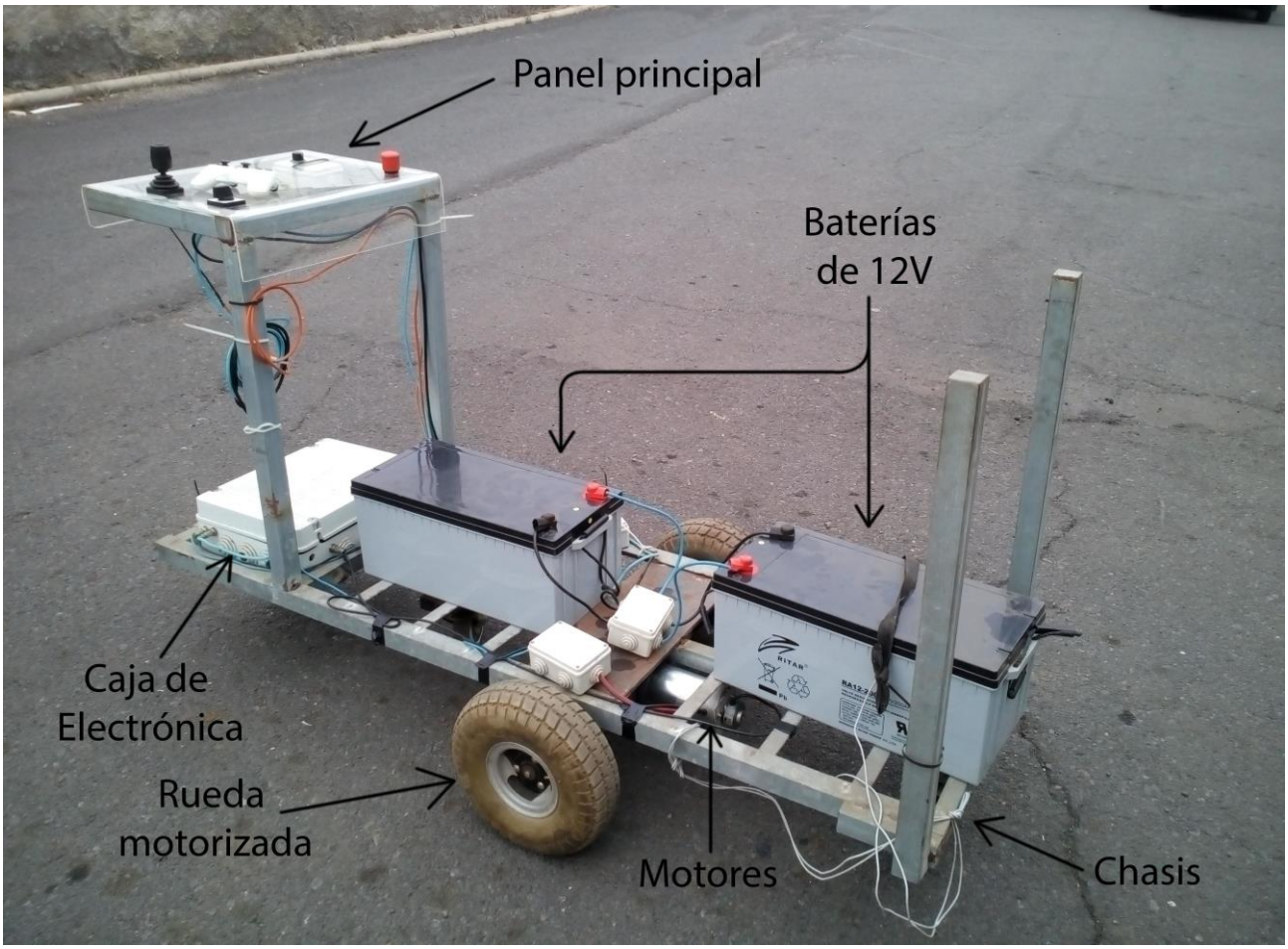


Figura 1 – Componentes principales del carro PACA

1.2 Para qué sirve

El principal propósito del carro PACA, es el de poder facilitar el transporte de mercancías a lo largo de la finca por zonas en donde otros vehículos no pueden acceder con tanta facilidad. Esto permite que el personal no sufra tanto estrés físico al tener que cargar con materiales pesados por toda la finca. Ya que cuenta con un sistema de control remoto, también podría servir para acercar ciertos elementos a otros empleados simplemente depositándolos en el carro y controlándolo remotamente hasta que llegue a su destino sin la necesidad de moverse de sitio, agilizando así la producción de la finca al no perder tanto tiempo en el transporte.

1.3 Dónde

Actualmente PACA se encuentra situado en la empresa *Finca Las Lucanas*, la cual se dedica, principalmente, a la producción de plantas verdes, flores, plantas aromáticas y hortalizas. Situada en Valle Guerra, cuenta con una alta tecnología tanto de riego, como de abonado y fertilización, para asegurar la calidad de sus productos, así como su durabilidad, a la vez que respetan el medio ambiente.

1.4 Estructura de la memoria de TFG

Ahora pasaremos a explicar cómo estará organizada principalmente la memoria del TFG. En primer lugar, comenzaremos con una pequeña descripción del carro, hablando de su funcionamiento y de la interfaz que posee para ser manejado. Seguidamente pasaremos a la parte más extensa de la memoria, en donde se desglosarán en secciones, las diferentes partes que componen al sistema, diferenciándolas según su naturaleza (electro-mecánicas, electrónicas o de software), y también se explicarán las tareas realizadas por el alumno a lo largo del transcurso del trabajo. Una vez concluida esta sección, pasaremos a las conclusiones generales del alumno sobre el proyecto realizado, así como futuras implementaciones que se podrían hacer al carro para mejorarlo. Por último, y para concluir la memoria, se representará un presupuesto aproximado del coste del sistema, separando las diferentes partes que lo componen, en secciones para mayor claridad.

Capítulo 2.
Descripción y Funcionamiento de PACA

A lo largo de este capítulo pasaremos a presentar tanto el funcionamiento como una descripción del carro PACA para conocer cómo funciona y que las partes que lo componen.

2.1 Quién lo controla

Debido a que el carro PACA no es un sistema autónomo, implica que para que pueda funcionar correctamente se requiere de un operario que esté atento en todo momento, bien a su lado, o a distancia, del recorrido que va realizando, para que, si fuese necesario, pueda detenerlo en caso de fallo.

2.2 Cómo funciona

El sistema PACA cuenta con un selector de estados en su panel principal y que nos permite indicar que sistema de control se va usar. A la hora de manejar el carro, existen dos opciones principales, mediante las cuales, el operario maneja el carro. Estas son:

- Control Manual: En este modo, el operario en cuestión, debe estar situado junto al carro, el cual posee un joystick manual, situado en el panel principal junto al selector. Aunque este modo se encuentra desde el comienzo del proyecto, su único propósito actual es la de tener una alternativa de reserva por si algo fallase en los sistemas remotos.
- Control Remoto: En este modo, por el contrario, no es necesario que el operario se encuentre pegado al carro, facilitando así el desplazamiento del mismo en largas distancias. Este tipo de control se divide a su vez en dos modos:
    - Modo PS3: Con este tipo de control, permitimos hacer uso de un mando de PS3, para dirigir el carro, haciendo uso de los gatillos y joysticks que incorpora. Para poder hacer uso del mando se debe estar en un radio de aproximadamente 20 metros como máximo, ya que la conexión con el carro se establece mediante Bluetooth. Este modo permite un control de manera muy natural debido al diseño de los mandos, lo cual facilita mucho el manejo del carro.

- Modo Móvil: Por otra parte, el modo móvil consiste en una aplicación disponible para el sistema Android, la cual contiene un Joystick virtual táctil, que permite enviarle ordenes al carro. Este modo permite una mayor distancia con el carro, ya que la conexión se establece mediante un dongle Wifi, lo cual aumenta la distancia teórica de conexión hasta los 50 metros. Por otra parte, esta aplicación también incorpora un sistema de monitorización(interface visual de usuario), el cual indica el estado en todo momento en el que se encuentra el sistema PACA actualmente y una lista de posibles logs a modo de depuración. Esto se debe, a que una vez montado todo el sistema, no existe ninguna retroalimentación hacia el usuario, por lo que, si sucediese algún fallo durante la ejecución del programa, no habría ninguna manera de conocer el posible error. Este tipo de control se hace bastante cómodo ya que al poder ser instalada en un móvil no hay que añadir ningún dispositivo de control aparatoso al usuario, su control es bastante orientativo y fácil de usar, permitiendo un control bastante preciso del carro.

2.3 Interfaz de usuario

Como ya se ha explicado, PACA cuenta con un total de tres estados, más uno de desconexión. Estos cuatro estados se controlan haciendo uso del selector situado en el panel frontal del carro (Figura 2). Cada estado se activa únicamente cuando el selector se encuentra en él, por lo que no podremos controlar el carro con el mando PS3 si no tenemos dicho modo seleccionado. Esto está diseñado como sistema de seguridad y control del carro, facilitando así la gestión que debe hacer el micro integrado.

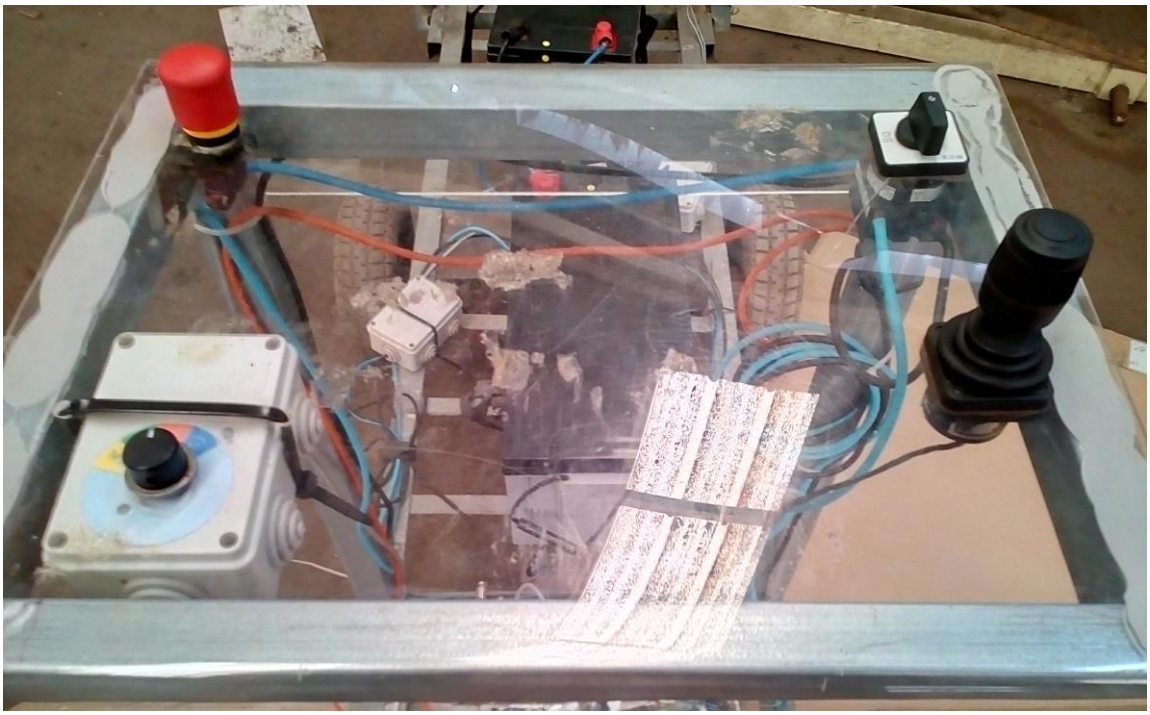


Figura 2 - Panel de Control de PACA

Modo Joystick Manual

A la hora de manejar el carro en este modo disponemos de un joystick situado en el panel principal del carro, el cual consta tal como se puede ver en la figura 3, de un botón de seguridad para evitar que algún posible atasco en el mecanismo del mismo, dejase el carro en funcionamiento sin ningún operario.

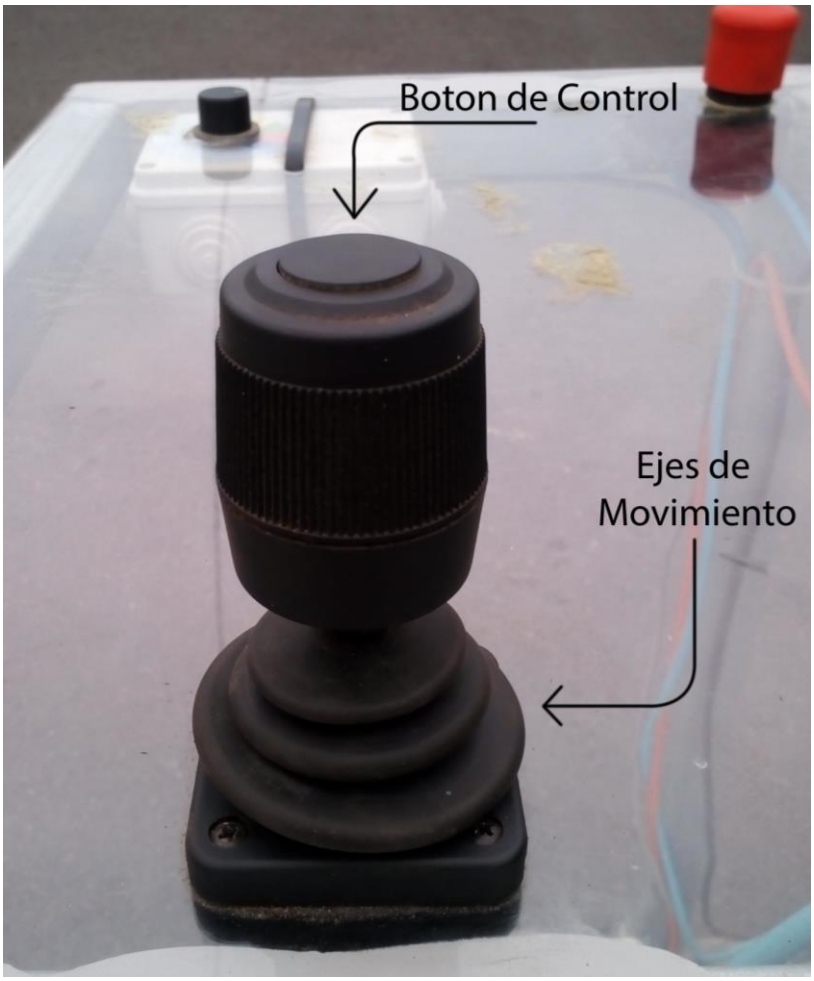


Figura 3 - Joystick manual de PACA

Para poner en marcha el carro simplemente debemos mantener pulsado el botón de control situado en la parte superior del joystick y después dirigir el mismo hacia donde se desee ir.

Modo PS3

En este modo disponemos de un mando de la consola PS3 que servirá como control haciendo uso de los gatillos y joysticks que incorpora (Figura 4). Este sistema también dispone de un sistema de seguridad similar al usado en el Joystick manual pero esta vez debemos mantener pulsado un gatillo del mando para activar el sistema.

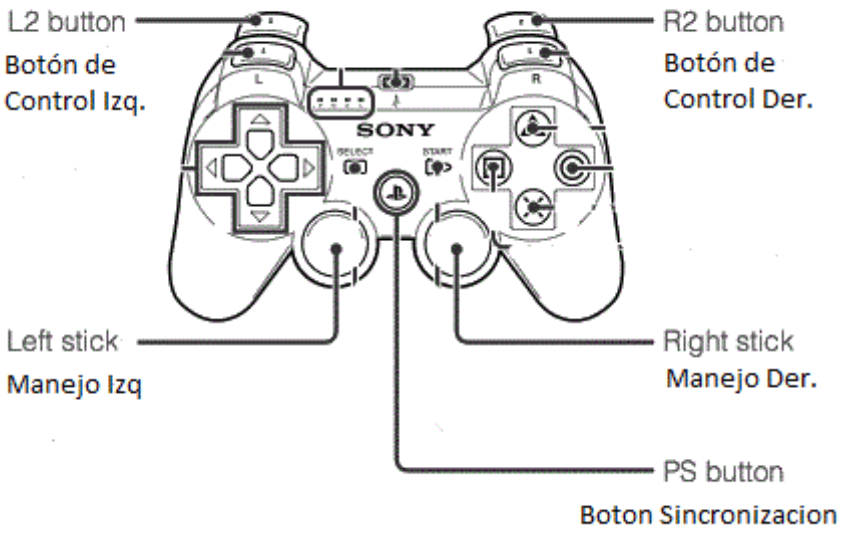


Figura 4 - Mando de control PS3

El control se puede llevar a cabo de dos maneras tal y como se puede ver en la figura 2.3.3. Debido a que existen dos joysticks en el mando, se decidió permitir el uso de ambos y dejar que el usuario decidiese cual prefería usar. Para manejar el carro solo debemos mantener pulsado un gatillo y usar el joystick del mismo lado para dirigirlo, algo muy similar al manejo con el joystick manual. Este sistema permite manejar el carro de manera cómoda con una sola mano al tener el gatillo de seguridad y el joystick en el mismo lado del mando, facilitando así el control del carro en caso de tener una mano ocupada.

Modo Smartphone

El último modo consta de una simple aplicación desarrollada para el sistema Android y que contiene un joystick virtual para el manejo del carro (Figura 5). Este sistema no posee un botón de control real como los otros modos, sino que, al detectar la pulsación del usuario en la pantalla del Smartphone, el control queda activado, permitiendo al usuario desplazar el dedo para manejar el sistema.

Figura 5 - Control virtual desde dispositivo móvil

Para manejar el carro basta con pulsar en cualquier parte de la pantalla de la aplicación, para que aparezca una coordenada de referencia desde la cual nos moveremos para acelerar el carro. Este modo dispone a su vez de un indicador de la coordenada con respecto a la de referencia, lo cual nos indica de forma numérica que dirección va tomando el carro.

Control de estado

Por último, y aunque no se trata de un estado de control del carro, tenemos la propia interfaz de la aplicación móvil, aun cuando el sistema no se encuentra en el estado Móvil (Figura 6).

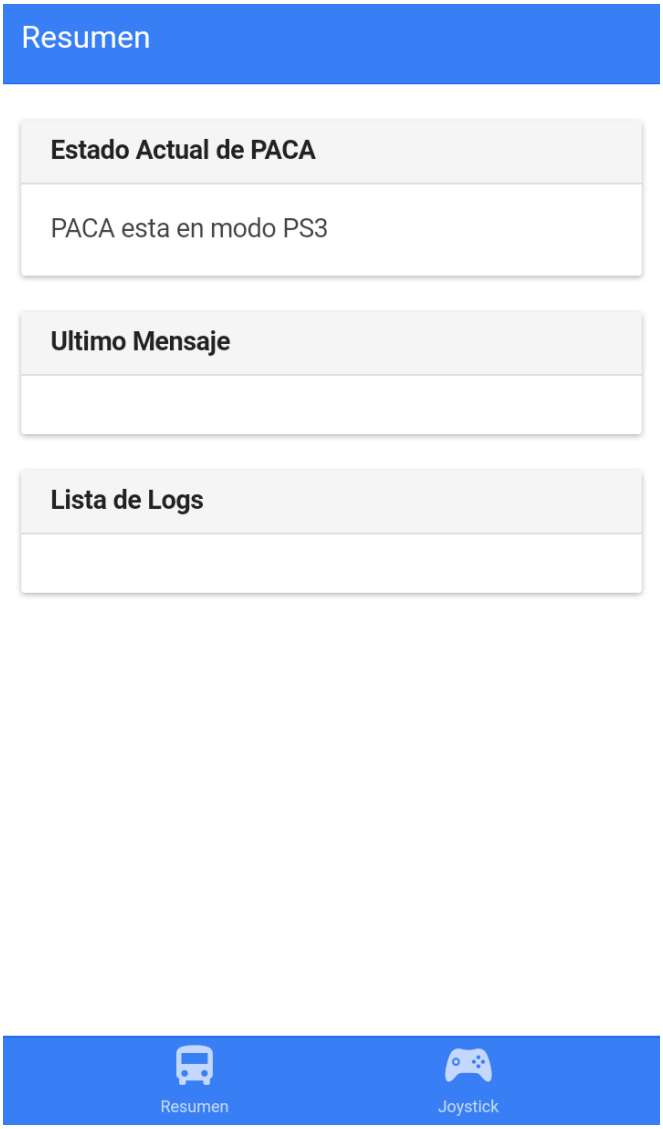


Figura 6 - Estado de control de PACA

En esta interfaz es completamente pasiva, y únicamente podemos observar en qué modo se encuentra el carro y si ha habido algún tipo de error durante la ejecución del programa principal. Esta interfaz se mantiene siempre activa, independientemente del estado en el que coloquemos el selector, y se actualiza en tiempo real para poder actuar en caso de fallo o simplemente como sistema de control.

Capítulo 3.
Sistemas de control hardware/software

A lo largo de este capítulo, se irán exponiendo las diferentes partes que componen el sistema PACA, desglosándolos según su categoría, y se explicarán las tareas realizadas a lo largo del trabajo, tanto a nivel hardware como software.

3.1 Componentes Electro-mecánicos

En lo que a la parte mecánica del aparato se refiere, podemos diferenciar tres partes principales:

- **Chasis:** Tal y como se observa en la figura 7, el chasis está compuesto principalmente por una estructura de metal, y su principal función es la de dar forma al carro y proporcionar un lugar donde pueda descansar todo el peso del sistema e interconectar los elementos que lo componen entre sí.

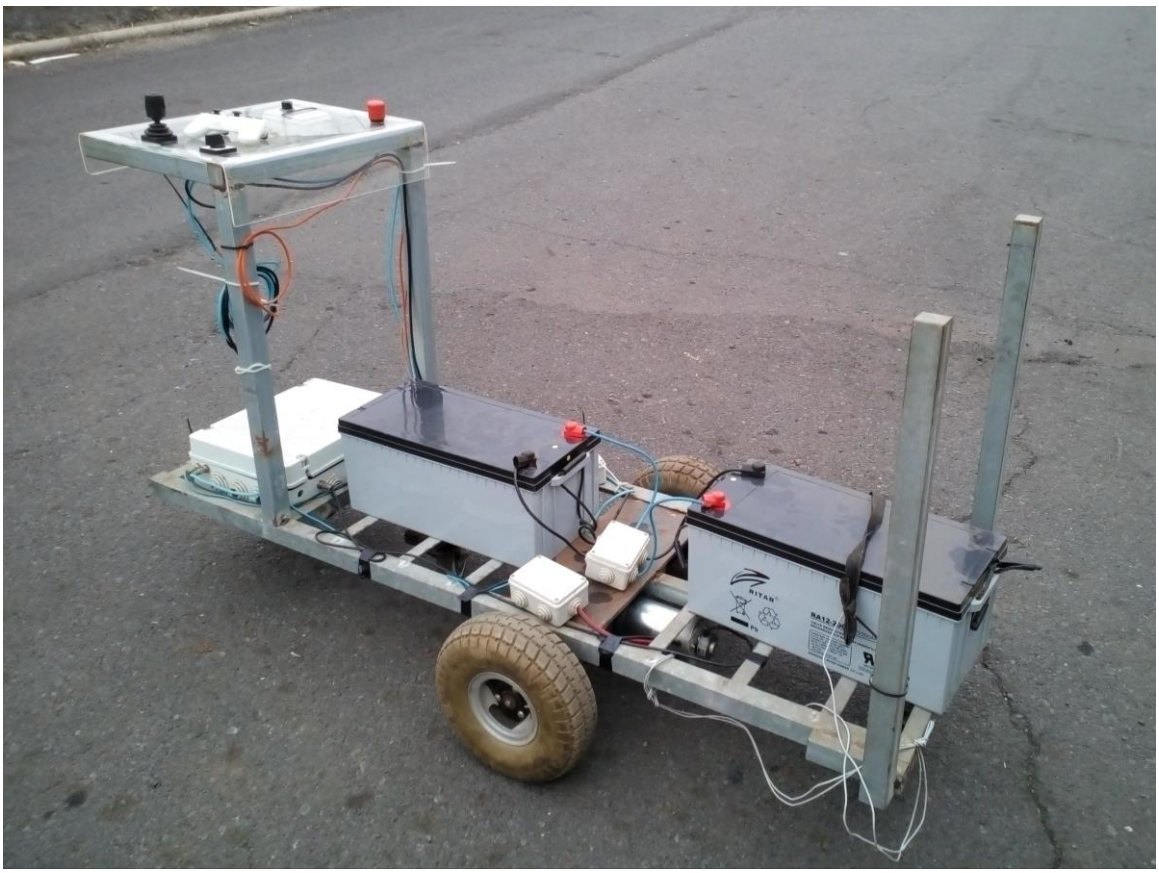


Figura 7 – Estructura principal del sistema PACA

- **Ruedas:** A lo largo del sistema, existen un total de cuatro ruedas, dos de las cuales se encuentran conectadas a los motores (Figura 8), y otras dos situadas en la parte delantera y trasera del cuadro. Estas dos últimas son ruedas locas y su principal función es la de proporcionar estabilidad al carro, mientras las motorizadas se encargan de aplicar empuje.

Figura 8 – Rueda Motorizada

- **Motores:** Para poder mover el carro poseemos un total de dos motores situados bajo el chasis y en donde se conectan las dos ruedas principales. Estos motores tienen una potencia de 500W cada uno a plena potencia, lo que permite desplazar el carro con mercancía sin problemas.

3.2 Componentes Electrónicos

Puesto que el carro se alimenta de un sistema eléctrico alimentado por baterías, la mayor parte del mismo es de tipo electrónico. Los principales componentes electrónicos que lo componen son:

- **Baterías:** En la parte delantera del carro existe un total de dos baterías dispuestas en serie, de 12V cada una (Figura 9), las cuales se encargan de alimentar todo el sistema. Estas baterías son de gel y tienen una capacidad de carga para alimentar el sistema completo a plena carga durante ocho horas seguidas, lo que las hacen perfectas para una jornada completa de trabajo.

Figura 9 - Batería de 12V

- **Placa reductora de voltaje:** En la caja de electrónica del carro contamos con una placa (Figura 10), que dispone de tres reguladores de potencia que convierten los 24V de las baterías, en 3-5-12V, ya que ciertos elementos de la circuitería del carro requieren estos voltajes.

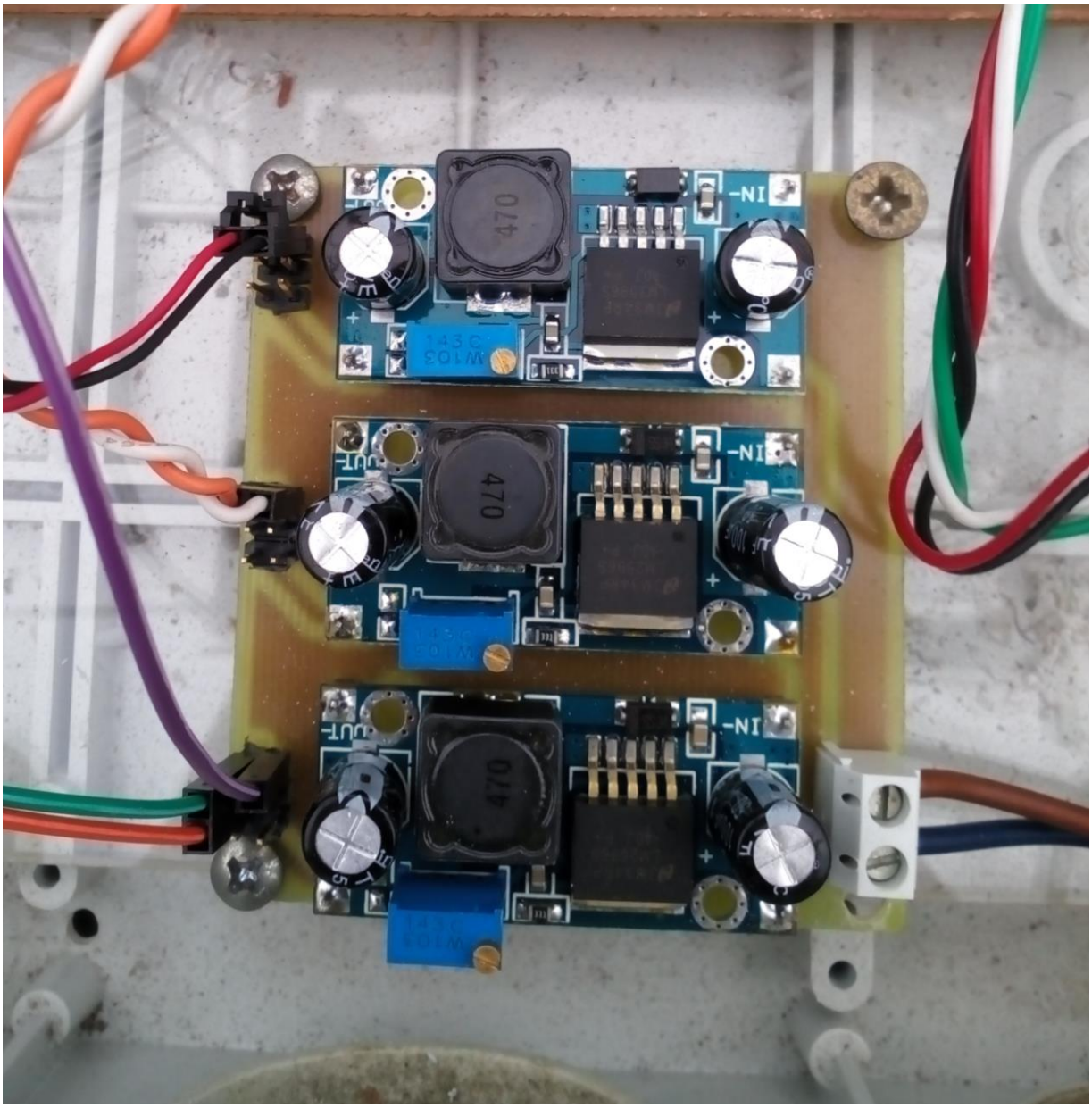


Figura 10 – Placa reductora de voltaje (3-5-12V)

- **Controladores de Potencia:** Para alimentar los motores contamos con dos unidades de control (Figura 11) que reciben la alimentación directa de 24V de las baterías y que estas a su vez están conectadas a unos controladores provenientes del micro, que le indican cuanta potencia deben aplicar a cada rueda, teniendo en cuenta la velocidad a la que se esté moviendo el carro.

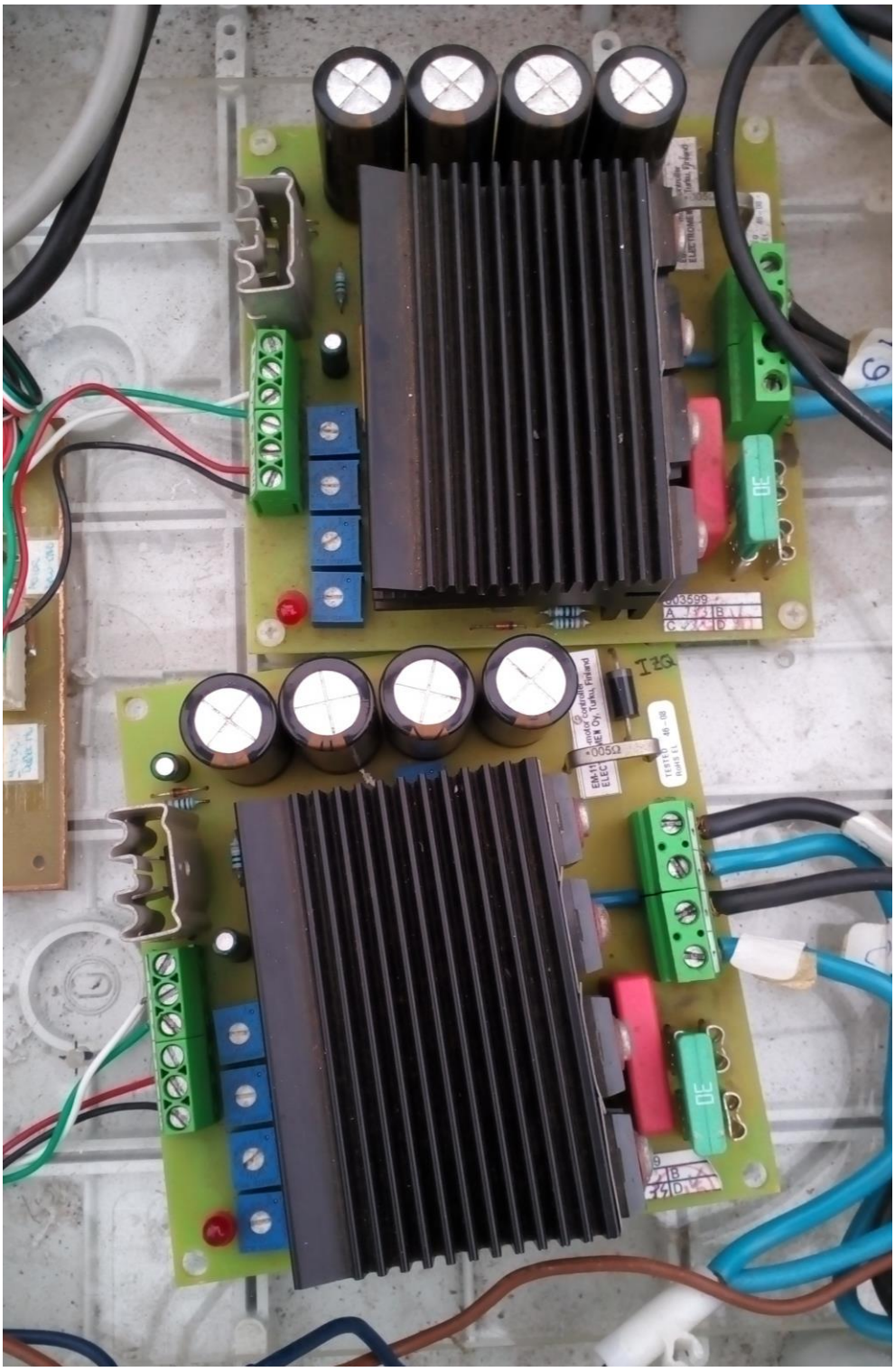


Figura 11 – Controladores de potencia para los motores

- **Microcontrolador MBed[2]:** Es el cerebro principal de todo el carro (Figura 12) y se encuentra en la caja de electrónica principal. Su principal función es la de manejar los eventos que vaya recibiendo, teniendo en cuenta el modo en el que se encuentre el carro, y enviar mensajes de control al Smartphone. Otra de sus funciones es la de indicarle a los controladores de potencia que velocidad debe aplicarse a las ruedas. En la parte inferior del micro, se encuentra una serie de pines, por donde se establece la comunicación con la mayoría de los elementos del sistema.

  Compuesto por un microcontrolador ARM Cortex-M de 32bits, 64KB de RAM y una memoria de 512KB, su principal objetivo es la de facilitar el acceso a los dispositivos del internet de las cosas. Dispone de diversas conexiones tales como ethernet, serial, señales analógicas, USB, así como una salida de 3.3V para alimentar a otros dispositivos.

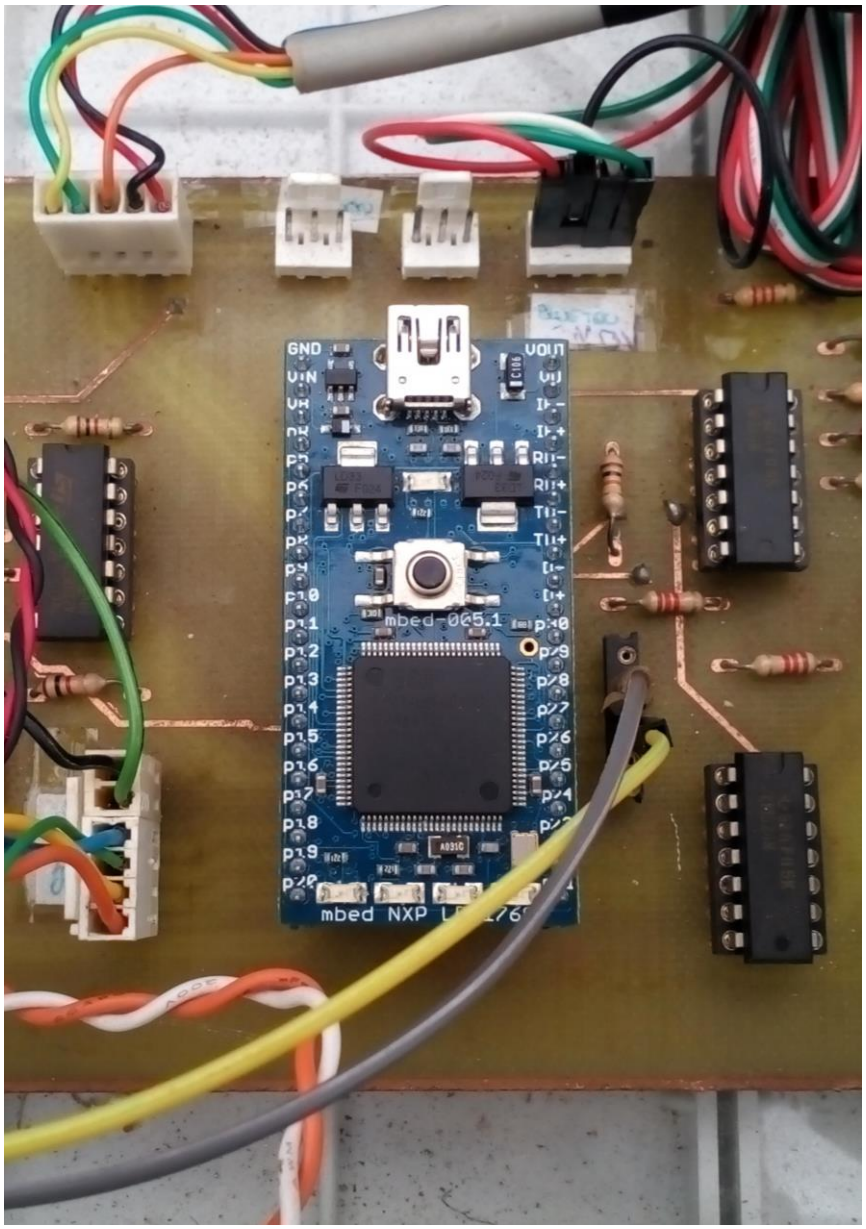


Figura 12 – Microcontrolador MBed

- **Wifi:** Para la comunicación con el Smartphone disponemos de un chip ESP8266[3] (Figura 13) conectado directamente a los pines del microcontrolador MBed por conexión serial y que se encarga de crear una red Wifi que sirva de puente entre el microcontrolador y el propio Smartphone. Este chip además posee una conexión a la placa de voltaje para alimentarse, aun cuando requiere 3.3V, evitando así sobrecargar el MBed.

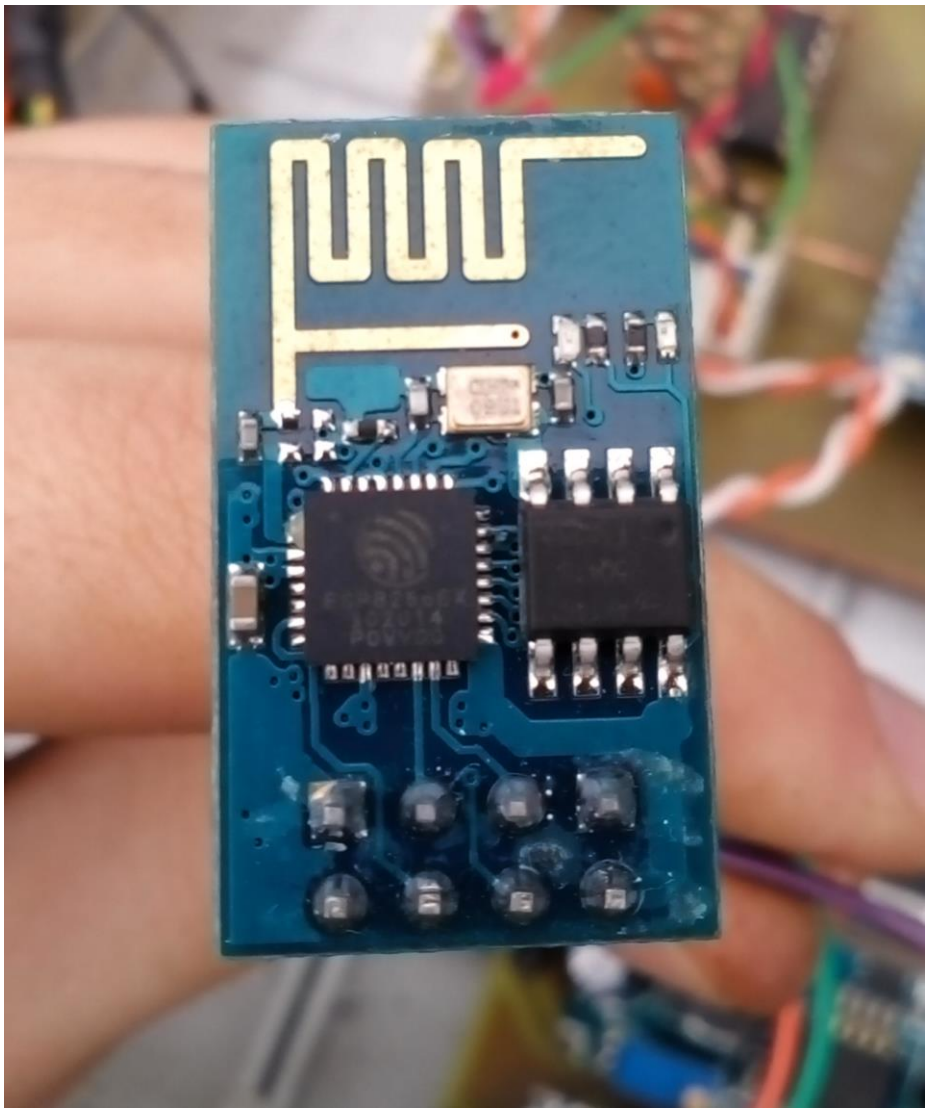


Figura 13 – Wifi ESP8266

- **Dongle Bluetooth:** A la hora de comunicarnos con el mando PS3 necesitamos un dispositivo Bluetooth conectado al MBed para manejar la conexión. Este dispositivo es un dongle (Figura 14) que se encuentra junto al microcontrolador en la caja de electrónica y que se comunica con el micro usando un conector hembra USB conectado utilizando los pines.

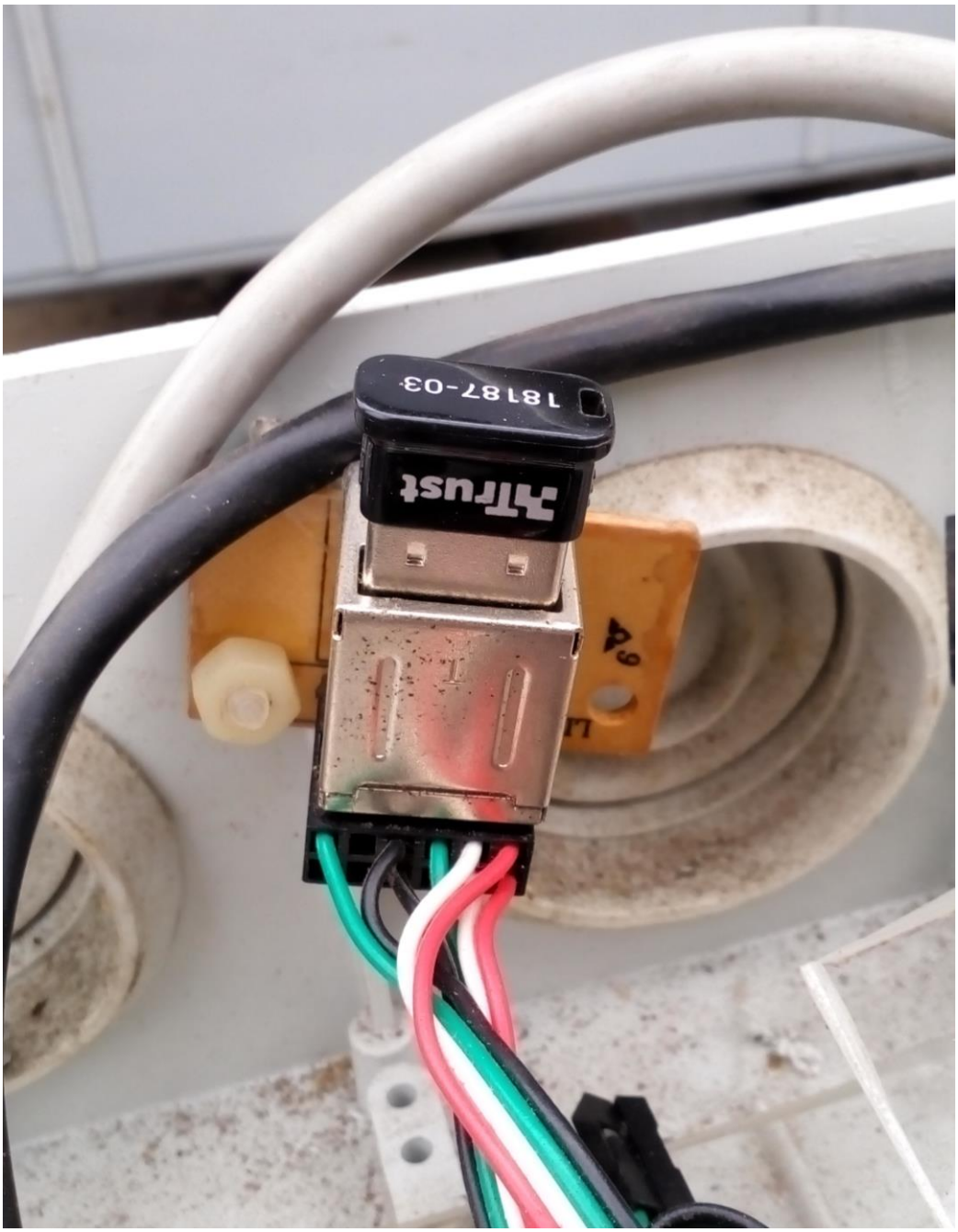


Figura 14 – Dongle Bluetooth

- **Mando PS3:** Se trata de un mando del sistema PS3 (Figura 15) que sirve de control a la hora de manejar el carro. Su funcionamiento es principalmente a distancia, aunque mediante el conector hembra USB disponible en el carro podemos comunicarnos con el MBed mediante un cable miniUSB que a su vez permite carga las baterías del mando. Utilizando este cable también es posible reestablecer la conexión con el sistema, si la sincronización no funcionase al pulsar el botón PS del mando.

Figura 15 – Mando PS3

- **Joystick manual:** En el cuadro principal del carro disponemos del joystick de control manual (Figura 16), el cual está conectado al microcontrolador mediante un par de cables que van desde el panel principal del carro, hasta la caja de electrónica situada en la parte inferior, y se conectan al micro directamente utilizando los pines.

Figura 16 – Joystick manual situado en el panel principal

- **Smartphone:** Se trata de cualquier Smartphone con sistema Android (Figura 17) y sirve tanto para manejar como para monitorizar el carro a distancia, haciendo uso del chip Wifi instalado en la caja de electrónica.

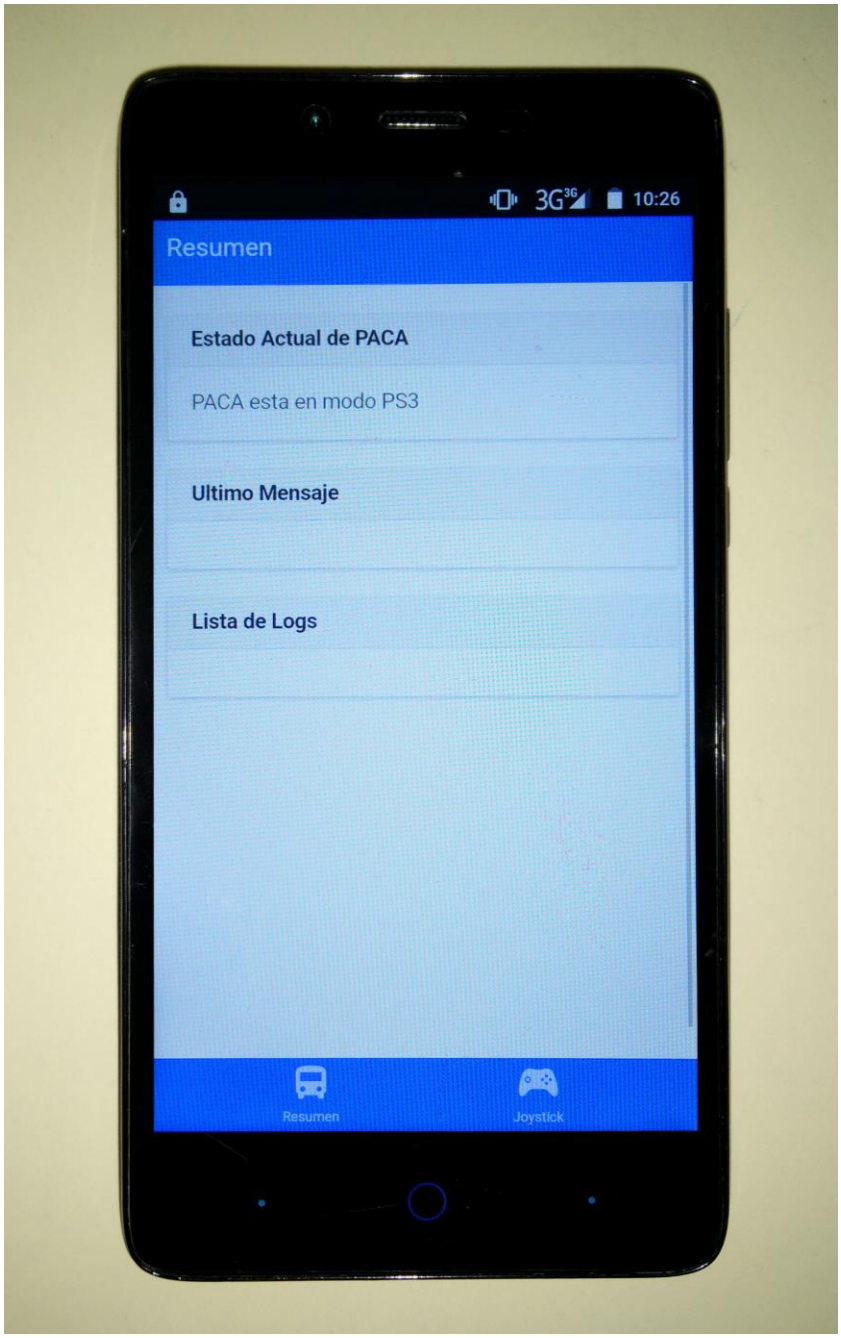


Figura 17 – Smartphone con la aplicación de control

- **Selector de estados:** A la hora de poder elegir de qué manera queremos manejar el carro, tenemos que utilizar un selector (Figura 18), situado en el panel principal del carro, y que cuenta con un total de cuatro estados. A la hora de poder elegir el estado actual del sistema, el MBed es el encargado de leer en qué posición se encuentra el selector, para después aplicar, a nivel software, las modificaciones necesarias para cambiar de modo. Esta lectura de estados se produce utilizando los pines presentes en la parte posterior del MBed. Aunque uno de los cuatro estados presentes en el selector, sirve para apagar el dispositivo MBed, esto no elimina la corriente de los demás elementos del sistema. Para poder cortar la corriente de todo el sistema, deberemos utilizar el interruptor correspondiente del panel general.

Figura 18 – Selector de estados del panel principal

- **Interruptor:** Junto al selector de estados, se encuentra situado un interruptor (Figura 19) que se encarga de cortar la corriente de todo el sistema, y no solo del MBed. A la hora de apagar el sistema completo este es el dispositivo que debemos utilizar y no el selector de estados.

Figura 19 – Interruptor de corriente general

- **Seta de seguridad:** Como medida de seguridad adicional, y aunque el sistema posea ya del interruptor general, existe una seta de seguridad (Figura 20) que también sirve para cortar la corriente de todo el sistema. Esto es así como medida extra de seguridad y para facilitar el apagado en caso de error, ya que es mucho más directo apretar esta seta que girar el interruptor.

Figura 20 – Seta de seguridad adicional

3.3 Componentes Software

En esta sección explicaremos las diferentes implementaciones que se hicieron a nivel software, así como aspectos del programa que ya existían y que se modificaron para añadirles mejoras. Aunque también se modificaron elementos electrónicos, debido a que el alumno que realizó el trabajo pertenece a la rama de programación de la Ingeniería Informática, la mayor parte del trabajo realizado se enfoca en esta sección. Los cambios e implementaciones principales se pueden dividir en tres categorías:

- **MBed:** Es el cerebro principal del sistema y el que contiene la mayor parte del código realizado. Este dispositivo posee, al ser una plataforma cerrada, su propia API y compilador online. Debido a esto todas las modificaciones que se realizaron se tuvieron que hacer mediante una herramienta situada en la nube y que permitía tanto modificar el código, como compilarlo y posteriormente descargarlo. El lenguaje utilizado por el sistema es principalmente C++. Las modificaciones realizadas al código se pueden dividir principalmente en:

    - **Wifi:** Para el correcto funcionamiento del Smartphone como sistema de control, se tuvieron que añadir modificaciones que soportasen la conexión con un dongle Wifi conectado al MBed. Esta conexión se estableció utilizado los pines RX y TX presentes en el MBed que sirvieron para crear una conexión serial entre ambos dispositivos.

      Una vez que la conexión estaba hecha, se tuvo que crear un modelo de datos para enviar y recibir información. Se optó por utilizar una tabla de códigos de estados y errores para indicarle al Smartphone en qué modo se encontraba el carro y los posibles errores que sucediesen en la ejecución del programa principal, seguidos de un carácter de control para verificar que el mensaje había llegado correctamente.

Nombre Código	Código a Enviar
Listo para recibir	-1
PS3	1
Joystick	2
Smartphone	3
Off	4

Tabla 3.3.1. Tabla de códigos de modos

Nombre Código	Código a Enviar
Sin Logs	5
Botón Ctrl NO Pulsado	6
Botón Ctrl Pulsado	7
Fallo Motores	8

Tabla 3.3.2. Tabla de códigos de logs

Ejemplo de código enviado desde el MBed al Smartphone indicando el modo actual:

```
1\r          //Código de PS3 Activado
```

Por otra parte, para que el Smartphone pudiese controlar el carro a distancia, se optó por enviar la posición del eje virtual con respecto a su centro, tanto del eje X como del Y, normalizándolos directamente en la propia aplicación, además de un estado de control que indicase si el usuario estaba pulsando la pantalla y un carácter de control para saber que el mensaje llegó correctamente.

Ejemplo de códigos enviados desde el Smartphone al MBed:

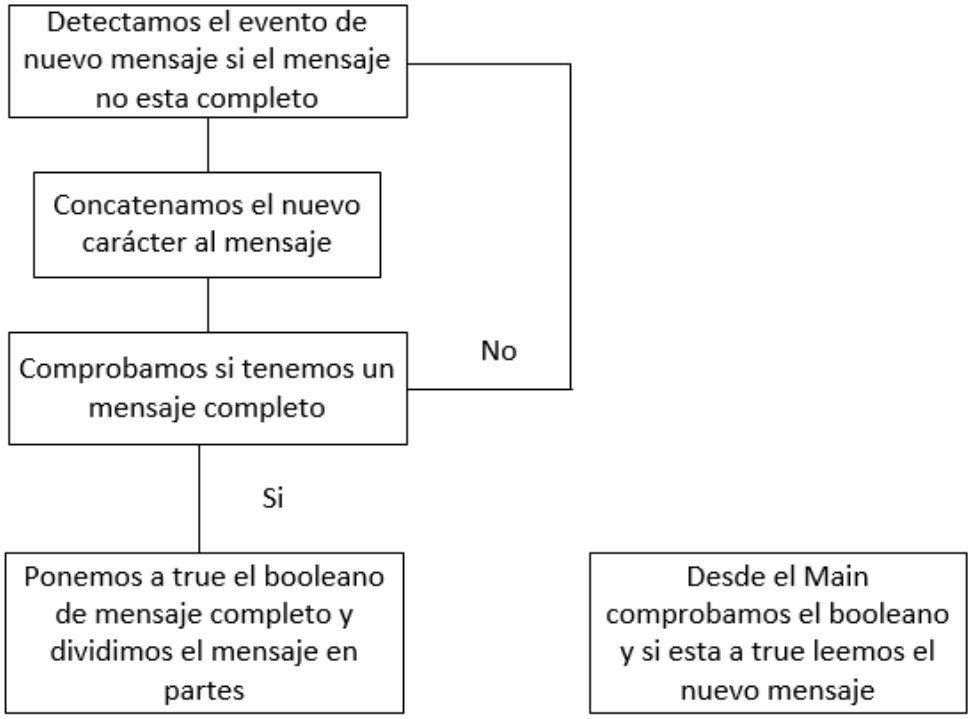
```
0.41,-0.28,1/ //Eje X, Eje Y, Botón de Control
```

En un principio se planteó que el MBed enviase el estado del carro al dongle en cada pasada del bucle principal del programa, pero se comprobó que el dongle no tenía potencia suficiente como para aguantar el estrés que esto suponía, con lo que no podía reenviar los datos a tanta frecuencia. Debido a esto se decidió crear un hilo a parte en el programa principal que se encargase de enviar el código al dongle, pero en un intervalo de medio segundo, con lo que se reducía considerablemente el trabajo que tenía que realizar el dongle. A su vez el MBed únicamente comprueba si existe nuevos mensajes por parte del Smartphone cuando el selector se encuentra en el modo correspondiente por lo que todos los posibles mensajes recibidos por el dongle en otro caso se desechan.

Cuando el selector se encuentra en modo Smartphone, para operar con los datos que le llegan, debe combinarlos. Esto es porque a la hora de recibir el mensaje a través del puerto RS232 , se hace byte a byte, por lo que debemos juntar todos los caracteres para formar el mensaje completo. Una vez que tenemos el mensaje lo dividimos en

secciones (Eje X, Eje Y y botón de Control) y pasamos estos valores a la función que se encargará de mover los motores.

Diagrama de flujo de la comunicación Wifi:



- **Mando PS3:** Aunque el mando PS3 ya se había implementado como sistema de control en otras revisiones del código, se tuvo que añadir a todo el proyecto de nuevo, ya que se había suprimido hacía tiempo. Para hacer que funcionase se tuvieron que implementar una serie de librerías que permitiesen la sincronización y emparejamiento con el mando, así como configurar el dongle Bluetooth conectado al MBed.

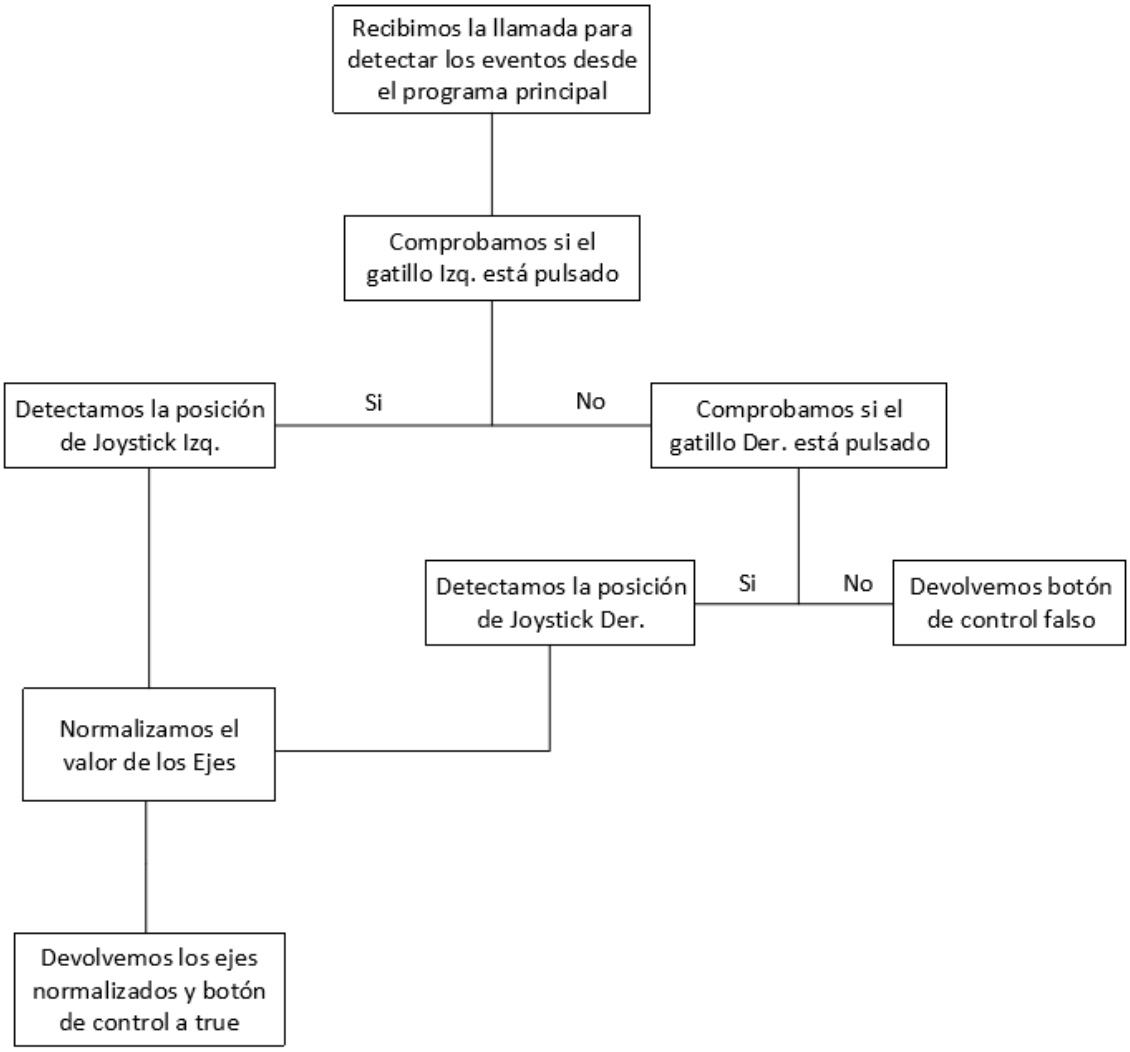
  A su vez, también se realizaron modificaciones en la propia librería para qué a la hora de recibir los datos, estos estuviesen normalizados con todo el programa principal, así como especificar que botones debían ser los que se usarían para el control del carro.

  Al comienzo del proyecto únicamente se configuró un joystick como sistema de control y un gatillo de seguridad, pero más adelante se decidió implementar ambos joysticks y que el usuario decidiese cual utilizar. Esto se hizo así, para dar al usuario la posibilidad de manejarlo como quisiese, pero sobretodo porque se comprobó que si el operario tenía una mano ocupada podía manejar el carro con la otra sin necesidad de cambiar el mando o la carga de sitio. Con este sistema, aunque se permiten utilizar ambos joysticks para el manejo

del carro, no se pueden utilizar a la vez, ni utilizar un gatillo con el joystick contrario.

Al igual que como sucede con el dongle Wifi y el Smartphone los posibles mensajes de manejo del carro únicamente se comprueban si el selector se encuentra en este modo, desechando todos los demás hasta que sean necesarios.

Diagrama de flujo de la función PS3:

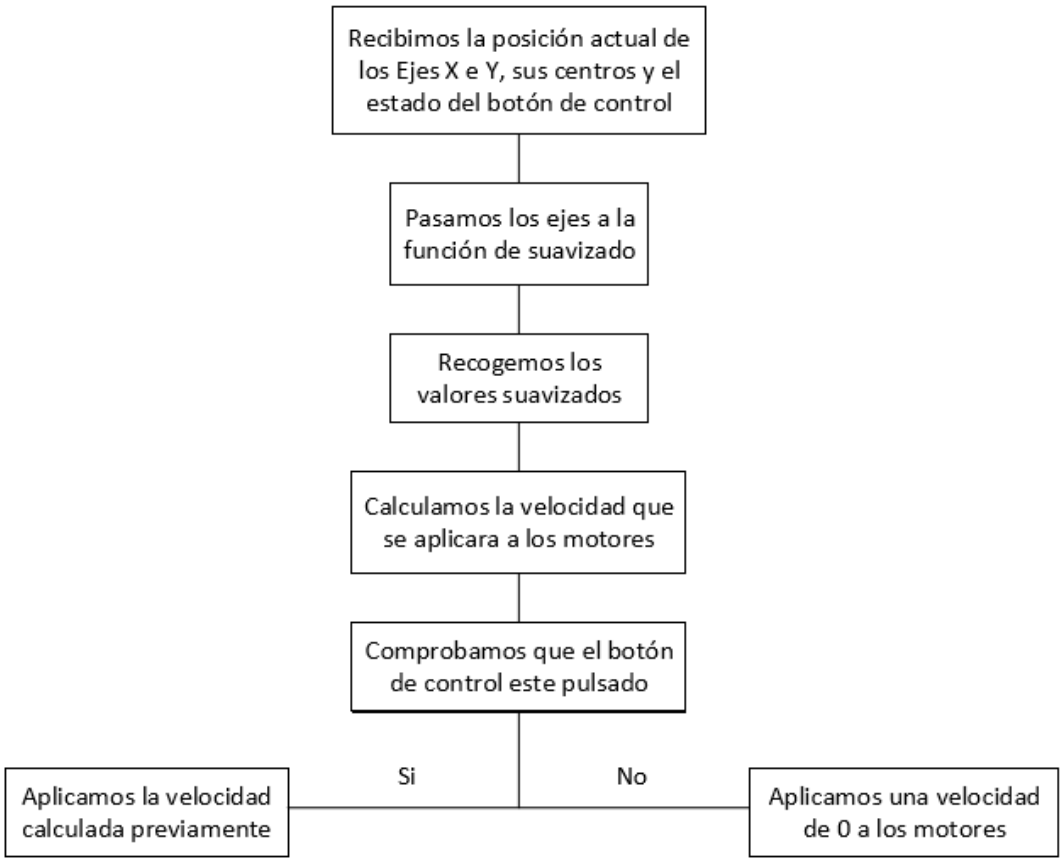


- **Modificaciones Función Joystick:** Debido a las modificaciones e implementaciones de los nuevos sistemas de control que se realizaron en el código, se planteó utilizar una función genérica que se encargase de mover los motores, independientemente del modo en el que estuviese el carro. Para ello se tuvieron que coger los datos que se recibían de los tres sistemas de control (Joystick, PS3 y Smartphone) y normalizar sus valores para que la función pudiese trabajar correctamente.

Se decidió colocar los posibles valores de los ejes (rango) en torno a 0.5 como máximo y -0.5 como mínimo, así como un booleano que indicase si el botón de seguridad estaba pulsado. Por otra parte, también se tuvieron que calcular la posición de los centros de los ejes, ya que el joystick manual debe calibrarse al comienzo del programa y la función que mueve los motores necesita dicho dato para calcular el desplazamiento relativo con respecto a su origen.

Una vez se tuvieron todos los valores procedentes de los tres sistemas normalizados, estos se pudieron pasar a la función joystick, centralizando el movimiento de los motores en un solo punto. Dicha función, aunque ya estaba presente en versiones anteriores del código, tuvo que ser modificada para añadirle una función de suavizado, que evitase acelerones bruscos a la hora de manejar el carro. Al finalizar esta función se obtienen los nuevos valores de los ejes que se aplicaran a los motores. A partir de estas nuevas posiciones se calcula la posición del eje para poder mover el carro en consecuencia, y además añadimos un índice de aceleración para incrementar la velocidad del sistema. Este índice ahora mismo posee un valor preestablecido, pero se ha planteado pasarle al Smartphone el control para poder aumentarlo o reducirlo dependiendo de si se desea recorrer una distancia larga o corta.

Diagrama de la función Joystick:



- **Función de Suavizado:** Cuando ya se tuvieron todos los modelos de datos de los sistemas de control normalizados y pasados a la función joystick, se decidió, después de hacer pruebas del manejo del carro, implementar una función que suavizase el control del carro. Esta idea surgió después de observar que el carro se aceleraba y frenaba de manera muy brusca, dificultando el poder manejarlo de manera fluida. Para este suavizado lo que se hizo fue una reducción del valor de entrada de los sistemas de control, incrementando el valor que se pasaba a la función joystick poco a poco, a lo largo del tiempo, en vez de asignarlo directamente.

  En este suavizado se distinguen dos tipos principalmente, que son el de aceleración y el de frenado. El suavizado de aceleración incrementa el valor que se aplica a los motores con una proporción menor que el de frenado. Con esto conseguimos que a la hora de detener el carro no tardemos tanto al hacerlo, conservando al mismo tiempo el suavizado que se buscaba.

  Diagrama de flujo de la función suavizado:

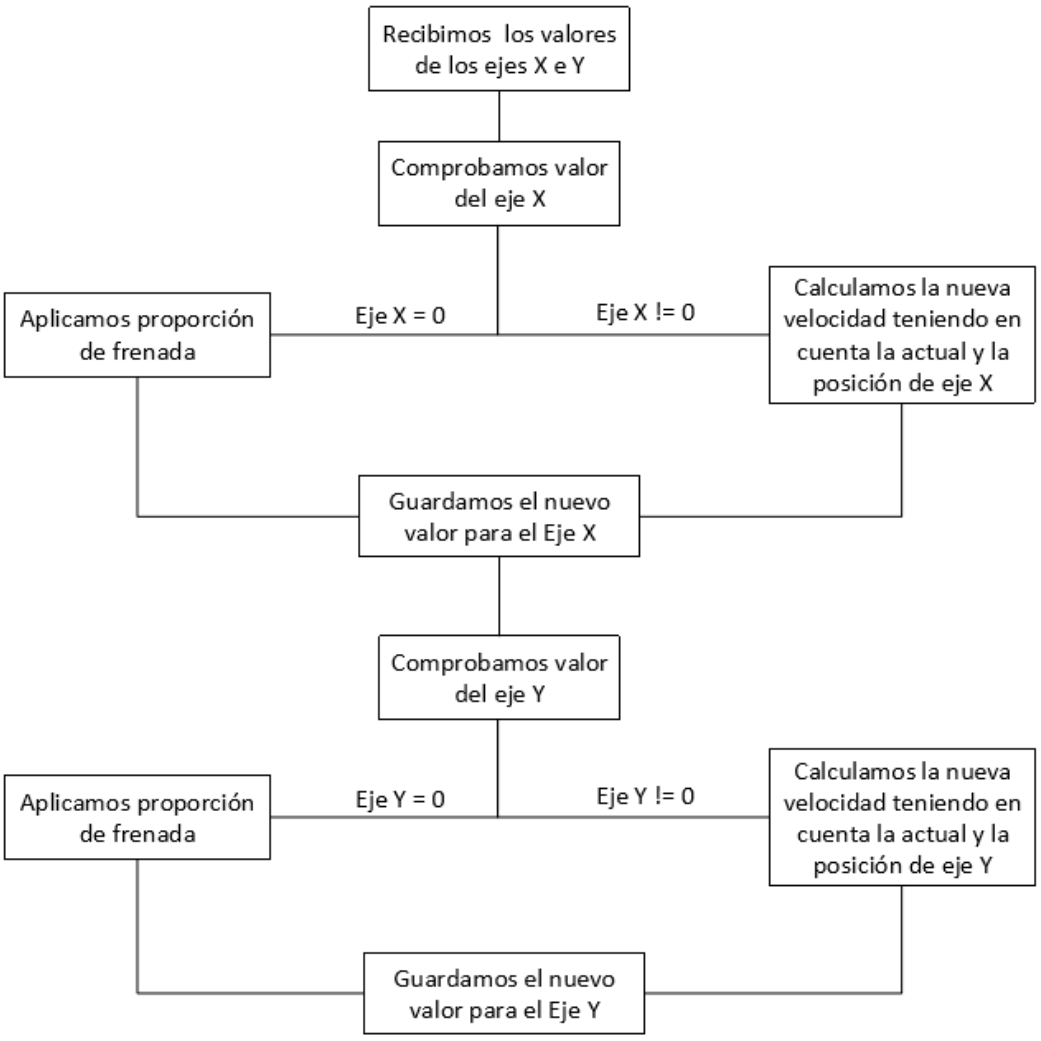
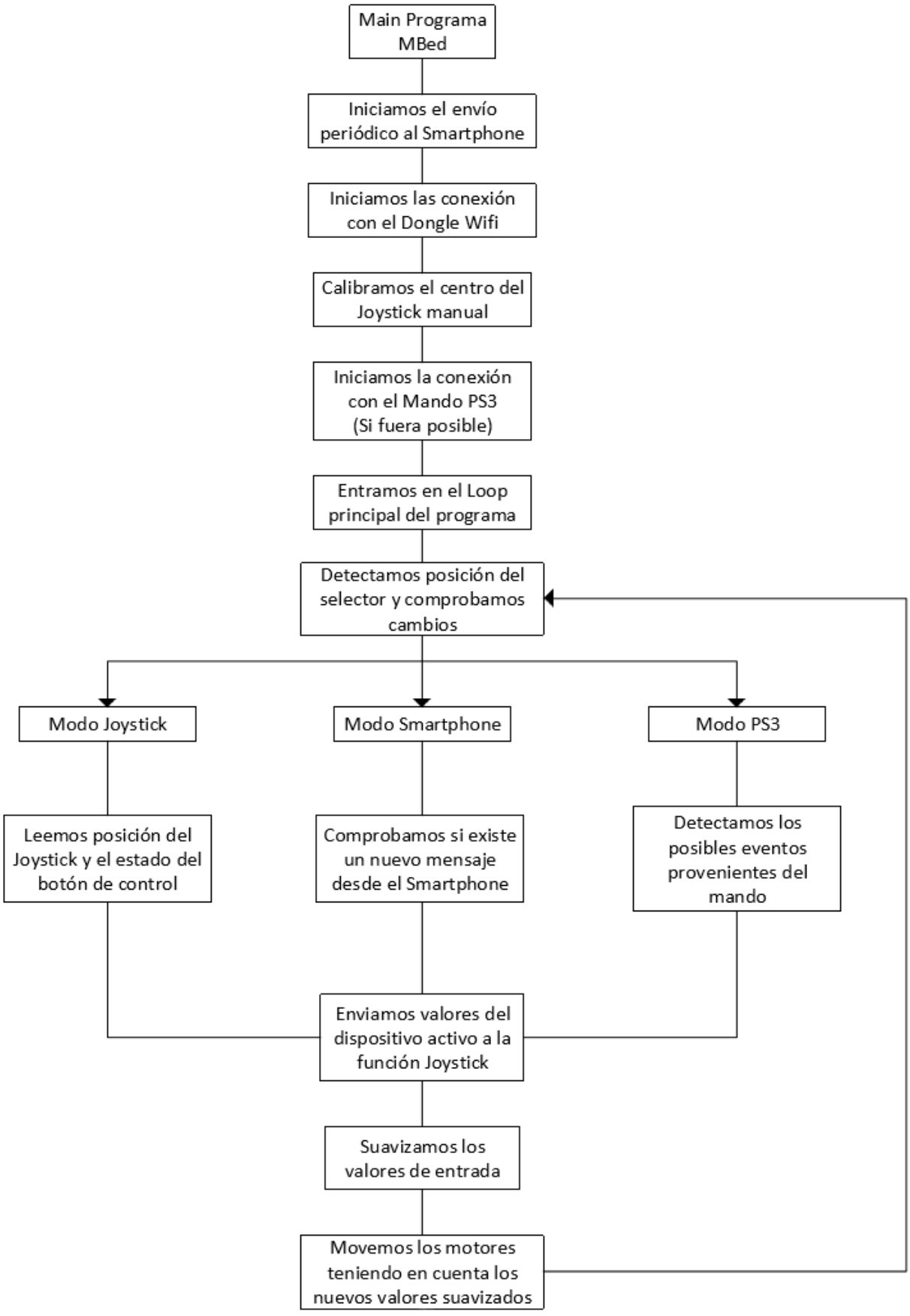


Diagrama de flujo del programa principal:



- **Aplicación Smartphone:** Para poder utilizar el Smartphone como un sistema de control remoto, se tuvo que diseñar una aplicación que estableciese una conexión Wifi y que enviase y recibiese mensajes en un formato especifico. Esta aplicación se implementó utilizando un framework llamado Ionic[4] que permite crear una aplicación multiplataforma en código HTML5/CSS3/JS y convertirla a un APK para instalarlo en cualquier dispositivo Android. Este framework, basado en Apache Cordova, añade elementos visuales y lógicos para facilitar el desarrollo y permite además incluirle módulos que añadan nuevas funcionalidades.

  La aplicación desarrollada se divide principalmente en dos secciones, la pantalla de control de estado del carro, y la de manejo del mismo. La primera pantalla es la que recibe todos los mensajes provenientes del MBed y contiene la tabla de posibles mensajes codificados, que una vez recibidos decodifica y muestra por pantalla en forma de texto. La segunda pantalla es la que se encarga de recibir los eventos del usuario y transmitirlos al MBed mediante la conexión Wifi. Esta sección se implementó adaptando un joystick virtual[5] de código libre en la interfaz del dispositivo que detectase los eventos táctiles de la pantalla y calculase su posición relativa con respecto al origen.

  Otro aspecto que se tuvo que implementar, pero que no forma parte de la interfaz de la aplicación, fue la conexión con el dongle Wifi. Debido a las características del dongle, se debía establecer una conexión que utilizase un Socket específico. El problema era que en HTML/JS no se permite abrir Sockets, sino que se debe hacer a nivel nativo. Este problema se consiguió solucionar incluyendo un módulo disponible para Cordova[6], el cual permite crear un Socket haciendo uso de código nativo y que añade funciones a JS, creando así, una capa de abstracción para el programador.

- **Dongle Wifi:[7]** A la hora de poder conectarnos con el dongle Wifi usando el Smartphone, se tuvo que configurar previamente el propio dongle para que proporcione un punto de acceso con una configuración específica. Para ello se tuvo que sustituir el firmware del dispositivo y añadirle todas las configuraciones de la red (Nombre, contraseña, seguridad, etc.) e indicarle que debía hacer al recibir un mensaje de un elemento conectado a la red. Todo este proceso se realizó utilizando una herramienta que permitía sustituir el firmware del dongle, el cual debía de estar escrito en el lenguaje Lua. Una vez hecho los cambios en el firmware, simplemente con conectar el dispositivo a la corriente, este crea una red Wifi accesible por cualquier dispositivo compatible sin, necesidad de configurar nada.

Capítulo 4.
Conclusiones, resultados y líneas futuras

Al finalizar el trabajo realizado, se ha llegado a la conclusión de que, en mayor o menor medida, todos los objetivos principales propuestos desde un comienzo, han sido realizados con éxito, dando como resultado, a falta de realizar algunas pruebas de rendimiento exhaustivo, un producto que bien podría usarse como dispositivo de carga en un entorno real.

En cuanto a experiencias y habilidades adquiridas, se deben resaltar los conocimientos más a nivel hardware que de software, puesto que, al provenir de la rama de computación, se tuvo más dificultad a la hora de abordar problemas de este tipo, lo cual a su vez sirvió para mejorar en esta rama.

Como principales objetivos alcanzados caben destacar:

- Estructuración del código.
- Inclusión del mando PS3.
- Mejora de la placa de voltajes.
- Instalación y configuración del módulo Wifi.
- Creación de la app de monitorización y control remoto desde Smartphone.
- Mejora de la función Joystick añadiéndole la función de suavizado

A lo largo del transcurso del trabajo se han ido planteando posibles mejoras o implementaciones a realizar en un futuro, tales como incluir un sistema de seguimiento por reconocimiento de patrones que consistiría en una cámara que buscase una imagen en concreto, situada por ejemplo en la parte posterior del chaleco de un operario, y que el carro siguiese a este de forma automática. Debido a la carga computacional que esto conllevaría, haría falta la inclusión de un microcontrolador más potente que el actual, como por ejemplo una Raspberry Pi.

Por otra parte, también se ha planteado, de cara a un producto más acabado a nivel hardware, sustituir las ruedas actuales por unas de oruga para evitar el hundimiento de estas en la tierra al llevar carga. Otra idea que se planteó, fue la de modular el sistema de electrónica principal, facilitando así los posibles cambios en el diseño del carro, a la vez que se aísla esta parte del sistema del resto de componentes. Por último se planteó el sustituir, o por lo menos tener como alternativa en el Smartphone, el cambio de estados del carro, implementándolo para que funcione remotamente y no tener que acercarse al carro para intercambiar los modos.

Capítulo 5.
Summary, Results and Conclusions

At the end of work developed, it has concluded that, all the major goals establish in the beginning have been successfully completed, resulting in a product that could be used in a real environment.

In terms of experiences and skills acquired, should be highlighted the knowledge acquired at hardware level, due to, as the student studied software computer science branch, it was much more difficult to face problems at hardware level than at software, which in turn, helped to improve in this area.

Main objectives are achieved should be highlighted:

- Structuring the code.
- Inclusion of the PS3 controller.
- Improved plate voltages.
- Installation and configuration of the wireless module.
- Creation of the app to monitories and remote control using the Smartphone
- Improved joystick function by adding the smoothing function

Throughout the course of the project, we have been considering possible improvements or implementations to perform in the future, such as including a pattern recognition system, which would consist of a camera that would be looking for a particular image, located for example, in the back of a vest, and the system would be able to follow it automatically. Due to computational burden that this improve would entail, it would be necessary the inclusion of a most powerful microcontroller, such as a Raspberry Pi.

Moreover, it has also been raised, the replacement of current wheels adding ones caterpillar wheels, which would prevent the collapse when the system is under a load. Another idea that was raised was modulate the main electronic system, thus facilitating futures changes in the design, while this part stays unchanged. Finally was proposed the idea of replacing, or at least have the possibility, of changing system state using the Smartphone.

Capítulo 6.
Presupuesto

En este capítulo desglosaremos todas las partes que componen el sistema tanto a nivel hardware como software, e indicaremos un precio por componentes aproximado, que servirá como guía a la hora de replicar el carro.

6.1 Componentes Mecánicos

Concepto	Unidades	Precio Unidad
Diseño y fabricación de un chasis	1	1500 €
Neumáticos y llantas	2	6 €
Ruedas locas	2	6 €
Batería	2	300 €
Selector de estados	1	10 €
Seta de seguridad	1	40 €
Interruptor	1	12 €
Total	--------	1874 €

Tabla 6.1.1. Tabla de componentes mecánicos

6.2 Componentes Electrónica de Potencia

Concepto	Unidades	Precio Unidad
Controlador electrónico de Potencia (EM-115 DC Motor Control Unit 12-36V 25A 4Quad)	2	160 €
Motores (PM90LWS)	2	400 €
Placa de conversión de voltaje (3-5-12V)	1	40
Total	--------	600 €

Tabla 6.2.2. Tabla de componentes de electrónica de potencia

6.3 Componentes de Interfaz

Concepto	Unidades	Precio Unidad
Joystick Manual (CH Product HF Series Hall effect joystick)	1	100 €
Mando de control PS3	1	25 €
Dongle Bluetooth	1	13 €
Módulo Wifi ESP8266	1	10 €
Microcontrolador MBed	1	45 €
Diseño de aplicación móvil Android	1	650 €
Diseño de software de control MBed	1	800 €
Total	--------	1643 €

Tabla 6.3.3. Tabla de componentes software y de interfaz

6.4 Componentes Varios

Concepto	Unidades	Precio Unidad
Metacrilato	1	48 €
Caja estanca de protección electrónica	1	60 €
Cableado	1	30 €
Total	--------	138 €

Tabla 6.4.4. Tabla de componentes varios utilizados en el sistema

Sumando todos los componentes necesarios para la creación de un carro completo, nos da un total de 4255 €, en donde los componentes mecánicos compondrían un 44% del presupuesto, los de electrónica de potencia un 14%, los de interfaz un 38% y los componentes varios un 3%.

Apéndice A.
Códigos implementados

A.1. MBed - Bucle principal

```
/*******************************************************************************
* Fichero main.cpp
*******************************************************************************/
/**
 *  En este metodo nos mantenemos en un bucle constante
 *  comprobando el estado del selector y llamando a joystick
 */
void loop(){
    while(1){
        selector(d1,d2);       //Detectamos el modo del selector
        checkCambioModo(modo); //Comprobamos los cambios de modo
        switch (modo) {        //Comparamos el modo para saber el estado actual
            case JOYSTICK:
                //Pasamos el boton del Joystick negado porque esta a baja
                joystick(ejeXJoystick, ejeYJoystick, Xcentro, Ycentro,
                         !botonJoystick);
                break;
            case MOVIL:
                //Si recibimos un mensaje nuevo con nuevos valores actualizamos
                if(mensajeCompletado()){
                    float *ejes = getEjesWifi();
                    //Actualizamos los ejes del Auto con los nuevos que nos llegaron
                    ejesAuto[0]=ejes[0];  ejesAuto[1]=ejes[1];  ejesAuto[2]=ejes[2];
                }
                joystick(ejesAuto[0], ejesAuto[1], CENTRO_PS3_Y_AUTO,
                         CENTRO_PS3_Y_AUTO, ejesAuto[2]);
                break;
            case PS3:
                RunPS3();      //Detectamos los eventos del mando PS3
                joystick(getEjeXPS3(), getEjeYPS3(), CENTRO_PS3_Y_AUTO,
                         CENTRO_PS3_Y_AUTO, getBotonCtrlPS3());
                break;
             case OFF:
                break;
            default:
                break;
        }//Final switch
    }//Final while
}//Final metodo loop()
```

A.2. MBed - Función Joystick

```
/*****************************************************************************
* Fichero main.cpp
*****************************************************************************/
/** Metodo para calcular el movimiento del joystick y asi mover los motores correctamente
 *      @param ejeX             Representa el valor del eje X
 *      @param ejeY             Representa el valor del eje Y
 *      @param centroX          Centro del Axis del eje X
 *      @param centroY          Centro del Axis del eje Y
 *      @param botonPulsado     Indica si el boton de control esta pulsado o no
 */

void joystick(float ejeX, float ejeY, float centroX, float centroY, bool botonPulsado){
    //Suavizamos los valores de los ejes de entrada
    calcularVelocidadEjesSuavizada(ejeX, ejeY);

    float y, x;
    y = yAct;
    x = xAct;

    y = y - centroY;
    x = x - centroX;

    float indiceAceleracion = 2.0;
    if (y > 0.1) {
        ri = indiceAceleracion * (2 * (y-0.1) - L * x)/2.5;
        rd = indiceAceleracion * (2 * (y-0.1) + L * x)/2.5;
    }
    else if (y < -0.1) {
        ri = (2 * (y+0.1) + L * x)/2.5;                 // Se puede cambiar los signos + - Lx
        rd = (2 * (y+0.1) - L * x)/2.5;                 // para corregir el giro marcha atras
    }
    else {
        ri = x/4;
        rd = -x/4;
    }

    vd = (botonPulsado) ? abs(rd)*4095 : 0;
    vi = (botonPulsado) ? abs(ri)*4095 : 0;

    DAC dac(rd,ri);
    dac.DAC_Output(vd, vi);
}// Fin metodo joystick()
```

A.3. MBed - Función Suavizado

```
/*******************************************************************************
* Fichero main.cpp
*******************************************************************************/
/**
 *  Calculamos la velocidad que vamos a aplicar a los motores
 *  teniendo en cuenta el paso del tiempo, para que no se
 *  acelere de manera brusca si no suavizada
 *
 *  @param x    Contiene la posicion actual del eje X
 *  @param y    Contiene la posicion actual del eje Y
 */
void calcularVelocidadEjesSuavizada(float x,float y){
    //Si el valor de X es 0 aplicamos la PROPORCION_FRENADA
    xAct = (x != 0) ? (xAct*(1-PROPORCION) + x*PROPORCION) : (xAct*(1-PROPORCION_FRENADA));

    //Si el valor de Y es 0 aplicamos la PROPORCION_FRENADA
    yAct = (y != 0) ? (yAct*(1-PROPORCION) + y*PROPORCION) : (yAct*(1-PROPORCION_FRENADA));
}//Final metodo calcularVelocidadEjesSuavizada(float x,float y)
```

A.4. MBed - Función EnviarCodigoWifi

```
/*******************************************************************************
* Fichero wifi.cpp
*******************************************************************************/
/**
 * Enviamos el codigo especificado por parametro al modulo Wifi
 *
 * @param code Contiene el codigo a enviar
 */
void enviarCodigoWifi(int code){
    char mensaje[5];
    sprintf (mensaje, "%d\r", code);

    //Si vamos a enviar un MODO
    if((code >= 1) && (code <= 4)){
        uart_wifi.printf(mensaje);
    }
    //Si vamos a enviar un LOG y es distinto al anterior
    else if((code >= 5) && (code <= 8) && (code != ultimoLogEnviado)){
        ultimoLogEnviado = code;
        uart_wifi.printf(mensaje);
    }
    else if(code == LISTO_PARA_RECIBIR_MENSAJE){
        uart_wifi.printf(mensaje);
    }
}
```

A.5. MBed - Función GetEjesWifi

```
/*****************************************************************************
* Fichero wifi.cpp
*****************************************************************************/
/**
 *  Parseamos el mensaje que hemos recibido y extraemos los
 *  valores de los ejes y del boton de control
 *
 *  @return pointerEjes Contiene una referencia al array de los
 *          ejes y boton de control
 */
float* getEjesWifi(){
    float *pointerEjes;

    pointerEjes = ejesWifi;

    int i=0;
    //Mientras encontremos comas, extraemos los valores
    while ((pos = mensajeRecibido.find(delimiter)) != string::npos) {
        ejesWifi[i] = atof((mensajeRecibido.substr(0, pos)).c_str());

        mensajeRecibido.erase(0, pos + delimiter.length());

        i++;
    }
    //Guardamos el ultimo valor (No existen comas al final del mensaje)
    ejesWifi[i] = atof((mensajeRecibido).c_str());

    mensajeCompleto = false;                        //Colocamos el mensaje completo a falso
    mensajeRecibido = "";                           //Limpiamos el mensaje anterior
    enviarCodigoWifi(LISTO_PARA_RECIBIR_MENSAJE);   //Indicamos al movil que podemos seguir
                                                    //recibiendo mensajes

    return pointerEjes;
}
```

A.6. MBed - Función ParsePS3Result

```
/*****************************************************************************
* Fichero Ps3USB.cpp
*****************************************************************************/
/**
 *  Con este metodo detectamos los eventos del mando PS3 y guardamos
 *  los valores de sus variables (Ejes y botones de control unicamente)
 *  A su vez identificamos que eje estamos utilizando mediante la
 *  deteccion de que gatillo esta pulsado actualmente, teniendo
 *  prioridad el gatillo izquierdo.
 *
 *  Las variables ejeX, ejeY y gatilloCtrl son variables globales en
 *  donde se almacenan los valores del eje y gatillo seleccionado
 */
int ParsePs3Result(const u8* data, int len, int count){
      //Contendran el valor de los ejes izquierdo y derecho, asi como sus gatillos
      int Lx, Ly, LbuttonCtrl;
      int Rx, Ry, RbuttonCtrl;

      //Recogemos los datos de entrada en una variable para acceder a cada elemento mas adelante
      ps3report* _ps3report = (ps3report*)data;

      //Recogemos los eventos del Joystick y gatillo IZQUIERDO
      Lx = (int) _ps3report->LeftStickX;
      Ly = (int) _ps3report->LeftStickY;
      LbuttonCtrl = (int) _ps3report->PressureL2;

      //Recogemos los eventos del Joystick y gatillo DERECHO
      Rx = (int) _ps3report->RightStickX;
      Ry = (int) _ps3report->RightStickY;
      RbuttonCtrl = (int) _ps3report->PressureR2;

      //Normalizamos el Joystick IZQ por defecto
      ejeX=(Lx-128);
      ejeY=(Ly-128);
      ejeX/=256;
      ejeY/=-256;

      //Si estamos pulsando el gatillo IZQ lo colocamos a true
      if(LbuttonCtrl > 100){
            gatilloCtrl = true;
      }
      //Si estamos pulsando el gatillo DER normalizamos el Joystick y ponemos el gatillo a true
      else if(RbuttonCtrl > 100){
            gatilloCtrl = true;
            ejeX=(Rx-128);
            ejeY=(Ry-128);
            ejeX/=256;
            ejeY/=-256;
      }
      //Si no hemos pulsado ningun gatillo lo colocamos a falso
      else{
            gatilloCtrl = false;
      }
}
```

A.7. Smartphone – Evento al recibir datos

```
/*****************************************************************************
* Fichero app.js
*****************************************************************************/
//Evento al recibir datos desde el MBed
socket.onData = function (data) {
	var c = String.fromCharCode.apply(null, data);
	datosRecibidos = parseInt(c);
	//Si recibimos el codigo para seguir enviando datos del Joystick
	if(datosRecibidos == -1)
		continuarEnvioDatos = true;

	//Si nos llega un log o un modo distinto lo cambiamos, sino no
	else if((ultimoLogGuardado != datosRecibidos) &&
		(ultimoModoGuardado != datosRecibidos)){
		//Si nos envian un modo
		if ((datosRecibidos >= 1) && (datosRecibidos <= (stringModos.length))){
			document.getElementById("paca-status").textContent = "PACA esta en modo " +
                                                                stringModos[datosRecibidos-1];
			ultimoModoGuardado = datosRecibidos;
		}
		//Si no nos envian un modo es un log por lo que comprobamos que exista
		else if(stringLogs[datosRecibidos-(stringModos.length+1)] != null){
			//Actualizamos el valor del ultimo log guardado
			ultimoLogGuardado = datosRecibidos-(stringModos.length+1);

			//Actualizamos el texto del ultimo log a mostrar
			document.getElementById("last-log").textContent = stringLogs[ultimoLogGuardado];

			var listaLogsRef = document.getElementById("listaLogs");

			//Insertamos el separador
			var separator = document.createElement("li");
			separator.className = "divider-vertical-second-menu";
			listaLogsRef.insertBefore(separator, listaLogsRef.firstChild);

			//Insertamos el log a la lista
			var item = document.createElement("li");
			item.innerHTML = stringLogs[ultimoLogGuardado];
			listaLogsRef.insertBefore(item, listaLogsRef.firstChild);
		}
	}
};
```

A.8. Smartphone – Evento al pulsar el joystick virtual

```
/*****************************************************************************
* Fichero main.js
*****************************************************************************/
//Evento al pulsar el joystick virtual
function joystickTouch(){
	var outputEl = document.getElementById('result');
	var ejeX, ejeY;

	var joystickX = joystick.deltaX();
	var joystickY = -joystick.deltaY();

	//Combrobamos que no nos salimos de rango
	if(joystickX > 150)
		ejeX = 0.5;
	else if(joystickX < (-150))
		ejeX = -0.5;
	else
		ejeX = (joystickX/300).toFixed(2);

	//Combrobamos que no nos salimos de rango
	if(joystickY > 150)
		ejeY = 0.5;
	else if(joystickY < (-150))
		ejeY = -0.5;
	else
		ejeY = (joystickY/300).toFixed(2);

	if (botonPulsado)
		mensaje = ejeX.toString() + "," + ejeY.toString() + ",1/";
	else
		mensaje = ejeX.toString() + "," + ejeY.toString() + ",0/";

	//Mostramos la posicion
	outputEl.innerHTML = '<b>Result:</b> ' + mensaje;

	enviarMensajeWifi(mensaje);
}
```

Apéndice B.
Planos y Esquemas de Conexiones

B.1. Plano Finca Las Lucanas

Se trata de un esquema general sobre cómo está estructurada la finca. En el se representan las distintas secciones que la componen, y además se representa una serie de caminos, por donde el carro puede desplazarse.

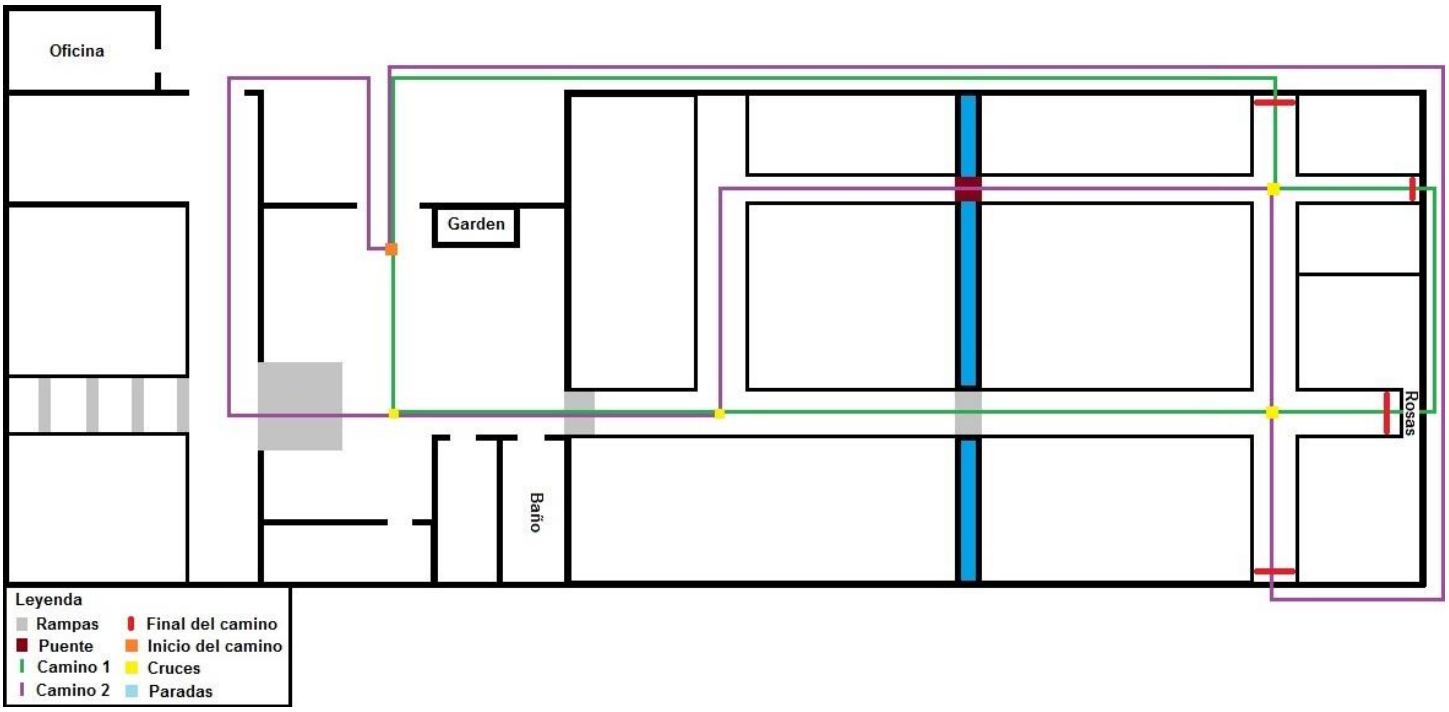


Figura 21 – Plano general de Finca Las Lucanas

B.2. Esquema de conexión MBed – ESP8266

Es una representación del esquema general de conexión entre el microcontrolador MBed y el dispositivo Wifi ESP8266.

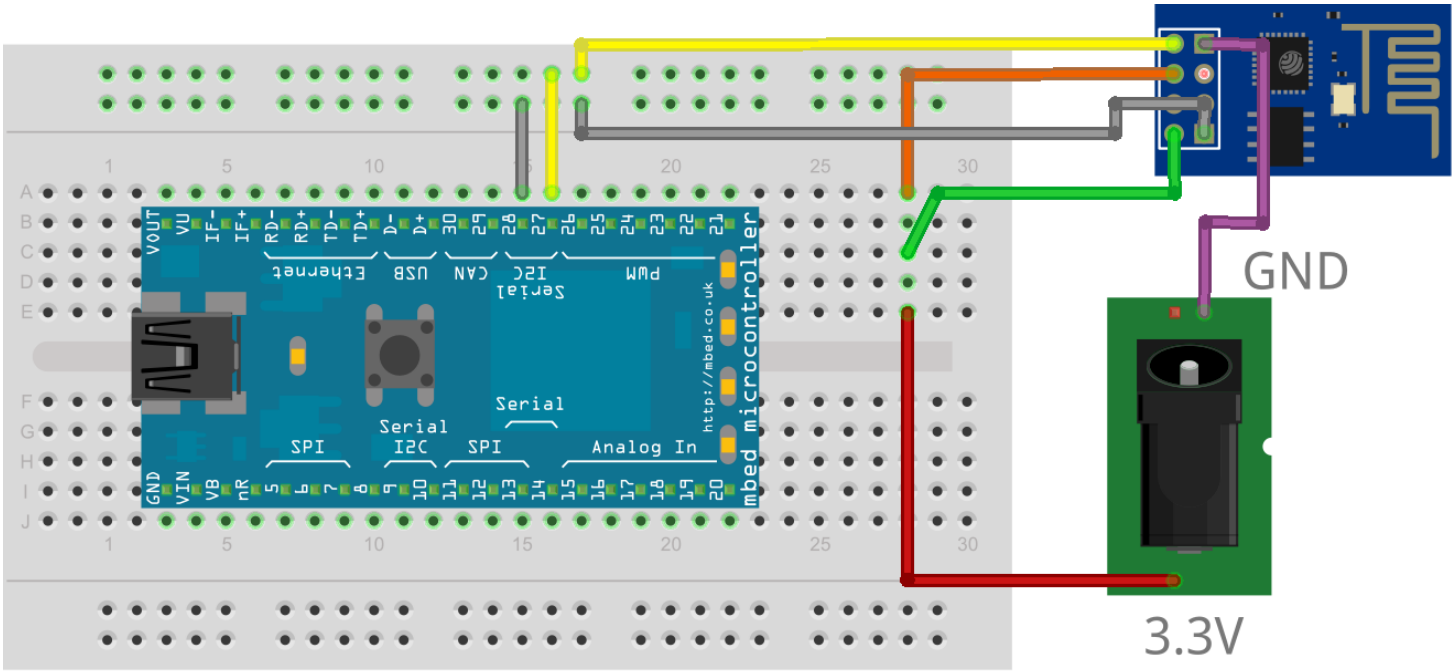


Figura 22 – Esquema de conexión MBed – ESP8266

Bibliografía

[1] Documento de referencia del Proyecto creado por Eduardo - https://drive.google.com/a/ull.edu.es/file/d/0B6wNOdTOPdq9czNZT3p5d1pGc00/view?usp=sharing

[2] Web oficial de MBed - https://www.mbed.com/en/

[3] Web oficial Wifi ESP8266 - http://www.esp8266.com/

[4] Web oficial Ionic - http://ionicframework.com/

[5] Virtual Joystick Smartphone - https://github.com/jeromeetienne/virtualjoystick.js

[6] Plugin Socket Ionic - https://github.com/blocshop/sockets-for-cordova

[7] Actualización del firmware Wifi - http://www.whatimade.today/loading-the-nodemcu-firmware-on-the-esp8266-windows-guide/