Localización de drones utilizando infrarrojos y RFID y creación de una API-JSON en el proyecto Games of Drones

Ander Pardo Martínez

Resumen—Actualmente, el mundo de las IoT (*Internet of Things*), la industria *gaming* y la industria de los drones están despegando con nuevas ideas, nuevos productos, nuevas invenciones y un mercado creciente. Algo que hace 10 años era desconocido para la mayoría, ahora mueve grande sumas de dinero alrededor del mundo. Por este motivo, el propósito de este proyecto es crear un producto el cual pueda contener un poco de estos tres mundos, *Game of Drones* es el resultado de esta unificación. Un proyecto originalmente propuesto por una empresa especializada en drones de carretas. El juego consiste en recrear el modo de juego comúnmente conocido en los videojuegos como 'captura la bandera', donde dos equipos combaten entre ellos para para capturar una serie de bases ganando puntos por ello, al final, el equipo con más puntos gana la partida. Este *paper* es uno de cuatro *papers* donde se explicará el desarrollo de una de esas bases. Seran expuestas y comparadas algunas de las tecnologías actuales y finalmente seleccionadas y desarrolladas las que mejor encajen en el proyecto.

Palabras clave— RFID, IRDA, Arduino, STM32, Microcontrolador, API JSON, Web, REST

Abstract—Nowadays the IoT world, the gaming and the drone industry are taken off with lots of new ideas, new products, new inventions a new and expanding market. Something which 10 ago was unknow for most, now moves big sums of money around the globe. For this reason, the aim of this project is to create a product that can have a bit of those three worlds, Game of Drones is the result of this unification, a project originally proposed by a specialized company on racing drones. The game consists on recreate a battle mode commonly named in videogames as *capture the flag* where two teams battle against them to catch a series of bases and earning points for that, at the end, the team with the most points will be the winner. This paper is one of four where it will be explained the development of one of those bases. There will be exposed and compared some of current technologies and finally selected and developed the ones that best fit this project.

Index Terms— RFID, IRDA, Arduino, STM32, Microcontoller, API JSON,Web, REST,

1 Introducción

ESTE TFG, juntamente a otros 3 tiene el propósito de recrear partidas de videojuegos de "capturar la bandera" implementándolo en drones reales. Esta propuesta originalmente nos vino de parte de una empresa especializada en el negocio de los drones de carreras.
Este sistema es capaz de simular una batalla entre drones, los cuales se "dispararán" entre ellos, capturarán bases con el fin de conseguir la máxima puntuación. Con este fin se ha dividido el proyecto en 4 partes, una de manejo del dron, otra de gestión de disparos y vida del dron, otra de captura de las bases y por último el control de todas ellas mediante un ordenador central que a su vez gestiona el juego.

Esta parte se ha centrado en el desarrollo de un prototipo de base que sea capaz de detectar si dentro de ella se encuentra un dron, identificarlo y comunicarlo al ordenador central, además, para mejorar la jugabilidad, esta cambia de colores en función de quien tenga la posesión de la misma.

- *E-mail de contacto: ander.pardo.martinex@e-campus.uab.cat*
- *Mención realizada: Ingeniería de Computadores + Electrónica*
- *Trabajo tutorizado por: Marius Montón Macias (Microelectrónica)*
- *Curso 2017/18*

2 Objetivo

El principal objetivo de este trabajo es crear un prototipo de base la cual nos pueda detectar que un dron está conquistándola. Para este propósito, se debe de tener en cuenta varios factores. El primero, es la infraestructura de la misma, la cual es un dodecaedro, por lo tanto un espacio claramente delimitado y de dimensiones no demasiado grandes; El segundo, el coste de la implementación del mismo, ya que existen muchos métodos de *trackeo* es necesario escoger una opción que no suponga un coste demasiado elevado para ser implementado, pero que permita

implementar un prototipo funcional, por ello se deberá tener en cuenta las ventajas que nos supone una base con los límites bien definidos; Tercero y último, es la latencia que esta aporte al juego, basándonos en la opinión de un experto en la industria se ha limitado la máxima latencia a 30 ms, para poder dotar al juego con la jugabilidad deseada.

Además, debido a que este juego está destinado a ser jugado en un espacio el cual no estará libre de interferencias, se ha dotado a la base de dos sistemas de control para verificar la posición de un dron, esto puede ser visto como un doble *check*. Los sistemas que se han implementado son el uso de Infrarrojos y RFID.

Por último, para poder permitir que des de fuera del juego se puedan implementar otras aplicaciones se ha desarrollado una API del tipo REST la cual nos permite obtener los datos que se generan los datos de la partida. Esta devuelve un fichero tipo JSON con los valores demandados.

3 Metodología

Para el desarrollo del este TFG se ha seguido una metodología del tipo scrum la cual se basa en reuniones periódicas cada una o dos semanas. En ellas hemos intervenido los 4 integrantes del proyecto, así como el tutor del mismo y un experto en el campo.

Esta metodología permite, entre otras cosas, el poder controlar el progreso individual de cada uno y poder definir de forma clara y sencilla los objetivos a cumplir a corto plazo, además también permite comprobar la integración entre los sistemas de cada integrante del grupo.

En la parte 4 del este trabajo se explica el estado del arte de las tecnologías utilizas. En el apartado 5 se hará una comparativa entre posibles tecnologías aptas para el proyecto. En el apartado 6 se describe la implementación y desarrollo del sistema. En el 7 y el 8 se exponen los resultados y se deducen las conclusiones del trabajo. Por último, en el apartado 9 se explican los posibles pasos a seguir para mejorar el proyecto en un futuro.

4 Estado del arte

En este paper se discuten y se desarrollan varias tecnologías las cuales están muy integradas con el mundo de las IOT, pero que hace unos años tenían otros propósitos.

- Bluetooth: Des de diciembre del 2016 esta tecnología cuenta con la versión 5.0 la cual nos ofrece más rango y más velocidad que sus predecesoras, está especialmente diseñada para dar soporte a aplicaciones de IoT y ya se encuentra disponible en algunos Smartphones.
- FPV: Este tipo de video se basa en la recepción des de un lugar remoto la visión que tiene algún objeto. Es de especial utilidad en el manejo de aviones o drones no tripulados ya que así se puede tener prácticamente el mismo control que si estuviéramos dentro de este.
- Ultra Sonido: Esta tecnología comúnmente se ha utilizado para medir distancias, pero, es en el mundo de la medicina dónde más se ha desarrollado.
- RFID: La gran ventaja de este sistema es que permite la compra de etiquetas RFID a un coste muy bajo y estas al estar tan miniaturizadas se pueden utilizar en una gran variedad de ámbitos cuotidianos, entre ellos se pueden destacar: seguimiento y autenticación de artículos de ropa, seguimiento de paquetes en almacenes o contenedores en puertos, identificación de mascotas mediante la aplicación subcutánea, control de acceso a edificios y pagos con tarjeta ente otros.
- GPS: Esta tecnología la encontramos en ámbitos tanto civiles como militares, ofrece cobertura a nivel mundial y consta de una constelación de como mínimo 24 satélites.
- Infrarrojos: Estos van des de comunicaciones de bajo consumo a velocidades de 9.600 bps hasta los 100Mbps, se pueden llegar a utilizar para el envio de archivos multimedia pero su uso más común es como mando a distancia.

5 Estudio previo

Primeramente, se tuvo que definir qué tipo de tecnología o tecnologías se podrían utilizar para la gestión de la base. Las posibles candidatas fueron los infrarrojos, el Bluetooth, ultrasonidos, etiquetas RFID, la video Recepción y el GPS.

5.1 Bluetooth

Esta conocida tecnología está ampliamente extendida y no es difícil encontrar información o proyectos relacionados con ella. Se basa en un enlace entre dispositivos mediante el uso del espectro de 2.4 GHz de radiofrecuencia.

En cuanto a la transmisión de datos no hay problema ya que de estas versiones tempranas como esta supera 1Mbit/s, velocidad más que aceptable para el envío del orden de las decenas de bits como sería en nuestro caso (estaríamos en torno las decenas de nanosegundos) para la transmisión de un mensaje.

El mayor problema que se observó con esta tecnología es su largo alcance (10 metros aproximadamente) comparado con el de la base (1-2) metros. Y el hecho que la emisión de la misma suele hacerse con antenas omnidireccionales y por lo tanto no nos podría delimitar la posición del dron en el área deseada.

5.2 FPV

La principal ventaja de este método es que se necesitaría acoplar ningún dispositivo extra al dron y sin embargo podríamos identificar de forma precisa a cada dron.

Esto se debe a que, para el control de los drones, especialmente los de carreras, se recurre al uso de la tecnología FPV (*First Person View*) para que el piloto des de el suelo pueda ver lo que "ve" el dron.

La FPV se basa en la emisión de video con un ancho de banda de unos 100MHz a frecuencias de 5.8GHz, de forma que se asocia un determinado ancho de banda a un emisor concreto y, por ende, a un dron concreto.

5.3 Ultra Sonido

Esta tecnología se utiliza comúnmente para medir distancias según el tiempo que tarde la señal en propagarse, rebotar contra el objeto que tenga delante y recibirla de

nuevo. Respecto a la direccionalidad y alcance del mismo, se adecua bien a las necesidades de la base, ya que se propaga en un rango de ± 30º como se puede observar en la ilustración 1 y cuentan con un rango que va des de pocas decenas de centímetros hasta distancias cercanas al metro/2 metros.

El principal problema que tiene es que no está pensado para la transmisión de paquetes. A pesar de esto se puede llegar a crear un protocolo para enviar datos a través de ultrasonido, esto lo vemos en alguna aplicación destinada a los *Smartphones*. Pero el desarrollo con Arduino sería demasiado complejo comparado con otras soluciones. Por lo tanto, utilizaríamos este sistema únicamente este sistema que algún objeto se encuentra dentro de la base, por lo tanto, nos obliga a la utilización de un segundo sistema para poder identificarlo.

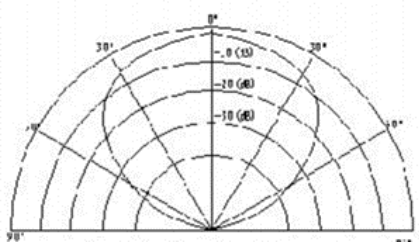


Ilustración 1. Diagrama de radiación de una antenna de ultra sonido

5.4 RFID

Esta tecnología se basa en la detección de unas etiquetas o tags mediante el envío de una ID asociada a cada uno de ellos. Estas etiquetas según su tecnología serán más pequeñas o más grandes, pero, en cualquier caso, estamos en torno los gramos/decenas de gramos, por lo que no nos supondría un peso a tener en cuenta para el dron que la lleve encima.

Estos tags actúan de antenas y permiten el envío del código identificador del mismo, además, poseen una pequeña memoria interna que varía entre las decenas bytes y los kilobytes, en este caso pero, este factor no es relevante para nuestro proyecto ya que solamente nos hace falta el envío de un identificador único para cada tag. El funcionamiento de este sistema se basa en 3 elementos principales estos son: un transceptor (comúnmente denominados *tags* o etiquetas) una antena y un lector.

- **Etiqueta o *tag***: Esta está dotada de un pequeño circuito y una antena los cuales le permiten enviar los datos que almacena en su pequeña memoria interna.
- **Antena**: Esta se encarga de emitir un campo a su alrededor el cual delimitará el radio de acción de los tags según su alcance. Además, también es la encargada de recibir la señal emitida por el *tag*.
  Estas antenas se clasifican según su patrón de radiación, el cual nos delimita el radio de acción comentado.
  - Según su alcance:
    - De largo alcance: Trabajan con frecuencias de RFID UHF (véase a continuación) y tienen un rango de acción de hasta 14 metros.
    - De corto alcance: Trabajan con frecuencias menores a las anteriores y tienen un alcance que suele llegar hasta 1,5 – 2 metros
  - Según su densidad de campo:
    - Alta densidad
    - Baja densidad
- Etiqueta o *tag*: Deben este nombra a que comúnmente estos circuitos suelen estar tan miniaturizados (incluso impresos) que estos se encapsulan como pegatinas o tarjetas. En su interior contienen una antena que le proporcionará la energía captada de las ondas electromagnéticas emitidas por la antena anteriormente comentada y, a su vez, servirá para emitir los datos almacenados en la memoria.

Existen varios tipos de RFID, los cuales se clasifican según el ancho de banda del espectro que utilizan y tienen asociadas una máxima potencia permitida. Estas frecuencias pueden variar ligeramente en función del país dónde se utilicen:

- *LF* - Baja frecuencia: de 125 y 134.2 KHz.
- *HF* - Alta frecuencia: 13,56 MHz.
- *UHF* - Frecuencia ultra-elevada: de 865 a 928MHz.
- *SHF* - Frecuencia Super altas (microondas): 2,4 GHz y 5,8GHz.

Además, según la fuente de alimentación que utilicemos, nos encontramos principalmente tres tipos de antenas, activas, pasivas y semipasivas:

- **Pasivas**: Obtienen la energía mediante la inducción de un campo magnético generado por la antena. Este campo magnético a su vez induce uno eléctrico en el circuito de la etiqueta y esta entonces puede enviar la información almacenada en ella a través de la antena.
- **Activas**: Estas etiquetas obtienen la alimentación de una pila que llenas incorporada, de forma que les otorga una gran potencia y son capaces de enviar la señal a más de 100 metros de distancia, y debido que el consumo de estas antenas es mínimo, estas baterías tienen una muy larga duración, (pueden durar años). Además este tipo de tags perite el establecimiento de sesiones entre la etiqueta y el lector.
- **Semipasivas**: Disponen de una pequeña alimentación y combinan las dos anteriores.

Observando el mercado actual vemos que tanto para las etiquetas semipasivas como las activas se podrían acoplar bien al sistema deseado, pero, dado que estas tienen un coste bastante elevado respecto los kits de etiquetas pasivas (en torno los 10€ para las pasivas frente aproximadamente 100€ para las otras).

5.4.1 RTLS

Esta aplicación de RFID en crear una constelación de antenas idénticas repartidas de forma equidistante entre ellas para poder delimitar una zona de "seguimiento". Este método se basa en la estimación de la posición de un cierto objeto según la probabilidad de pérdidas de paquetes entre el emisor y el receptor.

No nos centraremos mucho en el estudio de este sistema ya que su complejidad y su coste lo harían inviable para un proyecto como este. Pero podemos entender de forma aproximada como funciona si tenemos en cuenta que la probabilidad de recibir un paquete enviado por una etiqueta RFID a una distancia "d" ($p(d)$) disminuye a medida que nos alejamos de la antena. Obteniendo una esperanza de detectar un paquete a una distancia "d" sigue la forma

$$E(p(d)) = \frac{1}{1 + e^{a*(d-d_0)}}$$

Siendo:

- a: parámetro tomado de el estudio de Stony Brook
- d_0: distancia des de cuya probabilidad de detección se corresponde a 0,5

El principal problema de implementar este sistema es que tiene un coste de materiales demasiado elevado y es que es demasiado complejo para una aplicación tan sencilla en comparación como es la de situar únicamente los drones dentro del espacio de la base

5.5 GPS

El GPS o Sistema de Posicionamiento Global se basa en la trilateración de la posición gracias al uso de hasta 32 satélites.

Esta tecnología tiene la principal ventaja que toda la infraestructura ya viene dada, y nosotros solamente necesitamos de un sensor de GPS conectado a un microcontrolador para poder empezar a seguir, en este caso nuestros drones.

Por contra, es un sistema el cual sigue siendo bastante impreciso para aplicaciones como esta ya que dispone de una precisión en torno a los 2-3 metros, sin contar con la gran latencia que nos puede aportar al juego, lo cual se transmite en errores muy grandes en comparación con el terreno de juego si tenemos en cuenta que estos drones son capaces de alcanzar los 100Km/h. También, cabe destacar el alto consumo del propio GPS, lo que nos obligaría a ampliar las baterías del dron, el cual tiene una gran limitación de peso y espacio.

5.6 Infrarrojos

Los Infrarrojos están basados en la emisión de ondas electromagnéticas las cuales se sitúan entre la luz visible y las microondas. Estas se clasifican en dos tipos según si están más cerca o no del espectro visible:

- Luz infrarroja cercana (del inglés Near infrared)
- Luz infrarroja lejana (del inglés Farinfrared)

Para la comunicación mediante el uso de esta tecnología normalmente se recurre al uso de los protocolos RC-5 y SIRC, creados por Philips y Sony respectivamente.

En cuanto a la radiación de los mismos vemos que es la opción más direccional de todas ya que están entre los ±10º y los ±15º (Ilustración 2).

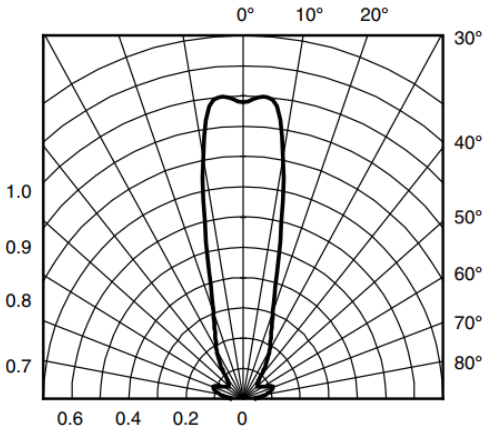


Ilustración 2. Diagrama de radiación de un LED infrarojo

5.6.1 RC-5

Este protocolo se basa en el envío de 128 comandos distintos, está destinado a utilizarse en equipos multimedia, ya que un mismo comando actuará de la misma forma sobre diferentes dispositivos (ej. Subir/bajar volumen). Utiliza una la modulación BPSK sobre una onda portadora de 36KHz con una codificación Manchester estándar utiliza palabras de 14 bits.

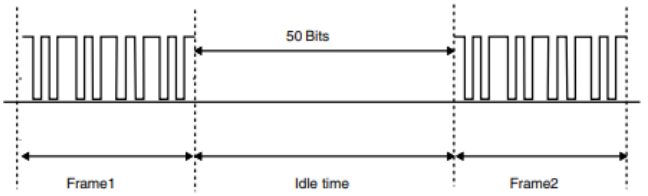


Ilustración 3.Gráfica en función del tiempo del envío consecutivo de 2 paquetes por RC-5.

La duración de cada bit es de 1.778ms haciendo una duración total de la palabra de 24.892 ms. Los 14 bits de la palabra se dividen en los siguientes campos:

- ***Start bit***: 1 lógico, duración de 1 bit
- ***Field bit***: duración de 1 bit
  - 1 lógico si se el comando enviado está comprendido entere el 0 y el 63 decimal.
  - 1 lógico si se el comando enviado está comprendido entere el 64 y el 127 decimal.
- ***Control bit:*** Este campo es útil a la hora de programar un mando a distancia ya que nos indica si se ha pulsado una vez el botón o se mantiene pulsado. 1 bit de duración
- ***Address:*** Dirección del dispositivo al que va dirigido el mensaje. 5 bits de duración
- ***Command***: junto con el bit "Field Bit" nos indica que número de comando se está enviando.

Este protocolo, además, para evitar colisiones tiene un

tiempo de reposo de 50 bits, como podemos observar en la Ilustración 3 que equivale a 88.9 ns.

Del estudio de este protocolo, aún y obviando este tiempo de reposo, podemos deducir que no nos es válido para el proyecto ya que a pesar de que se podría tratar de reutilizar los comandos del protocolo y adaptarlos a nuestro sistema, la latencia (24.892 ms) es excesiva para poder satisfacer las del proyecto.

5.6.2 SIRC

Este protocolo tiene 3 versiones del mismo según la longitud de la palabra que se envíe, estás son de 12, 15 y 20 bits. El código es transmitido mediante una portadora de 40 KHz utilizando una codificación de distancia de los bits. Se establece la duración de cada período de 600 us de forma que el '1' tiene una duración de 3T mientras que el 0 es de 2 T, tal y como se puede observar en la Ilustración 4:

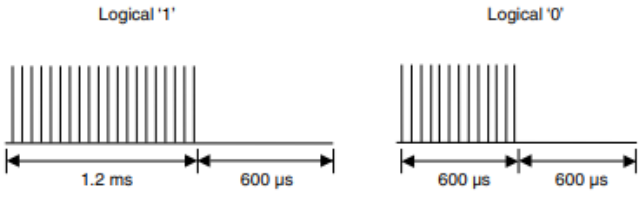


Ilustración 4. Representación del 1 y el 0 lógico para el protocolo SIRC

Los campos de este protocolo son los siguientes:

- ***Start Bit*:** Se basa en enviar un pulso de duración 4T seguido de un 0 de duración T
- ***Comand***: Comando a enviar de longitud 7 bits
- ***Adress:*** dirección del dispositivo de longitud 5 bits

Como se puede observar es muy similar al protocolo RC-5, pero con una duración variable. Para poder hacernos a una idea, estimamos el valor medio de la palabra para este protocolo:

$$T_{min} = 5T + (7+5)T_0 = 5 * 600\mu s + 12 * \ 1200\mu s = 17.400\ \mu s = 17{,}4\ ms$$

$$T_{max} = 5T + (7+5)T_1 = 5 * 600\mu s + 12 * \ 1800\mu s = 17.400\ \mu s = 24{,}6\ ms$$

$$T_{med} = (T_{min} + \ T_{max})/2 = (17{,}4\ ms + \ 24{,}6\ ms) = 21\ ms$$

Donde T_0 equivale al tiempo necesario para enviar un '0' lógico y T_1 al envío de un '1' lógico.

A la vista de estos resultados es evidente que tampoco se satisface el requisito del tiempo necesario para poder obtener una buena jugabilidad.

5.6.2 IrDA

Este estándar se basa en el uso de infrarrojos para el envío de datos entre dos dispositivos que se encuentren a una corta distancia (en torno 1 metro).

Este estándar está organizado en capas en capas. Se clasifican según si son obligatorias o no.

Dentro de las obligatorias se encuentran:

- IrPHY (*Phyical Signaling Layer*): El nivel más bajo de todas corresponde a la capa física, establece la distancia máxima y la velocidad de transmisión En esta definimos los 4 tipos de enlaces IR disponibles:
  - SIR (*Serial Infrared*): Velocidades soportadas por el puerto RS-232, hasta 115,2Kbps.
  - MIR *(Medium Infrared*): Des de 0,576 Mbps a 1,152Mbps.
  - FIR *(Fast Infrared*): Hasta 4 Mbps
  - VFIR (*Very Fast Infrared*): Des de 115,2Kbps a 14 Mbps
  - UFIR (*Ultra Fast Infrared*): Hasta 100Mbps.
- IrLAP (*Link Access Protocol*): Capa de enlace, en esta se establecen las conexiones entre los dispositivos. Estos dispositivos se dividen entre uno primario y otro o varios secundarios.
- IrLMP (*Infrared Link Management Protocol*): Esta es la se divide en dos partes:
  - LM-MUX (*Link Management Multiplexer*): Permite la multiplexación el cambio de dispositivo primario a secundario y viceversa.
  - LM-IAS (*Link Management Information Access Service*): Crea una lista de servicios.
- Tiny TP: Mejora la conexión y transmisión de los datos gracias al SAR (*Segmentation and Reassembley*)
- IrCOMM: Permite al canal trabajar de forma simultánea o paralela.
- OBEX (*Object Exchange*): Necesita *Tinny TP*. Este posibilita el intercambio de datos arbitrários.
- IrLAN (*Infrared Loacal Area Network*): Necesita *Tinny TP*. Permite conectarse por infrarrojos a una red de area local.

Todas estas capas las podemos ver esquematizadas en la Tabla 1 (en rojo las obligatorias y en amarillo las opcionales).

Tabla 1. Capas del estándard IrDA

IAS	irLAN	OBEX	IrCOMM
	Tiny TP		
IrLMP			
IrLAP			
Capa física			

Dado que el Infrarrojo irá encima del dron, y, por lo tanto, se utilizará una placa STM32 dado que esta placa contiene soporte para IrDA y UART, además de otras características necesarias para gestionar el disparo entre drones, la cual solamente nos deja configurar el IrDA en modo SIR (hasta 115,2 Kbps), y basándonos en la información proporcionada por el fabricante *Vishay* estaremos en torno a un tiempo mínimo de pulso de unos 1,63 us para un pulso (1 bit) por lo que por ejemplo, un mensaje de unos 20 caracteres tendría una duración aproximada de unos 2,6 ms. Tiempo el cual nos permite estar dentro del tiempo de latencia máximo.

5.7 Resumen

Como podemos observar en la Tabla 2 podemos concluir

que las opciones más validas son el NFC, el IrDA, y la video-recepción. Pero, como únicamente necesitamos dos sistemas para implementar un doble *check* descartamos el de videorecepción ya que no están disponibles tantas librerías e información.

Tabla 2. Tabla con los datos más relevantes a la hora de escojerlas

	Rango	Direccionalidad	Tipo comunicación	Precio (€)
RFID UHF	2-2.5 m	180º	Unidireccional	~90
RFID NFC	3 cm	180º	Unidireccional	~10
IR	25 m	± 15º	Unidireccional	~5
BT	9 m	360º	Bidireccional	~10
US	30 cm	± 30º	Unidireccional	~10
VR	≥ 88dBm	± 360º	Unidireccional	~10

6 DESARROLLO

En este apartado se describirán los distintos módulos implementados para poder cumplir con el objetivo principal de hacer una base la cual nos detecte y notifique de la presencia de un dron.

6.1. Base

Dado que este es un proyecto en común con otros tres, la parte de infrarrojos se hará juntamente con el proyecto de *gestión de disparo interdrónico*. De forma que el Dron no lleve 2 microcontroladores distintos más los sensores infrarrojos de los necesarios.

La idea de utilizar infrarrojos en la base hace que esta tenga un comportamiento muy similar la de un dron. Cuando un dron entre en la base, este "disparará" a la base y esta se lo comunicará al servidor del juego.

Por otro lado, se ha escogido un sensor de RFID gestionado por un Arduino para obligar al dron a estar un cierto tiempo dentro de la base y poder capturarla y, una vez estado el tiempo suficiente comunicárselo al ordenador central. El hecho que se haya utilizado un sensor RFID ha sido debido a que para realizar un pequeño prototipo es suficiente y únicamente deberíamos cambiarlo por un sistema más potente, y adaptarnos a posibles cambios en librerías a nivel de programación en caso de querer utilizar un sensor más "profesional".

El propósito de esto es tener un doble check para asegurarse que el dron se encuentra dentro de la base y no simplemente ha pasado cerca suyo. Y que además ha tenido que "perder" un determinado tiempo mientras para poder capturarla.

Esto hecho le añade más similitudes respecto al original "capturar la bandera" que vemos en algunos videojuegos, en los que vemos que para querer capturar un punto debemos exponernos al enemigo por un breve tiempo.

El comportamiento de la base podemos describirlo como un elemento pasivo, ya que este solamente se encargará de notificar al servidor central la presencia por dos medios distintos la presencia de un dron, pero en ningún momento toma otra decisión. Pero, una vez que el servidor del juego decide si una base pertenece a un equipo o a otro le notificará un cambió de color a la misma.

Este cambio de equipo de la base se simulará mediante el uso de 3 leds distintos, uno para indicar que la base no pertenece a ninguno de los dos equipos, y otros dos para indicar que le pertenece a uno de ellos.

6.1.1 RFID

La implementación del RFID se ha hecho mediante un Arduino y el sensor **MFRC522**. Se ha decidido cambiar de placa ya que este sensor además de tener la antena y el lector incorporados en la misma placa también goza de un buen soporte en cuanto a información y librerías.

Este lector tiene soporte para la lectura con el estándar ISO/IEC 14443 del tipo A, lo que significa que opera a 13.56MHz y que por lo tanto debemos utilizar tarjetas válidas para este ancho de banda, también permite usar etiquetas NTAG, las cuales tienen un funcionamiento muy similar, pero pudiendo imprimir circuitos en pegatinas del tamaño de una monea.

Las tarjetas una forma muy parecida a las típicas tarjetas de crédito.

6.1.1.1 Conexionado

Para comunicar el módulo MFRC522 a la placa Arduino se dispone de 2 protocolos para efectuar la comunicación.
El primero se trata del SPI (Serial Peripheral Interface) Este se basa en un host que actúa como master y una serie de periféricos que actúan como Slaves (para nuestro caso el master será la placa Arduino y el único Slave el lector).

El segundo es el I^2C, este protocolo también se basa en una comunicación máster-esclavo, con la diferencia que este permite la existencia de varios *masters* y la comunicación entre ellos. Para el caso de la base no es demasiado destacable esta última característica ya que solamente tenemos un microcontrolador y un sensor.

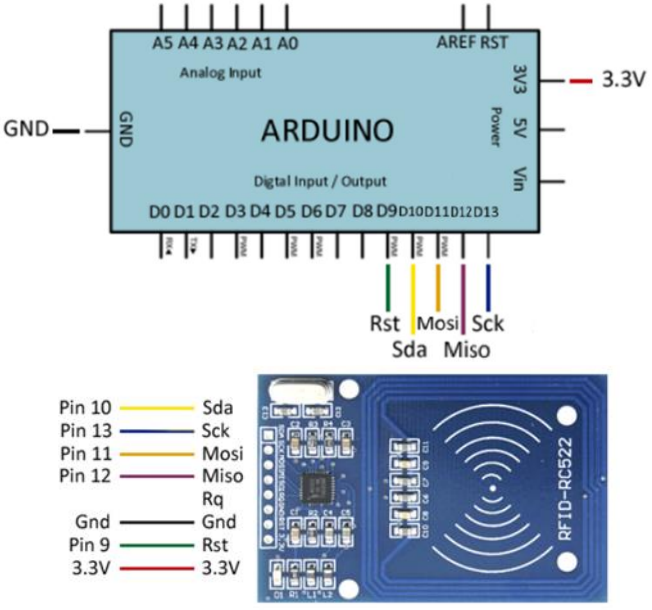


Ilustración 5.Conexionado RFID

Debido a las librerías que se utilizarán, se ha utilizado

el protocolo SPI. Cabe destacar que el pin físico SDA del sensor se comporta como pin SS para el modo SPI, como se puede apreciar en la Ilustración 5.

6.1.1.2 Librerías

Las librerías más relevantes que se han utilizado para este proyecto son la ***SoftwareSerial*** y la ***MFRC522*** ambas creadas específicamente para Arduino.

La funcionalidad que nos ofrece la primera es la configuración de los puertos de salida y entrada de la UART con 3 simples funciones:

- **mySerial()**: Nos permite definir la velocidad de comunicación (en bits/s) para la UART.
- **begin(rx,tx)**: Configura los pines de entrada y salida para una comunicación tipo serie.
- **serialEvent():** *CallBack* la cual saltará cuando recibamos algo por el puerto serie.

La segunda, MFRC522, está desarrollada para dar soporte a un protocolo de comunicación del tipo SPI (necesitaremos también esta librería). De esta librería cabe destacar las funciones:

- **PICC_IsNewCardPresent()**: Nos detecta si se ha detectado una tarjeta en el lector.
- **PICC_ReadCardSerial():** Guarda en la clase mfrc522.uid el identificador de la tarjeta detectada. Este identificador es único en cada tarjeta. Utilizaremos este código para poder definir que dron se ha detectado.

Para poder asociar cada una de estas tarjetas a su correspondiente dron se ha creado un código (lector.ino) el cual nos devuelve por la UART el código que lee. De forma que solamente debemos guardar en una librería el código de estos para poder almacenar el ID asociado a cada dron. Como se puede apreciar en la Ilustración 6, este fichero (Players.h) nos almacenará el listado de identificadores:

```
#define NUMDRONES 2
String DRONES[NUMDRONES]={"b60a159","1901d28"};
```

Ilustración 6.Libreria con listado de drones

6.1.1 Pruebas

Una peculiaridad de la base es que a pesar de que tengamos una latencia elevada a la hora de controlar si un dron se halla en ella es que podemos compensar este tiempo restándoselo al tiempo que debe permanecer en la base para capturarlo, e incluso obviarlo, ya que la latencia es del orden de los milisegundos frente a los segundos que se necesita para conquistar la base.

A pesar de esto, se han realizado dos pruebas para comprobar la latencia de la base:

- Tiempo des de que se detecta la presencia de un dron hasta que enviamos el mensaje de captura:
  - creamos un pulso cuando detectamos la función **PICC_IsNewCardPresent()** y generamos otro a la entrada del contador.

$$Tmedio = 10.71ms$$

- Tiempo des de que se conquista una base hasta que recibimos el mensaje de cambiar de color:
  - Se envía un pulso cuando se envía el mensaje por puerto serie de conquista de la base por el y otro justo después de leer el mensaje, decodificarlo y comprobar que color se ha de encender en la base.

$$Tmedio = 0{,}57ms$$

6.1.2 IrDA

Finalmente se ha optado por la opción del IRDA ya que esta opción se adecua a nuestras necesidades, además de que permite usarla tanto para la conquista de las bases como el disparo entre drones.

Esta parte, como bien ha sido mencionado en el apartado 5.6.2 se ha realizado en una placa STM32 ya que esta proporcionaba tanto soporte de IrDA como de UART y del resto de componentes necesarios para gestionar el disparo entre drones. El fabricante de esta, ST Micoelectronics, nos proporciona un conjunto de librerías y configuración de pines y RTOS, los cuales han ayudado a desarrollar el proyecto.

La estructura que se ha seguido para el IrDA ha sido la misma que se tiene en el proyecto *Disparo Interdrónico* ya que esta ha necesitado de otros elementos como el RTOS, la gestión de los buffers y la UART.

Dado que la IrDA actúa como una UART, las funciones definidas para esta serán muy parecidas a las de la UART.

Para el caso de la base, las funciones más importantes son las de escribir y leer. Aunque para el caso de la base la única que necesitamos es la de lectura y la escritura por la UART.

6.1.2.1 Conexionado

La idea original era conectar la STM32 a un transceptor IrDA, pero debido a que el encapsulamiento de este es de pocos milímetros se ha necesitado enviar a un experto para soldarlo y, debido a que el primer prototipo no llegó a ser funcional y faltó tiempo para soldar otro transceptor, para implementar el prototipo se ha recurrido al conexionado de con una placa *Cyclone II* la cual lleva integrado un transceptor IrDA los pines del cual pueden ser puenteados fácilmente hacia pines exteriores.

El transceptor escogido originalmente fue el CND0313A (véase el anexo A4), pero debido a que no disponía de un circuito de adaptación bien definido no se consiguió hacer funcionar.

El segundo transceptor escogido es el TFDU4101 ya que este cuenta con un circuito de adaptación mejor definido, pero, como se ha mencionado antes no ha sido posible implementar.

6.1.2.2 Prueba de radiación

Para verificar el radio de actuación del Infrarojo se ha ejecutado un código el cual nos hacia parpadear un LED de la placa STM32 cuando recibía un mensaje de IrDA, de forma que a menos potencia recibida más probabilidad de perder partes o mensajes enteros. Se ha hecho un conteo a varias distancias y varios ángulos y obtenemos el siguiente diagrama de radiación:

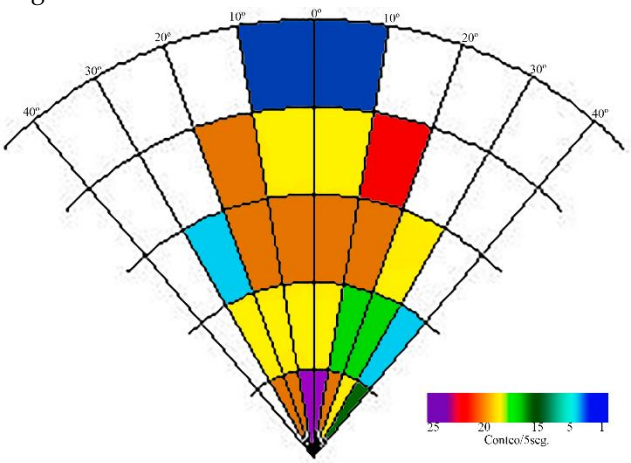


Ilustración 7. Diagrama de radiación obserbado mediante el conteo de mensajes recibidos entre infrarrojos.

6.1.2.3 Pruebas de latencia

A diferencia de la latencia del RFID, para el caso de la latencia del IrDA, si que era de especial importancia tener la mínima latencia ya que el dron los utilizará también en el disparo a otros drones. Las pruebas realizadas han sido las siguientes:

- Tiempo que se tarda en procesar una orden:
  - Medimos el tiempo que pasa des de que una vez que se ha decodificado una orden por la UART hasta que se acaba de enviar otro mensaje por el IrDA.

$$Tmedio = 2{,}16\ ms$$

- Tiempo de latencia de disparo:
  - Tiempo des de que se recibe un mensaje por la UART de un dron de disparar, este procesa la orden, envía un disparo por IrDA, un segundo dron recibe el disparo, lo decodifica y envía un mensaje por UART de que le han disparado.

$$Tmedio = 9{,}6\ ms$$

6.2 API-JSON

Debido a que el juego se monitoriza des de una web, se ha considerado que seria una buena opción crear una API que devuelva un JSON con los valores que se visualizan en la web

Esta API esta basada en el modelo REST (Representational State Transfer) el cual nos permite tener una aplicación fácilmente escalable. Esta se basa en una serie de puntos los cuales la deginen como tal:

- Protocolo cliente/servidor sin estado
- La gestión de datos se hace mediante 4 métodos:
  - **POST**: Crea un dato.
  - **PUT**: Edita un dato.
  - **GET**: Hace una petición de lectura sobre un dato.
  - **DELETE**: Elimina un dato.
- Todas estas operaciones se hacen mediante la URI
- Uso de Hipermedios, el sistema de envío de datos e información se hace mediante el el uso de objetos del tipo HTML o XLM.

En esto se basa el principio HATEOAS, el cual también ha sido aplicado, por el cual el cliente no necesita tener información previa de la arquitectura del sistema ya que además de los datos solicitados por el cliente se enviará información de enlaces a los cuales puede acceder.

A pesar de que como se ha mencionado una API REST-FULL utiliza cuatro métodos distintos para la gestión de los datos, solamente implementaremos métodos GET, de forma que aplicaciones de terceros solo puedan recibir información de la partida que se está desarrollando.

No obstante, para aplicaciones futuras de la pagina sería interesante añadir más funcionalidades a esta.

6.2.1 Libererias

Se ha utilizado la libreríia *flask* ya que, esta desarrollada en Python y por lo tanto nos permite una programación más sencilla, rápida y el uso de otras librerías igual de útiles. Además, esta librería está en un constante desarrollo y mantenimiento, por lo que es sencillo encontrar ejemplos y resolver dudas buscando información reciente sobre la misma.

Otro aliciente para usar esta librería es que nos permite utilizar un modo de *debug* el cual nos mostrará en el navegador los errores de código que se hayan podido cometer.

Otra funcionalidad que nos permite esta librería es añadir de forma sencilla el *HTML STATUS*. En este caso mayoritariamente utilizaremos el 200 OK para confirmar que se ha enviado el JSON correctamente, pero en caso de que se quiera ampliar y/o añadir nuevas funcionalidades será conveniente tener la posibilidad de añadir este campo.

Por último, como vemos en la Ilustración 8, podemos especificar en una línea el host, el modo *debug* y el puerto de la aplicación en una misma llamada a una función.

```
if __name__ == '__main__':
    app_api.run(host="127.0.0.1",debug=True, port=80)
```

Ilustración 8. Inicialización de la api

6.1.2 Pruebas

Debido a que la aplicación web con todos los datos de la partida está hospedada en la *RaspBerry* de uno de los integrantes del grupo se ha optado por "hardcodear" los valores devueltos por la API-JSON de forma que podamos comprobar su funcionamiento.

Como se puede comprobar en el Anexo A2, la api cumple con las condiciones de REST ya que nos devuelve un JSON con varios campos, además del *HTML STATUS* definido como vemos en el Anexo A3.

7 Resultados

Como se ha explicado y demostrado a lo largo de este paper, el sistema funciona correctamente dentro de las restricciones iniciales.

El sistema es capaz de detectar un dron, identificarlo y comunicárselo a la base.

Los tiempos de latencia de la parte *soft-time* son inferiores a 30ms, a pesar de que en comparación al tiempo de conquista de una base sean despreciables.

La base cuenta con dos sistemas individuales a modo de doble *check* para poder verificar de forma más segura la identificación del dron.

Se ha implementado una API-JSON, la cual funciona correctamente según el modelo REST, pero solamente se le ha implementado el método GET ya que hasta el momento no se requieren de más.

8 Conclusiones

El desarrollo del IrDA en la placa STM32 ha sido especialmente difícil y tedioso ya que esta tecnología no es muy popular hoy en día alargando esta parte mucho más de lo previsto, además, se han probaron de realizar otras placas para ver si se podía solucionar por otras vías, estas fueron la placa Freescae KE02Z64M20SF0RMF, y el transceptor MCP215. A pesar de ello, y gracias que esta parte se ha hecho en conjunto con la parte de *disparo interdrónico* y, por lo tanto, se ha implementado de forma conjunta ha supuesto una gran ayuda.

Además, se ha hecho notable la diferencia de programar con según que placa, en este caso, la programación de la STM32 es ha sido la más compleja de todas, seguido de la *Freescale* y finalmente *Arduino*, la cual goza de un amplio repertorio de ejemplos y librerías actualizadas distribuidos por Internet, perfectos a la hora de implementar un prototipo rápido.

La unión de los cuatro TFGs de *Game of Drones* ha cumplido con el cometido de implementar el juego a modo de prototipo, a pesar de ello, para que esto pudiera ser un producto comercial todavía necesíta pulir muchos detalles.

como serían en el caso de este proyecto, probar la vídeo-recepción o implementar la base con un sistema RFID más potente o incluso implementar el proyecto en un caso real para poder ver cuanto de efectivas son estas soluciones cuando se encuentran en un entorno menós controlado y a merced de más. A pesar de ello podemos estar satisfechos con el resultado obtenido ya que ha cumplido nuestras expectativas.

9 Futuros pasos

A pesar de que se puede dar por concluido este TFG, el proyecto se podría seguir desarrollando para conseguir un mejor producto. Esto pasaría por:

- Implementar el transceptor IrDA en un único circuito.
- Probar la video-recepción y comparar su precisióny coste con los ya desarrollados.
- Implementar todos los sensores en una única placa, en vez de en dos separadas.

Agradecimientos

Quisiera agradecer todo el apoyo y la ayuda recibida por los integrantes y, a la vez, amigos, del proyecto *Game of Drones*, ya que sin su ayuda no hubiese sido posible la realización del mismo. También a nuestro tutor Màrius Montón quien no se pensó 2 veces afrontar el reto de llevar 4 TFGs de doble titulacióncuando se lo propusimos. Por último, a David Matanzanzas, CEO de AIRK Drones, quien nos proporcinó la idea original.

Bibliografia

[1] CamdenBoss, "Ultrasonic Transducer"
[2] Vishay Semiconductors, "IRDC, Part 1: Physical Layer", Doc no.82513 – Set. 2006
[3] María Dolores Sancho Martín, "Evaluación de técnicas de localización para un sistema RFID", Universidad de Sevilla - 2016
[4] E-Globage Corporation, "What is infrared communication?"
[5] Vishay Semiconductors, "VSMF2890RGX01, VSMF2890GX01 Data sheet"-Ene. 2013
[6] Altera Corporation, "DE2 Development and Education Board"- 2005
[7] Altera Corporation, "Altera DE2 Board Pin Table" - 2005
[8] Vishay Semiconductors, "Infrared Transceiver Module (SIR, 115.2 kbit/s) for IrDA® TFDU4101 Datasheet -Feb 2017
[9] Panasonic, "CND0313A"- May. 2011
[10] St Microelectronics, "RM0091 Reference manual"- Ene 2017
[11] St Microelectronics, "STM32F051x4 STM32F051x6 STM32F051x8"- Ene 2017
[12] Alejandra María Reyes Villamil, Carolina Peña Morales, Leonardo Muñoz Muñoz, "Sistemas Embebidos 2011-I", Universidad Nacional de Colombia – Jun. 2011
[13] NXP Semiconductors, "MFRC522 Data sheet"- Abr.2016
[14] R. Fielding, J. Reschke, RFC7231, "Hypertext Transfer Protocol (HTTP/1.1): Semantics and Content", ISSN:2070-1721 – Jun. 2014
[15] BBVA, "API REST: qué es y cuáles son sus ventajas en el desarrollo de proyectos", mar. 2016
[16] Freescale Semiconductor, Inc., "KE02 Sub-Family Reference Manual"- Jul 2013
[17] MikroEleKtronika, "IrDA PROTO"
[18] Microchip Technology, "MCP2155 Data sheet" – Nov. 2012
[19] Microchip Technology, "MCP2120 Data sheet" – Oct. 2006

ANEXO

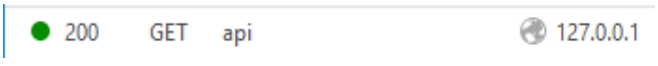
A1. FUNCIÓN GETAPI()

```python
class getApi(Resource):
    def get(self):
        # Default to 200 OK
        return [
            {
                "resource": "Drones",
                "link":{
                    "href":"/api/drones",
                    "rel":"self"
                },
            },
            {
                "resource": "bases",
                "link":{
                    "href":"/api/bases",
                    "rel":"self"
                },
            },
            {
                "resource": "Game",
                "link":{
                    "href":"/api/winner",
                    "rel":"self"
                },
            },
        ]
```

A2. JSON DEVUELTO POR LA APLICACIÓN


```
▼ 0:
  ▼ link:
      href:     "/api/drones"
      rel:      "self"
    resource:   "Drones"
▼ 1:
  ▼ link:
      href:     "/api/bases"
      rel:      "self"
    resource:   "bases"
▼ 2:
  ▼ link:
      href:     "/api/winner"
      rel:      "self"
    resource:   "Game"
```


A3. HTML STATUS



A4. TRANSCEPTOR IRDA SOLDADO A PLACA DE TOPOS