UNIVERSIDAD DE LAS PALMAS DE GRAN CANARIA
Instituto Universitario de Microelectrónica Aplicada
Sistemas de información y Comunicaciones

Máster en
Tecnologías de Telecomunicación

Trabajo Fin de Máster

SISTEMA DE TRANSMISIÓN DE DATOS ENTRE DISPOSITIVOS EN MOVIMIENTO PARA EL SEGUIMIENTO DE UN VEHÍCULO

Autor: Aythami Salvador Rodríguez Valentín
Tutor(es): José Francisco López Feliciano
Pablo Sebastián Horstrand Andaluz
Fecha: Julio de 2017

IUMA
t +34 928 451 086 | iuma@iuma.ulpgc.es
f +34 928 451 083 | www.iuma.ulpgc.es
Campus Universitario de Tafira
35017 Las Palmas de Gran Canaria

Versión: 2.0 – 14 junio 2011

UNIVERSIDAD DE LAS PALMAS DE GRAN CANARIA
Instituto Universitario de Microelectrónica Aplicada
Sistemas de información y Comunicaciones

Máster en Tecnologías de Telecomunicación

Trabajo Fin de Máster

SISTEMA DE TRANSMISIÓN DE DATOS ENTRE DISPOSITIVOS EN MOVIMIENTO PARA EL SEGUIMIENTO DE UN VEHÍCULO

HOJA DE FIRMAS

Alumno/a: Aythami Salvador Rodríguez Valentín Fdo.:

Tutor/a: José Francisco López Feliciano Fdo.:

Tutor/a: Pablo Sebastián Horstrand Andaluz Fdo.:

Fecha: Julio de 2017

IUMA
t +34 928 451 086 | iuma@iuma.ulpgc.es
f +34 928 451 083 | www.iuma.ulpgc.es
Campus Universitario de Tafira
35017 Las Palmas de Gran Canaria

Versión: 2.0 – 14 junio 2011

UNIVERSIDAD DE LAS PALMAS DE GRAN CANARIA
Instituto Universitario de Microelectrónica Aplicada
Sistemas de información y Comunicaciones

Máster en
Tecnologías de Telecomunicación

Trabajo Fin de Máster

SISTEMA DE TRANSMISIÓN DE DATOS ENTRE DISPOSITIVOS EN MOVIMIENTO PARA EL SEGUIMIENTO DE UN VEHÍCULO

HOJA DE EVALUACIÓN

Calificación: .....

Presidente Fdo.:

Secretario Fdo.:

Vocal Fdo.:

Fecha: Julio de 2017

IUMA
t +34 928 451 086 | iuma@iuma.ulpgc.es
f +34 928 451 083 | www.iuma.ulpgc.es
Campus Universitario de Tafira
35017 Las Palmas de Gran Canaria

AGRADECIMIENTOS

En primer lugar, quisiera agradecerle a toda mi familia por todo su apoyo y cariño, y en especial, a mi madre Juana Valentín Guillen y a mi padre Salvador Rodríguez Reyes, que no se encuentra conmigo después de su fallecimiento. Además, quiero dedicarle unas palabras a mi padre por su apoyo, su gran fuerza, su sonrisa, su cariño y su afán por vivir la vida, así como yo también lo haré por él.

En segundo lugar, quiero darle las gracias a mi hermana Irina Rodríguez Valentín por comprenderme y ayudarme en los buenos y malos momentos que hemos pasado, dándome siempre todo su amor de hermana para superar todas las facetas de la vida.

En tercer lugar, agradecerle a mi tutor José Fco. López Feliciano y mi cotutor Pablo Horstrand Andaluz por brindarme su apoyo y darme la oportunidad de realizar este Trabajo Fin de Máster que forma parte del proyecto europeo ENABLE-S3, co-financiado por el Gobierno de España a través del Programa Estatal de Investigación, Desarrollo e Innovación Orientada a los Retos de la Sociedad, en su convocatoria 2015-2.

ÍNDICE GENERAL

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

LISTADO DE ACRÓNIMOS

ASCII	American Standard Code for Information Exchange
EEPROM	Erasable Programmable Read-Only Memory
EGNOS	European Geostationary Navigation Overlay Service
ENABLE-S3	European Initiative to Enable Validation for Highly Automated Safe and Secure Systems
FTDI	Future Technology Devices International
GFSK	Gaussian Frequency Shift Keying
GPGGA	Global Positioning System Fix Data
GPGLL	Geographic Position, Latitude and Longitude
GPGSA	Global Positioning System Satellite Status
GPIO	General Purpose Input / Output
GPRMC	Global Positioning Recommended Minimum Sentence C
GPS	Global Positioning System
GPVTG	Global Positioning Vector Track and Speed Over Ground
HELICoiD	Hyperspectral Imaging Cancer Detection
ISM	Industrial, Scientific and Medical
I2C	Inter-Integrated Circuit
LORA	Long Range
MAC	Media Access Control
MSAS	Multi-functional Satellite Augmentation System
NMEA	National Maritime Electronic Association
RAM	Random Access Memory
RPI	Raspberry PI
RSSI	Received Signal Strength Indicator
SBAS	Satellite Based Augmentation System
SDK	Software Development Kit
SPI	Serial Peripheral Interface
SRAM	Static Random Access Memory
UART	Universal Asynchronous Receiver-Transmitter

USB	Universal Serial Bus
WAAS	Wide Area Augmentation System
WIFI	Wireless Fidelity

Capítulo 1

INTRODUCCIÓN

1.1. ANTECEDENTES

La Unión Europea aprobó en 2016 un proyecto de investigación dotado con más de 34 millones de euros, en el cual participa el Instituto Universitario de Microelectrónica Aplicada (IUMA) de la Universidad de Las Palmas de Gran Canaria. ENABLE-S3 (acrónimo del proyecto, de su nombre en inglés "European Initiative to Enable Validation for Highly Automated Safe and Secure Systems") está compuesto por un total de 74 socios de 15 países distintos, de los cuales 8 participantes son españoles. El IUMA, junto con el Centro de Electrónica Industrial (CEI) de la Universidad Politécnica de Madrid, son los dos únicos centros universitarios españoles que participan en este macro-proyecto, cuya duración estimada es de 3 años. En paralelo, este proyecto se ha visto complementado con fondos del Ministerio de Economía y Competitividad del Gobierno de España, a través de las acciones de Programación Conjunta Internacional del Programa Estatal de Investigación, Desarrollo e Innovación Orientada a los Retos de la Sociedad, en su convocatoria 2015-2.

ENABLE-S3 tiene un carácter claramente industrial, y su objetivo prioritario es el desarrollo de sistemas altamente automatizados, autónomos y seguros. En los últimos años se han venido desarrollando distintas iniciativas destinadas a la creación de coches inteligentes sin conductor, que permitan disminuir el número de accidentes y aumentar el grado de confort y la seguridad de sus ocupantes y de los peatones en general. No en vano, la mayoría de los accidentes de tráfico se deben a errores de los conductores, y no a defectos de los propios vehículos. Uno de los principales problemas a la hora de diseñar este tipo de sistemas, es que se necesita más de 1 millón de kilómetros de verificación hasta validar el producto, por lo que resulta deseable contar con sistemas que agilicen este proceso de verificación. La industria del automóvil es donde el proyecto ENABLE-S3 dedica más esfuerzos, pero también está dirigido hacia otros cinco segmentos en los cuales la seguridad y confortabilidad puede aumentarse haciendo uso de sistemas inteligentes basados en software y hardware. Así mismo, muchos de los avances que ha experimentado la industria del automóvil autónomo pueden ser transferidos a estos otros dominios industriales objeto de ENABLE-S3: el sector aeroespacial, el ferroviario, el sector hospitalario, el marítimo y el de la agricultura.

La principal aportación del IUMA en este proyecto es la de dotar de tecnología hiperespectral a varias de estas aplicaciones, mejorando de esta forma su funcionalidad y

prestaciones. La tecnología hiperespectral [1] surgió hace ya varias décadas en aplicaciones de satélites de observación de la tierra. Se basa en obtener una imagen, de una misma zona espacial, en distintas longitudes de onda del espectro electromagnético. En una imagen convencional, se capta solo una pequeña fracción del espectro: la correspondiente al rango del visible. Este rango se ve ampliado con la tecnología hiperespectral a otras longitudes de onda, como puede ser el infrarrojo o el ultravioleta. A partir de estos datos se puede obtener información físico-química del objeto que se está observando, y este es el principal beneficio que nos otorga esta tecnología, no solo la de detectar objetos sino la de identificar su composición. Uno de los mayores retos a los que nos enfrentamos a la hora de trabajar con esta tecnología es que tratamos con imágenes mucho más complejas, y por lo tanto requieren de un procesamiento también más complejo.

El IUMA lleva años desarrollando esta tecnología en sus laboratorios, y en la actualidad está transfiriendo mucho del conocimiento generado hacia diversas aplicaciones reales. Así, en el proyecto europeo HELICoiD, el IUMA aplica la tecnología hiperespectral para facilitar a los neurocirujanos la detección de los bordes de un tumor en el cerebro durante una operación, de forma que resulte más sencillo determinar qué zona debe ser extraída [2][3]. En otro proyecto financiado por la Agencia Espacial Europea (ESA), el IUMA desarrolla un sistema electrónico que permite comprimir los datos de una imagen hiperespectral antes de ser enviados desde un satélite a la tierra, para conseguir de esta forma una transmisión más rápida y eficiente [4][5].

En el proyecto ENABLE-S3 el IUMA está aplicando la tecnología hiperespectral en dos sectores: el espacio y la agricultura de precisión (o agricultura inteligente). En el primer caso, se pretende mejorar los actuales sistemas de compresión de imágenes hiperespectrales basándonos en estándares aprobados por las diversas agencias internacionales del espacio (como NASA en EEUU, CNSA en China, CNES en Francia o DLR en Alemania). Además, se desarrollarán sistemas de navegación autónomos para satélites espaciales basados en el procesado de imágenes.

Este Trabajo de Fin de Máster es una aportación al desarrollo del otro caso de uso del proyecto ENABLE-S3 en el que está involucrado el IUMA, el destinado a la agricultura de precisión [6][7]. Esta área de investigación se basa en aprovechar el potencial tecnológico actual para disminuir el esfuerzo del agricultor y aumentar la productividad en los campos agrícolas. Además de las connotaciones tecnológicas derivadas de este tipo de proyecto, hay otras de tipo socio-económico y medio ambiental. En cuanto al primer factor, el socio-económico, hay que indicar que según el Banco Mundial en los últimos 50 años se ha perdido casi la mitad de la tierra cultivable por persona (en el caso de España, hemos pasado de 0,53 hectáreas por persona en 1961 a 0,26 hectáreas en 2014) [8], a lo que se une el aumento de población que en el año 2030 pasará a ser de 8.500 millones de personas. Indudablemente esto afectará en gran medida a toda aquella población que tiene una gran dependencia del sector agrícola, y todo lo que sea mejorar los recursos y aumentar la productividad de sus tierras beneficiará a su economía y a su bienestar. Por otro lado, el uso de recursos hídricos para abastecer a los campos de cultivo, así como el de pesticidas para evitar la llegada de plagas o eliminarlas, hace necesaria una estrategia que permita optimizar su utilización,

disminuyendo el impacto negativo que pueda tener sobre el medio ambiente y la salud. Los riesgos que los pesticidas tienen para la salud (sobre todo en niños, adolescentes y mujeres embarazadas), hacen que debamos afrontar este problema de forma inteligente para evitar consecuencias fatales como pueden ser el provocar cáncer o acarrear consecuencias para los sistemas reproductivo, inmunitario o nervioso.

ENABLE-S3 pretende crear una prueba de concepto encaminada a desarrollar una maquinaria agrícola de recogida de cosecha (trigo, uvas, etc.) que sea autónoma y automática, de forma que se dirija por sí sola a aquellas zonas de un campo de cultivo en el que se haya detectado previamente que el grado de madurez es óptimo para realizar la recogida, o el grado de humedad está por debajo de un límite y haya que aumentar el riego, o que comienza a producirse un cambio en la calidad de las plantas debido a la llegada de una plaga y por lo tanto haya que echar pesticidas de forma selectiva en determinadas áreas. Para ello es necesario disponer de información (en forma de coordenadas GPS) de aquellas zonas con este tipo de necesidades, transmitir dichas coordenadas a la maquinaria agrícola y hacer que la misma realice su trabajo de forma automática.

Existen varias formas de acometer este reto. Si bien en la actualidad varias empresas afrontan este problema haciendo uso de una red de sensores distribuida por los campos agrícolas, la opción que seguimos en ENABLE-S3 es la de disponer de un único sensor hiperespectral embarcado en un dron que realice vuelos frecuentes a lo largo de la zona de cultivo. Para realizar la captura de las imágenes hiperespectrales, el IUMA dispone en sus laboratorios de varias cámaras, una de las cuales tienen unas características específicas para ser embarcada en un dron, debido a su pequeño volumen y su bajo peso: la FX10 de Specim (www.specim.fi). Esta cámara dispone de 224 bandas que van desde los 400 nm hasta los 1000 nm (el rango del visible está entre los 400 y los 700 nm), y tiene un peso de solo 1.4 Kg. Conjuntamente con esta cámara se ha adquirido un dron de altas prestaciones, el Matrice 600 de la empresa Dji (www.Dji.com), capaz de transportar hasta un máximo de 6 Kg durante un tiempo máximo de 20 minutos. Ambas infraestructuras se muestran en la Figura 1.

Figura 1. Dji Matrice 600 y cámara hiperspectral Specim FX10

A la hora de afrontar las tareas que conlleva la agricultura de precisión, en una primera fase se busca delimitar una zona en la que se quiera hacer el análisis, para posteriormente poner a volar el dron capturando imágenes, las cuales una vez procesadas permitan extraer información relativa a distintas características de la cosecha (madurez, enfermedades, humedad, etc.) y transmitir esa información al agricultor y/o a la maquinaria agrícola. Una vez que la maquinaria agrícola recibe las coordenadas a las que debe dirigirse, se inicia el movimiento de la misma de forma automática, y es en esta fase donde existe el riesgo de que animales o personas se encuentren frente a la maquinaria agrícola, produciéndose accidentes mortales en el peor de los casos, y roturas de elementos mecánicos de la propia maquinaria. Es por ello por lo que en este TFM se propone un sistema que permita que el dron siga a la maquinaria agrícola en su movimiento (misma velocidad, pero a una determinada altura) de forma que en un futuro una cámara hiperespectral o térmica instalada en el dron pueda detectar e identificar la presencia de un objeto extraño y enviar una señal que haga que se detenga todo el sistema. El sistema de seguimiento propuesto en este TFM se basa en el posicionamiento continuado del dron basado en las coordenadas GPS de la maquinaria agrícola. Si bien es cierto que en la actualidad las aplicaciones de Dji permiten un seguimiento basado en el procesamiento de imágenes, existen situaciones en las que este proceso falla por existir condiciones atmosféricas adversas (niebla, por ejemplo) o debido al polvo que se crea en el proceso de cosecha, y es por ello por lo se ha decidido crear un sistema basado en posicionamiento GPS. Por otro lado, la falta de redes de comunicación en muchos campos agrícolas hace necesaria la utilización de tecnologías alternativas que permitan la transmisión de las coordenadas GPS desde la cosechadora hasta el dron. La solución empleada en este TFM ha sido la de hacer uso de tecnología LORA, gracias a la cual se pueden asegurar comunicaciones de varios kilómetros en espacios abiertos.

1.2. OBJETIVOS

Conforme a los antecedentes explicados anteriormente, y los objetivos principales del proyecto europeo ENABLE-S3, en este proyecto se desarrollará un sistema capaz de posicionar un dron de la empresa Dji, sobre un vehículo agrícola que estará continuamente en movimiento a partir de unas coordenadas GPS dinámicas que serán dadas por un sistema de posicionamiento global. La lectura de las coordenadas GPS se leerá a través de un microcontrolador conectado con dicho sistema de posicionamiento global, encargado de formar la trama que se transmitirá a través de un canal inalámbrico hacia un dispositivo electrónico, y a su vez este tendrá una aplicación instalada que controla el sistema de vuelo del dron para su posicionamiento sobre el vehículo agrícola. El sistema se plantea para automatizar y mejorar la recogida de las cosechas como se comentó en los antecedentes de este capítulo. Para desarrollar este sistema se plantean los siguientes objetivos:

- Para aquellas situaciones de baja visibilidad en las que un sistema de seguimiento basado en procesamiento de imágenes puede inducir a errores, se propone un sistema de seguimiento en el cual se transmitan coordenadas GPS de forma continuada desde un dispositivo móvil a un dron y este se posiciona en las mismas coordenadas a una determinada altura.
- Se proponen distintos esquemas de funcionamiento y conexión entre los dispositivos a utilizar (GPS, emisor/receptor radio LORA, bluetooth, estación base del dron) y su localización, explorando las ventajas y desventajas de cada una de estas opciones y desarrollando la que más se ajuste a una aplicación real como la que se está desarrollando para el proyecto ENABLE-S3.
- Se desarrollará una aplicación en Xcode a partir de la SDK para móviles que ha creado Dji para manejar los datos y establecer la conexión con cualquier producto tecnológico que ofrece esta empresa. En principio esta solución está diseñada para ser utilizada con un dron Phantom 4 de Dji, pero puede ser fácilmente transferida a otro dron de la familia Dji, como es el caso de Matrice 600, plataforma que será utilizada en el proyecto ENABLE-S3.
- La aplicación que se desarrolla permitirá realizar el seguimiento, la representación y el posicionamiento del dron, leer y guardar un conjunto de puntos de un mapa con su información asociada y gestionar el bluetooth para escanear, conectar o desconectar los periféricos de los alrededores.
- Se desarrollarán distintas situaciones reales mediante las cuales se puedan verificar las prestaciones del sistema desarrollado, tanto en la resolución del posicionamiento del dron, así como en el alcance y calidad de la señal mediante los dispositivos de radio LORA.

1.3. ESTRUCTURA

Para conseguir los objetivos presentados en el apartado 1.2, se seguirá la siguiente estructura:

En el Capítulo 1 se hace una breve descripción de los objetivos principales que propone el proyecto europeo ENABLE-S3 en la agricultura inteligente, así como los problemas que se

presentan cuando los agricultores tienen que recoger sus cosechas, la automatización del proceso de recogida, y los sistemas que se plantean para el desarrollo de este trabajo.

En el Capítulo 2 se presentan todas las definiciones, esquemas, documentación, y explicación del hardware analizado para desarrollar un sistema que cumpla con todos los objetivos planteados.

En el Capítulo 3 se formulan tres esquemas para realizar el sistema, justificando cuál es el más adecuado para el proyecto europeo ENABLE-S3 y que cumplimenta todos los objetivos que se proponen en este trabajo. Una vez determinado el esquema que mejor compete al proyecto se explican cada una de las partes que componen todo el sistema, cómo se han realizado las comunicaciones, las conexiones y montaje, así como un manual de uso para el usuario.

En el Capítulo 4 se presentan todos los resultados y conclusiones de las pruebas realizadas al sistema para corroborar que se cumple con los objetivos propuestos.

Por último, en el Capítulo 5, se comentan líneas futuras que se formulan a partir de este proyecto.

Capítulo 2

DEFINICIONES Y HERRAMIENTAS EMPLEADAS

2.1. INTRODUCCIÓN

Como se ha comentado en el Capítulo 1, el sistema que se va a desarrollar está compuesto por diferentes componentes hardware que se explicarán detalladamente en este capítulo. En concreto, se ha adquirido un GPS, un microcontrolador y sistemas de comunicación inalámbrico. Además, se ha hecho uso de diversas herramientas software de Dji.

2.2. SISTEMA DE POSICIONAMIENTO GLOBAL

El sistema de posicionamiento global que se ha decidido utilizar es el RPI GPS ADD-ON V2.0 (módulo GPS personalizado para la RPI, versión 2.0) basado en el chip NEO-6, fabricado por la empresa u-blox, ya que incorpora las últimas tecnologías en posicionamiento GPS [9][10].

2.2.1. Características principales

El chip tiene un arranque en frío de 26-32 segundos cuando no conoce la posición y tiene una sensibilidad de -160dBm a -162dBm. Este parámetro indica el nivel de señal mínima que se necesita para detectar una señal y cuanto más negativo es, mejor es la recepción. Además, está preparado para trabajar con frecuencias L1, dispone de 50 canales (máxima cantidad de satélites que puede rastrear), y es capaz de captar los satélites del WAAS, EGNOS y MSAS. Este módulo se puede implementar en cualquier microcontrolador que disponga de puerto serie como la RPI, el Arduino Mega, el ATmega32u4 con radio LORA, etc. Esta decisión se explica en el Capítulo 3, pero a continuación se detalla la documentación relacionada con el GPS [11].

El chip NEO-6 tiene una precisión horizontal de 2.5m cuando solamente capta los satélites de posicionamiento GPS, pero cuando se comunica con el resto de satélites WAAS, EGNOS o MSAS es capaz de incrementar su precisión a menos de un 1m gracias a la nueva arquitectura definida por la empresa u-blox. Para este tipo de aplicaciones es uno de los mejores porque permite conocer las coordenadas GPS con precisión y cumple con los objetivos planteados.

En la Figura 2 se presenta el sistema de posicionamiento global. Este módulo dispone de una antena externa, un hardware para manejar tarjetas micro-SD, una adaptación para integrarse sobre la RPI y un regulador de tensión de 3.3V para alimentar todo el circuito.

Figura 2. Modulo RPI GPS ADD-ON V2.0

Los pines GPIO de este módulo solo están disponibles cuando se utiliza con la RPI porque hacen de puente cuando se coloca sobre la misma. En la Tabla 1 se muestra el esquema de pines disponibles sin tener en cuenta los pines GPIO.

Tabla 1. Pines disponibles en el RPI GPS ADD-ON V2.0

Pin / Pines	
6, 9, 14, 20, 25, 30, 34 y 39	GND
1 y 17	3.3V
2 y 4	5V
3	SDA
5	SCL
8	GPS_DIN (RX)
10	GPS_DOUT (TX)
19	SPI_MOSI
21	SPI_MISO
23	SPI_SCK
24	SPI_CE0 / SD_CS
26	SPI_CE1
27	ID_SD
28	ID_SC

2.2.2. Protocolos de salida

Los protocolos de salida del chip NEO-6 se especifican en la Tabla 2. Este chip dispone de unos pines que definen el protocolo de salida, y en este caso, son los pines SPI_MOSI (pin 19) y SPI_MISO (pin 21). Por defecto, el módulo arranca con el protocolo NMEA en ASCII cuando se

reciben datos por el puerto serie, y se aprovecha esta configuración para obtener la trama de datos del GPS realizando una conexión directa a través del puerto UART a una velocidad de 9600 baudios.

Tabla 2. Protocolos de salida del u-blox NEO-6

Protocolo	Tipo
NMEA	E/S, ASCII, 0183, 2.3 (compatible con 3.0)
UBX	E/S, binario, propio del u-blox
RTCM	E/S, 2.3

El estándar NMEA se compone de frases que definen tipos de datos como, por ejemplo, el GPGSV (satélites que están disponibles), GPRMC (datos específicos del GPS y su posicionamiento), GPGSA (satélites activos), GPGGA (datos del sistema de posición global), GPGLL (posición geográfica, latitud, longitud y tiempo) y GPVTG (velocidad) [12]. Para obtener las coordenadas GPS, así como otros parámetros, se decide leer solamente la trama el GPRMC porque esta nos proporciona todos los datos necesarios para la aplicación (fecha, hora, latitud, longitud, etc.).

2.3. MICROCONTROLADORES

2.3.1. Arduino Mega 2560

El microcontrolador ATmega2560 [13] se ilustra en la Figura 3, diseñado por Arduino, dispone de multitud de pines y conexiones para realizar todo tipo de tareas. También, contiene diferentes accesorios que se pueden utilizar para programar y diseñar una gran variedad de circuitos según el proyecto que se desee realizar, así como, sensores de humedad, sensores de temperatura, leds, servos, relés, infrarrojos, potenciómetros, etc.

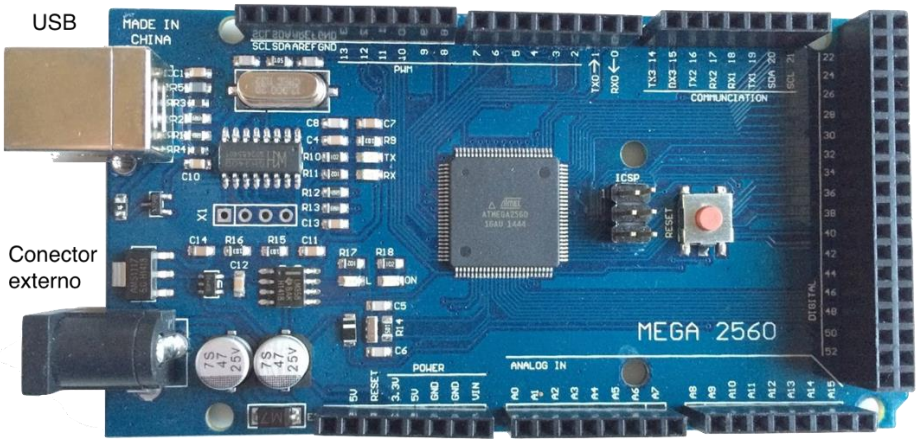

Figura 3. Arduino Mega 2560

2.3.1.1. Características principales del Arduino Mega 2560

El esquemático del Arduino Mega 2560 se representa en la Figura 4. Este módulo incorpora un chip FTDI FT232RL que permite realizar vía USB la comunicación con el IDE de Arduino en un sistema operativo de Windows o macOS.

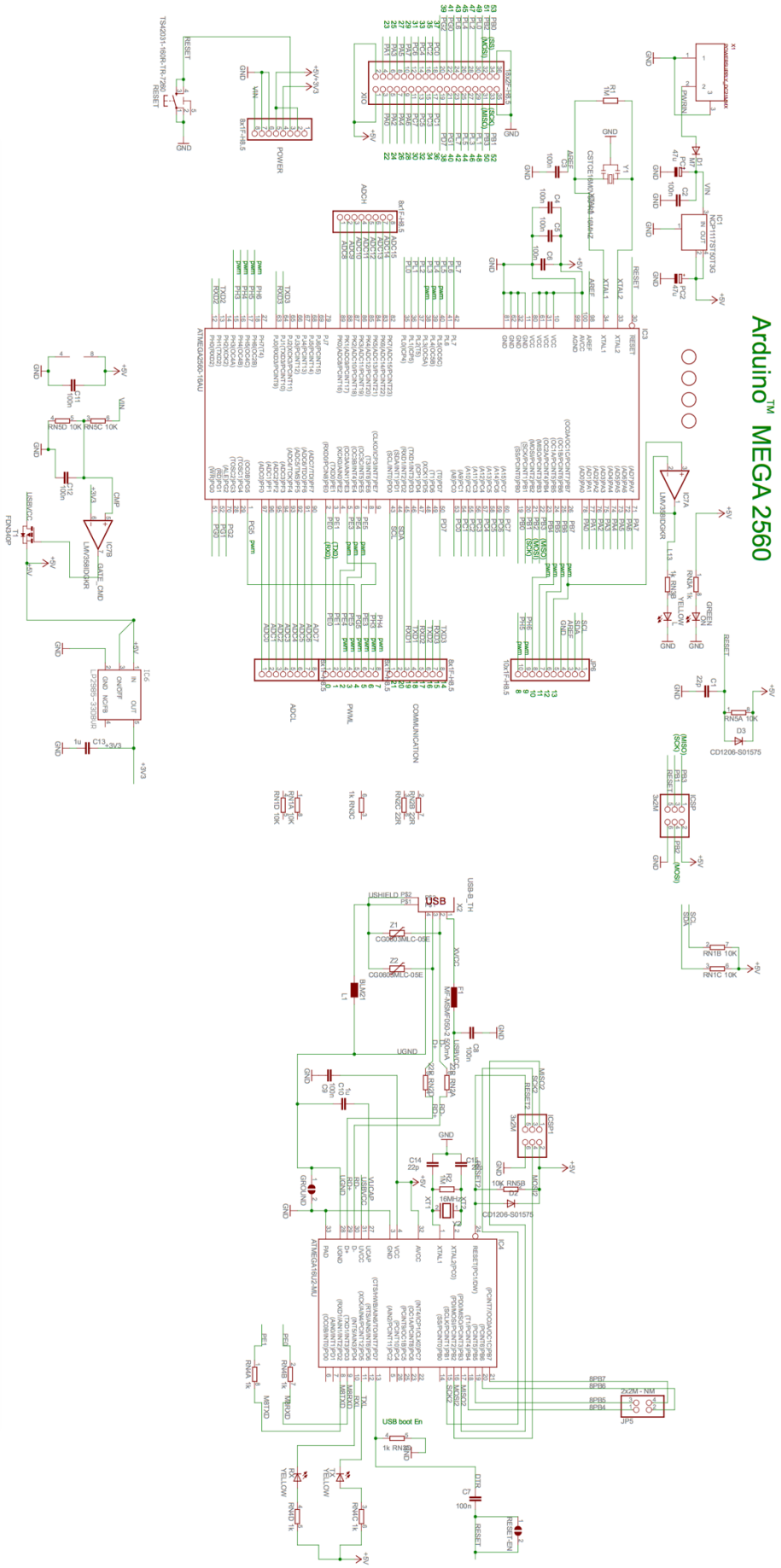

Figura 4. Esquemático del Arduino Mega 2560

A pesar de que dispone solamente de 4 puertos series, se puede aumentar el número de puertos utilizando un par de pines llegando a un máximo de 20 puertos UART gracias a la librería desarrollada por Arduino y denominada “SoftwareSerial”.

Otras de las características del Arduino Mega 2560 es su regulador de tensión, que soporta una tensión máxima de 20V, aunque su fabricante recomienda un máximo de 12V. El microcontrolador ATmega2560 dispone de una memoria flash de 128KB, una SRAM de 8KB y una EEPROM de 4KB para ejecutar los códigos de programación. Estas especificaciones técnicas, así como el resto de información se documentan en la Tabla 3.

Tabla 3. Especificaciones técnicas del Arduino ATmega2560

Regulador de tensión	Voltaje de operación de 5V Recomendado de 7-12V Máximo de 6-20V
Puerto serie	4
Otros buses de comunicación	SPI, I2C
Memorias	FLASH 256KB SRAM 8KB EEPROM 4KB
Entradas y salidas digitales (E/S)	54, de las cuales 15 pines disponen de un modulador de ancho por pulsos en sus salidas (PWM)
Oscilador de cristal	16MHz
Corriente DC por cada pin de E/S	20mA
Corriente de salida a 3.3V por cada pin de E/S	50mA
Entradas analógicas	16
Dimensiones	101.52x53.3mm
Peso	37 gramos

El Arduino Mega 2560 se puede alimentar a través de un conector de alimentación externo con una pila de 9V o un adaptador de corriente, respetando los límites de tensión especificados por el regulador de tensión.

2.3.1.2. Software de programación del ATmega2560

Para programar el microcontrolador ATmega2560 se utiliza un entorno de programación desarrollado por la misma empresa y que recibe el nombre de Arduino IDE [14] como se ilustra en la Figura 5. Este entorno permite realizar la programación de todos los microcontroladores de esta empresa, y a su vez, incorpora un monitor serie con el que podemos interactuar para conocer datos que se transmitan o se reciban a través del puerto serie.

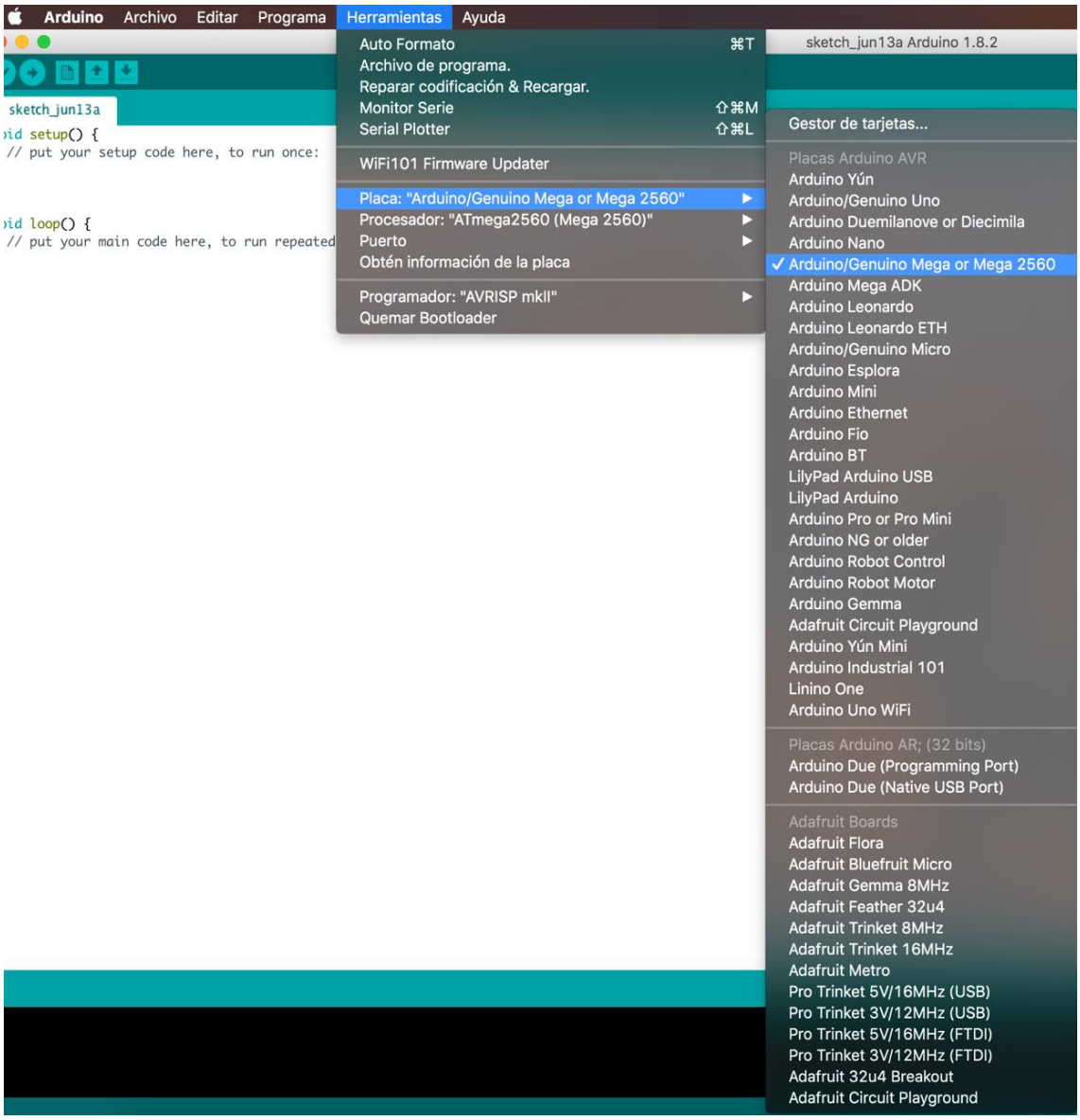

Figura 5. Selección de la placa en el Arduino IDE

En la Figura 5, se aprecia una primera vista del Arduino IDE donde se observa cómo se elige el hardware, pues permite programar multitud de placas de Arduino, e incluso añadir nuevos hardware que no se encuentren en la lista. Además, es necesario definir el puerto serie donde se encuentra el microcontrolador y el procesador que incorpora.

2.3.2. Microcontrolador ATmega32u4 con radio LORA

La tecnología LORA permite trabajar con redes de largo alcance y bajo consumo. En este caso, se estudia el microcontrolador ATmega32u4 con radio LORA representado en la Figura 6. Este módulo funciona con frecuencias inferiores al WiFi, en concreto, entre 868MHz y 915MHz. Esta versión dispone de un microcontrolador que trabaja a una frecuencia de reloj de 8MHz y una radio LORA que alcanza los 2 kilómetros con antenas estándar, aunque puede alcanzar distancias de 15 a 20 kilómetros tanto en la emisión como en la recepción si se utilizan antenas bidireccionales, de gran potencia, gran tamaño, y adaptadas para las frecuencias del módulo [15].

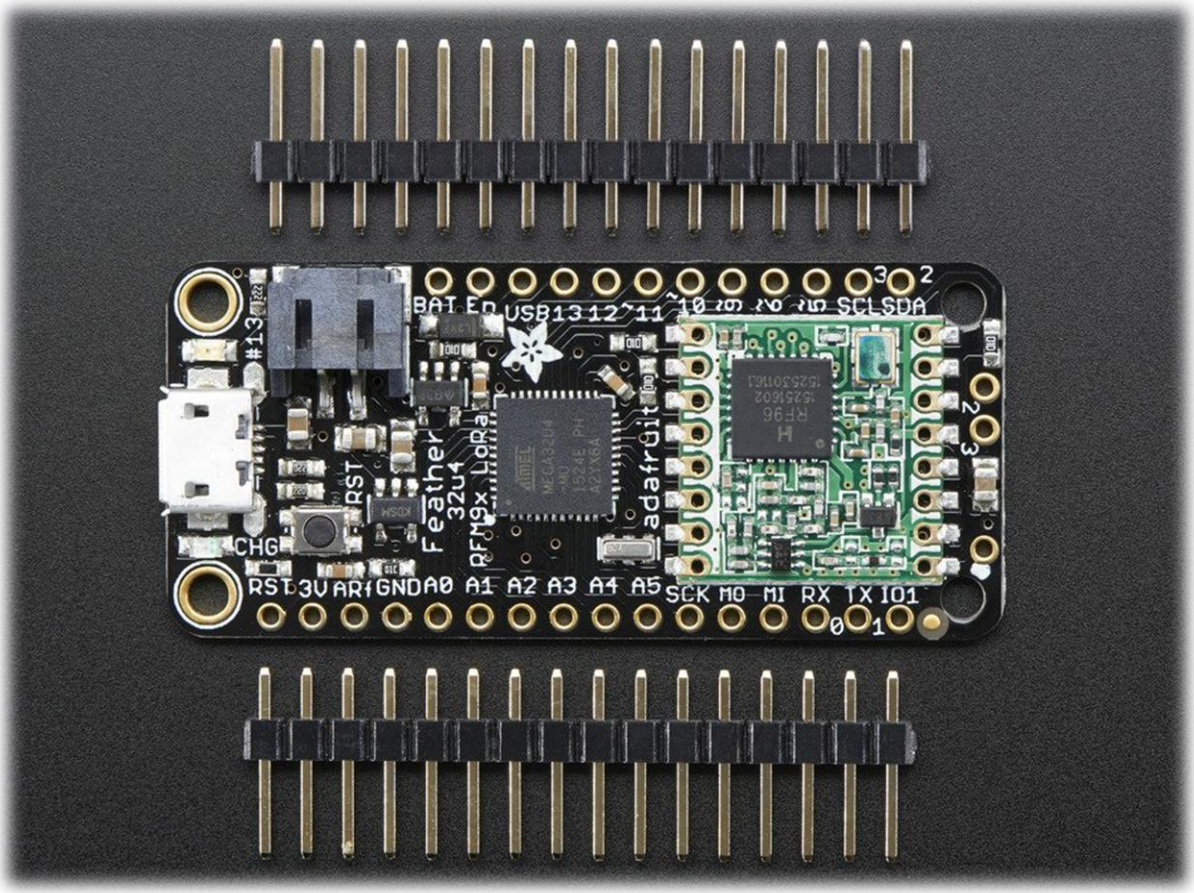

Figura 6. Microcontrolador ATmega32u4 con un módulo de radio LORA

2.3.2.1. Características principales del microcontrolador ATmega32u4 con radio LORA

El microcontrolador que incorpora este circuito se desarrolló por Adafruit, se denomina ATmega32u4 y su programación es igual a la del Arduino. Este chip contiene una memoria flash de 32KB y una memoria RAM de 2KB para ejecutar los códigos de programación. Además, en las especificaciones técnicas del fabricante se documenta que el microcontrolador ATmega32u4 con radio LORA realiza las comunicaciones directamente con el ordenador a través del micro-USB y, por tanto, solamente cuenta con un único puerto serie.

Este diseño incorpora un conector externo para alimentar el circuito mediante una batería de litio de 3.7V. Esta batería está conectada al regulador de tensión de 3.3V y su diseño permite activarlo o desactivarlo a través de un pin denominado *EN*, ubicado a la derecha del conector de alimentación externo. Este dispositivo se puede alimentar a través del micro-USB a 5V o mediante el conector de batería mencionado. Además, dispone de un led que indica cuando se está cargando la batería de litio y un botón para reiniciar el microcontrolador ATmega32u4 con radio LORA.

A continuación, en la Figura 7 se observa el esquema de conexiones del microcontrolador ATmega32u4 con radio LORA, diseñado y proporcionado por la empresa de Adafruit, y en la Tabla 4 se detallan sus especificaciones técnicas.

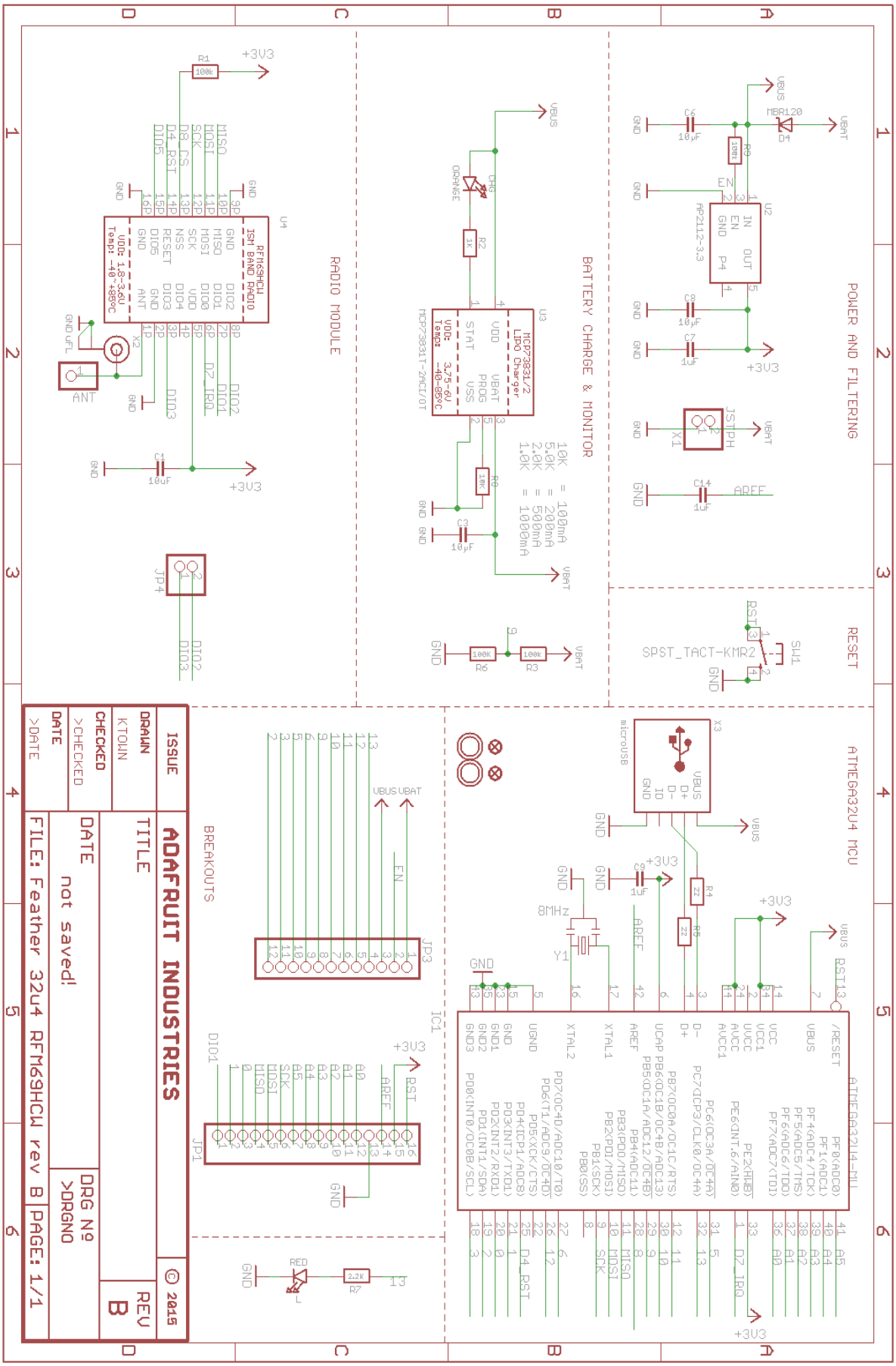

Figura 7. Esquemático del microcontrolador ATmega32u4 con radio LORA

Tabla 4. Especificaciones técnicas del microcontrolador ATmega32u4 con radio LORA

Regulador de tensión	Voltaje de operación de 3.3V Recomendado de 5V Máximo de 10V
Conectores externos	2 Conector micro-USB Conector externo para una batería de litio
Potencia de salida	Hasta 100mW (Desde +5dBm hasta +20dBm)
Consumo	Modo activo a +20dBm: 120mA Modo escucha: 40mA Modo sueño: 300uA
Puerto serie	1
Memorias	FLASH 32KB RAM 2KB
Entradas/Salidas digitales	20 GPIO, de los cuales 7 disponen de un modulador por acho de pulsos (PWM)
Otros buses de comunicación	SPI, I2C
Oscilador de cristal	8MHz
Alcance máximo	1.2/2 Kilómetros con un cable de cuarto de onda 15/20 kilómetros con antenas potentes y bidireccionales
Entradas analógicas	10
Frecuencias de funcionamiento	868 a 915MHz
Dimensiones	51x23x8mm
Peso	5.5 gramos

Con respecto al rango de alcance de las radios LORA, en los test realizados por los desarrolladores utilizando los ajustes por defecto de las bibliotecas de programación se alcanzan distancias de 1.2-2 kilómetros al enviar y recibir tramas de datos con una antena básica medida a un cuarto de onda o antenas estándares como las que se utiliza para el WIFI. Sin embargo, con antenas bidireccionales, unos ajustes en la biblioteca de programación y en campo abierto, se pueden alcanzar los 20 kilómetros de distancia.

A continuación, se muestra en la Figura 8 el diseño del microcontrolador ATmega32u4 con radio LORA donde aparecen todos los pines disponibles, así como los tipos de conectores que se necesitan.

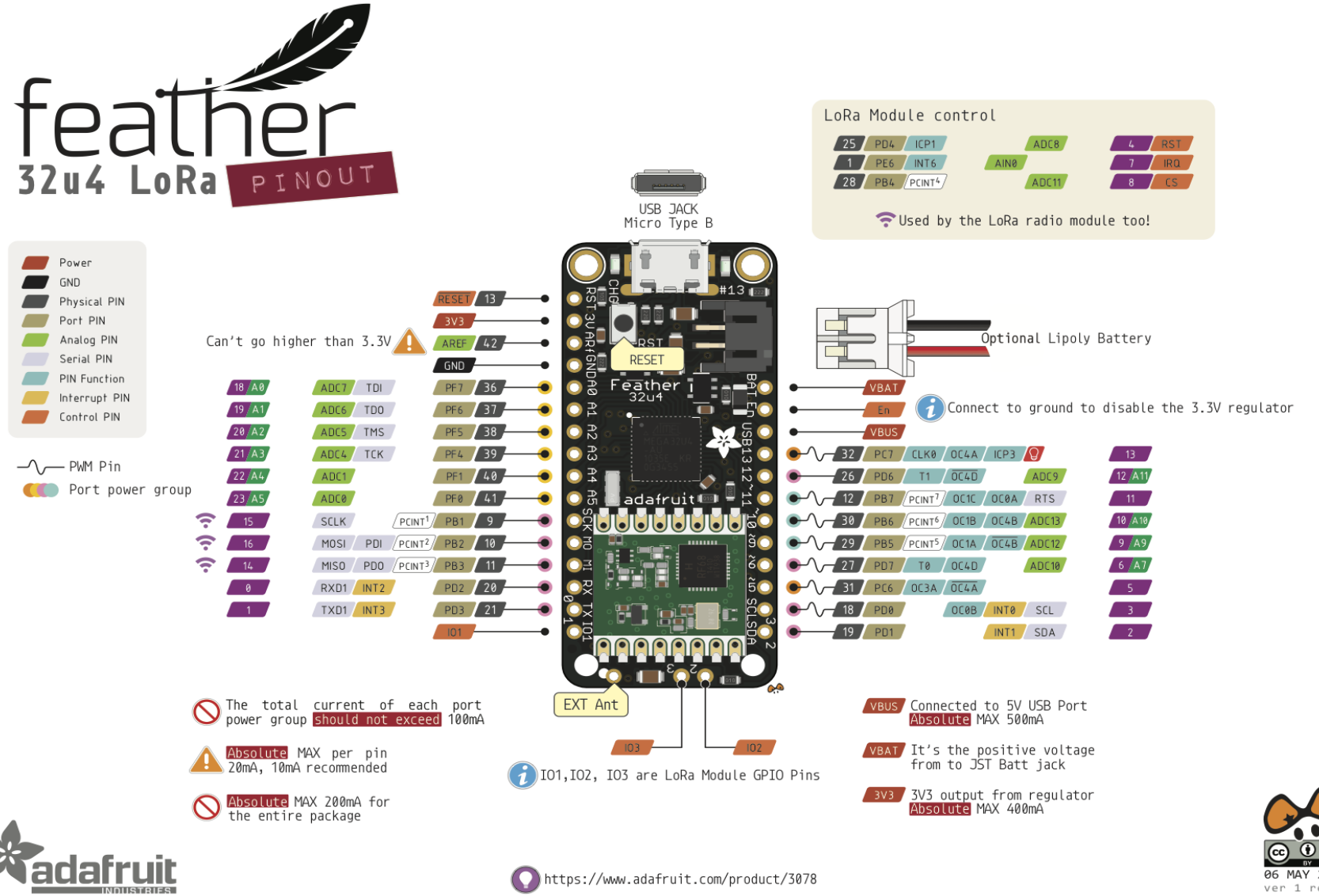

Figura 8. Esquema de pines del microcontrolador ATmega32u4 con radio LORA

2.3.2.2. Software de programación del microcontrolador ATmega32u4 con radio LORA

El entorno de programación que se utiliza para el ATmega32u4 con radio LORA es el mismo que el Arduino mega mencionado anteriormente en el apartado 2.3.1.2. Este entorno permite realizar las comunicaciones y descargar los códigos de programación en el microcontrolador ATmega32u4, pero antes hay que seguir una serie de pasos para instalar las tarjetas de Adafruit en el IDE de Arduino.

Una vez terminado todos los pasos anteriores, podemos empezar la programación del módulo. Para ello, existen algunos ejemplos en la página web que explican cómo se realizan las comunicaciones y uno de estos ejemplos se representa en la Figura 9. En este Trabajo Fin de Máter se partirá de este código como base, realizándose posteriormente las modificaciones pertinentes para adaptarlo a los objetivos del trabajo.

```
void setup()
{
  pinMode(LED, OUTPUT);
  pinMode(RFM95_RST, OUTPUT);
  digitalWrite(RFM95_RST, HIGH);

  while (!Serial);
  Serial.begin(9600);
  delay(100);

  Serial.println("Feather LoRa RX Test!");

  // manual reset
  digitalWrite(RFM95_RST, LOW);
  delay(10);
  digitalWrite(RFM95_RST, HIGH);
  delay(10);

  while (!rf95.init()) {
    Serial.println("LoRa radio init failed");
    while (1);
  }
  Serial.println("LoRa radio init OK!");
  if (!rf95.setFrequency(RF95_FREQ)) {
    Serial.println("setFrequency failed");
    while (1);
  }
  Serial.print("Set Freq to: ");
  Serial.println(RF95_FREQ);
}
```

```
void loop()
{
  if (rf95.available())
  {
    // Should be a message for us now
    uint8_t buf[RH_RF95_MAX_MESSAGE_LEN];
    uint8_t len = sizeof(buf);

    if (rf95.recv(buf, &len))
    {
      digitalWrite(LED, HIGH);
      RH_RF95::printBuffer("Received: ", buf, len);
      Serial.print("Got: ");
      Serial.println((char*)buf);
       Serial.print("RSSI: ");
      Serial.println(rf95.lastRssi(), DEC);
      delay(10);
      // Send a reply
      uint8_t data[] = "And hello back to you";
      rf95.send(data, sizeof(data));
      rf95.waitPacketSent();
      Serial.println("Sent a reply");
      digitalWrite(LED, LOW);
    }
    else
    {
      Serial.println("Receive failed");
    }
  }
}
```

Figura 9. Código de ejemplo del ATmega32u4 con radio LORA

2.4. SISTEMA DE COMUNICACIÓN INALÁMBRICO

El sistema de comunicaciones inalámbrico utilizado en este proyecto es el módulo Bluetooth HM-10 [16] que se muestra en la Figura 10 porque permite realizar las comunicaciones mediante un puerto serie a partir de un lenguaje de alto nivel, en concreto, los comandos AT. Este hardware fue desarrollado por los fabricantes para establecer comunicación con los nuevos dispositivos electrónicos del mercado que incorporan una versión más reciente, en concreto, el Bluetooth 4.0 en adelante. Los móviles de última generación incorporan versiones superiores, pero con este hardware se puede establecer comunicación con los mismos porque pertenecen a la versión 4.x.

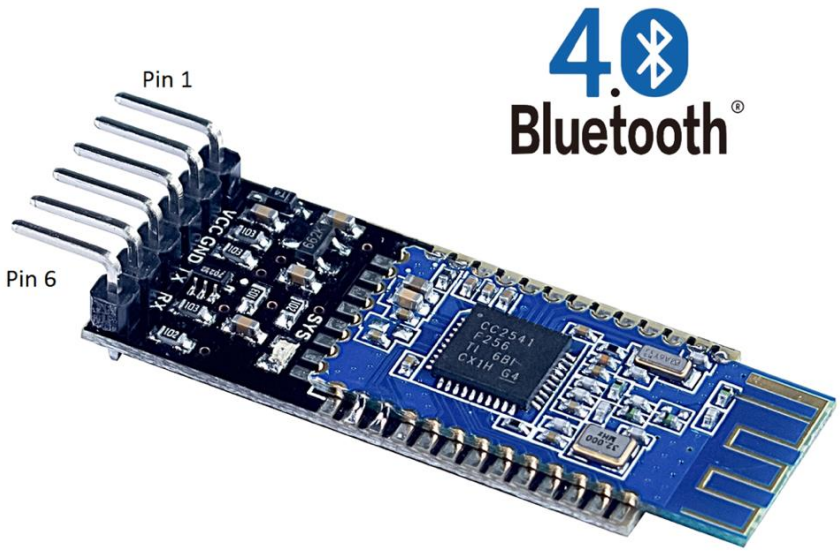

Figura 10. Bluetooth HM-10

El Bluetooth HM-10 transmite y recibe la información en una frecuencia de trabajo de 2.4GHz en la banda ISM y tiene una modulación GFSK (Gaussian Frequency Shift Keying). Además, el alcance máximo es de 100 metros según los test realizados con un Iphone 4S. El microcontrolador que incorpora este modelo se denomina CoreBlue CSR o TI CC2540.

2.4.1. Características principales

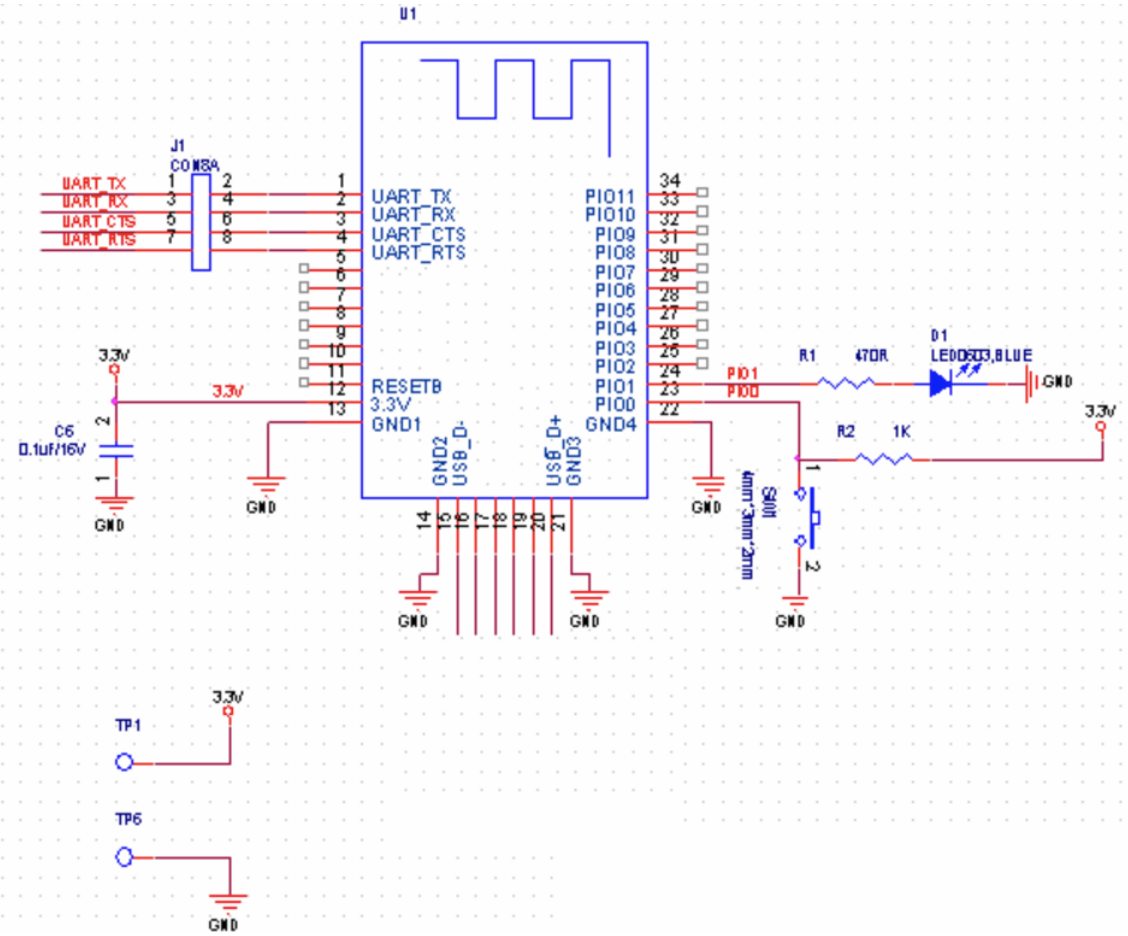

Figura 11. Esquemático del Bluetooth HM-10

El esquema de conexiones publicado por el fabricante se muestra en la Figura 11 y se denomina Bluetooth HM-10. Este chip no puede maniobrar por sí solo y no incluye un regulador de tensión para alimentar el circuito de manera segura con la batería de 5V que se utilizará para el sistema. Por tanto, se usa el circuito impreso que se representa en la Figura 10 porque el chip está insertado en un nuevo diseño que se puede manejar con facilidad e incorpora un regulador de tensión para alimentar el Bluetooth HM-10. El regulador de tensión es de 3.3V y permite alimentar el circuito dentro del intervalo de tensión de 3.6-6V. Además, es posible transmitir y recibir información sin límite porque el software que integra envía la información divida en paquetes de 20 en 20 bytes. A continuación, se detallan las especificaciones técnicas del Bluetooth HM-10 en la Tabla 5.

Tabla 5. Especificaciones técnicas del Bluetooth HM-10

Regulador de tensión	Voltaje de operación: 3.3V
Consumo	Modo sueño: 400uA hasta 1.5mA Modo activo: 8.5mA Máximo: 50mA
Puerto serie	1
Alcance	100 metros
Temperatura de trabajo	De -5°C hasta + 65°C
Entradas/Salidas digitales	20 GPIO, de los cuales 7 disponen de un modulador por acho de pulsos (PWM)
Velocidad	Asíncrona 6KB Síncrona 6KB
Seguridad	Autenticación y encriptación
Potencias de salidas	-23dBm, -6dBm, 0dBm y 6dBm
Frecuencia de funcionamiento	2.4GHz en la banda ISM
Dimensiones	26.9x13x2.2mm

Por otro lado, este modelo cuenta únicamente con 6 pines y corresponden con la alimentación y el puerto serie del mismo como se especifica en la Tabla 6. Para este proyecto no son necesarios el Pin 1 y el Pin 6. Este módulo se programa utilizando los comandos AT especificados en el Anexo I.

Tabla 6. Pines del Bluetooth HM-10 representado en la Figura 10

Pin 1	STATE
Pin 2	VCC – Tensión 3.6-6V
Pin 3	GND
Pin 4	TXD – Tensión 3.3-5V
Pin 5	RXD – Tensión 3.3-5V
Pin 6	EN

2.5. HERRAMIENTAS DE DJI

2.5.1. Introducción

Dji ha creado una comunidad de desarrolladores que permiten realizar proyectos con drones en las plataformas de IOS o Android a partir de sus SDK [17]. Las SDK disponibles son:

- El Mobile SDK tiene como objetivo dar al usuario un gran abanico de posibilidades para aprender a desarrollar todo tipo de aplicaciones en IOS o Android a partir de una serie de conocimientos y librerías que proporciona la tecnología de Dji.
- El Onboard SDK le permite al usuario manejar los productos de Dji a partir de los puertos UART y USB, realizando una comunicación serie directa con el controlador de vuelo. Las plataformas compatibles con los controlados de Dji son Windows, Linux o sistemas embebidos con ARM.
- El Guidance SDK da al usuario la posibilidad de desarrollar fácilmente todo tipo de aplicaciones basadas en la visión a partir de la concesión de un control total de su orientación.

En este proyecto utilizaremos el Mobile SDK porque es un kit de desarrollo que permite a los programadores acceder a un mayor abanico de productos de Dji. La SDK que se utiliza simplifica el proceso de desarrollo de las aplicaciones a bajo nivel tales como la estabilización del vuelo, la gestión de la batería, transmisión de las señales y las comunicaciones. Gracias a esta tecnología, los desarrolladores pueden centrarse en la automatización del vuelo del dron, el control de la cámara, realizar misiones, recoger datos de sensores en tiempo real y monitorizar el estado de otros componentes.

Figura 12. Esquema de comunicaciones del Mobile SDK

Las comunicaciones que se realizan entre el producto de Dji y su mando remoto son automáticas. El protocolo de comunicaciones está basado en el Dji Lightbridge, una tecnología capaz de transmitir grandes cantidades de información a grandes distancias. Por ejemplo, el producto de Dji que se ilustra en la Figura 12, el Phantom 4, ofrece una transmisión de video

en vivo y HD en cualquier dispositivo móvil hasta una distancia máxima de 3 millas (4.8 kilómetros aproximadamente).

Para comunicar el IPad con el mando y realizar una correcta comunicación con los productos de Dji hay que seguir los pasos del diagrama de la Figura 13. La aplicación móvil está construida a partir del Mobile SDK, la plataforma SDK (iOS o Android) y se ejecuta en un dispositivo móvil.

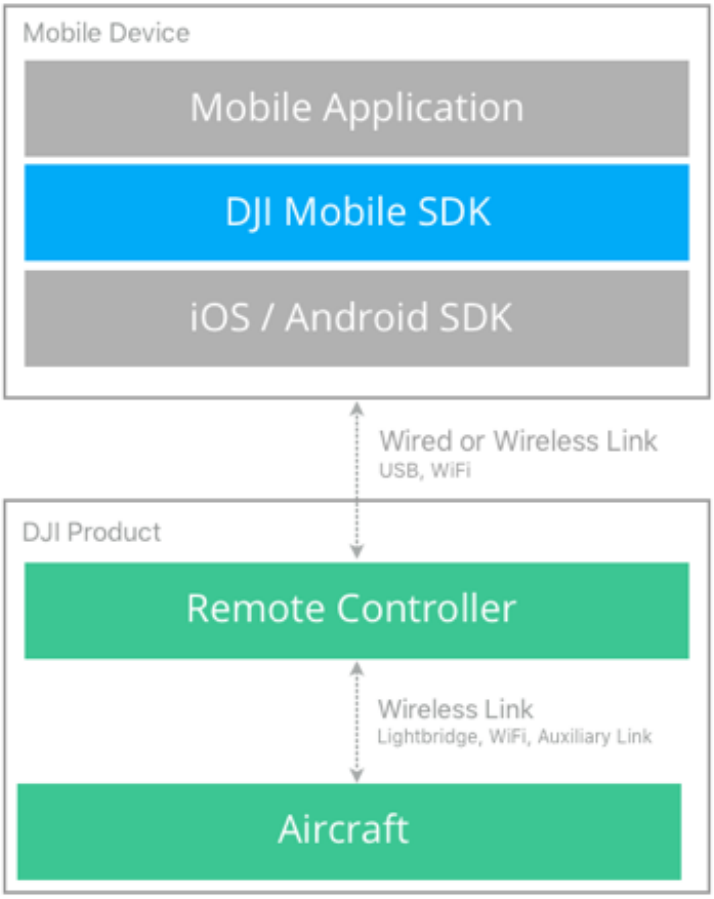

Figura 13. Mobile SDK en un dispositivo móvil y su conexión con los productos de Dji

2.5.2. Simulador Dji Assistant

El simulador fue desarrollado por la empresa Dji y se denomina Dji Assistant. Este simulador permite verificar y validar las aplicaciones en el dispositivo móvil sin necesidad de salir con el dron al exterior y ponerlo en vuelo, corregir errores que surjan durante el vuelo y fallos propios del programador. Para empezar a utilizar el simulador es necesario conectar el producto con el simulador como se representa en la Figura 14. Luego, instalamos la aplicación en el dispositivo móvil y lo conectamos al mando del dron mediante el cable USB.

Al abrir el simulador introducimos los parámetros básicos antes de comenzar (latitud, longitud, etc.). Después de realizar los anteriores pasos, solamente falta ejecutar la aplicación y la simulación como se muestra en la Figura 14.

Figura 14. Esquema de conexiones para iniciar el simulador Dji Assistant

2.5.3. Librerías de programación

Una biblioteca de programación es un conjunto de implementaciones funcionales, codificadas en un lenguaje de programación, que ofrece una interfaz bien definida para la funcionalidad que se invoca.

El Mobile SDK ofrece un conjunto de librerías que permiten programar todas las funcionalidades de los productos de Dji y se descargan a través de la página oficial Dji Developer [17]. Estas librerías permiten realizar misiones predefinidas durante el vuelo del dron. Las misiones se utilizan para facilitar la automatización del vuelo y permiten realizar multitud de actividad de una manera eficaz. Las misiones disponibles en el Mobile SDK son las siguientes:

- Dji WayPoint se utiliza para volar hacia una serie de ubicaciones predefinidas. Para ello, se necesita conocer la latitud, la longitud, la altitud y el rumbo de la aeronave en cada punto.
- Dji HotPoint es una misión que permite girar el dron alrededor de una coordenada concreta con un radio específico. Hace falta definir la altura del vuelo del dron.
- Dji FollowMe hace el seguimiento de un objeto móvil mediante unas coordenadas GPS dinámicas. Es necesario decirle si queremos que se oriente hacia el mando o hacia las coordenadas antes de iniciar la misión.
- Dji ActiveTrack permite realiza un seguimiento de un objeto móvil a partir del procesamiento de imágenes, sin necesidad de utilizar un GPS.
- Dji TapFly hace que la aeronave vuele hacia un punto definido en una secuencia de video en directo. El usuario define dicho punto en el dispositivo móvil.

- Dji Panorama gira la cámara 180 o 360 grados para realizar una serie de fotos durante el vuelo. Si la cámara se gira 180 grados se capturan 5 fotos y cuando la cámara se gira 360 grados se toman 8 fotos. Estas imágenes se guardan automáticamente en la micro-SD y el usuario puede extráelas para generar una foto panorámica.
- Dji Custom es una misión personalizada que permite ejecutar varias misiones en un orden predefinido. Además, incorpora otro tipo de misiones secundarias que solamente se pueden iniciar usando el Dji TimeLine:
  - Dji GoHome (volver a casa).
  - Dji GoToPoint (ir hacia un punto).
  - Dji GoToAltitude (elevarse o descender).
  - Dji ShootPhoto (capturar una foto).
  - Dji RecordVideo (grabar un video).
  - Dji AircraftTakeOff (despegar el dron).
  - Dji ChangeAircraftYaw (cambiar la orientación)
  - Dji ChangeGimbalAttitude (modificar la orientación del gimbal).

Para este Trabajo Fin de Máster se tomarán como ejemplos algunos de los códigos de programación que aporta la SDK, los cuales serán modificados posteriormente para adaptarlos a nuestros objetivos. Esto permitirá entender cómo funcionan las librerías de la SDK y el camino que se debe de seguir a la hora de realizar la programación. En la Figura 15 se muestra un ejemplo de cómo se inicia y finaliza la misión de Dji FollowMe en Objective-C con la versión SDK 4.0.

```
- (IBAction)onStartButtonClicked:(id)sender
{
    WeakRef(target);
    DJIFollowMeMission* mission = (DJIFollowMeMission*)[self initializeMission];
    [self.followMeOperator startMission:mission withCompletion:^(NSError * _Nullable error) {
        if (error) {
            ShowResult(@"Start Mission Failed:%@", error);
        } else {
            [target missionDidStart:error];
        }
    }];

    [self.followMeOperator addListenerToEvents:self withQueue:nil andBlock:^
        (DJIFollowMeMissionEvent * _Nonnull event) {
        [target onReciviedFollowMeEvent:event];
    }];
}

- (IBAction)onStopButtonClicked:(id)sender
{
    WeakRef(target);
    [self.followMeOperator stopMissionWithCompletion:^(NSError * _Nullable error) {
        if (error) {
            ShowResult(@"Stop Mission Failed:%@", error.description);
        } else {
            [target startUpdateTimer];
            [target.followMeOperator removeListener:self];
        }
    }];
}
```

Figura 15. Programación del inicio y parada de la misión Dji FollowMe en Objective-C

Para compilar las librerías de Dji es necesario instalar el Dji SDK CocoaPods. Por tanto, se abre la consola en el macOS, se accede a la carpeta donde se encuentra el proyecto que se acaba de crear, se introduce los comandos que aparecen en la Figura 16, y la instalación habrá terminado cuando aparezcan los mensajes de la Figura 17.

```
sudo gem install cocoapods

pod install
```

Figura 16. Comandos para instalar los Dji SDK CocoaPods

```
Analyzing dependencies
Downloading dependencies
Installing DJI-SDK-iOS (4.1.1)
Installing DJI-UILibrary-iOS (4.1.1)
Generating Pods project
Integrating client project

[!] Please close any current Xcode sessions and use `UILibraryDemo.xcworkspace` for this project from now
Pod installation complete! There is 1 dependency from the Podfile and 1 total pod
installed.
```

Figura 17. Mensaje final de la instalación del CocoaPods

2.6. SOFTWARE DE PROGRAMACIÓN XCODE

2.6.1. Introducción

El Xcode [18] es un entorno de desarrollo integrado para macOS que contiene un conjunto de herramientas software que permite el desarrollo de software para MacOS, iOS, watchOS y TVOS. Este entorno es compatible con el código fuente de los lenguajes de programación de C, C++, Objective-C, Objective-C++, Java, AppleScript, Python, Rubí, Resedit y Swift.

El Xcode 1.0 comenzó en el otoño de 2003 con las primeras versiones y ha ido evolucionando a lo largo de los años. El 13 de junio de 2016 se lanzó el Xcode 8.0 y actualmente se utiliza las versiones 8.x.

El entorno Xcode 8.3 dispone de 6 partes como se puede observar en la Figura 18:

- **Área de navegación**. En este panel existes diferentes navegadores. El primer navegador es el navegador de proyectos y como su nombre indica, permite manejar los diferentes archivos. El segundo es el buscador y se utiliza para realizar una búsqueda de fragmentos de texto del proyecto. El tercer navegador es el editor de navegación, y permite conocer los errores de compilación del código.
- **Área de edición**. En esta área es donde se introduce el código de programación. Este editor se puede dividir en dos editores utilizando las opciones de la barra de herramientas. Además, permite buscar y seleccionar los ficheros del código que se quieren tener a la vista.

- **Panel inspector**. Este panel se utiliza para mostrar los detalles del archivo resaltado en el navegador.
- **Panel de librerías**. Este panel solamente se utiliza cuando estamos buscando archivos XIB/Storyboard en el *storyboard*.
- **Área de depuración.** Muestra el estado de las diversas variables que se están utilizando en el código y permite depurarlo paso a paso.
- **Área de la consola.** La consola sirve para conocer los errores que suceden durante la ejecución de la aplicación. Además, se pueden mandar mensajes a la consola con el comando NSLog para ver el estado de variables o mensajes.
- **Barra de herramientas.** Observando la barra de herramientas de izquierda a derecha en la Figura 18, en primer lugar, tenemos un acceso directo para iniciar la ejecución de la aplicación. En segundo lugar, podemos elegir el dispositivo con el que se va a trabajar y el proyecto. En tercer lugar, se dispone de un visualizador para saber en todo momento que se está ejecutando en el Xcode. En cuarto lugar, hay una pequeña barra para manejar las diferentes ventanas del área de edición donde se puede seleccionar una sola área de edición o dos. Por último, tenemos otra pequeña barra de opciones que permiten ocultar el área del navegador, el panel de inspector y el panel de librerías, y el área de depurado junto con la consola.

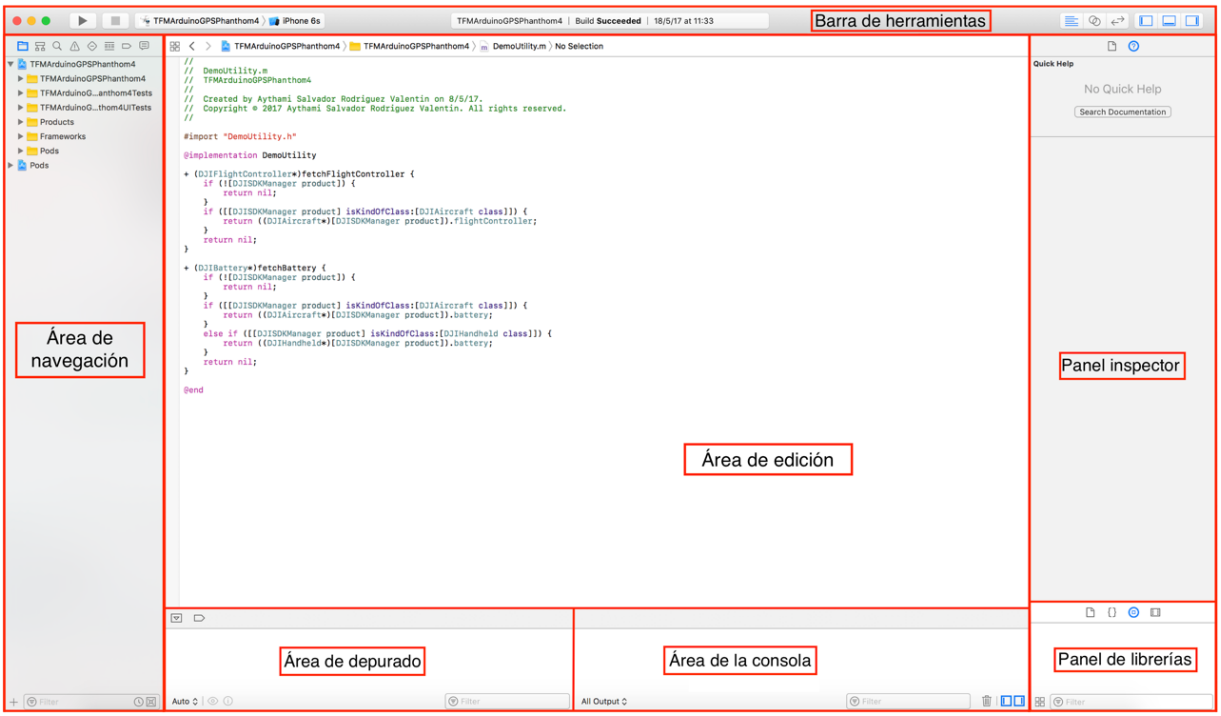

Figura 18. Entorno de programación Xcode 8.3

El Xcode 8.3 incorpora un simulador en la barra de herramientas que permite ejecutar las aplicaciones en un dispositivo virtual si no tenemos un dispositivo móvil de iOS conectado en alguno de los puertos del ordenador.

Uno de los pasos más importantes que se realizan en el Xcode cuando desarrollamos una aplicación desde cero es crear un nuevo repositorio para guardar todas las actualizaciones que

se van haciendo a lo largo de todo el proyecto. Para este caso, se genera un repositorio en GitLab a través de la cuenta de correos del IUMA para el proyecto europeo ENABLE-S3 y se realiza el control de cambios en la misma. Para definir el enlace del repositorio en el programa hay que seguir los pasos de la Figura 19.

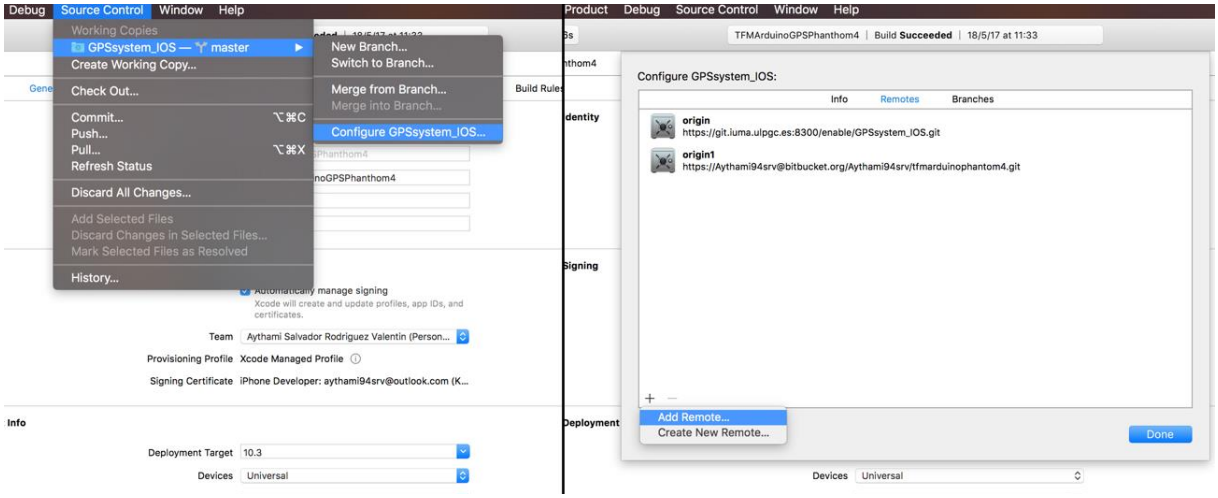

Figura 19. Pasos para agregar un repositorio en la aplicación

2.6.2. Lenguaje de programación

En este proyecto se utiliza el Xcode 8.3 para desarrollar una aplicación y el lenguaje de programación será Objective-C. Este lenguaje de programación está orientado a objetos y fue creado por un superconjunto de C para implementar un modelo de objetos parecidos al Smalltalk. Su creador fue Brad Cox y la corporación StepStone en 1980 [19].

Cuando se empieza desde cero una aplicación es importante entender cómo se implementa un patrón de diseño para utilizar las variables en cualquier parte del código cuando sea necesario. En este caso, se habla del patrón de diseño *singleton* representado en la Figura 20. Este modelo contiene una clase estática propia del modelo y una función instanciada para inicializar las variables usadas en él. Con estas líneas de código ya somos capaces de manejar estas variables en toda la aplicación de forma global.

```
#import "MiClase.h"

@implementation MiClase

#pragma mark Métodos

+ (id)sharedInstance {
    static MiClaseSingleton *sharedInstance = nil;
    static dispatch_once_t onceToken;
    dispatch_once(&amp;onceToken, ^{
        sharedInstance = [[self alloc] init];
    });
    return sharedInstance;
}

- (id)init {
  if (self = [super init]) {
      miPropiedad = [[NSString alloc] initWithString:@"Valor por defecto"];
  }
  return self;
}

@end
```

Figura 20. Patrón de diseño singleton

Capítulo 3

APORTACIONES

3.1. INTRODUCCIÓN

Para el desarrollo de este capítulo se tendrán en cuenta todos los conocimientos y objetivos descritos en el Capítulo 1 y el Capítulo 2. En este capítulo se representa tres esquemas de conexionado y se justifica de manera razonada la elección del más adecuado para esta aplicación en concreto.

Inicialmente se presenta una estructura de todo el sistema, se documentan las implementaciones en tres bloques diferenciados, y este a su vez se divide en tres partes: un primer bloque compuesto por el módulo GPS y el microcontrolador ATmega32u4 con radio LORA transmisora, un segundo bloque compuesto por el microcontrolador ATmega32u4 con radio LORA receptora y el Bluetooth HM-10, y finalmente un tercer bloque que comprende el desarrollo de la aplicación para iOS.

Para concluir el capítulo, se presenta un manual de uso que se divide en tres apartados: el primer apartado se centra en el funcionamiento del sistema, el segundo apartado explica la interfaz gráfica para el usuario y el tercer apartado representa gráficamente la misión desarrollada para el vuelo del dron en el simulador.

3.2. ESQUEMAS DE CONEXIONADO PARA EL SISTEMA

En este apartado se representan en las Figura 21, Figura 22 y Figura 23 los tres esquemas a analizar para ver cuál es el que mejor se adapta a nuestras necesidades. La diferencia entre cada uno de estos tres esquemas radica en el hecho de que los componentes están dispuestos en distintas localizaciones y que dependiendo de cada una de esas localizaciones se utilizarán unos componentes u otros.

En el primer esquema, Figura 21, el sistema está compuesto por 6 elementos. El primer elemento es el módulo GPS, RPI GPS ADD-ON V2.0, y se encarga de transmitir a través del puerto serie la trama de datos según el protocolo NMEA en formato ASCII. El segundo elemento es el Arduino Mega, el cual compone la trama de datos recibida en el puerto serie 1 donde está conectado el sistema de posicionamiento global y la transmite al puerto serie 2. El tercer elemento es el Bluetooth HM-10, que está conectado en el puerto serie 2 del Arduino Mega, y se utiliza para transmitir la información hasta el IPad. El cuarto elemento es el IPad, y en él, se instalará una aplicación que se desarrolla haciendo uso del Mobile SDK. El quinto y

sexto elemento son el mando y el dron, pues el siguiente paso consiste en conectar el IPad con el mando del dron para ejecutar la aplicación desarrollada con la SDK de Dji, y así posicionar el dron sobre las coordenadas que correspondan. Para finalizar el primer esquema se comenta que todos los elementos nombrados excepto el dron se incorporan dentro del camión agrícola.

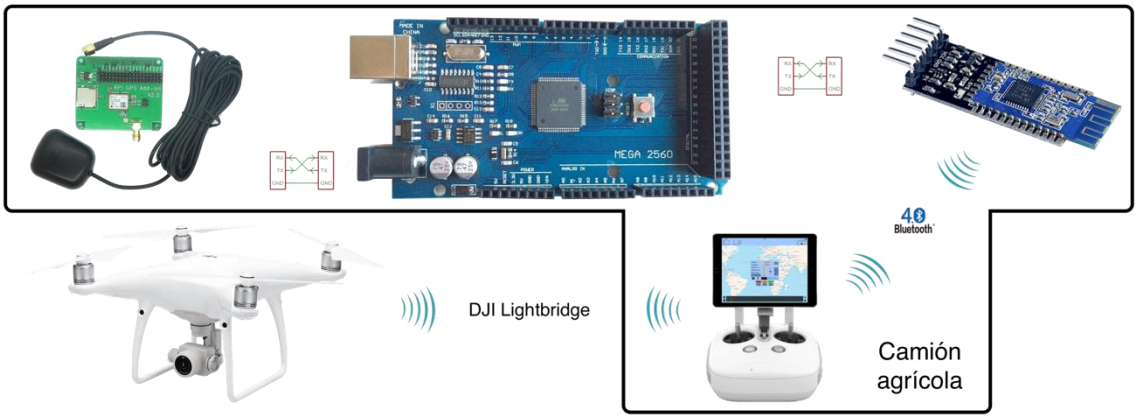

Figura 21. Esquema 1 (RPI GPS ADD-ON V2.0, Arduino Mega, Bluetooth 4.0, Phantom 4)

Uno de los problemas detectados con esta opción una vez analizada es que no cumple con uno de los objetivos importantes de este Trabajo Fin de Máster y el proyecto europeo ENABLE-S3, que es la conducción automática del camión agrícola, sin personas físicas en su interior, y por lo tanto, no podemos colocar todo el sistema dentro del mismo, ya que en caso de emergencia es necesario que alguien tomase el control del dron de forma manual, y esto no sería posible ya que el mando estaría dentro del vehículo sin ningún operario junto a él. Sería equivalente a utilizar la función DjiFollowMe, pero sin personas junto al control remoto. Es por ello por lo que desechamos esta opción.

En el segundo esquema, Figura 22, el sistema está compuesto por 4 elementos, ya que el quinto y sexto (el IPad y el mando del dron) se separan de la unidad de control para desarrollar una nueva unidad inteligente. El primer elemento es el módulo GPS, RPI GPS ADD-ON V2.0, y se encarga de transmitir a través del puerto serie la trama de datos según el protocolo NMEA en formato ASCII. El segundo elemento es el microcontrolador ATmega32u4 con radio LORA, la unidad transmisora, que compone la trama de datos recibida en el puerto serie donde está conectado el sistema de posicionamiento global y la transmite por radio a 915MHz al tercer elemento que también es un microcontrolador ATmega32u4 con radio LORA, unidad receptora. El tercer elemento se comportará como una nueva unidad inteligente encargada de desarrollar las tramas de datos con un formato específico para controlar el dron (cuarto elemento) a través del puerto serie. Para finalizar con el segundo esquema se comenta que el primer y segundo elemento se incorporan dentro del camión agrícola.

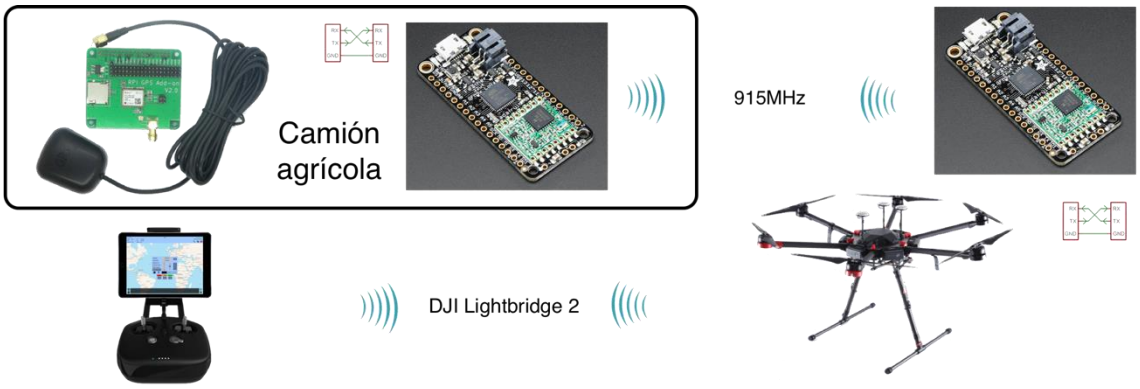

Figura 22. Esquema 2 (RPI GPS ADD-ON V2.0, ATmega32u4 con radio LORA, Matrice 600)

Esta segunda opción es válida, pero surgen algunas dificultades cuando se requiere programar el Matrice 600 a través del puerto serie porque el Onboard SDK no dispone de todos los comandos necesarios para realizar todas las operaciones, lo que añadiría aún más complejidad al sistema. Por otro lado, se presenta otro problema, pues no se pueden tener dos unidades inteligentes controlando el dron porque se producen conflictos y errores al intentar manejar los controles del mando a la vez que se maneja con el microcontrolador ATmega32u4 con radio LORA por el puerto serie.

Cuando nos referimos a una unidad inteligente estamos definiendo quién va a controlar el dron. El mando del dron es de por sí una unidad inteligente porque se puede controlar todos los movimientos del mismo. De la misma forma, si se realiza la comunicación con el dron a través del puerto serie a partir de los comandos que nos especifican los desarrolladores de Dji se puede manejar una parte de las habilidades que tiene el dron, pero como se comentó, se podrían crear conflictos a la hora de intentar tomar el control con el mando si surge algún problema durante el vuelo del mismo. Por tanto, es recomendable que solo dispongamos de una unidad inteligente que realice los objetivos deseados y resuelva todos los problemas presentes durante la misión personalizada. Es por ello por lo que hemos desechado este esquema de funcionamiento.

En el tercer esquema, Figura 23, el sistema está compuesto por 7 elementos. El primer elemento es el módulo GPS, RPI GPS ADD-ON V2.0, y se encarga de transmitir a través del puerto serie la trama de datos según el protocolo NMEA en formato ASCII. El segundo elemento es el microcontrolador ATmega32u4 con radio LORA, la unidad transmisora, que compone la trama de datos recibida en el puerto serie donde está conectado el sistema de posicionamiento global y la transmite a través de la unidad transmisora a 915MHz hacia el otro microcontrolador ATmega32u4 con radio LORA, unidad receptora. La unidad receptora reenvía la información recibida del transmisor hacia el puerto serie donde está conectado el Bluetooth HM-10. Este es el cuarto elemento y permite encaminar la información recibida hacia el IPad. El quinto elemento es el IPad, y este controla el vuelo del dron a partir de una aplicación que se desarrollará desde cero en el entorno de programación Xcode 8.3 con la SDK de Dji. El sexto elemento es el mando del dron, y en él estará conectado el IPad vía USB. Y el último elemento es el dron, que volará hacia el punto de coordenada correspondiente en cada momento. Para finalizar el tercer esquema se comenta que el primer y segundo elemento se localizan dentro del camión agrícola y el tercero, cuarto y quinto elemento formarán la estación base.

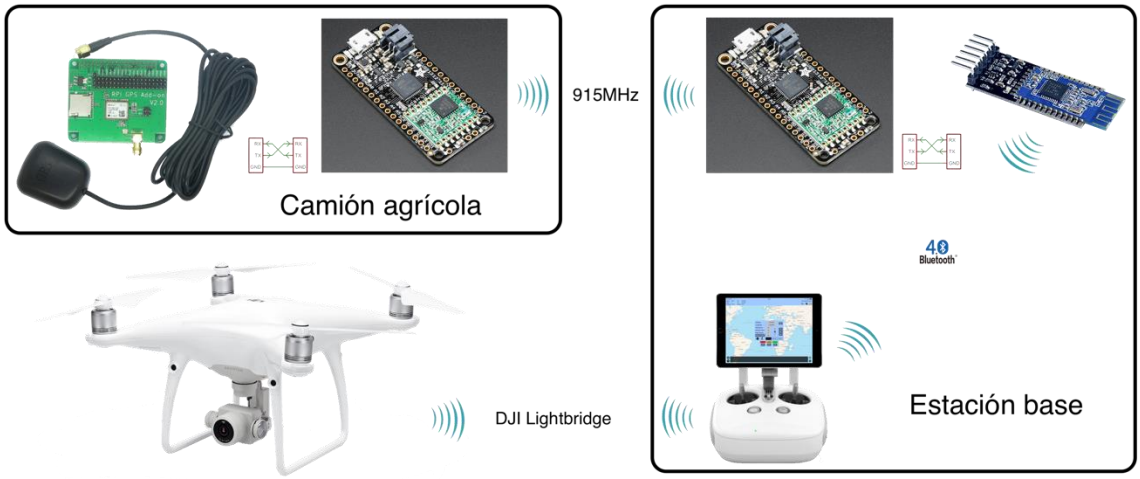

Figura 23. Esquema 3 (RPI GPS ADD-ON V2.0, ATmega32u4 con radio LORA, Bluetooth 4.0, Phantom 4)

Este último esquema se adecua más a nuestra aplicación y se utiliza porque cumple con todos los objetivos del TFM y el proyecto europeo ENABLE-S3:

- El GPS va instalado en el camión agrícola junto con la unidad transmisora, el primer microcontrolador ATmega32u4 con LORA, y envía la información hasta la estación base a varios kilómetros de donde se ubica el vehículo agrícola. La estación base está formada por la unidad receptora, el segundo microcontrolador ATmega32u4 con LORA, el Bluetooth HM-10, el IPad y el mando del dron, y estarán siempre en manos de un supervisor que podrá hacerse con el control del dron en caso de emergencia.
- Se trata de un sistema trasversal para todas las plataformas de Dji, es decir, que puede usarse tanto para un Phantom 4 como para un Matrice 600, además de otras plataformas.

3.3. ESTRUCTURA DEL SISTEMA

Como se explicó anteriormente, el sistema elegido corresponde a la Figura 23, y para su desarrollo en este Trabajo Fin de Máster se divide en 3 etapas. En la primera etapa se realiza el montaje y la programación del microcontrolador ATmega32u4 con el GPS. En la segunda etapa, se repite el mismo proceso, pero esta vez para el segundo microcontrolador ATmega32u4, se realizan las conexiones con el Bluetooth HM-10 y se procede a programar el microcontrolador ATmega32u4 con radio LORA. Además, en la primera etapa se programa la radio LORA como un emisor y en la segunda etapa se programa como un receptor. Y en la tercera etapa, se crea desde cero una aplicación en Xcode 8.3 para desarrollar un gestor de bluetooth que permita escanear, conectar o desconectar los periféricos de los alrededores, un conjunto de puntos (coordenadas, título, subtítulo, hora, fecha, etc.) con su información asociada para representar en el mapa el trayecto de la unidad transmisora, el dron y el dispositivo móvil, una tabla para representar el conjunto de puntos con su información asociada de la unidad transmisora, un menú de opciones y la programación del vuelo del dron con la SDK 4.0 que se encarga de realizar el seguimiento.

3.3.1. Desarrollo de la primera etapa: unidad transmisora y módulo GPS

El comienzo del sistema fue el montaje del microcontrolador ATmega32u4 con radio LORA y el RPI GPS ADD-ON V2.0 a través del puerto serie. La antena utilizada para la unidad transmisora trabaja a una frecuencia de 2.4-2.5GHz, pero también es apta para trabajar con esta radio LORA ya que su frecuencia de funcionamiento es inferior a la de la antena, en concreto, 868-915MHz. Estos dispositivos hardware se encargan de obtener y transmitir una trama de datos.

El siguiente paso consiste en descargarse la IDE de Arduino y realizar los pasos que se han explicado anteriormente en el apartado 2.3.2.2**,** para descargar las tarjetas del microcontrolador ATmega32u4 y las librerías de la radio LORA. La programación del ATmega32u4 se divide en varias partes. Por un lado, se realiza la lectura del módulo GPS por separado para entender cómo funciona la trama de datos, posteriormente se analiza y se separan los comandos necesarios para obtener las posiciones, y se guarda en memoria. En la Figura 24 se representa un fragmento del código donde se guarda cada uno de los parámetros del comando GPRMC en diferentes posiciones de un array de string para componer la trama de datos más adelante. El comando GPRMC sirve para conocer la fecha y hora actual, las coordenadas y la orientación del GPS, la velocidad en nudos, y otros parámetros que se explican en [12].

```
switch(I){
  case 0 :  if (idx >= 8) break; if (idx==2 || idx==5) { Datos_GPRMC[I] = Datos_GPRMC[I] + ":";
    idx++; }; Datos_GPRMC[I] = Datos_GPRMC[I] + Buffer[E+1]; break;
  case 1 :  Datos_GPRMC[I] = Buffer[E+1]; break;
  case 2 :  Datos_GPRMC[I] = Datos_GPRMC[I] + Buffer[E+1]; break;
  case 3 :  Datos_GPRMC[I] = Buffer[E+1]; break;
  case 4 :  Datos_GPRMC[I] = Datos_GPRMC[I] + Buffer[E+1]; break;
  case 5 :  Datos_GPRMC[I] = Buffer[E+1]; break;
  case 6 :  Datos_GPRMC[I] = Datos_GPRMC[I] + Buffer[E+1]; break;
  case 7 :  Datos_GPRMC[I] = Datos_GPRMC[I] + Buffer[E+1]; break;
  case 8 :  if (idx==8) break; if (idx==2 || idx==5) { Datos_GPRMC[I] = Datos_GPRMC[I] + "/";
    idx++; }; Datos_GPRMC[I] = Datos_GPRMC[I] + Buffer[E+1]; break;
  case 9 :  Datos_GPRMC[I] = Datos_GPRMC[I] + Buffer[E+1]; break;
  case 10 : Datos_GPRMC[I] = Datos_GPRMC[I] + Buffer[E+1]; break;
  case 11 : Datos_GPRMC[I] = Datos_GPRMC[I] + Buffer[E+1]; break;
  case 12 : Datos_GPRMC[I] = Buffer[E+1]; Datos_GPRMC[I] = Datos_GPRMC[I] + Buffer[E+2]; break;
}
```

Figura 24. Código para guardar los parámetros del comando GPRMC

El siguiente paso consiste en generar una trama de datos específica que sea pequeña y con un formato claro para transmitirla a través de la radio LORA, pero antes, se realiza la programación básica de la unidad transmisora para iniciar la radio LORA y se establece las configuraciones iniciales (potencia de salida, frecuencia, etc.). Para enviar la trama por la radio a una frecuencia de 915MHz se utiliza como base los ejemplos de la página oficial y se modifican los fragmentos de código necesarios para llevar a cabo el proceso como se muestra en la Figura 25, donde aparecen los comandos que permiten enviar la información a través de la radio LORA.

```
Serial.print("Mensaje enviado(");Serial.print(sizeof(GPS));Serial.print("): ");Serial.println(GPS);
radioLoRa.send(GPS, sizeof(GPS));
delay(10);
radioLoRa.waitPacketSent();
```

Figura 25. Código modificado para enviar la trama de datos

Para terminar el montaje se desarrolló con una impresora 3D una caja donde ubicar el microcontrolador ATmega32u4 con radio LORA y el RPI GPS ADD-ON V2.0 dejando los habitáculos para pasar el conector micro-USB, la antena del módulo GPS y la antena del radio LORA como se ilustra en la Figura 26.

Figura 26. Montaje final de la primera etapa (RPI GPS ADD-ON V2.0 y microcontrolador ATmega32u4 con radio LORA)

3.3.2. Desarrollo de la segunda etapa: unidad receptora y módulo bluetooth

En la segunda etapa del proyecto es necesario realizar algunos pasos similares a la primera etapa. En primer lugar, se realiza el montaje del microcontrolador ATmega32u4 con radio LORA y se utiliza la misma antena que se comentó anteriormente en el apartado 3.3.1. Una vez realizado el montaje, se realizan las conexiones con el Bluetooth HM-10 mediante el puerto serie y se comienza con la programación.

En segundo lugar, se configura los parámetros iniciales de la unidad receptora, se recibe la información de la unidad transmisora y se guarda en memoria. Para ello, se utilizan los ejemplos de la página oficial y se modifican los fragmentos de código necesarios para llevar a cabo el proceso como se ilustra en la Figura 27, donde se almacena la información recibida.

```
if (radioLoRa.available())
{
  // Variables para recoger los datos.
  uint8_t buf[RH_RF95_MAX_MESSAGE_LEN];
  uint8_t len = sizeof(buf);
```

Figura 27. Código para recibir la información del radio LORA

Una vez almacenada la información en memoria, automáticamente se transmite por el puerto serie para que el Bluetooth HM-10 la reenvíe hacia el dispositivo móvil. La configuración inicial del Bluetooth HM-10 se realiza a través del puerto serie donde se establece el nombre del periférico y la velocidad de transmisión como se representa en la Figura 28.

```
// Estas ahí?
Serial1.println("AT");
EsperaPuertoSerie();
Blink(LED, 500, 1);
// Definimos el nombre del Bluetooth 4.0.
Serial1.print("AT+NAME");
Serial1.println(NOMBRE);
EsperaPuertoSerie();
Blink(LED, 500, 1);
// Definimos la velocidad del Bluetooth 4.0.
Serial1.print("AT+BAUD");
Serial1.println(BPS);
EsperaPuertoSerie();
Blink(LED, 500, 1);
```

Figura 28. Configuración inicial del Bluetooth HM-10

Para finalizar la estación base, se realiza una segunda caja con la impresora 3D. La caja se construye de manera que el microcontrolador ATmega32u4 con radio LORA y el Bluetooth HM-10 estén adaptados a la perfección dentro de la misma dejando los habitáculos para pasar el conector micro-USB y la antena del radio LORA como se muestra en la Figura 29.

Figura 29. Montaje final de la segunda etapa (microcontrolador ATmega32u4 con radio LORA y Bluetooth HM-10)

3.3.3. Desarrollo de la tercera etapa: aplicación en Xcode 8.3

Para comenzar con el desarrollo de la aplicación en Xcode, se crea un nuevo proyecto y se ejecuta el Dji iOS SDK CocoaPods a través de la consola del macOS para manejar y compilar las librerías del Mobile SDK como se explicó anteriormente al final del apartado 2.5.3.

Una vez finalizado el paso anterior, se crea una nueva clase para desarrollar un gestor de bluetooth donde se programan todas las librerías y protocolos del CoreBluetooth para el dispositivo móvil. Luego, se desarrolla el código para gestionar el estado del bluetooth (si está encendido, apagado, reiniciando, etc.) y sus métodos. De la misma forma, se procede a insertar las funciones que permiten conocer los estados de los periféricos (si está disponible, conectado o desconectado), así como los métodos necesarios para recibir la información de la estación base como se representa en la Figura 30.

```
// Services of peripheral connected.
- (void)peripheral:(CBPeripheral *)peripheral didDiscoverServices:(NSError *)error {
    if (error) return;
    for (CBService *service in peripheral.services) [peripheral discoverCharacteristics:nil forService:
        service];
}

// Characteristics of peripheral connected.
- (void)peripheral:(CBPeripheral *)peripheral didDiscoverCharacteristicsForService:(CBService *)service
    error:(NSError *)error {
    if (error)  return;
    for (CBCharacteristic *characteristic in service.characteristics)   [peripheral setNotifyValue:YES
        forCharacteristic: characteristic];
}

// Notification about state of characteristic.
- (void)peripheral:(CBPeripheral *)peripheral didUpdateNotificationStateForCharacteristic:
    (CBCharacteristic *)characteristic error:(NSError *)error {
    if (error) [error localizedDescription];
    if (characteristic.isNotifying) [peripheral readValueForCharacteristic:characteristic];
}
```

Figura 30. Métodos para recibir los servicios, las características y las notifiaciones de los periféricos.

Después de terminar el gestor de bluetooth, se crea una nueva clase para los puntos con unas propiedades específicas (coordenadas, título, subtítulo, etc.) que cumpla con el protocolo de los mapas y un método para inicializar un punto con su información asociada como se ilustra en la Figura 31. Como se puede observar, este modelo no incorpora el patrón *singleton* porque no se utiliza para guardar variables globales.

```
@property (nonatomic) CLLocationCoordinate2D coordinate;
@property (nonatomic, copy) NSString * title;
@property (nonatomic, copy) NSString * subtitle;
@property (nonatomic, copy) NSString * Date;
@property (nonatomic, copy) NSString * Time;
@property (nonatomic, copy) NSString * RSSIRadio;

-(ModeloPuntos*)initWithPropertyCoordinate: (CLLocationCoordinate2D) coordenada
                                 withTitle: (NSString*) titulo
                              withSubtitulo: (NSString*) subtitulo
                                  withDate: (NSString*) date
                                  withTime: (NSString*) time
                                  withRSSI: (NSString*) RSSILoraRadio;
```

Figura 31. Inicialización de un punto con su información asociada

Ahora se procede a crear una nueva clase para el modelo de coordenadas con el patrón *singleton* donde se guardará un conjunto de puntos con su información asociada para la unidad transmisora, el dron y el dispositivo móvil. En la Figura 32 se representa un fragmento de esta clase con el patrón de diseño *singleton* y el método utilizado para inicializar las variables en él.

```
+ (ModeloCoordenadas *) getInstance {
    static  ModeloCoordenadas* modeloCoordenada = nil;
    static dispatch_once_t onceToken;
    dispatch_once(&onceToken, ^{
        modeloCoordenada = [[self alloc] initWithGetInstance];
    });
    return modeloCoordenada;
}

- (ModeloCoordenadas *) initWithGetInstance {
    if (self = [super init]) {
        self.ArrayCoordinateLORA = [[NSMutableArray alloc] init];
        self.ArrayCoordinateDron = [[NSMutableArray alloc] init];
        self.ArrayCoordinateMovil = [[NSMutableArray alloc] init];
    }
    return self;
}
```

Figura 32. Patrón de diseño *singleton* implementado en la clase *ModeloCoordenadas*

Para guardar los puntos con su información asociada se crea una nueva clase para desarrollar dos métodos que permitan leer y guardar ficheros de datos como se ilustra en la Figura 33.

```
#pragma mark - Ficheros TXT

/**
 Guarda un formato de datos dados en NSMutableString en un fichero de datos.

 @param dataSave Formato de datos a guardar en el fichero
 @param file Nombre del fichero
 @return NSError
 */
- (NSError*)saveOneFileManager:(NSMutableString*)dataSave withPath:(NSString*)file;

/**
 Devuelve un formato de datos dados en NSMutableString de un fichero de datos.

 @param file Nombre del fichero
 @return Devuelve los datos del fichero en NSMutableString si existe. En caso contrario, devuelve nil.
 */
- (NSMutableString*)readOneFileManager:(NSString*)file;
```

Figura 33. Métodos para leer y guardar datos en ficheros.

Una vez que tenemos todas las clases necesarias para comenzar con el desarrollo de la aplicación se crean los *storyboards*. En esta aplicación se diseñan 3 *storyboards* para un IPhone 6S como se ilustra en la Figura 34, pero se añaden los *constrains* necesarios para utilizar la aplicación en cualquier otro dispositivo móvil como el IPad. Por un lado, se utiliza el primer *storyboard* como vista principal y, por otro lado, se utilizan dos tablas para las vistas secundarias que corresponden con el segundo y tercer *storyboard*. Asimismo, dentro de cada uno de ellas se programan los elementos necesarios para su desarrollo.

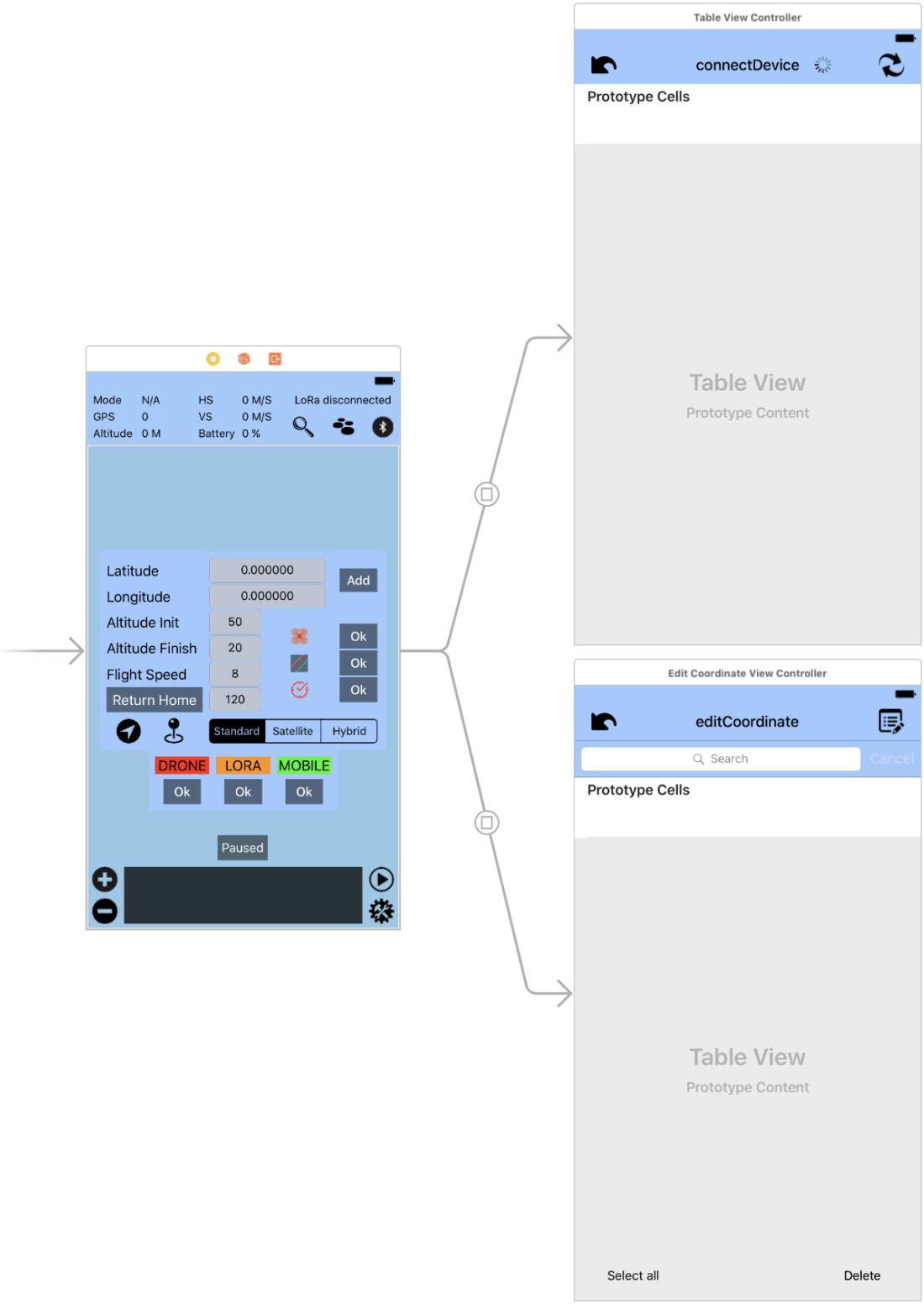

Figura 34. Diseño de la aplicación

A continuación, se explica el proceso de desarrollo para el diseño de la aplicación a partir de la Figura 34. En el primer *storyboard*, se incorpora un mapa para representar el último punto recibido por la radio LORA, la posición del dron y la ubicación actual del dispositivo móvil. Además, se dibujan unas líneas a lo largo de la superficie del mapa que corresponden con el trayecto del módulo GPS, el dron, y el dispositivo móvil. En el segundo *storyboard*, se

inserta una tabla para escanear, conectar o desconectar los periféricos de los alrededores. En el tercer *storyboard* también se incluye una tabla y se utiliza para mostrar el conjunto de puntos de la radio LORA con su información asociada. Esta tabla permite editar los elementos del array, seleccionar o deseleccionar los objetos de las celdas con un botón, o incluso eliminarlos. Además, incorpora un buscador donde se puede navegar para encontrar un dato específico. En el primer *storyboard* se integra una vista y será el menú de opciones junto con un menú secundario. Aquí aparecen las opciones de vuelo del dron, un botón para activar o desactivar el trayecto de los puntos, un botón para el icono del mapa, un botón para centrar el mapa cuando llega un dato nuevo y otros elementos que se comenta más adelante en el manual de uso. Y el menú secundario se usa para activar o desactivar las líneas de la radio LORA, el dron y el dispositivo móvil.

Una vez explicados los *storyboards* de la aplicación, se desarrolla la programación del dron. Para ello, se comienza insertando en la zona superior del mapa una nueva vista, y en ella, se incorpora los parámetros de vuelo del dron (modo, GPS, altitud, batería, velocidad en el eje x e y, y la velocidad en el eje z). Asimismo, se añaden otros elementos para mostrar el estado en el que se encuentra la radio LORA, un icono para conocer el estado de conexión del bluetooth del dispositivo móvil y dos botones para abrir las tablas explicadas anteriormente. También, en el *storyboard* del mapa se incluye en la zona inferior algunos elementos para mostrar el estado de la misión personalizada, un botón para acercar y alejar el mapa cuando llega un dato nuevo, un botón para abrir o cerrar el menú de opciones, y un botón para comenzar o finalizar la misión personalizada. El botón que inicia o finaliza esta misión realiza una serie de pasos para ejecutar el vuelo del dron o devolverlo a casa. Por ello, se explica paso por paso cada una de las misiones que se utilizan durante el despegue, el vuelo y el aterrizaje del dron.

En el inicio, se ejecuta una misión denominada Dji AircraftTakeOff que permite despegar el dron hasta una altura de un metro. La segunda misión se denomina Dji GoToAltitude y se encarga de elevar el dron hasta una cierta altura que se define en el menú de opciones y que por defecto es de 50 metros. La tercera misión se denominada Dji GoToPoint y da una orden al dron utilizando la última coordenada recibida por la radio LORA, y en caso de no tener ninguna coordenada, el dron se coloca en la posición del dispositivo móvil. Una vez realizado el trayecto de ida, se vuelve a ejecutar la misión de Dji GoToAltitude, pero esta vez se posiciona en una altura que también se define en el menú de opciones y por defecto es de 20 metros. La velocidad de vuelo durante el inicio del recorrido hasta el dispositivo LORA se define en el menú de opciones y por defecto es de 8m/s. Cuando el dron este posicionado sobre el dispositivo LORA se inicia la misión de Dji FollowMe. En estos momentos, el dron seguirá a la unidad transmisora que en teoría estaría colocada en el camión agrícola. En caso de ocurrir algún error durante el vuelo, se pierda la señal del módulo GPS durante un intervalo de tiempo definido en el menú de opciones o la batería alcance un nivel igual o inferior al 20%, el dron ejecutará la misión de vuelta a casa. El método que inicia la misión de vuelta a casa se divide en dos misiones. La primera de todas eleva el dron hasta la altura inicial con el método Dji GoToAltitude y la segunda ejecuta la misión de vuelta a casa que se denominada Dji GoToHome.

3.4. MANUAL DE USO

El manual de uso se divide en 3 partes. En primer lugar, se explica el funcionamiento del sistema donde se detalla paso por paso cómo se debe proceder para conectar correctamente con la estación base y el producto de Dji (en este caso, se usa el Phantom 4). En segundo lugar, se profundiza en la aplicación explicando todos los componentes de la misma. Y, en tercer lugar, se muestra todo el sistema en el simulador Dji Assistant para observar de forma representativa la misión personalizada.

3.4.1. Funcionamiento del sistema

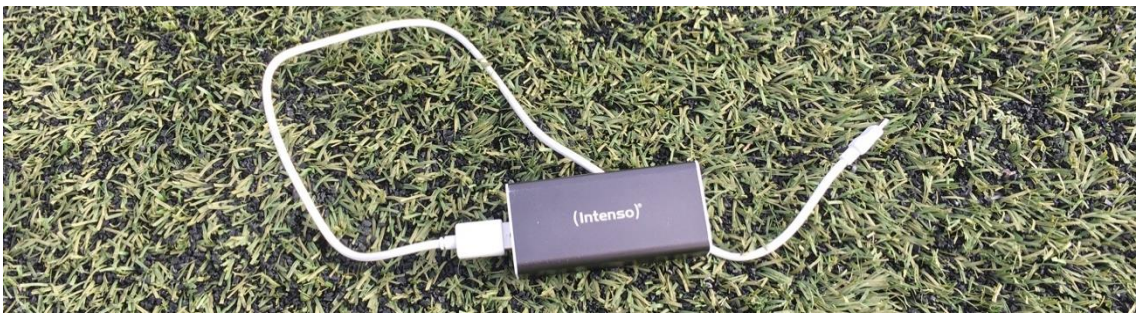

Figura 35. Batería externa para el sistema

En este apartado se explica el funcionamiento global del sistema y cuáles son los pasos a seguir para usarlo de forma eficaz en el entorno:

1. Se colocan las antenas de las radios, la antena del GPS y se alimentan el sistema utilizando dos baterías externas para móviles como la que se ilustra en la Figura 35.
2. Se conecta el dispositivo móvil vía USB con el mando del dron y se enciende el Phantom 4 según las especificaciones de Dji.
3. Se ejecuta la aplicación en el dispositivo móvil y se verifica que el producto se registre correctamente como se representa en la Figura 36.
4. Se accede a la tabla de dispositivo, se busca el periférico con el nombre “TFMARDUINOPHANTOM4” y se conecta.
5. Ahora se debe recibir información a través de la unidad transmisora y su estado debe cambiar. Si se está recibiendo información aparece en la zona superior el estado de la señal de la unidad transmisora. En caso contrario, avisa que los datos no son válidos, y esto se debe a que el módulo GPS está buscando satélites.
6. Se abre el menú de opciones y se configura los parámetros de vuelo del dron (altura inicial, altura final, velocidad y tiempo de vuelta a casa).
7. A continuación, se pulsa el botón de inicio para ejecutar la misión personalizada y se observa en la zona inferior de la vista el estado de la misión.
8. Se recuerda que la aplicación debe quedar abierta en todo momento. Para ello, se avisa con un mensaje inicial como se ilustra en la Figura 37. Esto es necesario porque la misión personalizada comprobará contantemente el estado de vuelo y actualizará la posición del dron en todo momento.

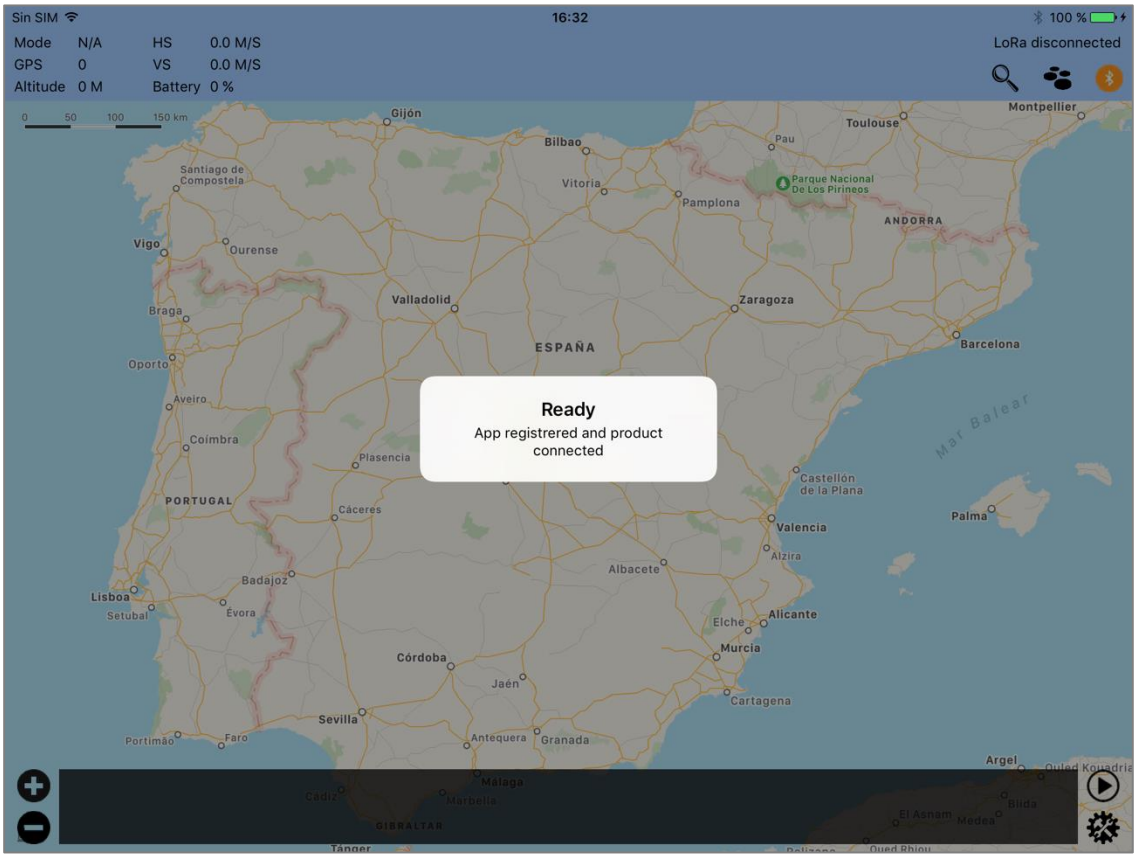

Figura 36. Producto de Dji registrado correctamente

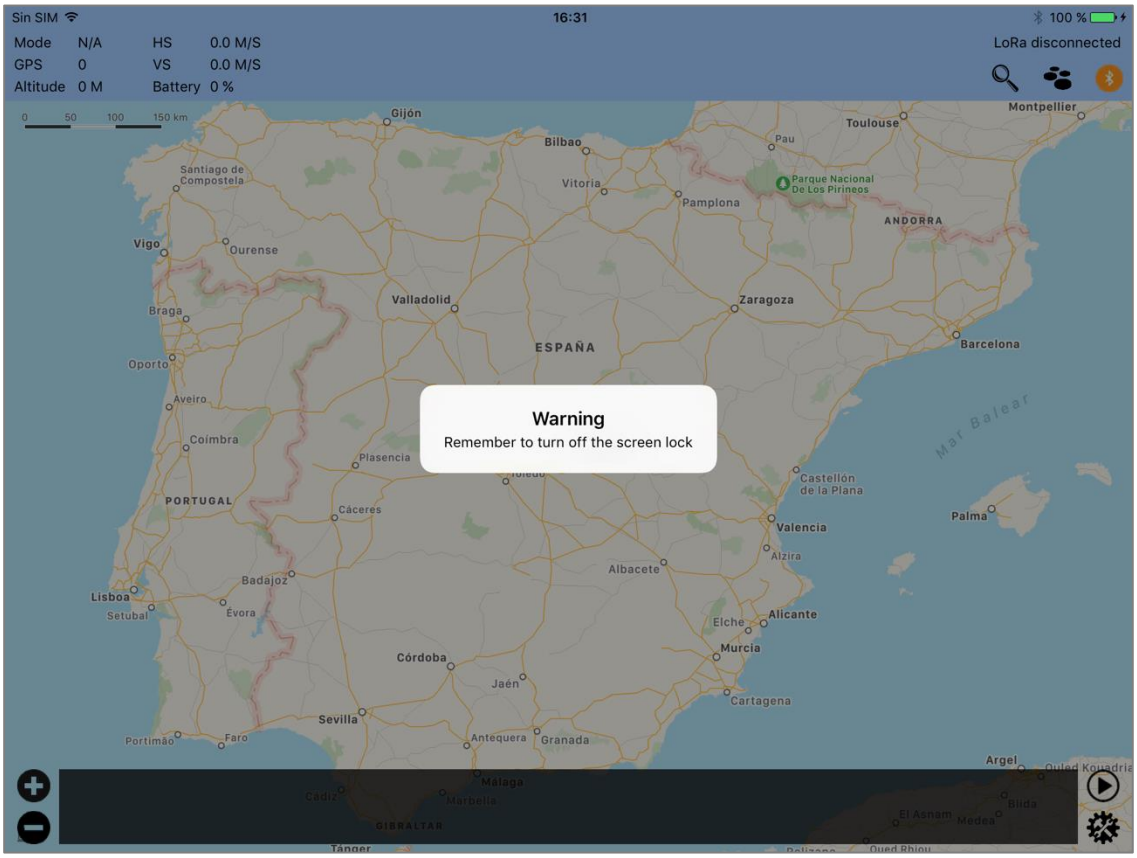

Figura 37. Recordatorio inicial para utilizar la aplicación

3.4.2. Funcionamiento de la aplicación

En este apartado se comienza explicando la barra de herramientas superior, los elementos de la zona inferior, el menú de opciones y el menú secundario del primer *storyboard*, luego se continúa con el segundo *storyboard* que corresponde con la tabla de dispositivos y se finaliza con el tercer *storyboard* que es la tabla de puntos.

La barra de herramientas se ubica en la zona superior del primer *storyboard* como se ilustra en la Figura 38.

Figura 38. Barra de herramientas de la zona superior del primer *storyboard*

Los datos de vuelo del dron proporcionan información acerca del modo en el que se encuentra el dron en cada momento (GPS, TakeOff, WP: WayPoint, FM: FollowMe…), el número de satélites de posicionamiento GPS captados en cada instante, la altitud en cada momento, la velocidad con la que se mueve tanto en el eje vertical (VS) como en el eje horizontal (HS) y el porcentaje de batería restante. El estado de la unidad transmisora describe cómo se encuentra su conexión, el valor de la señal en RSSI y si hay coordenadas GPS o no. Cuando no hay datos es porque el sistema de posicionamiento global está buscando los satélites. El estado de conexión del bluetooth del dispositivo móvil es un icono que modifica su color cuando el bluetooth está encendido (verde) o apagado (rojo), y si está encendido, se muestra si están disponibles (naranja) o conectados (azul) los periféricos de los alrededores.

En la Figura 39 se muestran los elementos de la zona inferior del primer *storyboard*. Aquí se encuentran los botones para ampliar o reducir la vista centrada en el último punto recibido por la unidad transmisora (o añadido desde el menú de opciones), un panel para mostrar el estado de la misión del dron y un botón para abrir o cerrar el menú de opciones.

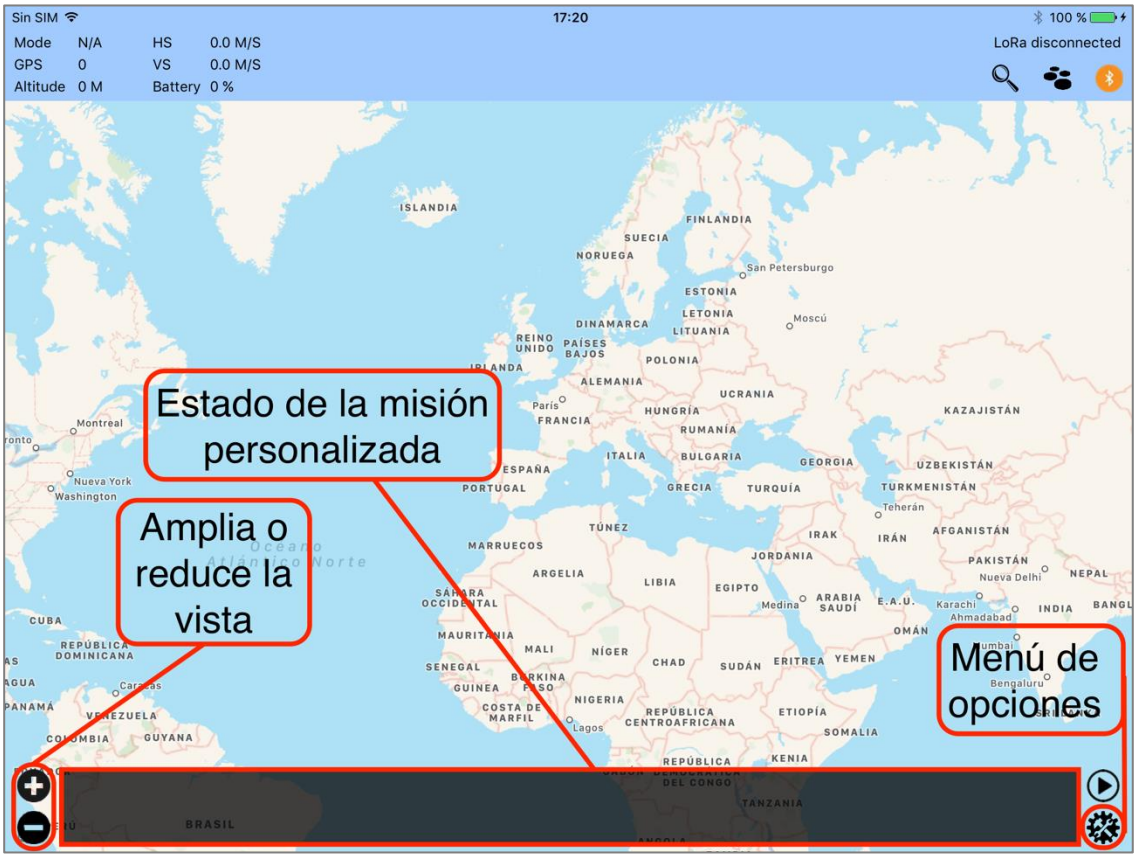

Figura 39. Elementos de la zona inferior del primer *storyboard*

Figura 40. Botón para iniciar la misión personalizada

El primer *storyboard*, ilustrado en la Figura 40, contiene la ubicación actual del dispositivo móvil que se representa con el punto blanco y se implementa utilizando los métodos de la librería CoreLocation.

Cuando se pulsa el botón de inicio, el programa tiene en cuenta el estado de la batería y los motores. En caso de tener menos de un 21% de batería el dron no despega, y en caso de estar volando, detecta en qué posición está ubicado y ejecuta la misión que le corresponde en ese momento. Esto puede ocurrir si se cierra de repente la aplicación en el dispositivo móvil. Entonces, se vuelve a iniciar la aplicación, y se ejecuta de nuevo la misión o se presiona el botón de vuelta a casa en el menú de opciones.

El menú de opciones se utiliza para modificar diferentes parámetros de control de la aplicación como se muestra en la Figura 41. Por un lado, se puede añadir nuevos puntos al mapa manualmente y presionando sobre el botón *Add*. Por otro lado, están los parámetros que definen el vuelo del dron, como la altitud inicial y final del trayecto, la velocidad de vuelo y un botón para volver a casa.

La aplicación incorpora un temporizador que se ejecuta cuando se inicia la misión personalizada. Este temporizador se agota cuando se pierde la señal con la radio LORA y ha transcurrido un tiempo definido en el menú de opciones (por defecto son 120 segundos, 2 minutos). Los elementos representados a la derecha del menú y definidos como opciones del mapa, permiten decidir cuándo queremos mostrar el icono, dibujar el trayecto y centrar el punto en el mapa.

Figura 41. Herramientas del menú de opciones

Las herramientas restantes del menú de opciones son una barra lateral para manejar las vistas del mapa (estándar, satélite o híbrido), un botón para representar todos los puntos y un botón para centrar el mapa en la ubicación actual del dispositivo móvil. Estas se ilustran por separado en la Figura 42.

El botón para la ubicación actual tiene una peculiaridad y es la siguiente: cuando se pulsa este botón, se separa esta coordenada en latitud y longitud para actualizar en el menú de opciones estos valores. Al pulsar el botón *Add* se añaden las coordenadas de la ubicación actual al conjunto de puntos de la radio LORA con su información asociada. Unos ejemplos sencillos se representan en la Figura 44 y la Figura 45.

El menú secundario se abre presionando el botón especificado en la Figura 42 y se utiliza para representar o no las líneas correspondientes a la unidad transmisora, el dron y el dispositivo móvil.

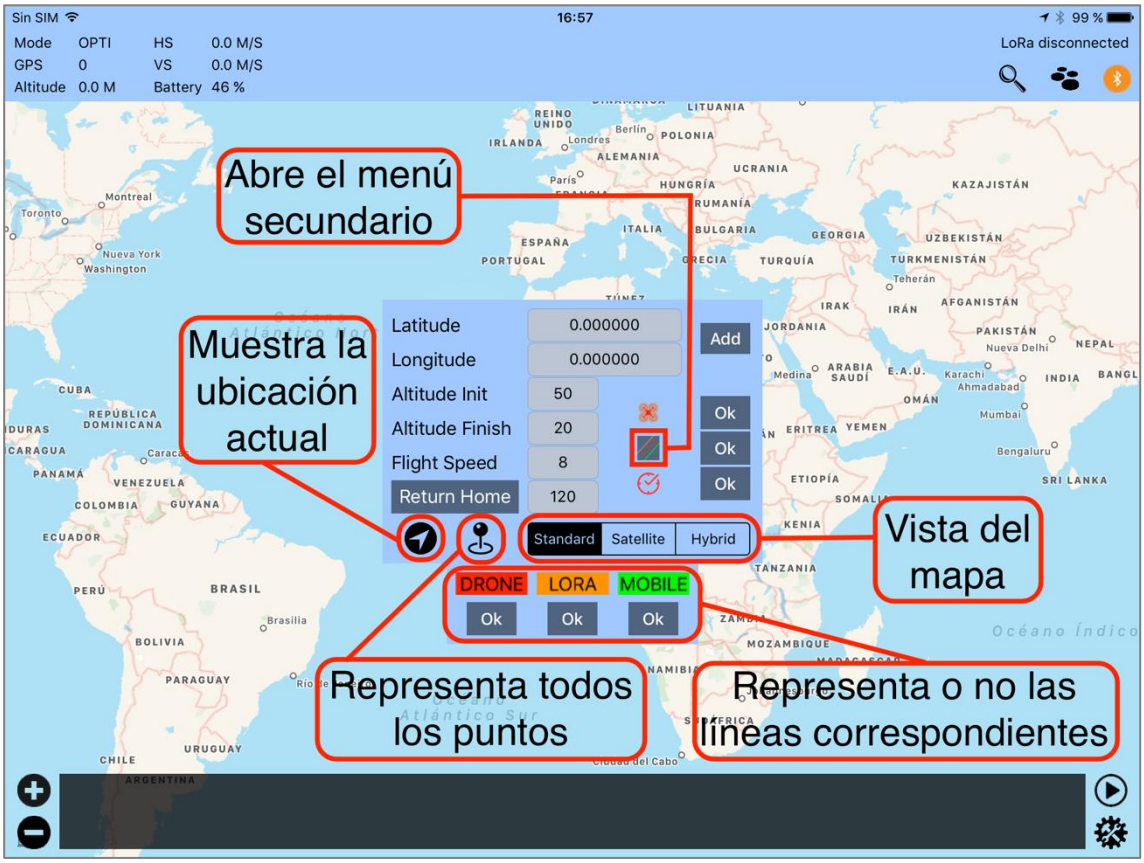

Figura 42. Otras herramientas del menú de opciones

Para abrir la tabla de dispositivos se pulsa sobre la lupa en la barra de herramientas superior como se representa en la Figura 38. Después de pulsar este botón, se abre la tabla de dispositivos que se muestra en la Figura 43.

Esta tabla permite escanear los periféricos disponibles utilizando el gestor de bluetooth explicado anteriormente en la página 34. En ella se encuentran dos botones, un título y una lista donde aparecen los periféricos. El botón de la derecha permite realizar un escaneo que dura 5 segundos, el botón de la izquierda nos devuelve al primer *storyboard* y el título indica el estado del bluetooth y la conexión actual de los periféricos. Para conectar los periféricos, se

hace un clic sobre la celda correspondiente, o se desconecta deslizando el dedo hacia la izquierda.

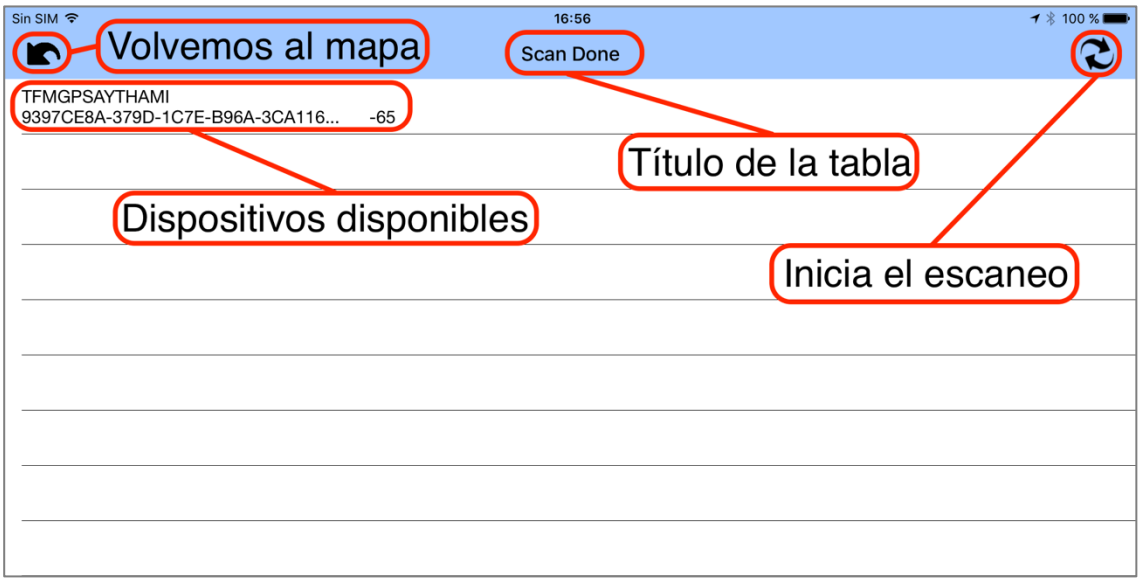

Figura 43. Tabla para escanear, conectar o desconectar los periféricos

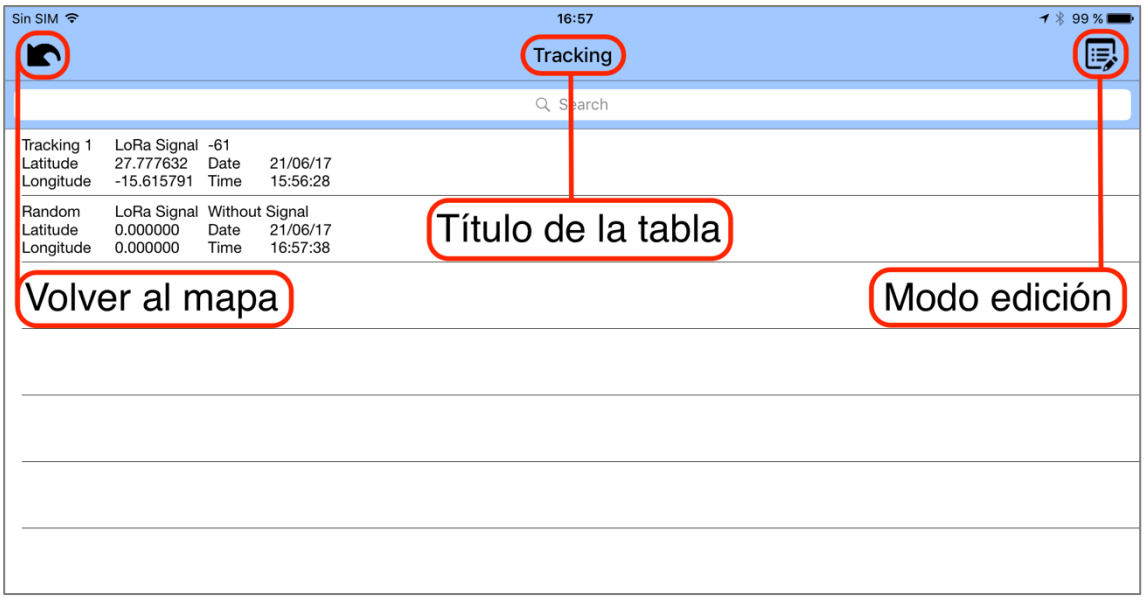

Figura 44. Barra superior de la tabla de puntos para el radio LORA

La tabla de coordenadas se encarga de representar un conjunto de puntos de la radio LORA con su información asociada como se representa en la Figura 44. En la zona superior del *storyboard*, se encuentra un botón para volver al primer *storyboard* como en la tabla de dispositivos, y un botón para editar los puntos. Además, se incluye un buscador para navegar sobre los mismos, pues se utilizará para realizar una búsqueda cuando se guarden muchos puntos. El conjunto de puntos añadidos desde el menú de opciones aparece en la vista como se representa en la Figura 45. Asimismo, en la primera celda aparece un punto con su información asociada guardado a través de la radio LORA y en la segunda celda se observa un punto aleatorio con su información asociada añadido desde el menú de opciones.

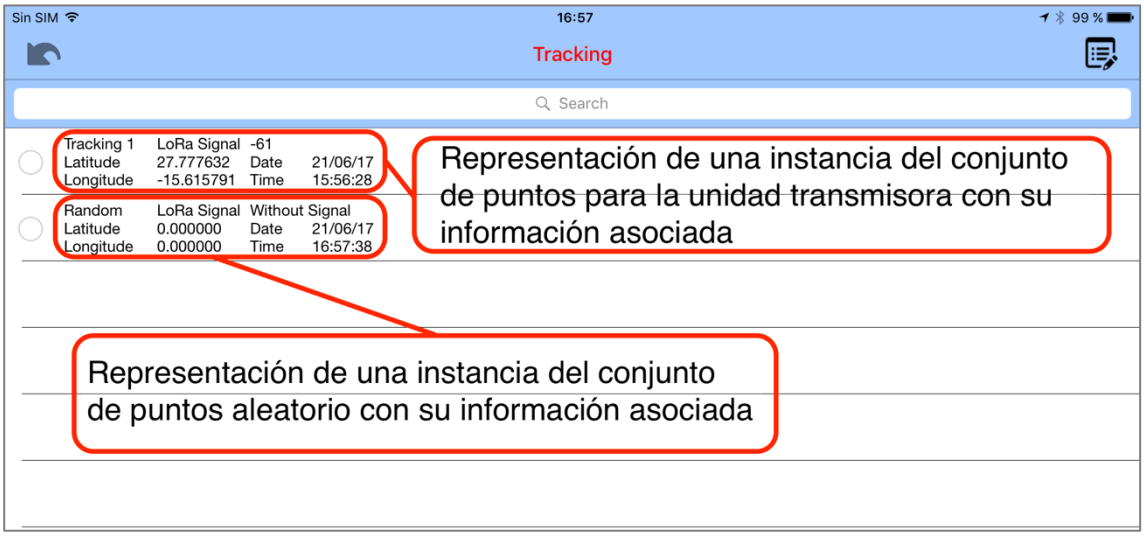

Figura 45. Conjunto de puntos para la unidad transmisora.

Para terminar con la tabla de puntos de la radio LORA con su información asociada, se explican las utilidades del menú de edición. Cuando se pulsa sobre este botón, se puede seleccionar o deseleccionar cualquier conjunto de puntos. Además, incluye un botón para seleccionar o deseleccionar los puntos, así como un botón para eliminar todos los puntos seleccionados. Un dato a tener en cuenta es que no se puede salir de la tabla si se está editando el conjunto de puntos de la radio LORA con su información asociada como se representa en la Figura 46.

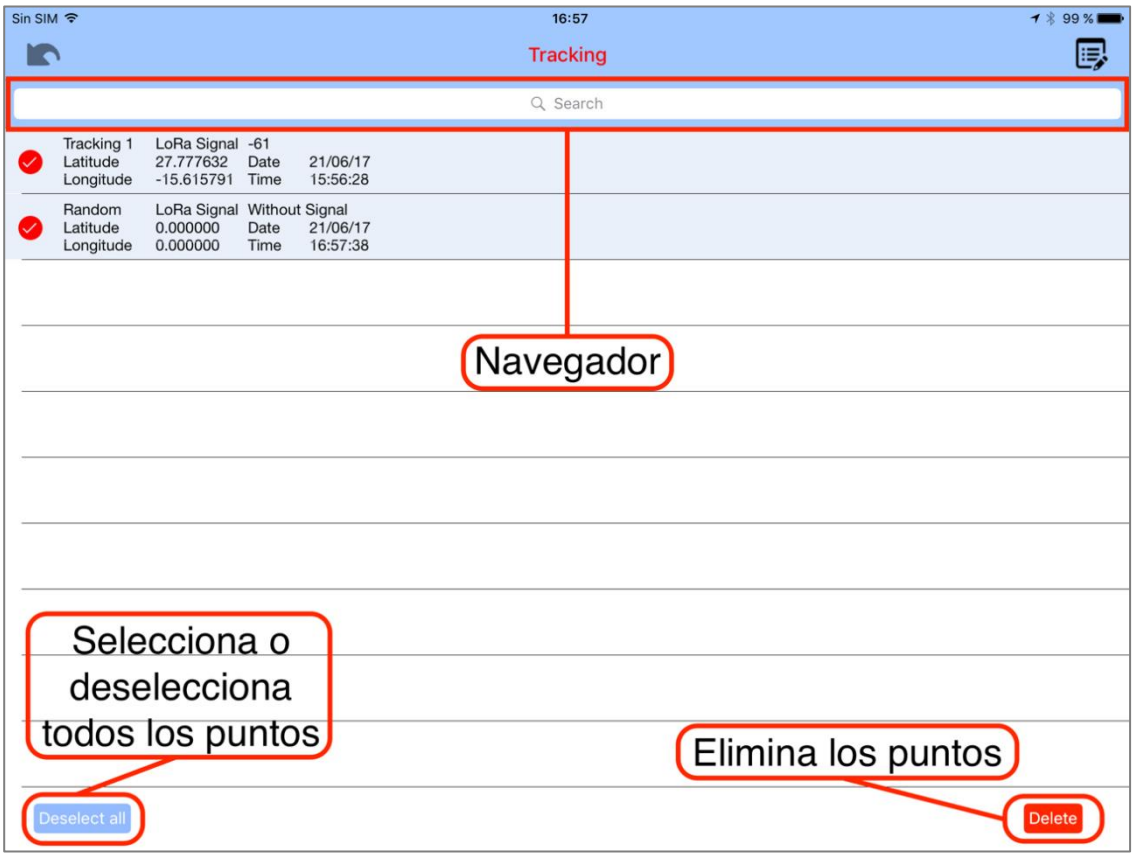

Figura 46. Edición de la tabla de puntos

3.4.3. Funcionamiento en el simulador

Una vez explicado el funcionamiento del sistema y la aplicación, se recrea el vuelo del dron en el simulador para explicar el recorrido que realiza desde el despegue hasta el inicio de la misión de Dji FollowMe como se ilustra en la Figura 47, y el recorrido desde que se finaliza esta misión hasta la vuelta a casa como se muestra en la Figura 48. Además, es necesario realizar los pasos explicados anteriormente en el apartado 2.5.2 para conectar y ejecutar el simulador correctamente con el producto de Dji. En este caso se utiliza el Phantom 4.

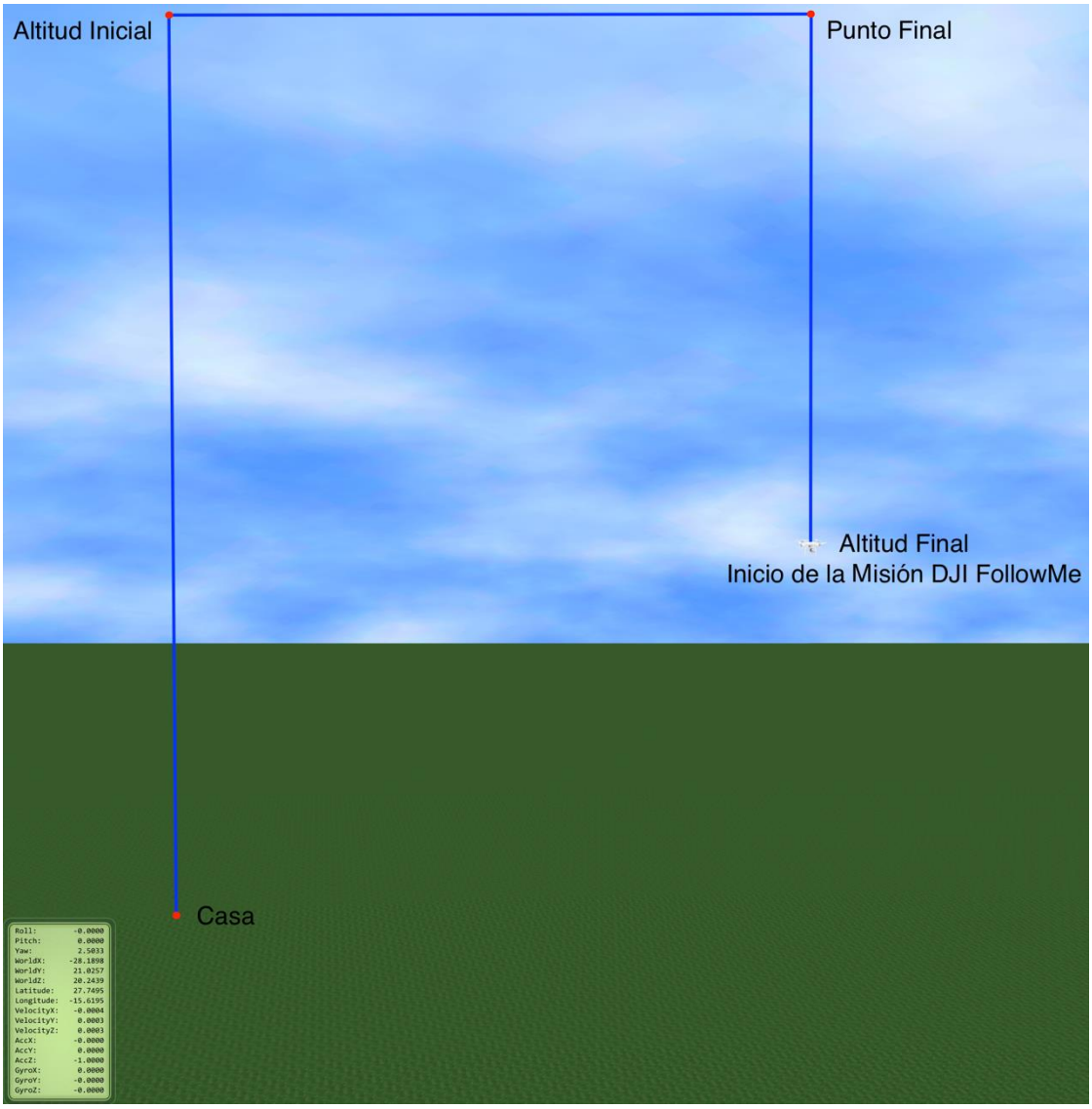

Figura 47. Trayecto de la misión personalizada en el simulador

La misión personalizada comienza cuando el agricultor pulsa el botón para ejecutar la misión personalizada. La aeronave despega del suelo y aumenta su altura hasta una altitud inicial dada en el menú de opciones. Luego, se dirige hacia un punto final que corresponde con la ubicación del camión agrícola y reduce su altura hasta una altitud final establecida en el menú de opciones. En ese momento la aeronave comienza con la misión de Dji FollowMe y sigue al camión agrícola durante la recogida de la cosecha basándose en las coordenadas GPS recibidas.

Cuando la aeronave alcanza un nivel de batería del 20%, se finaliza la misión u ocurre algún error, la aeronave vuelve inmediatamente a casa. Para ello, se finaliza la misión de Dji FollowMe y se aumenta su altura hasta alcanzar la altitud inicial. Luego, vuelve al punto inicial donde despegó y aterriza dentro de un radio establecido en las especificaciones de Dji.

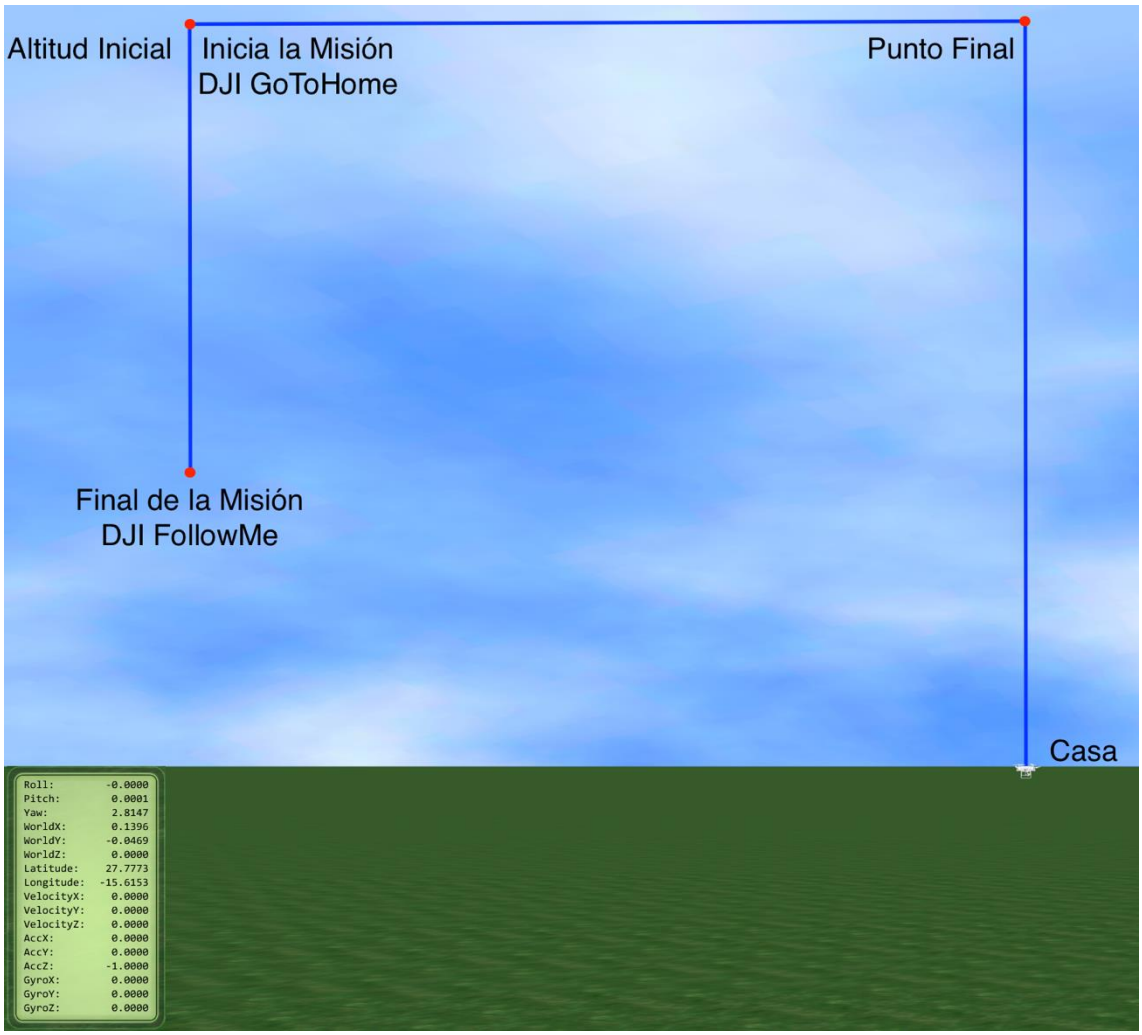

Figura 48. Misión personalizada para devolver el dron a casa en el simulador

Capítulo 4

RESULTADOS Y CONCLUSIONES

4.1. INTRODUCCIÓN

En este capítulo se presentan los resultados y conclusiones del sistema desarrollado en el Capítulo 3 e ilustrado en la Figura 49 para verificar el correcto funcionamiento del mismo. En el inicio del capítulo se explican las pruebas y los resultados realizados que a su vez se divide en tres sub-apartados. En primer lugar, los resultados de atenuación de la señal en función de la distancia entre ambos módulos de radio LORA. En segundo lugar, se muestran los resultados de la precisión de seguimiento del Phantom 4 con respecto al módulo GPS. Por último, se muestra la prueba de funcionamiento de la aplicación con el Matrice 600, utilizando para ello el simulador de Dji.

Para finalizar el capítulo se describen las conclusiones del Trabajo Fin de Máster conforme a los objetivos planteados en el Capítulo 1 y los resultados obtenidos que se muestran en este apartado.

Figura 49. Sistema gobal del proyecto en el campo de fútbol de Tafira

4.2. RESULTADOS

4.2.1. Atenuación y alcance de la señal de la unidad transmisora

Para estudiar el comportamiento de la radio LORA transmisora se mide la caída de intensidad de la señal mediante el indicador de intensidad de la señal recibida. Este indicador, denominado RSSI, es una escala de referencia que mide el nivel de potencia de las señales inalámbricas y su unidad de medida es en dBm (potencia referenciada a 1mW). El intervalo de medida usual para transmitir y recibir la información a través de este tipo de señales es de +20dBm a -137dBm, correspondiendo el valor de +20dBm con una potencia de señal de 100mW, y -137dBm con un valor de 0.2pW.

La prueba se realiza en la Avenida de la playa de Las Canteras en Las Palmas de Gran Canaria, caminando desde el Auditorio Alfredo Kraus en sentido Norte aproximadamente 1.5Km como se muestra en la Figura 50. La ruta comienza y termina en la parte posterior del Alfredo Kraus a una distancia de 75 metros de la unidad receptora. Esta unidad se coloca junto al IPad.

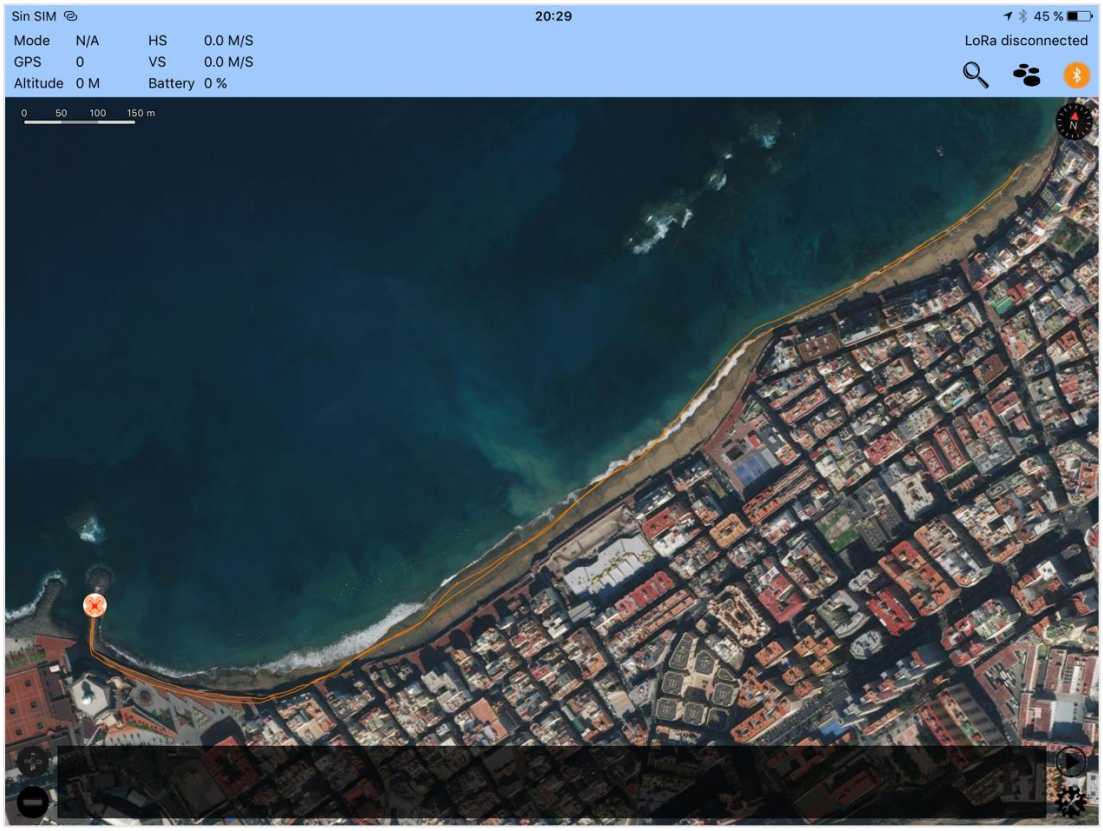

Figura 50. Representación del recorrido de la unidad transmisora en la Avenida de Las Canteras

Los resultados obtenidos se representan en la Figura 51. Como se puede apreciar en la gráfica, estos son más que aceptables para una distancia máxima alcanzada de 1400 metros. Los valores de RSSI de la señal se mantienen prácticamente siempre por encima de -100dBm, un valor de intensidad de señal aceptable, dentro del rango de trabajo definido por el fabricante y que permite transmitir las coordenadas de manera estable durante prácticamente todo el recorrido realizado, con pocas pérdidas de señal en el transcurso del mismo.

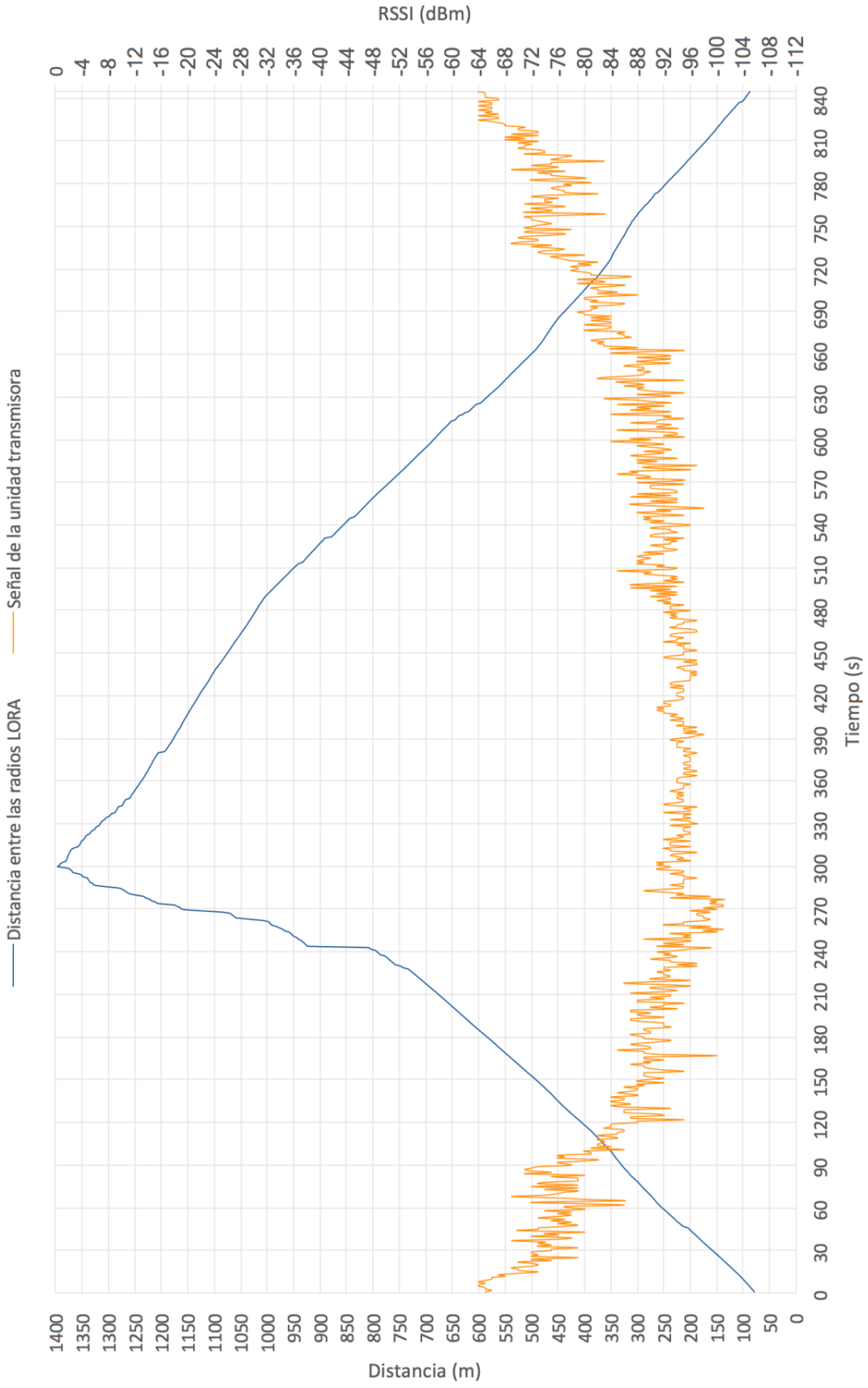

Figura 51. Representación gráfica de la caída de intensidad de la señal en la unidad transmisora y la distancia entre ambas radios LORA

La atenuación de la señal de la unidad transmisora varía constantemente en el tiempo porque depende de varios factores como el medio de propagación de la onda, el ruido de la señal, la distancia entre ambas radios…, que no son objeto de estudio en este Trabajo Fin de Máster.

En esta prueba se alcanzan aproximadamente 1,4 kilómetros según los resultados obtenidos, pero existe la posibilidad de obtener una mayor distancia entre ambas radios LORA ya que estos dispositivos están preparados para trabajar con intensidades de señal aún más débiles que los obtenidos en esta prueba. El fabricante establece que el intervalo de alcance con este tipo de antenas puede llegar hasta los 2 kilómetros de distancia.

4.2.2. Precisión y distancia entre la unidad transmisora y el Phantom 4

La siguiente validación consiste en comprobar el correcto funcionamiento de todo el sistema, es decir, el posicionamiento del dron sobre las coordenadas GPS. La prueba se realiza en las instalaciones deportivas del Campus Universitario de Tafira, concretamente en el campo de fútbol. Se desea analizar lo siguiente: la precisión del módulo GPS observando a simple vista sobre el mapa la superposición del trazado del recorrido realizado por la unidad transmisora y la línea que delimita el campo de fútbol, la separación que existe entre la unidad transmisora y el dron, que se denominó *offset*, y la caída de intensidad de la señal de la unidad transmisora. La prueba comienza cuando se inicia la misión de Dji FollowMe, y a su vez, se acaba cuando se finaliza esta misión. Los resultados obtenidos en las pruebas se muestran en la Figura 52, Figura 53 y Figura 54.

El recorrido que se realiza con la unidad transmisora y el Phantom 4 se representa en la Figura 52, mostrando en rojo las coordenadas del dron y en naranja las de la unidad transmisora. Como se puede apreciar en el mapa, el dron sigue en todo momento la trayectoria de la unidad transmisora, la cual se mueve sobre las líneas que delimitan el campo de fútbol.

La distancia entre ambas radios LORA se ilustra en la Figura 53 junto con la caída de intensidad de la señal de la unidad transmisora. Como se puede observar, la pendiente de la distancia presenta una linealidad considerable, esto se debe a que no se aprecian prácticamente pérdidas de la señal con respecto al transmisor, excepto en dos ocasiones que se analizan con la gráfica del *offset*, a continuación. El intervalo de la caída de intensidad de la señal es de -54dBm hasta -75dBm durante toda la prueba.

El *offset* que se obtiene entre la unidad transmisora y el Phantom 4 se muestra gráficamente en la Figura 54. Se pueden observar dos picos diferenciados en los instantes de tiempo 154 y 285 que se deben a la pérdida de señal con el transmisor durante 2 segundos, lo que implica una desorientación puntual del dron. Cuando la señal se vuelve a estabilizar, el *offset* vuelve a los valores normales. El valor medio de *offset* durante toda la prueba es de 2,39 metros. Sin embargo, hay que resaltar que el vuelo se realiza a 20 metros de altura, y a esa distancia un offset como el mostrado, conjuntamente con el FOV de la cámara, no afecta al objetivo principal de este proyecto, como es poder visualizar lo que hay alrededor de una maquinaria agrícola que se mueve de forma automática por una cosecha. Aun así, este offset es algo a corregir en versiones futuras de esta aplicación.

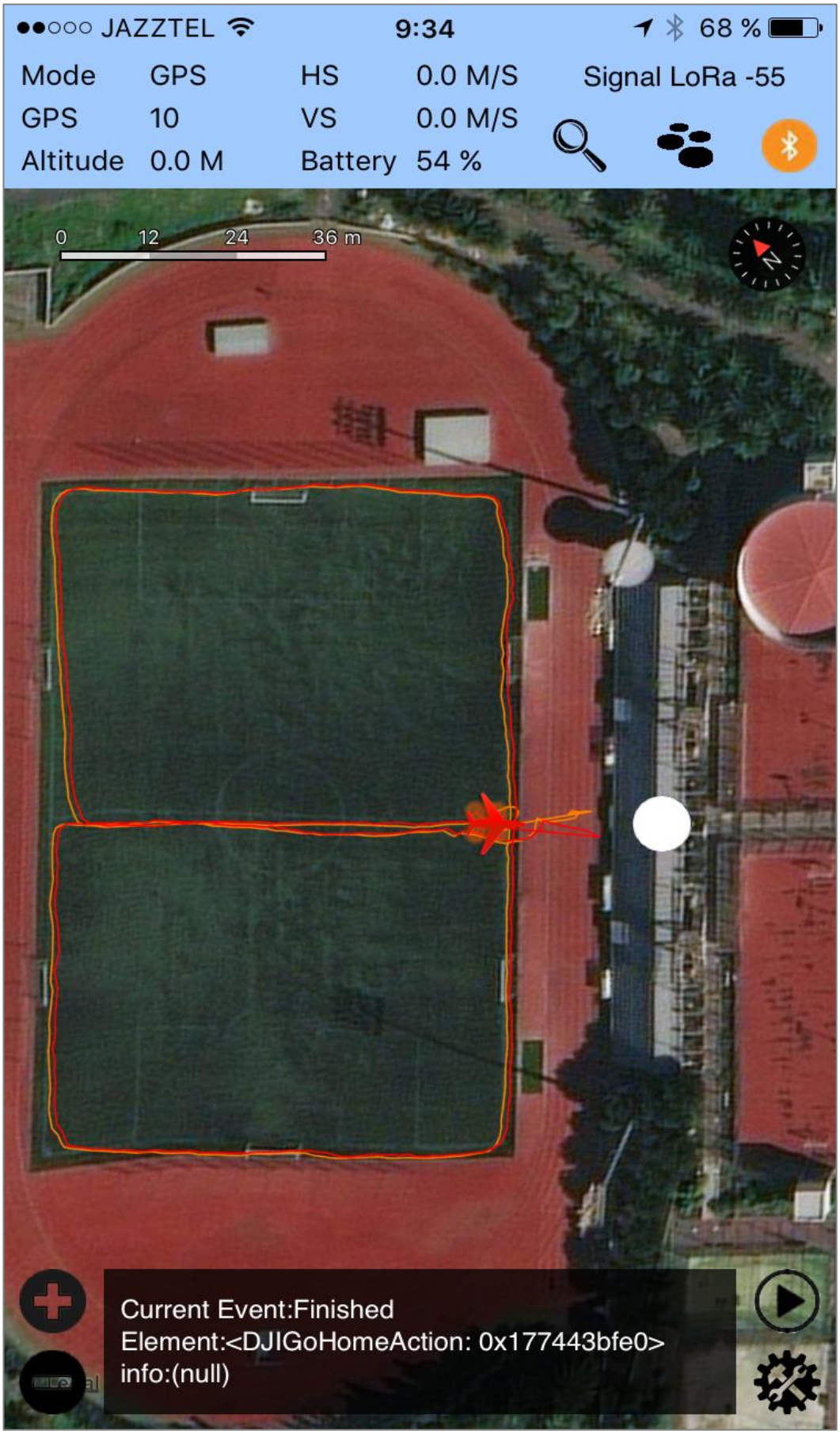

Figura 52. Representación de la prueba realizada en el campo de fútbol de Tafira

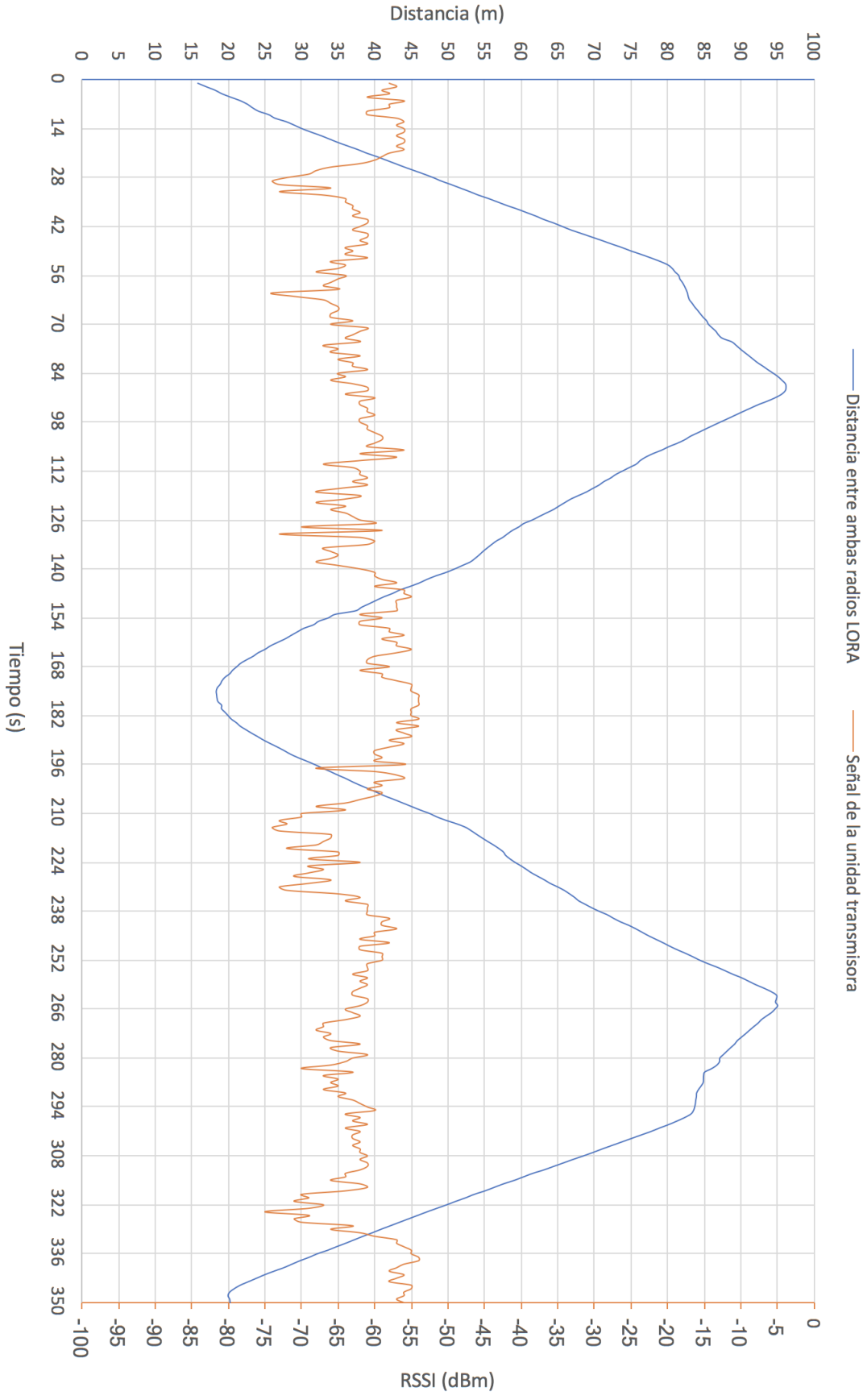

Figura 53. Representación gráfica de la atenuación de la señal de la unidad transmisora y la distancia entre ambas radios LORA

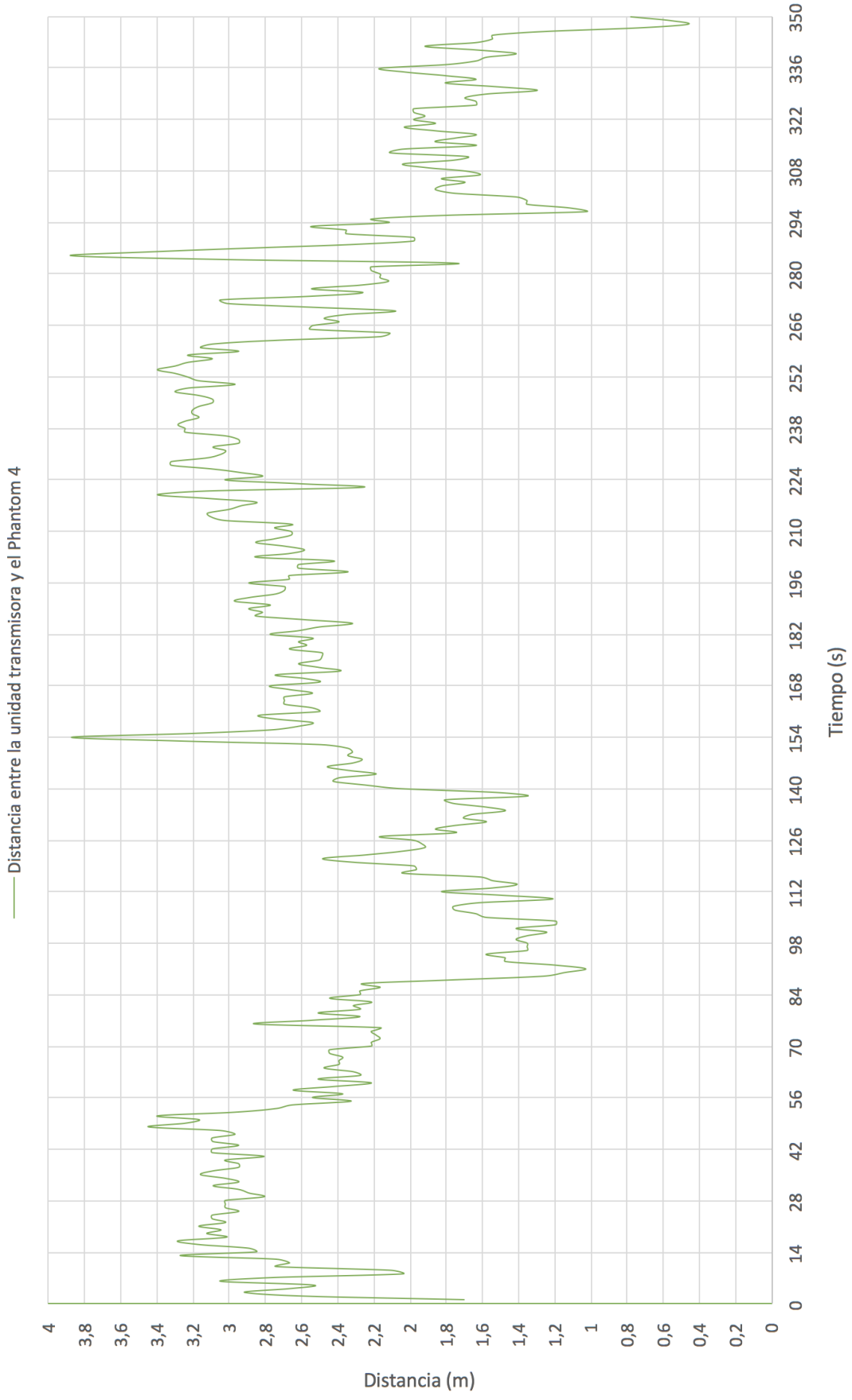

Figura 54. Representación gráfica del *offset* entre la unidad transmisora y el Phantom 4

4.2.3. Implementación de la aplicación en el Matrice 600

Los productos de Dji que incorporan la plataforma para Mobile SDK son compatibles entre ellos. La aplicación que se desarrolla y verifica con el Phantom 4 ha sido probada y validada en el Matrice 600 utilizando el simulador Dji Assistant. Para conectar el Matrice 600 al simulador se siguen los mismos pasos que se explican en el apartado 2.5.2.

El Matrice 600 tiene una peculiaridad a diferencia de los otros de producto de Dji y es la siguiente: este dron incorpora 6 baterías independientes a diferencia del resto, como el Phantom 4, que tiene una única batería. Por tanto, es necesario modificar el método que se utiliza para obtener el nivel de batería porque este método solo es válido para los productos de Dji que tengan una única batería. Para obtener los niveles de carga del Matrice 600 es necesario utilizar la propiedad que se representa en la Figura 55 que está relacionada con la gestión de las 6 baterías.

```
/**
 *  Returns the overview of batteries in the battery group. When a battery is not
 *  connected, the `isConnected` property is `NO` and the `chargeRemainingInPercent`
 *  is zero. For Matrice 600, there are 6 elements in this array.
 */
@property(nonatomic, readonly) NSArray<DJIBatteryOverview *> *_Nullable batteryOverviews;
```

Figura 55. Propiedad que devuelve información acerca de las baterías del Matrice 600

Una vez iniciada la misión en el simulador, se comprueba que todo funciona correctamente y que la misión personalizada se ejecuta satisfactoriamente. En la Figura 56 se muestra todo el montaje realizado para probar la aplicación en el Matrice 600.

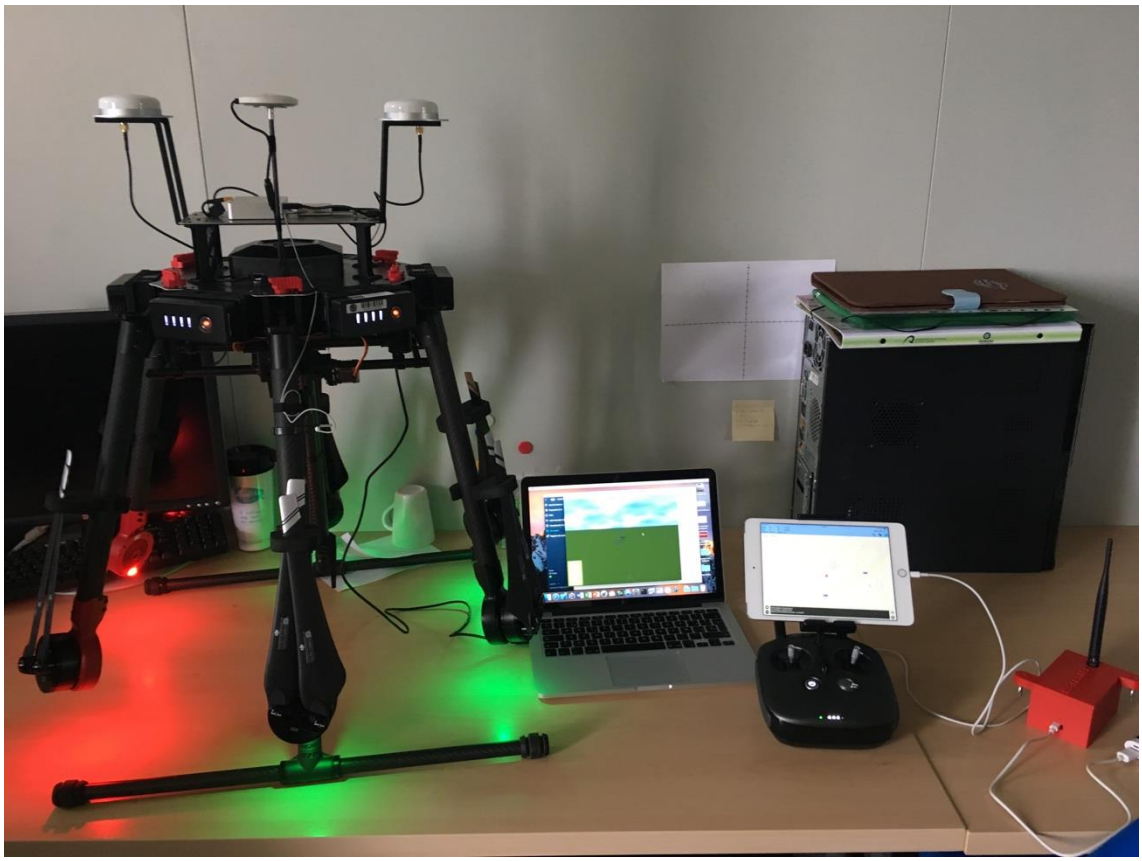

Figura 56. Prueba del sistema en el simulador Dji Assistant con el Matrice 600

4.3. CONCLUSIONES

Una vez finalizado el desarrollo del sistema, la aplicación y los resultados obtenidos anteriormente, se presentan las conclusiones de este Trabajo Fin de Máster:

- ✓ Se ha desarrollado un sistema capaz de posicionar un dron de Dji (en este caso un Dji Phantom 4) sobre unas coordenadas GPS dinámicas mediante un sistema de posicionamiento global que es capaz de trabajar en un rango de distancias superior a los 1400 metros como se ha demostrado gracias a la tecnología LORA, que es más que suficiente para la aplicación agrícola que se desarrolla en el proyecto ENABLE-S3. Con esta solución se consigue hacer frente a las situaciones adversas que pueden ser provocadas debido a baja visibilidad en el terreno como la niebla o el polvo.
- ✓ Se ha desarrollado una aplicación en Xcode con el Mobile SDK de Dji y se ha logrado el seguimiento, la representación y el posicionamiento del dron controlándolo de forma automática mediante la misión desarrollada que se puede modificar con los parámetros establecidos en el menú de opciones, así como otras funciones en la aplicación para escanear, conectar o desconectar los periféricos de los alrededores, y representar el conjunto de puntos de la radio LORA con su información asociada a través de una tabla.
- ✓ La aplicación contempla los errores que se pueden producir durante el vuelo y devuelve el dron a casa en cualquiera de las siguientes circunstancias: se pierde la señal de la unidad transmisora durante un intervalo de tiempo definido en el menú de opciones, el nivel de batería es igual o inferior al 20%, o se produce algún error durante la misión.
- ✓ Los resultados obtenidos en las pruebas son satisfactorios y verifican el correcto funcionamiento del sistema y la aplicación con el Phantom 4, así como la extrapolación de la misma al Matrice 600 para corroborar la correcta ejecución del mismo en esta plataforma de Dji.

Capítulo 5

LÍNEAS FUTURAS

En este capítulo se presentan las líneas futuras que se proponen a partir de este Trabajo Fin de Máster para continuar con los objetivos del proyecto europeo ENABLE-S3:

- Desarrollar un algoritmo que sea capaz de corregir el *offset* del dron durante el vuelo, y a su vez, un método predictivo para que el dron se coloque a una cierta distancia por delante del vehículo agrícola, con la cámara apuntando siempre en la misma dirección.
- Implementar una misión para que mientras el dron realiza el seguimiento del vehículo agrícola, en caso de detectar cualquier anomalía por delante del mismo, baje de forma automática a una cierta altura para analizar la zona con una mayor resolución espacial.
- Integrar esta solución con la electrónica que tiene el vehículo Renault Twizy disponible en la empresa Tecnalia con la cual el IUMA está participando en el proyecto ENABLE-S3. Este vehículo será utilizado como prototipo en el proyecto ENABLE-S3.
- Integrar este software en una plataforma de desarrollo en vuelo que incluya una cámara hiperespectral y que junto algoritmos de detección de anomalías permitan detectar personas o animales para evitar cualquier accidente no deseado frente al vehículo al que se le está haciendo el seguimiento.
- Introducir seguridad en las comunicaciones para evitar conflictos en las misiones que realiza el dron.

BIBLIOGRAFÍA

[1] "Introduction to hyperspectral imaging", en www.microimages.com/documentation/Tutorials, última visita el 08.06.2017.

[2] H. Fabelo, S. Ortega, S. Kabwama, G. M Callico, D. Bulters, A. Szolna, J. F. Pineiro, R. Sarmiento, "HELICOID Project: a new use of hyperspectral imaging foir brain cáncer detection in real-time during neurosurgical operations", SPIE Commercial+ Scientific Sensing and Imaging, Baltimore, EEUU, 17-21 abril 2016.

[3] S Ortega, H Fabelo, R Camacho, ML Plaza, GM Callico, R Lazcano, D Madroñal, R Salvador, E Juárez, R Sarmiento, "Detection of human brain cáncer in pathological slides using hyperspectral images", Neuro-oncology, vol. 19, mayo 2017.

[4] L. Santos, L. Berrojo, J. Moreno, J. F. López, R. Sarmiento, "Multispectral and hyperspectral lossless compressor for space applications (HyLoC): a low complexity FPGA implementation of the CCSDS 123 standard", IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing, vol. 9, num. 2, enero 2015.

[5] A. García, L. Santos, S. López, G. M. Callicó, J. F. López, R. Sarmiento, "Efficient lossy compression implementations of hyperspectral images: tolos, hardware platforms and comparisons", SPIE Sensing Technology + Applications, Baltimorer, EEUU, 5-9 mayo 2014.

[6] M. Teke, H.S. Deveci, O. Haliloglu, S.Z. Gurbuz, U. Sakarya, "A short survey of hyperspectral remote sensing applications in agriculture", 6th International Conference on Recent Advances in Space Technologies (RAST), Estambul, Turquía, 12-14 junio 2013.

[7] R.N. Sahoo, S.S. Ray, K.R. Manjunath, "Hyperspectral remote sensing of agriculture", Current Science, vol. 108, num. 5, marzo 2015.

[8] J. Pedraza, "La revolución que nos dará de comer (y cuidará el planeta)", Periódico El País, 21 abril, 2017.

[9] Floreano, D., & Wood, R. J. (2015). Science, technology and the future of small autonomous drones. Nature, 521(7553), 460-466.

[10] Pham, H. D., Drieberg, M., & Nguyen, C. C. (2013, December). Development of vehicle tracking system using GPS and GSM modem. In Open Systems (ICOS), 2013 IEEE Conference on (pp. 89-94). IEEE.

[11] U-blox, NEO-6 series: https://www.u-blox.com/en/product/neo-6-series, fecha última visita el 4 de julio de 2017.

[12] NMEA DATA: http://www.gpsinformation.org/dale/nmea.htm, fecha última visita el 4 de julio de 2017.

[13] Mega, A. 2560: http://www.mantech.co.za/datasheets/products/A000047.pdf, fecha última visita el 4 de julio de 2017.

[14] Arduino IDE: https://www.arduino.cc/en/Main/OldSoftwareReleases, fecha última visita el 4 de julio de 2017.

[15] Adafruit Feather 32u4 LoRa Radio: https://learn.adafruit.com/adafruit-feather-32u4-radio-with-lora-radio-module/overview, fecha última visita el 4 de julio de 2017.

[16] Technology Company, Bluetooth Low Energy, modelo HM-10, Datasheet V507 (2013): ftp://imall.iteadstudio.com/Modules/IM130614001_Serial_Port_BLE_Module_Master_Slave_HM-10/DS_IM130614001_Serial_Port_BLE_Module_Master_Slave_HM-10.pdf, fecha última visita el 4 de julio de 2017.

[17] Dji Developer: https://developer.Dji.com/, fecha última visita el 4 de julio de 2017.

[18] Allan, A. (2010). Learning iPhone Programming: From Xcode to App Store. " O'Reilly Media, Inc.".

[19] Hernández, F. L. (2012). Objective-C. Curso práctico para programadores Mac OS X, iPhone y iPad. Rc Libros.

ANEXO I

COMANDOS AT

El Bluetooth HM-10 tiene instalado un software que permite trabajar con él a través de los comandos AT (lenguaje de alto nivel). La programación se realiza a través del puerto serie con un rango de tensión entre los 3.3-5V. A continuación, en la Tabla 7 se explican algunos de los comandos más importantes para controlar el Bluetooth HM-10.

Tabla 7. Comandos AT del Bluetooth HM-10

Comando	**Recepción**	**Parámetro**
AT	OK	-
AT+RESET	Ok+RESET	-
AT+VERS?	Versión del módulo	-
AT+NAME? AT+NAME[par1]	OK+NAME[par1] OK+Set[par1]	[Par1]: Nombre del módulo Por defecto: HMSoft Máximo 12 caracteres
AT+TYPE? AT+TYPE[par1]	OK+Get:[par1] OK+Set:[par1]	[Par1]: De 0 a 2 0 → No es necesario el código PIN (Por defecto) 1 → No necesita PIN 2 → Enlaza con el PIN
AT+PASS? AT+PIN[par1]	OK+GET:[par1] OK+Set:[par1]	[par1]: Contraseña Por defecto: 000000 Rango: 000000 - 999999
AT+PWRM? AT+PWRM[par1]	OK+Get:[par1] OK+Set:[par1]	[par1]: Modo sueño 0 → Automático 1 → Manual
AT+SLEEP	OK+SLEEP	-
AT+MODE?	OK+Get:[par1] OK+Set:[par1]	[par1]: De 0 a 2 0 → Modo transmisión 1 → Modo control remoto 2 → Modo 0 + Modo 1
AT+POWE? AT+POWE[par1]	OK+Get:[par1] OK+Set:[par1]	[par1]: Potencia de salida 0 → -23dBm 1 → -6dBm 2 → 0dBm (Por defecto) 3 → 6dBm
AT+CLEAR	OK+CLEAR	Limpiar las direcciones de los dispositivos conectados anteriormente

Comando	Recepción	Parámetro
AT+PWRM? AT+PWRM[par1]	OK+Get:[par1] OK+Set:[par1]	[par1]: Modo sueño 0 → Automático 1 → Manual
AT+RENEW	OK+RENEW	Restauración de fábrica
AT+BAUD? AT+BAUD[par1]	OK+Get:[par1] OK+Set:[par1]	[par1]: Velocidad en Baudios del bus. 0 → 9600 (Por defecto) 1 → 19200 2 → 38400 3 → 57600 4 → 115200 5 → 4800 6 → 2400 7 → 1200 8 → 230400
AT+ADDR?	OK+ADDR:	Dirección MAC del módulo
AT+PAR? AT+PARI[par1]	OK+Get:[par1] OK+Set:[par1]	[Par]: Paridad del bus 0 → Ninguno (Por defecto) 1 → Par 2 → Impar
AT+CON[par1]	OK+CONN[par2]	[par1]: Dirección del dispositivo [par2]: A, E, F A → Conectando E → Error al conectar F → Conexión fallida
AT+TCON? AT+TCON[par1]	OK+TCON:[par1] OK+Set:[par1]	[par1]: Tiempo de espera Por defecto: 000000 Rango: 000000 – 999999 (ms)
AT+ROLE? AT+ROLE[par1]	OK+Get:[par1] OK+Set:[par1]	[par1]: 0 ó 1 1 → Periférico 2 → Central
AT+RSSI?	OK+RSSI: Señal	-
AT+NOTI? AT+NOTI[par1]	OK+Get:[par1] OK+Set:[par1]	[par1]: Notificaciones 1 → No (Por defecto) 1 → Si
AT+HELP?	Información de ayuda	-
AT+FILT? AT+FILT[par1]	OK+Get:[par1] OK+Set:[par1]	[par1]: 0 ó 1 0 → Filtra el comando AT 1 → No filtra el comando AT
AT+START	OK+START	-