Universidad de Alcalá
Escuela Politécnica Superior

GRADO EN INGENIERÍA EN TECNOLOGÍAS DE TELECOMUNICACIONES

Trabajo Fin de Grado

Gestión telemática, mediante canal inalámbrico, de drivers de lámparas inteligentes para emisión/recepción de códigos ópticos.

Autor: Luis Javier Gutiérrez Gómez

Tutor/es: José Luis Lázaro Galilea

Álvaro de la Llana Calvo

2021

UNIVERSIDAD DE ALCALÁ

Escuela Politécnica Superior

GRADO EN INGENIERÍA EN TECNOLOGÍAS DE TELECOMUNICACIONES

Trabajo Fin de Grado

Gestión telemática, mediante canal inalámbrico, de drivers de lámparas inteligentes para emisión/recepción de códigos ópticos.

Autor: Luis Javier Gutiérrez Gómez

Tutor/es: José Luis Lázaro Galilea

Álvaro de la Llana Calvo

TRIBUNAL:

Presidente: Daniel Pizarro Pérez

Vocal 1º: Mª del Rosario Fernández Ruiz

Vocal 2º: José Luis Lázaro Galilea

FECHA: 27/07/2021

A mis tutores, José Luis y Álvaro, que han estado ayudándome en todo lo que he necesitado durante la realización de este proyecto, ya que sin ellos no habría sido posible sacarlo adelante.

A mis padres Javier y Mª Carmen, a mi yayo Luis, a mi hermana Yolanda y a Gabriel, porque este trabajo es el final de un largo recorrido en el que ellos me han acompañado en mis mejores momentos, pero también en los más difíciles.

A mi tío Joaquín, y a los que ya no están, por haberme dado el primer empujón que necesitaba en el Grado, y porque estoy seguro de que estarían tremendamente orgullosos de que haya alcanzado este objetivo.

A mis amigos, por haber pasado los mejores años de mi vida de estudiante junto a ellos, y porque, de una manera u otra, lo hemos logrado juntos.

Al Programa Erasmus, porque me ha cambiado la perspectiva de mi futuro y he aprendido que la vida hay que vivirla, hoy y ahora.

RESUMEN

El presente Trabajo Fin de Grado tiene por objetivo la realización de una herramienta de gestión telemática capaz de desarrollar un algoritmo para el emplazamiento óptimo de bombillas o focos y de los códigos que estos tienen que emitir, para desarrollar un sistema de posicionamiento local en interiores (IPS) mediante la iluminación artificial del edificio. Este trabajo es parte de un proyecto que pretende desarrollar una aplicación de posicionamiento en interiores (a modo de GPS en interiores) y que para ello utilice como balizas las propias bombillas de iluminación del edificio, que serán moduladas convenientemente. Como ejemplo de edificio para el desarrollo del prototipo se usará el Museo Provincial de Guadalajara.

En primer lugar, realizaremos un estudio sobre la detección de las frecuencias a utilizar (códigos a emitir desde las bombillas) en la cámara de dispositivos móviles. Utilizaremos MATLAB para estudiar la información captada y llegar a una conclusión sobre qué frecuencias y combinación de estas debemos emitir.

En segundo lugar, nos dedicaremos al desarrollo de un algoritmo de ubicación eficiente y fiable, con ayuda de MATLAB, que nos permita distribuir los diferentes focos emisores de códigos a lo largo y ancho de las plantas del museo. Realizaremos una base de datos que sea accesible para todos los dispositivos de los visitantes al museo, ya que será necesario que acceda a ella la aplicación de posicionamiento. Dicha base de datos contendrá la asignación de códigos a las bombillas de cada posición del entorno.

Una vez asignados los códigos, se programará cada bombilla (cuyo controlador driver ha sido diseñado en el proyecto y cuenta con comunicación wifi) mediante la red wifi del edificio.

Por último, realizaremos una pequeña interfaz gráfica que sea capaz de gestionar y editar la base de datos para el envío de códigos hacia las lámparas inteligentes.

PALABRAS CLAVE

- Comunicación por Luz Visible.
- Algoritmo de ubicación
- Lámparas inteligentes
- Interfaz gráfica

SUMMARY

The objective of this Final Degree Project is to create a telematics management tool capable of developing an algorithm for the optimal placement of bulbs or spotlights and the codes that they have to emit, to develop a local indoors positioning system (IPS) by artificial lighting of the building. This work is part of a project that aims to develop an indoor positioning application (like indoor GPS) and to use the building's own lighting bulbs as beacons, which will be suitably modulated. As an example of a building for the development of the prototype, the Provincial Museum of Guadalajara will be used.

First, we will carry out a study on the detection of the frequencies to be used (codes to be emitted from the bulbs), in the camera of mobile devices. We will use MATLAB to study the information captured and reach a conclusion about which frequencies and combination of these we should emit.

Secondly, we will dedicate ourselves to the development of an efficient and reliable location algorithm, with the help of MATLAB, that allows us to distribute the different code-emitting sources throughout the museum's floors. We will create a database that is accessible to all museum visitors' devices, since it will be necessary for the positioning application to access it. This database will contain the assignment of codes to the bulbs of each position in the environment.

Once the codes have been assigned, each bulb (whose driver has been designed in the project and has Wi-Fi communication) will be programmed through the building's Wi-Fi network.

Finally, we will create a small graphical interface that can manage and edit the database for sending codes to smart lamps.

KEYWORDS

- Visible Light Communication (VLC).
- Location algorithm
- Smart lamps
- Graphic interface

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

ÍNDICE

1. INTRODUCCIÓN

Las nuevas tecnologías han cambiado nuestra perspectiva y nuestra manera de manejar distintos ámbitos de la vida cotidiana. Hoy en día es muy difícil encontrar a alguien que no disponga de un dispositivo smartphone, que es la herramienta más versátil de la que dispone el ser humano, con su infinidad de usos y aplicaciones. Por otro lado, es cada vez más usual ver a la gente utilizarlo para posicionamiento en exteriores mediante GPS. Pero en ocasiones GPS no es del todo eficaz, como por ejemplo dentro de un edificio en el que las señales de los satélites no entran directamente o tienen diferentes plantas, salas y pasillos.

En cuanto al posicionamiento en un entorno determinado, hemos pasado muchísimos años peleándonos con mapas en papel de edificios grandes con muchas salas y pasillos, como museos. Ha llegado la hora de revolucionar este concepto, y realizar alguna aportación para lograrlo es el objetivo. Y queremos realizarlo sin incorporar una gran carga de sistemas, de una manera simple, sencilla de manejar y a un coste bajo.

En esta línea, y con el afán de poder facilitar la visita de las personas que acuden a un museo, por ejemplo, hemos decidido crear un sistema mediante el que un dispositivo móvil pueda, a través de un sensor de señales ópticas que han sido emitidas por la iluminación en cualquiera de las salas del museo, determinar la posición. En concreto, se va a trabajar en el prototipo en el Museo Provincial de Guadalajara y, posteriormente, podremos reproducirlo en otros ambientes.

El desarrollo del driver controlador para las lámparas inteligentes, su sistema de comunicación y control y la modulación de códigos forman parte de otro proyecto fin de grado. En el TFG que nos ocupa se abordará el control de la iluminación y emisión de códigos desde cada una de las lámparas que intervienen el sistema, de manera inalámbrica. Además, se desarrollará una aplicación que distribuya los diferentes códigos disponibles de la manera más eficiente posible.

En este caso, inicialmente, el trabajo se centrará en la gestión telemática de los códigos a emitir por los diferentes elementos de la iluminación del edificio, que incluirá:

- la comunicación inalámbrica desde un nodo central (ordenador)con cada lámpara inteligente;
- la gestión de los códigos o secuencias a emitir cada lámpara y la creación de una base datos con la información de diferentes parámetros para las lámparas y su código correspondiente;
- la algoritmia para determinar la mejor opción de códigos en lámparas vecinas, cuando múltiples lámparas estén en zonas cercanas y cuando se deban reutilizar códigos en un mismo edificio;
- y una interfaz gráfica de usuario para gestionar todo lo anterior.

El punto de mayor interés de este trabajo es el algoritmo de determinación de posición de bombillas emisoras de códigos en las diferentes partes del edificio, a partir de disponer de un mapa de este, y la determinación de las señales (códigos) más adecuados a emitir por cada una de ellas y sus vecinas, para facilitar la determinación de la posición del receptor.

A continuación, se pretende hacer un análisis de los principales conceptos que se van a tratar en este Proyecto. Se empezará hablando de las tecnologías conocidas como Comunicación por Luz Visible (VLC) y Posicionamiento por Luz Visible (VLP). Además, se hablará de forma más pormenorizada de los protocolos Wi-Fi, IP y TCP, y sobre bases de datos, que son partes muy importantes en este proyecto.

1.1 COMUNICACIÓN Y POSICIONAMIENTO POR LUZ VISIBLE

La Comunicación por luz visible, más conocida como VLC (*Visible Light Communication*), es un medio transmisor de datos que utiliza la luz como señal portadora de la información. VLC es un subconjunto de tecnologías de comunicaciones ópticas inalámbricas. Existen dispositivos diseñados para recibir señales de fuentes de luz (televisores, carteles luminosos, señales de tráfico, etc.) mediante un sensor óptico que llevan incorporado, como por ejemplo la cámara de fotos de un smartphone.

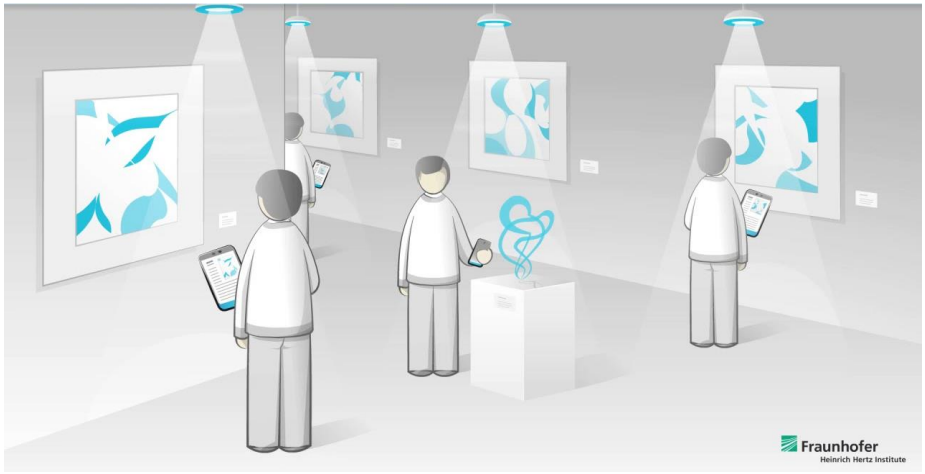

Figura 1. *Ilustración sobre VLC (Visible Light Communication). Group Franhofer.*

Estos sistemas que son bastante sencillos, a priori, pero conllevan mucho trabajo y experimentación detrás, ya que influyen muchos factores tales como la potencia, la intensidad, la dirección o la frecuencia de envío de la información a través de la luz. Cuando la información emitida mediante luz es utilizada en el receptor para determinar la posición en lugar de, o además de, para obtener información se habla de tecnología y sistemas de VLP.

Todavía se están diseñando y perfeccionando, pero desde luego es una apuesta muy interesante dado que se tratará de sistemas muy económicos, eficientes y de provecho ampliando otras aplicaciones ya existentes, debido a que en los edificios ya se dispone de la infraestructura necesaria (iluminación) que solo debe ser modificada cambiando

los elementos de iluminación por otros que puedan emitir códigos. Cabe esperar que los sistemas de comunicación y posicionamiento por luz visible sean los sistemas por excelencia en la próxima generación.

El proyecto utiliza la luz como emisor de información a través de diferentes frecuencias. Dependiendo de las frecuencias utilizadas en los focos emisores, se transmite una información u otra. No es información directa, pues posteriormente necesita una comparación contra la base de datos anteriormente creada.

1.2 PROTOCOLO DE INTERNET Y PROTOCOLO DE CONTROL DE TRANSMISIÓN (TCP/IP)

Una dirección IP es un número que identifica de manera lógica y jerárquica los dispositivos de una red o subred. Un usuario al conectarse a la red utiliza una dirección IP que puede cambiar de una sesión a otra. Por ejemplo, podemos tener una pequeña red doméstica a la que se quieran conectar diferentes dispositivos. Cada uno de ellos tendrá una dirección IP privada que los identifique en el momento de estar conectados.

- **Router:** 192.168.0.1
- **Mi móvil:** 192.168.0.11
- **Mi impresora:** 192.168.0.12
- **Mi ordenador:** 192.168.0.98

Tendremos una primera parte que identifica a la red (192.168.), y una segunda parte que identifica a los dispositivos.

En cuanto a la cantidad de dispositivos, esta se dictamina con la máscara de subred. Dicha máscara puede ser del tipo 255.255.255.0, por ejemplo, y significa que podremos tener entre 0 y 254 dispositivos conectados a nuestra red doméstica. Podremos aumentar la cantidad de dispositivos o disminuirla, simplemente cambiando la máscara de subred. Si queremos ampliarla al doble, por ejemplo, debemos poner 255.255.254.0, o bien si queremos reducirla a la mitad, 255.255.255.128.

El protocolo de Internet IP será de gran ayuda en el proyecto, ya que nos permitirá identificar de forma segura con cada luminaria y tener interconectada y ordenada nuestra iluminación inteligente de algún modo. En nuestra aplicación, cada una de las lámparas tendrá un código IP fijo que estará relacionado con la posición que ocupa en el edificio y que será utilizado para comunicar con dicha lámpara, de forma univoca, y remitirle los códigos que tiene que emitir. Esto es importante ya que los códigos que emite cada lámpara y sus vecinas serán los que se utilicen para determinar la posición.

El protocolo TCP es un protocolo fundamental en las comunicaciones por Internet para enviar flujos de datos de manera fiable. Es decir, garantizando la entrega sin errores y en el mismo orden en el que se transmitieron. Este protocolo tiene una asociación directa con el protocolo que hemos explicado en anterior apartado, el protocolo IP.

En la pila de protocolos, TCP actúa en la capa de transporte, entre la capa de red y la capa de aplicación. Está orientado a conexión, pues tanto el cliente como el servidor tienen que aceptar dicha conexión antes de empezar a enviar y recibir los datos requeridos. El lado cliente es quien inicia la apertura a través de un puerto y posteriormente es el servidor el que acepta o no esa conexión. Una vez ambos se han sincronizado, la transferencia de datos es totalmente confiable y segura. La conexión se debe terminar en ambos lados, primero cerrando en el lado cliente o servidor y seguidamente desde el lado servidor o cliente.

El concepto de puerto permite identificar las aplicaciones emisoras y receptoras. Cada lado tendrá un número entre 0 y 65535 puertos posibles asignado por una de las dos aplicaciones.

1.3 OBJETIVOS Y CAMPO DE APLICACIÓN

El objetivo general de este proyecto es la creación de una herramienta que, junto a otros desarrollos, ayude a posicionar un dispositivo móvil dentro de un edificio a través de un sensor óptico de un Smartphone (cámara o sensor de iluminación, por ejemplo). Que contribuya a una mejora de la calidad y de la experiencia cultural de los asistentes a centros de ocio o comerciales, utilizándose como ejemplo para el prototipo el Museo Provincial de Guadalajara dentro del proyecto “GUIA”, financiado por la Junta de Comunidades de Castilla-La Mancha.

Trataremos de gestionar todos los códigos a emitir por la iluminación de un edificio, en concreto mediante bombillas con inteligencia y conectividad Wi-Fi, a través de una base de datos. La gestión implicará la asignación de frecuencias, agregar y quitar bombillas al sistema, la generación de unos códigos que serán los que se trasmitan por cada luminaria, y todo ello desde un PC conectado con la misma tecnología Wi-Fi. Hacer notar que el funcionamiento y reasignación debe ser en tiempo real y que cada vez que se sustituya, elimine o añada una lámpara el sistema ha de reconfigurarse en tiempo real.

Entre los objetivos específicos, que hemos ido depurando, adaptando e incrementando a lo largo del desarrollo del proyecto, están los siguientes:

- Establecer una comunicación inalámbrica unívoca con cada lámpara del entorno.
- Gestión de códigos o secuencias a emitir por cada una de ellas.
- Crear, controlar y manejar una base de datos con diferentes parámetros de cada lámpara, como su localización, y códigos a emitir en cada momento.
- Generar la algoritmia para determinar la mejor opción de códigos cuando múltiples lámparas estén en zonas cercanas y cuando se deban reutilizar códigos en un mismo edificio.
- Manejar la tecnología Wi-Fi, anteriormente mencionada, para la trasmisión de códigos.

• Desarrollar una interfaz para el manejo de la base de datos, las comunicaciones y el establecimiento de códigos de cada lámpara.

1.4 DESCRIPCIÓN DEL TRABAJO

Se propone el desarrollo de una herramienta que permita distribuir, de la manera más eficiente, un número de frecuencias determinadas disponibles y combinaciones de las mismas (a elección de los diseñadores), pudiendo repetir dichas frecuencias con una estrategia de carácter complejo que permita obtener la localización de la lámpara (aunque su código -combinación de frecuencias- sea reutilizado por otra), cuando se capta con el sensor del Smartphone un código, y posicionarse de forma local.

Con una base de datos, almacenada en un PC con toda la información de la iluminación del edificio (siendo posible la modificación de esta, como agregar/quitar bombillas o actualizar sus características), se podrán establecer diferentes estrategias de reparto de los diferentes códigos a emitir desde cada luminaria y desplegar un mapa del edificio, para que desde el dispositivo móvil receptor se pueda determinar su localización exacta. Nos basaremos en el esquema de la Figura 2.

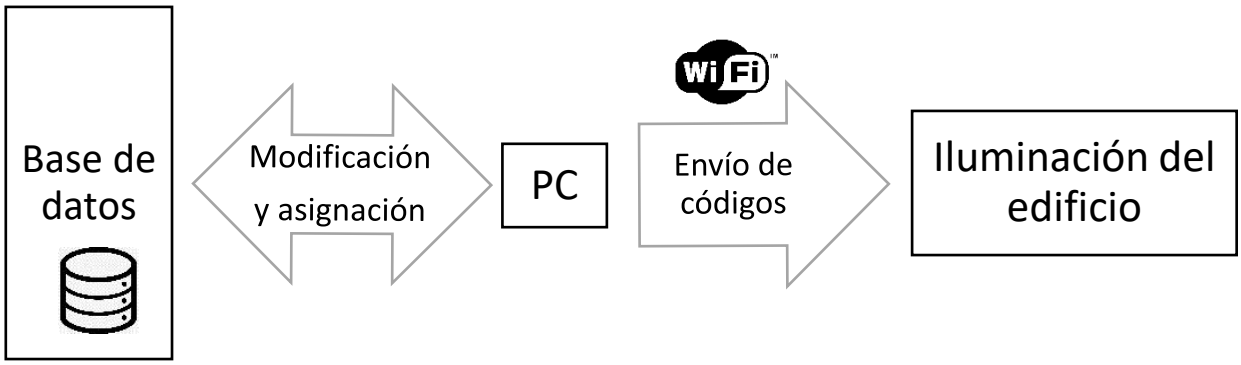

Figura 2. *Esquema objetivo del proyecto.*

Los focos, inicialmente, se prevé que puedan emitir secuencialmente 2 o 3 frecuencias cada uno, constituyendo esa secuencia su código (pudiendo ser cambiada en tiempo real). Las combinaciones de 2 o 3 frecuencias, en función del número de lámparas necesarias (que a su vez es función del tamaño del edificio a cubrir) es probable que deban ser reutilizadas en diferentes lámparas, pero la algoritmia y el sistema de asignación de códigos ha de ser capaz de asignarlas de manera que no puedan ser localizadas simultáneamente por un mismo dispositivo móvil. Dichas bombillas inteligentes, cuyo desarrollo es objeto del otro proyecto, dispondrán de un microcontrolador, además de conectividad Wi-Fi que servirá como medio inalámbrico para la comunicación con el PC mencionado anteriormente y para que cada bombilla emita los códigos asignados.

Para la gestión de la base de datos y la transferencia de los códigos generados se plantea utilizar como soporte la herramienta Microsoft Excel, combinada y manejada con ayuda de MATLAB.

En los siguientes apartados trataremos de explicar todo el proceso que hemos llevado a cabo para la realización del proyecto.

Comenzaremos por el estudio de las frecuencias a emitir desde las luminarias y que captaremos a través de un smartphone. Tras capturar las señales observaremos detenidamente si somos capaces de detectar dichas frecuencias transmitidas por los focos experimentales. Para ello vamos a utilizar las características del rolling shutter de las cámaras de los dispositivos móviles (sensor CMOS), donde el obturador capta la imagen a partir de un barrido. El objetivo de usar este método de captura de imágenes es extraer el par o la terna de frecuencias del foco lo más fácilmente posible.

Después, explicaremos el algoritmo que hemos desarrollado para la asignación de códigos y cómo lo vamos a utilizar a la hora de repartir las frecuencias en el plano. Una vez hayamos decidido el lugar y la posición de dichas lámparas inteligentes pasaremos a la evaluación de la configuración de cada lámpara. Para ello elaboraremos una base de datos con diferentes tablas con toda la información que necesitemos.

Tras la elaboración de la base de datos, discutiremos la forma de enviar los códigos que van a ser capaces de configurar todas las lámparas inteligentes del museo. Además, tendremos que decidir qué parámetros hay que enviar en dichos códigos y qué formato deben tener para que se pueda llevar a cabo la configuración.

Seguidamente realizaremos una interfaz gráfica para que cualquier usuario pueda ser capaz de elegir la plantilla del entorno, lanzar el algoritmo y distribuir las frecuencias por los diferentes focos inteligentes de su edificio. Buscaremos que sea sencillo, muy intuitivo y fácil de manejar.

Por último, redactaremos un capítulo con los resultados obtenidos, comentando tablas e imágenes sobre el Museo de Guadalajara, ya que hacer que funcione en este escenario es el objetivo principal.

2. ANÁLISIS DE SEÑALES A EMITIR

2.1 ELECCIÓN DE FRECUENCIAS

En un primer momento quisimos saber cómo se comportaría la cámara de los dispositivos móviles ante la iluminación de los focos prototipo. Como nuestra idea inicial fue enviar pares de frecuencias construimos un circuito muy sencillo capaz de alterar la iluminación del foco para que, con corriente alterna, lográramos tener un cierto tiempo una de las frecuencias y otro periodo de tiempo la otra frecuencia. La modulación utilizada fue modulación por anchura de impulsos (PWM). Decidimos elegir 10 frecuencias porque es un número no muy elevado que es capaz de darnos una cantidad considerable de emparejamientos posibles.

El siguiente paso sería probar, desde la misma distancia al foco, diferentes pares de frecuencias; en concreto las 45 posibilidades disponibles si utilizábamos frecuencias de 1kHz a 10kHz, siguiendo la fórmula estadística de las combinaciones sin repetición de m elementos tomados de n en n:

$$N_{combinaciones} = \frac{m_{frecuencias}!}{n_{posibles}! \cdot (m_{frecuencias} - n_{posibles})!}$$

$$N_{combinaciones} = \frac{10!}{2! \cdot (10-2)!} = 45$$

En el futuro se podrá analizar, por ejemplo, el caso en el que tengamos tres frecuencias en un mismo foco, que aumentaría nuestras posibilidades de una manera considerable y podríamos abarcar superficies más extensas. En este caso obtendríamos, siguiendo la fórmula anterior:

$$N_{combinaciones} = \frac{10!}{3! \cdot (10-3)!} = 120$$

En las fotografías y tablas que se presentan a continuación se hace un estudio exhaustivo de la viabilidad de todos los emparejamientos posibles entre 1kHz y 10 kHz. El motivo por el cual debemos hacer este estudio es que en primer lugar debemos saber si detectamos bien las frecuencias, y en segundo lugar saber la eficiencia de los diferentes emparejamientos, ya que no será lo mismo detectar 1 kHz junto con 3 kHz, o 9 kHz con 10 kHz. A partir de los archivos *.mat* generados con MATLAB de las imágenes que recogimos hemos podido extraer los siguientes resultados a través de un script. Los *.mat* son archivos que están en el formato contenedor de datos binarios, y pueden contener variables, matrices, cadenas de caracteres, etc.

En la fotografía de la Figura 3 vemos un ejemplo de lo que hemos experimentado en el laboratorio de la universidad.

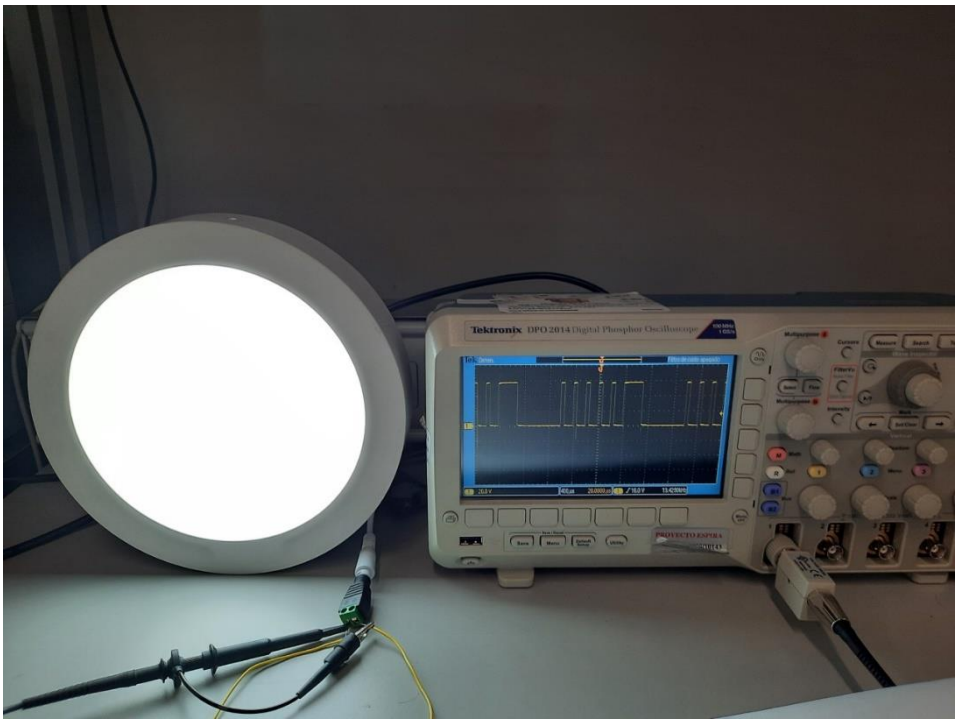

Figura 3. *Fotografía del foco utilizado y osciloscopio.*

Tenemos conectado el circuito a nuestro foco prototipo en el que se puede apreciar a su derecha, en el osciloscopio, y en la Figura 4, una señal PWM al 30% con 5 kHz y 1 kHz. Buscaremos explotar la señal PWM ya que, dependiendo de su porcentaje, podremos regular la intensidad de luz a placer.

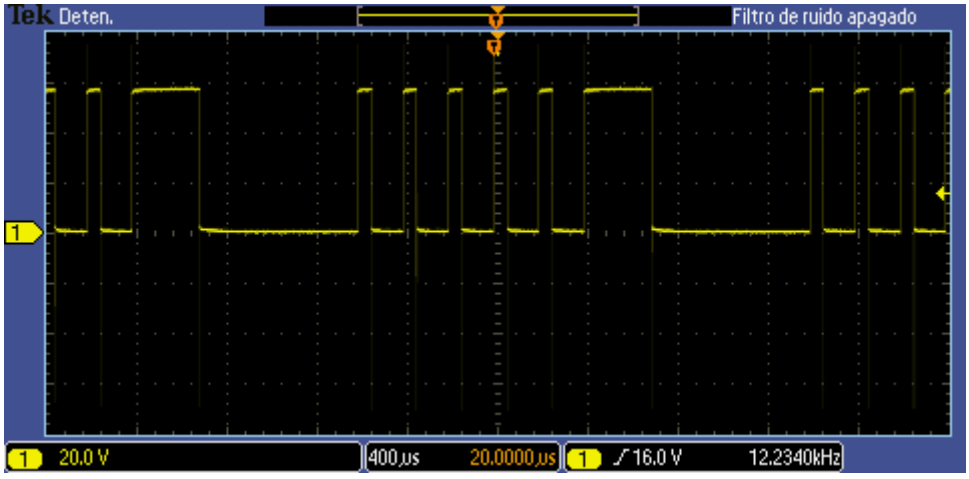

Figura 4. *Captura de osciloscopio. Señal PWM al 30% con 1 kHz y 5 kHz*

Estas imágenes y capturas fueron tomadas con ayuda del otro trabajo fin de grado que va en concordancia con nuestro proyecto: el diseño del driver controlador para manejar las modulaciones y los códigos a emitir.

En la Tabla 1 vemos ocho columnas. La primera columna es simplemente un identificador del archivo *.mat* que hemos utilizado para extraer los datos. Será la misma columna en todas las tablas posteriores. Las dos columnas siguientes muestran los pares de frecuencias generados en el foco en Hercios [Hz]. En cuanto a la cuarta y quinta columna son parámetros que tuvimos que introducir en el sistema para generar los pares. Es decir, el campo *F_central* refleja la frecuencia central entre ambas frecuencias y el campo *Deviation* refleja la variación positiva y negativa que necesitamos para obtener, a partir de la frecuencia central, las frecuencias requeridas en Hercios [Hz]. Por último, las tres columnas que finalizan esta tabla examinan las frecuencias que se reciben con potencias máximas de cada archivo *.mat*. Esto nos dará una idea de cuáles de todas las posibilidades serán las más eficientes, y cuáles son óptimas para utilizar.

En la Tabla 2 realizamos los siguientes cálculos previos a la decisión final. Inicialmente, examinamos el valor de los máximos de potencia lumínica de las frecuencias con valores máximos (con ellos se desea conocer si al comprobar las frecuencias presentes se pueden discriminar fácilmente las 2 que se deseaban emitir como código respecto del resto que se capten). Calculamos la diferencia entre primer y segundo máximo, y segundo y tercer máximo. En el primer caso interesa que sea lo más pequeño posible, debido a que serán los más sobresalientes respecto al resto, y estarán uno muy próximo al otro en lo que a potencia recibida se refiere. En cuanto al segundo caso, nos interesa que sea lo más grande posible, porque eso significará un margen grande entre el segundo y el tercer máximo, que corresponderá a una frecuencia que no nos interesa. Este apartado solo será reseñable si primer y segundo pico son los picos que buscamos en el archivo *.mat*.

Finalmente, en la Tabla 3 realizamos de nuevo unos cálculos sencillos de relación entre los dos primeros máximos y los dos segundos. Si la relación entre el segundo máximo y el primero es superior al 75%, damos por hecho que el emparejamiento es muy eficiente y los marcamos como "BIEN" subrayados en verde. En cambio, si la relación entre el tercer y segundo máximo es superior al 75%, significa que el emparejamiento podría darnos problemas con el tercer pico que se detecta. Marcamos en rojo y escribimos "Menos Preferente" los que vamos a dejar como combinaciones no prioritarias, y subrayamos en amarillo escribiendo "3ER_PICO" en los que el segundo valor de frecuencias que buscamos aparece en el tercer máximo del archivo *.mat*. Estos últimos son casos especiales que deberemos estudiar en el futuro, ya que existe al menos una frecuencia no deseada que sobresale por encima de la una de las que necesitamos identificar. El resto, tras haber estudiado las imágenes y las gráficas aportadas, los daremos por buenos ("BIEN").

Cabe destacar que la mayoría de los que marcamos como "Menos Preferente" tienen 7kHz como una de las frecuencias que buscamos. Esto puede ser debido a múltiples motivos como, por ejemplo, la frecuencia que utilizamos para el método rolling shutter, la propia cámara de la Tablet utilizada, el mismo foco, etc. Al comparar estos casos con el resto, y no obtener los picos de frecuencia que buscamos a la altura que deseamos, hemos decidido que se deben estudiar en profundidad en el futuro.

Tabla 1. *Emparejamientos de frecuencias. Picos destacados.*

.mat	f1 [Hz]	f2 [Hz]	F_central [Hz]	Deviation [Hz]	Fr1 [Hz]	Fr2 [Hz]	Fr3 [Hz]
b101	1	2	1,5	0,5	2	1	6
b102	1	3	2	1	1	3	9
b103	1	4	2,5	1,5	4	1	3
b104	1	5	3	2	5	1	3
b105	1	6	3,5	2,5	1	6	3
b106	1	7	4	3	3	5	1
b107	1	8	4,5	3,5	1	8	7
b108	1	9	5	4	1	9	3
b109	1	10	5,5	4,5	1	10	20
b110	2	3	2,5	0,5	3	2	9
b111	2	4	3	1	4	2	15
b112	2	5	3,5	1,5	5	2	6
b113	2	6	4	2	6	2	1
b114	2	7	4,5	2,5	2	1	6
b115	2	8	5	3	2	8	7
b116	2	9	5,5	3,5	2	9	6
b117	2	10	6	4	2	10	8
b118	3	4	3,5	0,5	4	3	9
b119	3	5	4	1	3	5	19
b120	3	6	4,5	1,5	3	1	12
b121	3	7	5	2	3	1	6
b122	3	8	5,5	2,5	3	8	15
b123	3	9	6	3	3	9	18
b124	3	10	6,5	3,5	3	10	19
b125	4	5	4,5	0,5	5	4	15
b126	4	6	5	1	4	6	1
b127	4	7	5,5	1,5	4	1	3
b128	4	8	6	2	4	8	6
b129	4	9	6,5	2,5	4	9	17
b130	4	10	7	3	10	4	8
b131	5	6	5,5	0,5	5	1	6
b132	5	7	6	1	5	1	10
b133	5	8	6,5	1,5	5	8	9
b134	5	9	7	2	5	9	15
b135	5	10	7,5	2,5	5	10	21
b136	6	7	6,5	0,5	20	4	18
b137	6	8	7	1	8	7	6
b138	6	9	7,5	1,5	9	1	6
b139	6	10	8	2	10	1	6
b140	7	8	7,5	0,5	8	9	17
b141	7	9	8	1	9	1	16
b142	7	10	8,5	1,5	10	1	19
b143	8	9	8,5	0,5	9	8	17
b144	8	10	9	1	10	8	5
b145	9	10	9,5	0,5	9	10	19

Tabla 2. *Emparejamientos de frecuencias. Picos máximos de potencia y diferencias.*

.mat	Maximo1 [ad]	Maximo2 [ad]	Maximo3 [ad]	Diferencia1-2 [ad]	Diferencia2-3 [ad]
b101	0,63071094	0,629283489	0,194882715	0,001427451	0,434400774
b102	0,623493663	0,576807714	0,242226142	0,046685949	0,334581572
b103	0,654258901	0,572971837	0,205522649	0,081287064	0,367449189
b104	0,580542685	0,561784571	0,169181877	0,018758113	0,392602694
b105	0,895119923	0,394837555	0,278998051	0,500282368	0,115839504
b106	0,381348413	0,349092952	0,331578579	0,032255461	0,017514373
b107	0,762952416	0,417203328	0,263716586	0,345749088	0,153486742
b108	0,639534548	0,556116195	0,346408714	0,083418353	0,209707481
b109	0,761987421	0,488612753	0,143499684	0,273374668	0,345113069
b110	0,620058681	0,615041945	0,198769363	0,005016736	0,416272582
b111	0,606454539	0,59650264	0,187104011	0,009951899	0,409398629
b112	0,667350695	0,637118148	0,229133489	0,030232548	0,407984659
b113	0,682228453	0,494964424	0,140602645	0,187264029	0,35436178
b114	0,514717019	0,453891311	0,317778955	0,060825708	0,136112355
b115	0,693093674	0,535350893	0,17674862	0,157742781	0,358602273
b116	0,69786595	0,435558596	0,278757652	0,262307354	0,156800944
b117	0,737081101	0,46839911	0,177007581	0,268681991	0,291391529
b118	0,600097692	0,572655111	0,227018978	0,027442581	0,345636133
b119	0,673339132	0,530125867	0,176891989	0,143213264	0,353233879
b120	0,632952204	0,266077763	0,250979112	0,366874441	0,01509865
b121	0,679303484	0,376138138	0,233557323	0,303165347	0,142580815
b122	0,57155516	0,516251019	0,199365632	0,055304141	0,316885387
b123	0,69077879	0,509304704	0,263086942	0,181474086	0,246217762
b124	0,654335761	0,445234302	0,194538197	0,209101459	0,250696105
b125	0,699514856	0,547737706	0,204605888	0,151777151	0,343131817
b126	0,718658217	0,359864922	0,302319909	0,358793295	0,057545013
b127	0,603403126	0,556970723	0,181086829	0,046432403	0,375883894
b128	0,656569354	0,359068454	0,20490102	0,2975009	0,154167434
b129	0,665091969	0,411200946	0,183929327	0,253891023	0,227271619
b130	0,594288568	0,587774513	0,176995746	0,006514055	0,410778767
b131	0,531496735	0,387071711	0,375084742	0,144425024	0,011986969
b132	0,626822228	0,469909784	0,248077649	0,156912444	0,221832135
b133	0,620467233	0,340618193	0,219184764	0,27984904	0,121433428
b134	0,656859194	0,541207432	0,204836859	0,115651762	0,336370573
b135	0,555781821	0,526124082	0,178581227	0,029657739	0,347542855
b136	0,118665836	0,112864402	0,108151832	0,005801434	0,00471257
b137	0,439870706	0,39429705	0,270983555	0,045573656	0,123313495
b138	0,594685583	0,296612335	0,228216484	0,298073248	0,068395851
b139	0,54962987	0,336132566	0,298396034	0,213497303	0,037736533
b140	0,307865204	0,217514108	0,202111431	0,090351096	0,015402677
b141	0,538900814	0,402441214	0,180482791	0,1364596	0,221958423
b142	0,558514576	0,430605511	0,212968756	0,127909064	0,217636755
b143	0,590091288	0,494102946	0,182898597	0,095988342	0,311204348
b144	0,646070964	0,361751045	0,19911442	0,284319919	0,162636625

Tabla 3. *Emparejamientos de frecuencias. Relaciones y decisiones.*

.mat	% Relación 2/1 [ad]	% Relación 3/2 [ad]	Validez	Validez_Definitiva
b101	99,7736759	30,96898589	BIEN	BIEN
b102	92,51220147	41,99426183	BIEN	BIEN
b103	87,57570382	35,86958996	BIEN	BIEN
b104	96,76886577	30,11508071	BIEN	BIEN
b105	44,11001752	70,66147769	BIEN	BIEN
b106	91,54173457	94,98289132	Menos Preferente	Menos Preferente
b107	54,68274549	63,21056616	BIEN	BIEN
b108	86,95639615	62,29070778	BIEN	BIEN
b109	64,12346711	29,36879619	BIEN	BIEN
b110	99,19092563	32,31801745	BIEN	BIEN
b111	98,35900331	31,36683705	BIEN	BIEN
b112	95,46976609	35,96405	BIEN	BIEN
b113	72,55112597	28,40661624	BIEN	BIEN
b114	88,18268953	70,01212576	PROBLEMAS	Menos Preferente
b115	77,24077033	33,01547122	BIEN	BIEN
b116	62,41293125	64,00003449	BIEN	BIEN
b117	63,54783885	37,78990554	BIEN	BIEN
b118	95,42698109	39,64322913	BIEN	BIEN
b119	78,73088647	33,36792255	BIEN	BIEN
b120	42,03757583	94,32547454	PROBLEMAS	Menos Preferente
b121	55,37114799	62,09349685	PROBLEMAS	Menos Preferente
b122	90,32391887	38,61796391	BIEN	BIEN
b123	73,72905932	51,656099	BIEN	BIEN
b124	68,04370602	43,69344327	BIEN	BIEN
b125	78,30251219	37,35472037	BIEN	BIEN
b126	50,07455746	84,00927414	BIEN	BIEN
b127	92,30491175	32,51280927	PROBLEMAS	Menos Preferente
b128	54,68857968	57,06461195	BIEN	BIEN
b129	61,82617825	44,72979174	BIEN	BIEN
b130	98,90389021	30,1128651	BIEN	BIEN
b131	72,82673356	96,90316576	PROBLEMAS	3ER_PICO
b132	74,96699425	52,79261197	PROBLEMAS	Menos Preferente
b133	54,89704768	64,34910668	BIEN	BIEN
b134	82,39321873	37,84812385	BIEN	BIEN
b135	94,66378026	33,94279667	BIEN	BIEN
b136	95,11111687	95,82457375	Menos Preferente	Menos Preferente
b137	89,63930638	68,72573727	PROBLEMAS	3ER_PICO
b138	49,87716928	76,94099581	PROBLEMAS	3ER_PICO
b139	61,15616801	88,77331842	PROBLEMAS	3ER_PICO
b140	70,65238476	92,91876871	PROBLEMAS	Menos Preferente
b141	74,67816033	44,84699502	PROBLEMAS	Menos Preferente
b142	77,09834801	49,45797265	PROBLEMAS	Menos Preferente
b143	83,73330642	37,01629366	BIEN	BIEN
b144	55,99246291	55,04183684	BIEN	BIEN
b145	90,37131605	30,36336957	BIEN	BIEN

2.2 PROCESAMIENTO

En este apartado vamos a comentar, de manera detallada, todo el proceso de toma de imágenes de la señal procedente de los focos prototipo y el estudio de dichas imágenes con ayuda de MATLAB.

Basándonos en el experimento que se realizó en el artículo [22] *Measuring rolling shutter with a strobing LED*, realizamos los cálculos y experimentos necesarios para obtener tanto el tiempo de exposición como la frecuencia de muestreo y así obtener estas imágenes que, posteriormente, vamos a tratar con MATLAB para su estudio.

$$t_{exposición} = \frac{1}{6400}[s];\ f_{muestreo} = 94476\ [Hz]$$

Como ya hemos explicado en el capítulo anterior, no es una toma de fotografía convencional. Al estar utilizando el método rolling shutter, la fotografía se ha tomado en diferentes momentos temporales.

Para que la cámara captase solamente la circunferencia del foco bajamos la iluminación de la sala para tener mejor experiencia, encontrándonos a dos metros de distancia, aproximadamente, desde la toma de la fotografía hasta el foco. Utilizamos una tableta Sansung Galaxy Tab S3, modelo SM-T820. Estos son algunos ejemplos captados en el laboratorio del departamento de Electrónica, en la Universidad de Alcalá:

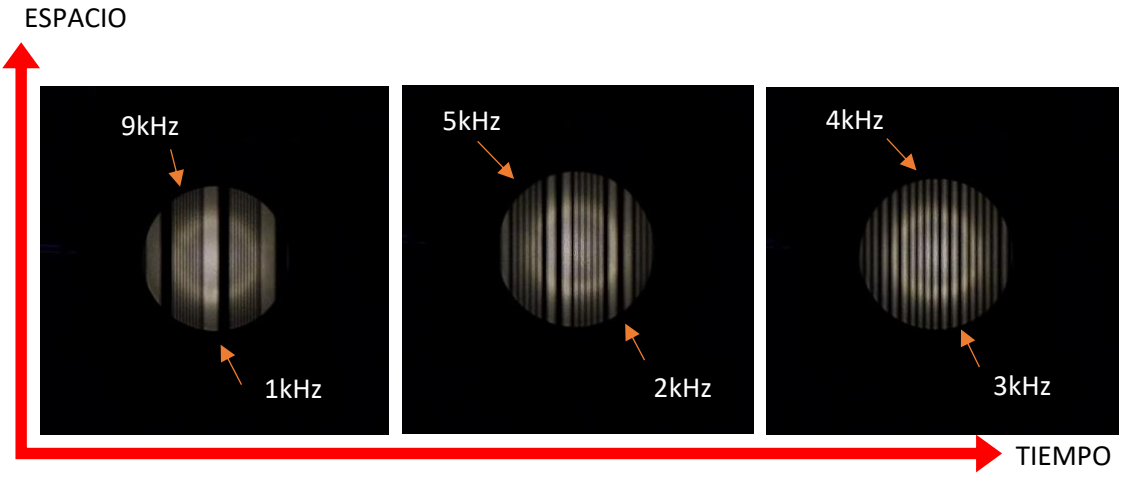

Figura 5. *Foco prototipo con diferentes pares de frecuencias. 1-9 kHz 2-5 kHz y 3-4 kHz*

En el primer caso de la Figura 5, las rayas más gruesas corresponden a la frecuencia más baja, 1 kHz, y las más finas corresponden a la frecuencia más alta, en este caso 9 kHz. Repetimos el proceso con el resto y el resultado es de carácter similar en todos los casos.

A continuación, el objetivo sería captar las frecuencias a partir de estas imágenes realizando un script en MATLAB que nos ayudase a detectarlas. Para ello nos basamos en la teoría básica de la FFT. Convertimos la imagen en una matriz de pixeles en escala

de grises con valores entre 0 y 255 y, de una manera aproximada, estraemos el diámetro de la circunferencia. Con ello logramos tener una fila de bits con la informacíon del color que a su vez nos da la información para sacar las frecuencias. Aplicamos la FFT a dicho vector junto con otros tratamientos de la señal para conseguir los siguientes resultados vistos en distintas gráficas que nos muetran la potencia en función de la frecuencia. Estas gráficas van en corcondancia con los resultados mostrados en el apartado que hemos estudiado anteriormente. Como se pude observar, para la frecuencia de rolling shuter de estos dispositvos, las frecuencias inferiores a 1 kHz ocuparían una gran parte del tiempo de captura no dejando margen para otras frecuencias, y las superiores a 10 kHz serían muy difíciles de detectar en imágenes pequeñas de los focos.

Estamos midiendo el nivel de luz a partir a la información de los valores de los pixeles que van de 0 a 255. Este valor depende del tiempo de exposición, del parámetro de sensibilidad lumínica ISO de la cámara, etc. Por lo tanto, normalizando los valores obtenidos de la potencia en función de la frecuencia obtenemos figuras como las mostradas desde Figura 5 a Figura 7.

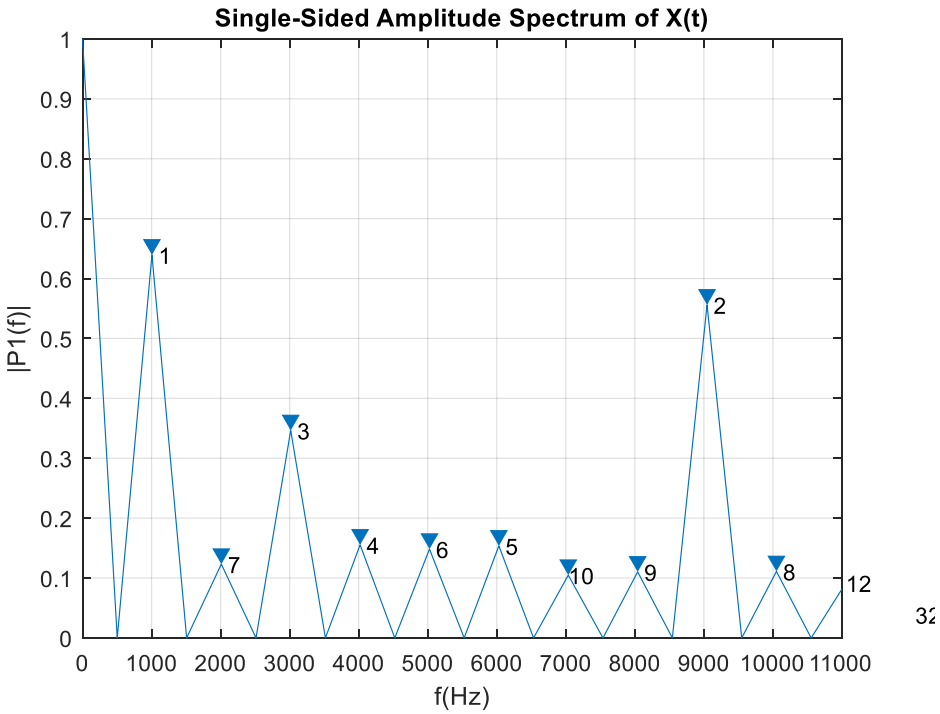

Figura 6. *Potencia en función de la frecuencia. Caso 1-9 kHz.*

En la Figura 6 se aprecia perfectamente que la potencia transmitida sobresale en las frecuencias de 1 kHz y 9 kHz por encima del resto. Es un caso en el que la relación entre ambas frecuencias máximas es aceptable y el tercer pico está alejado de las frecuencias referencia que buscamos obtener.

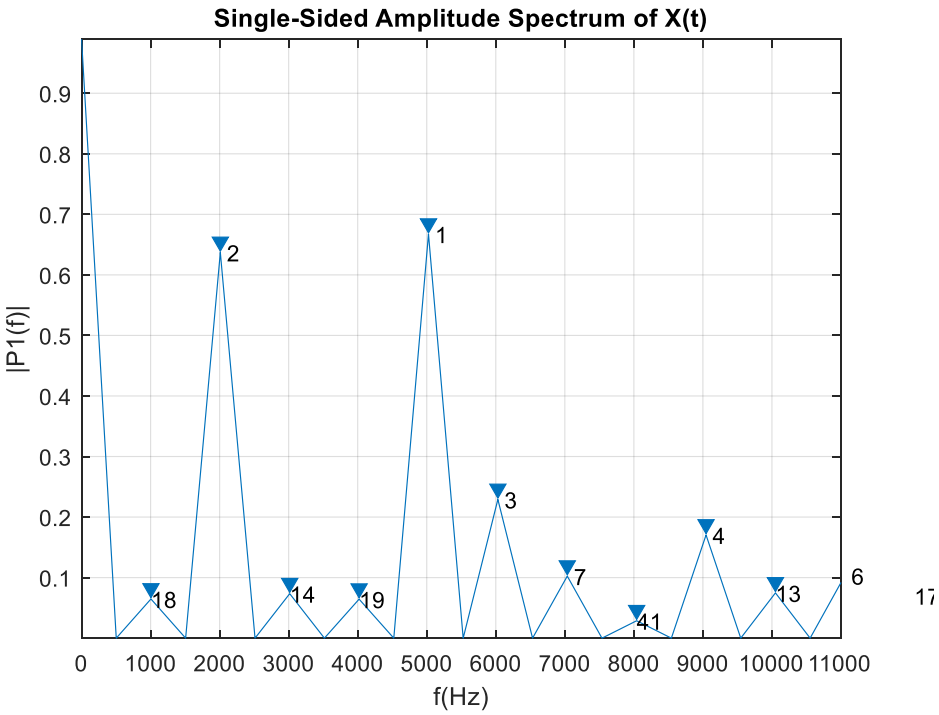

Figura 7. *Potencia en función de la frecuencia. Caso 2-5 kHz.*

En la Figura 7 observamos dos picos que sobresalen por encima del resto. Este es un caso que roza la perfección, porque la relación entre las frecuencias máximas sobrepasa el 95% y el tercer máximo está muy alejado de las frecuencias objetivo que deseamos tener, 2 kHz y 5 kHz.

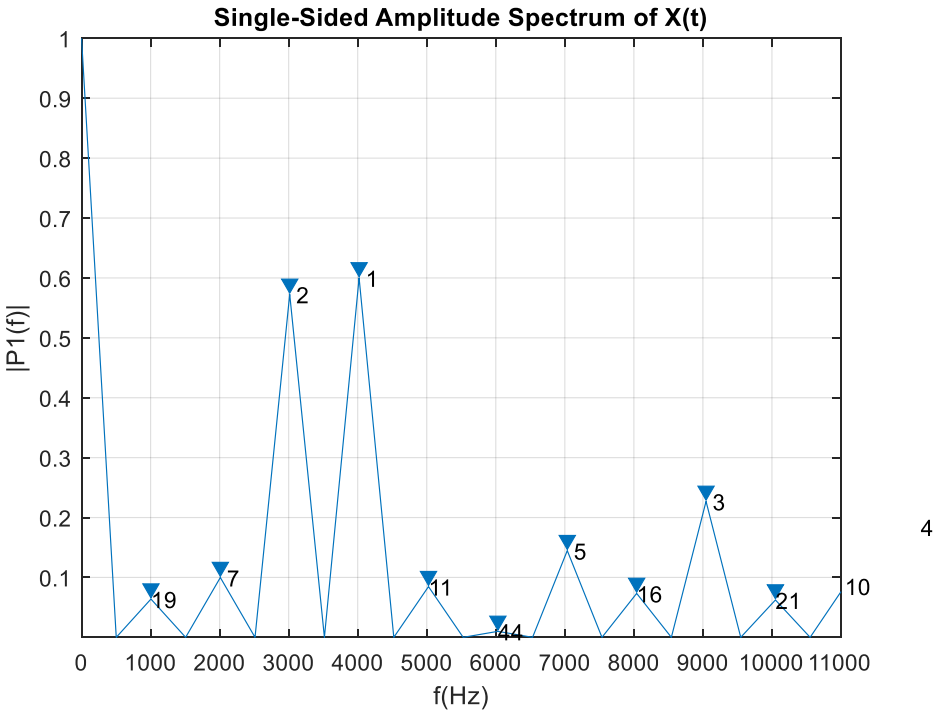

Figura 8. *Potencia en función de la frecuencia. Caso 3-4 kHz.*

Al igual que en la anterior Figura 7, en la Figura 8 obtenemos un caso similar, con ambas frecuencias dominantes sobresaliendo por encima del resto, y el tercer pico muy alejado de las dos frecuencias máximas, 3 kHz y 4 kHz.

Como se puede apreciar en las tres figuras anteriores, observamos la disposición de las frecuencias más representativas del foco. La potencia luminosa en función de la frecuencia, y por orden de potencia, siempre veremos en las dos primeras posiciones el par de frecuencias del código. Tendremos en cuenta el caso de armónicos que pueden ser un problema a la hora de la detección.

Una vez comprobado que la detección funciona correctamente, es hora de ponerse manos a la obra y desarrollar un algoritmo para asignar pares de frecuencias, de una manera ordenada y a la vez eficiente, que no nos de problemas a la hora de ubicarnos, puesto que es el propósito más importante de este proyecto.

3. POSICIONAMIENTO DE LUMINARIAS Y ASIGNACIÓN DE CÓDIGOS A LAS MISMAS

3.1 DESCRIPCIÓN DE LA PROPUESTA

A la hora de colocar la iluminación del edificio nos encontramos con el problema de cómo detectar la posición del dispositivo móvil. El objetivo primordial era detectar las frecuencias con este dispositivo y automáticamente plasmar la posición, pero ¿cómo distribuir dichas frecuencias? ¿Podíamos repetirlas? Y si podíamos repetirlas, ¿cuántas veces?

Se optó por generar una cuadrícula, como si fuera un plano de coordenadas, colocar el plano del edificio por debajo de esa cuadrícula y detectar las posiciones donde podemos, o no, poner luminarias (que tendrán asignados pares de frecuencias). Nótese que, en el caso de patios, zonas ciegas no transitadas, paredes, pasillos estrechos, etc., las luminarias no se asignarán de forma homogénea o no se asignarán.

Se decidió que en el dispositivo podíamos detectar dos focos con 2 frecuencias cada uno, en principio. Más adelante podríamos investigar si cada foco podía obtener 3 o incluso 4 frecuencias, y el algoritmo sería más provechoso y eficiente. Además, debemos considerar que en el caso en que los focos estén separados de manera que no se detecten 2 simultáneamente, siempre podemos saber el último que vimos. Así, para proseguir se considera que tenemos la información de 2 focos vecinos en la posición que ocupamos.

El siguiente paso sería realizar combinaciones para adjudicar códigos a focos adyacentes. Es decir, en ningún momento podamos tener dos focos adyacentes con mismos códigos (en dos lugares diferentes), y así no tener la duda de si estamos en un sitio u otro del espacio (en otras palabras, que no hay 2 focos vecinos que tengan idénticos códigos más que en una parte del edificio). Por tanto, al ser una cuadrícula, debemos tener en cuenta los focos vecinos de cada uno de los focos; cada foco tendrá 8 vecinos (norte, sur, este, oeste, noreste, sureste, noroeste y suroeste) como máximo, puesto que podrá tener menos si está cercano a un lugar donde no hay que asignar pares de frecuencias a los focos, como por ejemplo una pared, un patio o un límite del plano.

Para aclarar el concepto: podremos tener un código asignado a dos focos en dos sitios diferentes del plano si sus vecinos adyacentes poseen códigos totalmente diferentes unos de otros. Si algún vecino tratara de repetirse, este caso se descartaría.

3.2 DISTRIBUCIÓN DE LUMINARIAS

A la hora de desarrollar una plantilla del plano que necesitábamos rellenar con zonas a cubrir por luminarias y zonas vacías, buscamos una herramienta capaz de diseñarla. Inkscape es una herramienta editor de gráficos vectoriales libre y de código abierto. Inkscape puede crear y editar diagramas, líneas, gráficos, logotipos, e ilustraciones complejas con herramientas flexibles de dibujo. Posee una amplia compatibilidad de formatos de archivo, por ejemplo .pdf y .png, que son los formatos utilizados en el proyecto.

Así, a partir de disponer de un plano de las plantas del edificio a cubrir en cualquiera de los formatos aceptados, usamos Inkscape para colocar encima del plano origen las salas y lugares en los que se desea que haya luminarias emitiendo códigos, y los lugares donde no tendríamos que colocar focos, ya fuera porque no interesaba o porque, por ejemplo, es un lugar al aire libre o cuarto privado cerrado. En la Figura 9 se muestra un ejemplo.

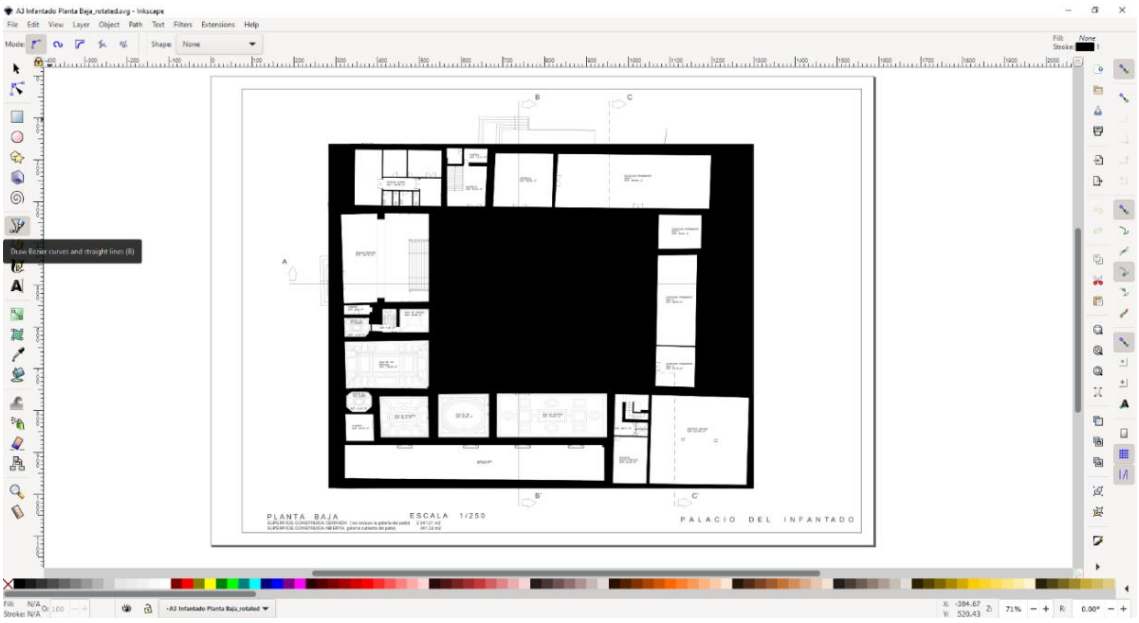

Figura 9. *Captura de InkScape. Diseño de la plantilla sobre el plano del museo.*

En la Figura 9 observamos una captura de la página principal de diseño en InkScape. En ella aparecen herramientas en el lado izquierdo, de las cuales las que nos han servido de mayor utilidad han sido *Draw Bezier curves* y *Measurement*. La primera nos ha proporcionado la posibilidad de realizar rectas y con ellas polígonos para poder rellenarlos de negro posteriormente y obtener las celdas ciegas que necesitamos (que marcaremos posteriormente como -1). Por otro lado, la segunda herramienta permite realizar mediciones. Al tener la escala del plano, podemos realizar los cálculos necesarios para dictaminar el tamaño de las futuras celdas.

Por tanto, las ubicaciones sin iluminación se rellenarían de negro y el resto de blanco, como se puede observar en la Figura 10. A partir de ella explicamos el procedimiento para calcular el número de píxeles que vamos a exportar en formato imagen .png.

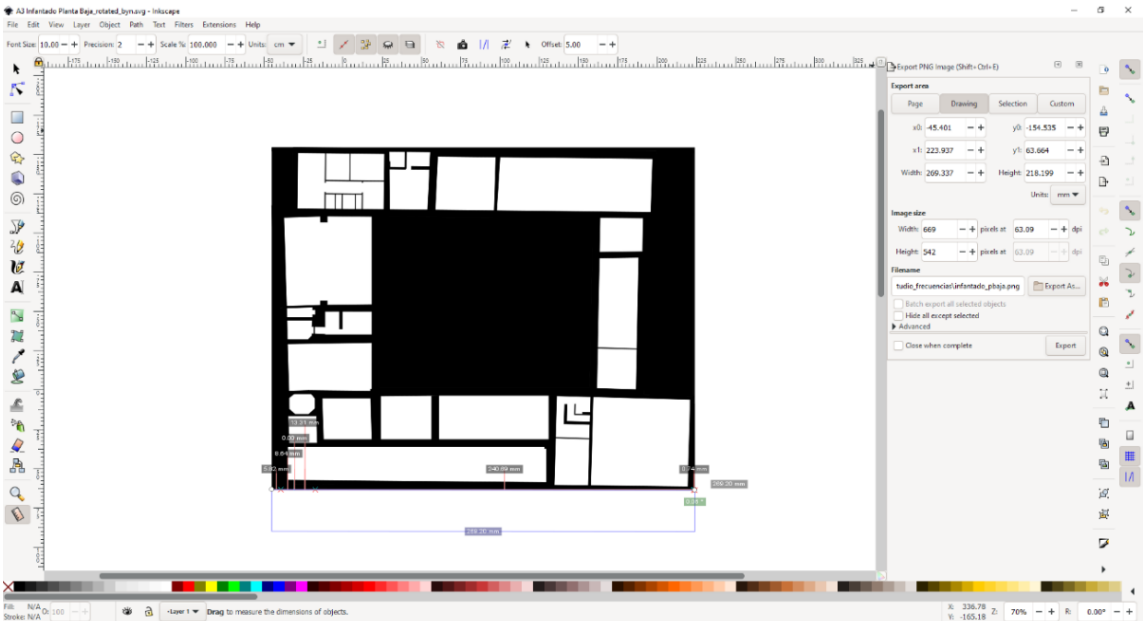

Figura 10. *Captura de InksCape. Exportar plantilla en formato .png.*

Con ayuda de la herramienta *Measurements* clicamos en dos puntos del diseño y el programa nos dará automáticamente la distancia entre dichos puntos. La escala del plano proporcionado por el museo, por ejmplo, es de 1:250. El ancho de la figura del museo son 268 mm, aproximadamente. Y el largo, 210 [mm]. Por tanto:

$$268\ [mm] * 250 \approx 67.000\ [mm] = 67\ [m]$$

$$210\ [mm] * 250 \approx 52.500\ [mm] = 52{,}5\ [m]$$

Calculados los datos de las dimensiones aproximadas del museo, hacer equivaler un metro a un píxel iba a estar muy desajustado y desproporcionado. Por tanto, quisimos obtener una mayor resolución a la hora de exportar la imagen y decidimos pasar las unidades a decímetros [dm]. Es por ello por lo que, finalmente, el resultado de las dimensiones de la imagen exportada fue 669x542 [dm^2].

A partir de aquí, comienza el trabajo en MATLAB para conseguir la matriz definitiva que usaremos en el algoritmo de ordenación. En la Tabla 4, vemos el script de MATLAB que transforma el archivo .png que hemos exportado desde InkScape.

En primer lugar, recogemos el archivo en la variable *alpha*, que será una matriz con valores entre 0 y 255 donde, tras haberla invertido debido a la función *imread()*, 0 serán las zonas blancas y 255 serán las zonas negras. Existirán valores intermedios en las transiciones de blanco a negro.

Tabla 4. *Script de MATLAB. Reducción del archivo .png*

```
%% Leemos el archivo .png y lo guardamos en alpha
[X,map,alpha] = imread('infantado_pprimera.png');

dist_celdas_dcm=20; %2 metros de distancia entre focos
pixeles_activos_min=dist_celdas_dcm*dist_celdas_dcm*0.90; %Porcentaj
e de blanco para tener en cuenta y decidir tipo de celda

[row,col]=size(alpha);

menos_unos = zeros(floor(row/dist_celdas_dcm),
floor(col/dist_celdas_dcm)); %Matriz de '0's tras la redimension

unos_ceros = alpha<127; %Invertir el fichero que genera imread
invertido

indice_x_tabla_01=1; %Índices del bucle
indice_y_tabla_01=1;

%% Recorremos la matriz original y rellenamos con 0 y -1 la matriz
redimensionada
for k = 1 : dist_celdas_dcm : row-dist_celdas_dcm
    for s = 1 : dist_celdas_dcm : col-dist_celdas_dcm

        unos_ceros20 = unos_ceros(k : dist_celdas_dcm+k-1, s :
dist_celdas_dcm+s-1);

        if sum(sum(unos_ceros20)) < pixeles_activos_min
            menos_unos(indice_x_tabla_01, indice_y_tabla_01) = -1;
        else
            menos_unos(indice_x_tabla_01, indice_y_tabla_01) = 0;
        end

        indice_y_tabla_01 = indice_y_tabla_01 + 1;

    end

    indice_y_tabla_01 = 1;
    indice_x_tabla_01 = indice_x_tabla_01 + 1;

end
```

Una vez conseguida la matriz en *alpha*, el objetivo es redimensionarla, de manera que nos quede una matriz mucho más reducida y compacta, en la que cada celda tenga una equivalencia de 2x2 [m] (esta es la distancia que se ha decidido en este ejemplo para separar las luminarias, pero este es un parámetro a introducir a MATLAB. Por tanto, recorreremos la matriz en sub matrices 20x20 (recordemos que la matriz *alpha* está en decímetros [dm]) y asignaremos el valor 0, o el valor -1, a una única casilla o celda, dependiendo del porcentaje de blanco que obtengamos en esa sub matriz.

El resultado del script será una matriz de reducidas dimensiones, como la que se observa en la Figura 11, conseguida mediante el plano del museo de Guadalajara que tiene unas dimensiones de 27x33:

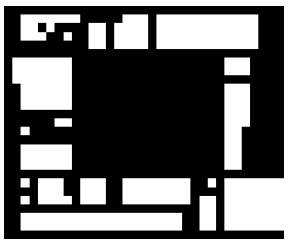

Figura 11. *Ejemplo de plantilla reducida final para el algoritmo de distribución.*

Antes de tratar de explicar el algoritmo que hemos desarrollado, vamos a realizar un pequeño análisis con ejemplos de lo que queremos. Para ello usaremos una matriz 4x4 que iremos rellenando con 9 identificadores posibles que harán referencia a las luminarias, para aclarar nuestro propósito.

Para ello tenemos que explicar los diferentes tipos de celdas que nos podemos encontrar en la matriz. Dos tipos:

- ***Celda a iluminar (0).*** Serán las celdas en las queremos introducir y configurar un foco inteligente. Posteriormente, pasarán a tener el valor de un identificador ID.
- ***Celda ciega (-1).*** Serán celdas donde no necesitamos o no disponemos de iluminación inteligente. Estas casillas mantendrán sus características hasta que decidamos cambiarla, si se necesita y se puede, por algún motivo concreto.

El algoritmo principal tratará de rellenar una cuadricula (matriz n x m) con un número concreto de posibilidades (identificadores), pudiendo repetir todos ellos con las siguientes restricciones:

- *Sea un número repetido 2 o más veces en la cuadrícula cuyos vecinos adyacentes (izquierda, derecha, arriba, abajo y diagonales) sean totalmente diferentes unos de otros -por ejemplo, el identificador número 5 tiene como vecinos: 1, 2, 3, 4, 6, 7, 8, 9-, podremos repetir dicho identificador -número 5- si ninguno de sus vecinos es igual a los anteriores.*

- *Sea un identificador repetido 2 o más veces en la cuadrícula, la distancia entre ellos debe ser mayor o igual que una distancia mínima impuesta (en principio esta distancia será de 3 cuadrículas). De esta manera evitaremos casos a-b-a en tres casillas consecutivas de una manera más estricta.*

La matriz original que vamos a utilizar para explicar para explicar a modo de ejemplo será la que se presenta en la Figura 12.

-1	0	0	0
0	0	-1	0
0	0	0	-1
0	-1	0	0

Figura 12. *Matriz original ejemplo.*

En el primer paso, como se observa en la Figura 13, de manera aleatoria se han colocado todos los identificadores en las casillas donde había algún 0.

-1	1	0	7
0	8	-1	2
6	3	4	-1
5	-1	9	0

Figura 13. *Matriz ejemplo tras la primera iteración.*

Seguidamente, como se aprecia en la Figura 14, de manera aleatoria se han intentado colocar de nuevo todos los identificadores. En este caso, al llegar al identificador número 3, hemos visto que no ha sido posible escribirlo en la matriz, porque no cumple con las restricciones anteriormente descritas: ni la condición de identificadores vecinos ni la condición de distancia mínima. Por tanto, hemos pasado a intentar colocar los siguientes identificadores, y este número 3 queda descartado.

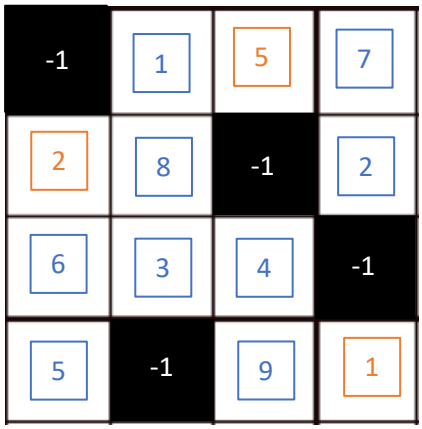

Figura 14. *Matriz ejemplo tras finalizar el algoritmo de distribución.*

Al escribir el identificador 5 en la segunda ronda, hemos acabado con la asignación, es decir, ya no quedan sitios libres donde colocar identificadores. En este ejemplo el resultado ha sido favorable porque hemos conseguido nuestro objetivo. Sin embargo, no siempre será posible porque tenemos una gran dependencia de lo grande que sea la matriz, las celdas ciegas que existan y la cantidad de identificadores posibles.

De esta forma sencilla, pero a la vez muy eficiente y efectiva, y tras realizar diferentes pruebas y combinaciones de lo que podría suponer una matriz de grandes dimensiones, hemos desarrollado el siguiente algoritmo de reparto de luminarias.

3.3 ASIGNACIÓN DE CÓDIGOS

Teniendo en conocimiento las restricciones, procedemos a elaborar los pasos a seguir en el algoritmo de reparto de ubicaciones, que podemos ir apreciando en el diagrama de flujo de la Figura 15:

1. Cálculo de una lista de posibles parejas *list_pairings* (total de combinaciones) siguiendo la fórmula anteriormente explicada de combinaciones sin repetición de m elementos tomados de n en n. Con ello tendremos el número total de identificadores.
2. Creación de la matriz objetivo Z. En primera instancia estará vacía con todo 0 en los espacios donde sí debe haber luminarias que emitan códigos. Los lugares anteriormente mencionados que no debemos tener en cuenta porque no se van a asignar tendrán un -1 en su espacio (por ejemplo, cuartos ciegos, patios, paredes muy amplias). Se irá rellenando poco a poco con los identificadores.
3. Creación de una matriz *used_pair*. Contiene todos los identificadores en ambos ejes. En primera instancia estará vacía con todo a 0. Se irá completando con 1 si el cruce de vecinos ha sido utilizado. Por ejemplo, si por el momento el identificador 5 tiene como vecinos el 14 y el 23, estas casillas cruzadas contendrán un 1 en su interior.

4. Mientras exista al menos un 0 en la matriz Z, trataremos de buscar una solución. Estableceremos un límite de iteraciones por si la solución que intentamos encontrar es imposible.
5. En la primera iteración, mediante un bucle, escribimos aleatoriamente todas los identificadores posibles, sin posibilidad de error porque todavía no hay ningún identificador repetido.
   Actualizamos la matriz *used_pair* cada vez que asignemos un identificador a la matriz Z, para guardar los vecinos utilizados que ya no se pueden volver a dar.
6. En la segunda iteración buscamos aleatoriamente un sitio libre en la matriz Z. A continuación, buscamos el mismo identificador que escribimos en el paso anterior en la matriz Z. Tendremos en cuenta las restricciones anteriormente mencionadas, tanto la distancia mínima como la repetición de vecinos adyacentes.
   Actualizamos la matriz *used_pair* cada vez que asignemos un identificador a la matriz Z, para guardar los vecinos utilizados que ya no se pueden volver a dar.
7. En la tercera iteración se realiza el mismo procedimiento que en la segunda, pero comparando con todos los identificadores iguales repetidos anteriormente escritos, es decir, el de la primera y el de la segunda si se llegó a escribir.
8. Y así sucesivamente, hasta poder completar la matriz, o por el contrario llegamos al número límite de intentos *(Timeout)* de relleno y abortamos el desarrollo del algoritmo.

En la Figura 15 se muestra el diagrama de flujo del proceso a seguir.

A continuación, procedemos a explicar las condiciones del código. En la Tabla 5 se muestra un código del proceder marcado en la primera restricción. Este fragmento del código utilizado en MATLAB nos muestra la lógica de decisión a la hora de colocar un identificador a la plantilla en cuestión. De manera ordenada vamos recorriendo los diferentes posibles vecinos. La variable no_asignar será la encargada de tomar la decisión final.

Si el valor del identificador vecino de la izquierda es distinto de 0 o -1, entramos en la condición. Seguidamente, si ese valor está en la lista de pares usados, used_pair, volvemos a entrar en la condición y colocamos en la variable no_asignar un 1, dando a entender que ese identificador no vamos a poder utilizarlo. En caso contrario, saltaremos al resto de condiciones para evaluarlas del mismo modo que con la primera condición.

Finalmente llegaremos, si todo va bien según las condiciones vecinales, a asignar el identificador en el lugar elegido aleatoriamente por el algoritmo.

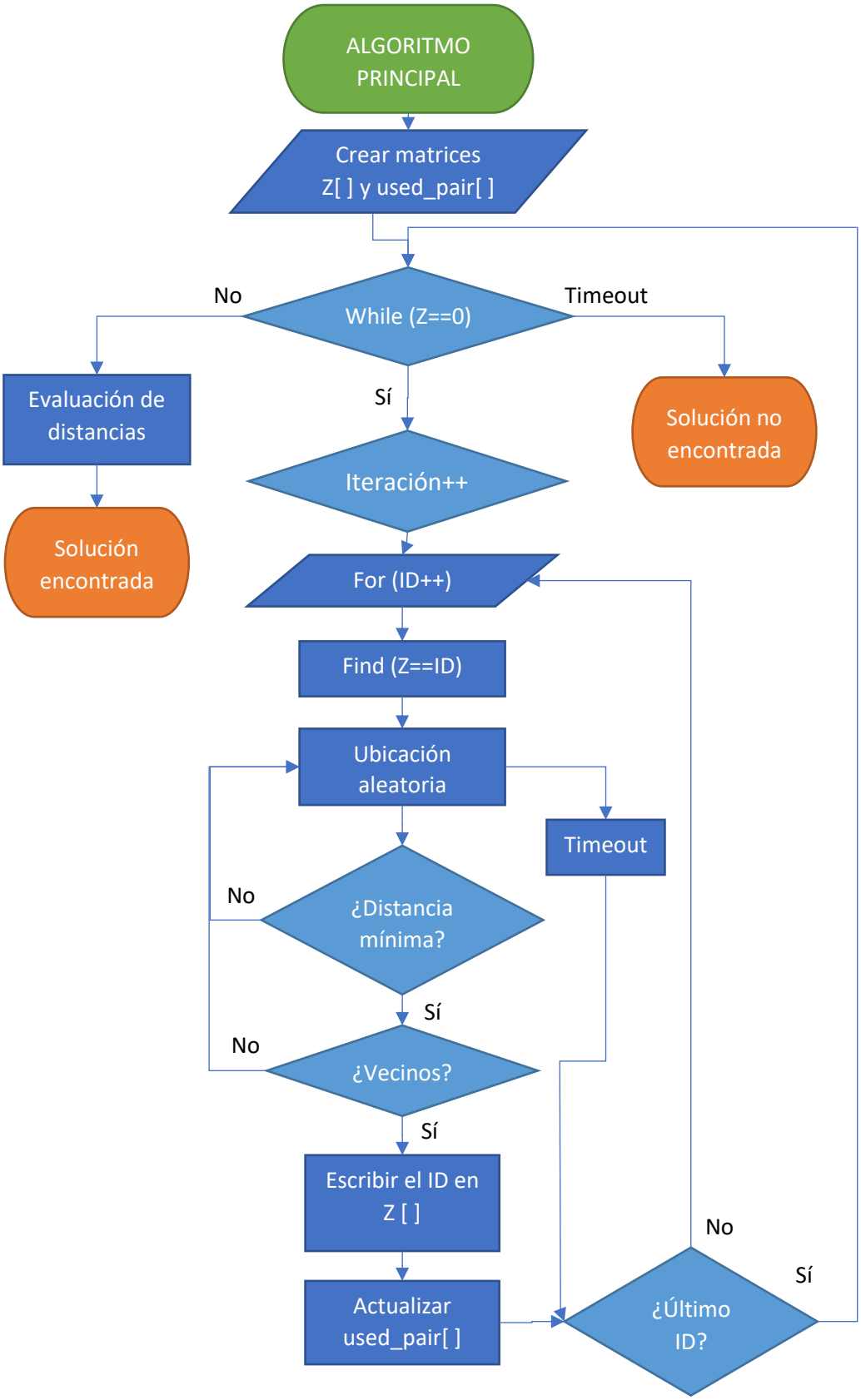

Figura 15. *Diagrama de flujo. Algoritmo de distribución.*

***Tabla 5**. Script de MATLAB. Comparaciones con identificadores ID vecinos.*

```
%% Comparaciones con adyacentes (arriba, abajo, izquierda, derecha y
diagonales)
no_asignar = 0;

%% Laterales
if ((xx-1)>0 && Z(yy, xx-1) ~= 0 && Z(yy, xx-1) ~= -1)
      if (used_pair(i,Z(yy, xx-1)) == 1)
                  no_asignar = 1;
      end
end
if ((xx+1)<=ladox && Z(yy, xx+1) ~= 0  && Z(yy, xx+1) ~= -1)
      if (used_pair(i,Z(yy, xx+1)) == 1)
            no_asignar = 1;
      end
end
if ((yy-1)>0 && Z(yy-1,xx) ~= 0 && Z(yy-1, xx) ~= -1)
       if (used_pair(i,Z(yy-1, xx)) == 1)
            no_asignar = 1;
       end
end
if ((yy+1)<=ladoy && Z(yy+1, xx) ~= 0 && Z(yy+1, xx) ~= -1)
       if (used_pair(i,Z(yy+1, xx)) == 1)
            no_asignar = 1;
       end
end
%% Diagonales
if ((xx-1)>0 && (yy-1)>0 && Z(yy-1, xx-1) ~= 0 && Z(yy-1, xx-1) ~= -
1)
      if (used_pair(i,Z(yy-1, xx-1)) == 1)
            no_asignar = 1;
       end
end
if ((xx+1)<=ladox && (yy+1)<=ladoy && Z(yy+1, xx+1) ~= 0 && Z(yy+1,
xx+1) ~= -1)
       if (used_pair(i,Z(yy+1, xx+1)) == 1)
            no_asignar = 1;
       end
end
if ((xx+1)<=ladox && (yy-1)>0 && Z(yy-1, xx+1) ~= 0 && Z(yy-1, xx+1)
~= -1)
            if (used_pair(i,Z(yy-1, xx+1)) == 1)
            no_asignar = 1;
       end
end
if ((xx-1)>0 && (yy+1)<=ladoy && Z(yy+1, xx-1) ~= 0 && Z(yy+1, xx-1)
~= -1)
      if (used_pair(i,Z(yy+1, xx-1)) == 1)
                  no_asignar = 1;
       end
end
if (no_asignar == 0) % Se asigna el identificador
            Z(yy,xx) = i;
      asignado = 1;
end
```

El siguiente fragmento de código de la Tabla 6, utilizado en MATLAB, nos muestra la lógica de decisión a la hora de decidir si cumplimos con la distancia mínima impuesta. En primer lugar, buscamos donde se encuentran los identificadores iguales y, seguidamente, con ayuda del Teorema de Pitágoras calculamos la distancia entre los puntos a través de las coordenadas cartesianas del plano. Si cumplimos con la condición, la ubicación será óptima y correcta.

Tabla 6. *Script de MATLAB. Comprobaciones sobre la distancia mínima.*

```
%% Comprobamos distancia mínima con mismo id en la distribución
value_min_distR = 3;
[idi_y0, idi_x0] = find(Z==i);
%% Nos devuelve las coordenadas de las casillas de la matriz Z en
las que se encuentra el valor del identificador
W = [idi_y0, idi_x0];
V = [yy,xx].*ones(length(idi_y0),1);
R = V-W;
min_distR = min(sqrt(sum((R.*R)')));
%% Teorema de Pitágoras: hip=sqrt(a^2+b^2)
if ((min_distR >= value_min_distR))
      ubicando = 0; % Ubicación correcta
end
```

Como mejora del algoritmo, el objetivo será distribuir las frecuencias en zonas menos amplias que se puedan cubrir con una sola cuadricula y con dicho número de frecuencias (es decir dividir un espacio amplio en subespacios más pequeños). En principio trataremos 2 zonas, la este y la oeste de cada planta del edificio utilizado para las pruebas iniciales y depuración del algoritmo, pero se podrá dividir en más zonas en versiones futuras como, por ejemplo, 4 zonas, aunque otra opción de futuro es llegar a tener el suficiente número de frecuencias para poder juntarlas y así ampliar nuestras posibilidades en un solo espacio más extenso.

Con la ayuda de la tecnología GPS podremos seleccionar *grosso modo* una de las cuadrículas (zonas) y a partir de seleccionar dicha zona posicionarnos en la cuadrícula correcta a través del algoritmo principal. Al tener una exactitud de posicionamiento con el GPS bastante alta, no será difícil concretar la zona en el edificio, aunque nos encontremos en su interior. No es el objetivo de este proyecto realizar una aplicación con GPS, pero es una aproximación más que correcta para desarrollarla en el futuro.

4. BASE DE DATOS Y COMUNICACIÓN CON LAS LUMINARIAS.

Este apartado lo dedicaremos a explicar cómo y por qué hemos realizado la base de datos que, a posteriori, utilizaremos para la comparación y definición final de la ubicación del usuario en el escenario elegido.

4.1 DESCRIPCIÓN DE LAS DIFERENTES TABLAS DE DATOS

Una vez conseguida la matriz final con todos los identificadores ID, tras la ejecución del algoritmo de reparto de ubicaciones, ha llegado el momento de caracterizar a nuestras lámparas inteligentes. Para ello debemos realizar una serie tablas, que en su conjunto formarán una base de datos, que nos muestren toda la información necesaria de dichas lámparas inteligentes.

Siguiendo con la explicación del algoritmo de reparto de ubicaciones, necesitamos tres tablas. La primera de ellas sólo contendrá información sobre el identificador ID y qué frecuencias le corresponden (cada lámpara tiene asignado un indicador).

La segunda, Tabla 7, contendrá todos los identificadores ID tanto en el eje vertical como en el horizontal, y en ella iremos apuntando las direcciones IP de los vecinos que ya han sido asignados a uno dado, en cualquier parte del espacio. Si nos fijamos en la tabla 4, podemos ver un ejemplo de cómo se comportaría la tabla. Por ejemplo, el identificador ID 1 tendrá asignada la dirección IP .44 compartiendo unos vecinos de su alrededor, pero a su vez el mismo identificador ID 1 tendrá asignada la dirección IP .97 cuando tiene otros vecinos de su alrededor en otro lugar del espacio. Esto quiere decir que al menos tendremos dos mismos ID 1 con dos direcciones IP diferentes en el plano que hemos desarrollado. Por tanto, será una matriz de decisión en la que escojamos primero un identificador ID y, posteriormente, el identificador ID de su vecino. Por tanto, no tendremos la posibilidad de equivocarnos a la hora de distinguir su dirección IP.

A continuación, vamos a describir la tercera tabla de nuestra base de datos, que contendrá los siguientes campos:

- ***Dirección IP*.** Dirección de la red local.
- ***PWM o Seno (0-1).*** Tipo de señal que va a emitir la luminaria (de momento solo se va a usar PWM).
- ***Número de frecuencias (1, 2 o 3).*** En caso de ser seno se emite una única frecuencia.
- ***Frecuencia 1.*** Primera frecuencia del foco.
- ***Frecuencia 2.*** Segunda frecuencia del foco.
- ***Frecuencia 3.*** Tercera frecuencia del foco.
- ***Porcentaje del ciclo de trabajo de la señal PWM.***

- ***Identificador ID.*** Número que identifica la combinación de frecuencias.
- ***Coordenada X.***
- ***Coordenada Y.***
- ***Planta.*** Baja, 0, o Primera, 1 (en este caso es un edificio con 2 plantas).
- ***Zona del plano.*** Este, 0 u Oeste, 1.
- ***LANZAMIENTO (VERDADERO o FALSO).*** Campo de tipo lógico que nos servirá, posteriormente, para elegir qué códigos vamos a enviar, en un momento dado, ayudados por una interfaz gráfica.

El motivo por el cual realizamos esta base de datos será para poder controlar lo que queremos configurar y tener acceso a toda la información registrada en cuanto la iluminación inteligente del edificio. De esta forma podemos encontrar de una forma muy sencilla cualquier lámpara que nos esté dando problemas, o que simplemente queramos sustituir o cambiar su configuración específica.

Por ejemplo, si observamos la Tabla 8, vemos que las primeras filas están vacías y el campo dirección IP con valor -1. Esto significa que en esas zonas del plano no tendremos ningún foco que colocar o que configurar, ya sea porque no hay iluminación o no interesa tener nada configurado en esa zona del plano. Aun así, debemos mantener guardadas esas filas porque en el futuro esa situación puede cambiar y nos puede interesar configurar algo en esas zonas del plano ahora descartadas. Podremos ordenarlas como nos apetezca, por zonas, por plantas e incluso por identificadores, siempre respetando el algoritmo.

Finalmente, todas estas tablas creadas en MATLAB las exportaremos a hojas de cálculo Excel para que sean más manejables y visuales, también con la intención de poder subir los archivos a un servidor que posteriormente utilizarán (accederán a ellas de forma automática) las aplicaciones en los dispositivos de los usuarios para descargarlos y así poder detectar la ubicación.

Tabla 7. *Base datos. Identificadores ID con sus respectivas direcciones IP.*

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45
1		0	44	0	0	97	0	0	97	44	145	145	0	97	0	97	67	0	0	0	0	145	0	67	97	0	44	97	67	97	0	44	44	67	0	67	0	0	145	97	0	0	145	0	0
2	0		0	0	0	12	52	140	0	12	8	140	12	8	52	140	0	0	0	0	0	140	8	140	0	0	0	0	0	140	0	0	140	8	140	0	52	0	8	0	0	0	0	0	0
3	31	0		0	0	0	0	0	34	31	61	0	0	136	0	0	61	0	34	34	136	34	34	31	0	31	31	0	61	0	61	31	31	34	31	0	136	136	34	136	61	34	0	0	0
4	0	0	0		66	0	27	0	100	27	100	27	0	0	66	0	66	100	27	100	100	0	0	0	100	27	0	92	0	100	0	0	92	66	92	92	0	0	92	0	66	0	0	0	100
5	0	0	0	63		0	0	0	0	2	63	2	0	0	0	0	63	0	50	0	0	63	0	0	0	0	2	2	50	50	0	2	0	121	121	50	0	121	0	50	63	121	121	0	0
6	108	11	0	0	0		80	138	0	11	0	138	11	108	138	0	80	80	138	0	80	0	0	0	138	0	0	0	108	80	11	138	0	0	0	0	0	0	0	138	0	0	11	138	
7	0	53	0	41	0	79		41	53	0	0	0	0	0	53	0	79	79	41	41	79	0	13	0	0	41	0	0	0	0	79	0	0	13	0	0	53	0	0	0	0	0	0	53	0
8	0	139	0	0	0	139	42		0	42	130	139	0	0	139	139	0	0	0	42	0	0	0	139	0	42	42	0	0	0	0	0	139	0	0	0	0	130	0	0	139	0	130	0	0
9	110	0	33	88	0	0	54	0		0	88	0	0	110	54	110	0	110	0	0	0	33	0	0	0	33	0	0	0	88	0	33	33	33	0	0	0	0	33	0	0	33	0	54	110
10	30	25	30	14	14	25	0	30	0		0	14	25	0	0	0	0	0	25	30	0	0	0	30	0	14	30	30	0	25	0	14	0	0	30	0	0	0	0	0	0	0	0	0	0
11	144	9	59	87	59	0	0	144	87	0		144	0	0	0	0	0	87	0	0	0	59	9	0	87	0	0	0	0	87	59	9	0	0	0	0	0	144	9	0	59	0	144	9	0
12	132	152	0	15	15	152	0	152	0	15	132		0	0	0	0	0	0	0	15	0	132	0	152	0	15	15	15	0	0	0	15	0	0	90	90	0	0	132	0	152	0	132	90	0
13	0	24	0	0	0	24	0	0	0	24	0	0		115	0	119	0	0	24	0	115	0	115	0	0	0	0	119	119	24	0	24	119	39	119	24	0	119	119	0	0	119	0	24	0
14	109	7	122	0	0	109	0	0	109	0	0	0	122		0	109	0	0	0	0	122	0	122	0	0	7	0	0	0	109	0	0	0	7	0	0	122	0	7	0	0	0	0	0	7
15	0	56	0	69	0	125	56	125	56	0	0	0	0	0		0	69	0	125	0	0	0	0	69	125	0	0	0	125	0	0	0	125	69	0	0	56	0	125	0	69	0	125	56	0
16	98	127	0	0	0	0	0	127	98	0	0	0	127	98	0		0	98	0	0	134	134	0	0	98	0	0	98	127	127	0	0	127	0	127	0	0	0	134	98	0	127	0	0	98
17	65	0	65	65	65	77	77	0	0	0	0	0	0	0	65	0		155	0	0	77	155	4	4	77	0	4	4	65	155	0	0	0	65	4	0	0	0	0	0	65	77	0	0	0
18	0	0	0	99	0	81	81	0	99	0	99	0	0	0	0	99	142		0	99	0	142	0	0	99	0	0	0	0	142	81	0	0	0	142	0	0	0	0	99	0	0	142	0	99
19	0	0	48	40	38	137	40	0	0	38	0	0	38	0	137	0	0	0		48	0	48	0	0	137	40	0	0	38	38	0	0	0	0	0	48	0	0	0	48	137	0	0	0	137
20	0	0	35	112	0	0	29	29	0	29	0	29	0	0	0	0	0	112	35		112	35	35	29	0	29	29	29	0	112	0	35	0	0	0	35	0	0	35	35	0	0	0	0	112
21	0	0	123	113	0	78	78	0	0	0	0	0	123	123	0	147	78	0	0	113		147	123	0	78	0	0	0	0	113	0	0	0	0	0	0	123	0	147	0	0	78	0	0	0
22	146	154	47	0	60	0	0	0	47	0	60	146	0	0	0	146	154	154	47	47	146		0	154	0	0	0	0	0	154	0	0	0	0	0	0	0	146	0	60	47	0	0	0	
23	0	22	22	0	0	0	0	0	0	0	22	0	114	114	0	0	5	0	0	22	114	0		5	0	5	0	0	0	0	0	22	0	0	5	22	0	0	22	0	0	0	0	22	5
24	71	153	17	0	0	0	0	153	0	17	0	153	0	0	71	0	17	0	0	17	0	153	17		0	71	17	17	0	153	0	0	71	17	71	71	0	0	0	71	0	0	0	0
25	83	0	0	86	0	124	0	0	0	0	86	0	0	0	124	86	76	86	124	0	76	0	0	0		76	0	83	0	83	0	0	0	0	0	0	76	0	124	86	0	76	124	0	0
26	0	0	19	28	0	0	28	28	19	28	0	28	0	19	0	0	0	0	28	28	0	0	19	73	73		0	28	0	0	0	19	0	19	19	73	73	0	0	0	0	73	0	0	19
27	43	0	43	0	3	0	0	43	0	43	0	3	0	0	0	0	3	0	0	43	0	0	0	3	0	0		3	0	0	0	0	94	94	94	0	0	94	94	0	0	0	0	0	0
28	84	0	0	104	16	0	0	0	0	16	0	16	104	0	0	84	16	0	0	16	0	0	0	16	84	16	16		104	84	0	0	104	0	104	104	0	0	104	84	0	0	104	0	0
29	64	0	64	0	51	0	0	0	0	0	0	0	118	0	118	118	64	0	51	0	0	0	0	0	0	0	0	118		51	0	0	118	64	0	118	0	0	118	0	64	0	118	0	0
30	96	141	0	101	37	96	0	0	101	37	101	0	37	96	0	141	141	141	37	101	101	141	0	141	96	0	0	96	37		0	37	0	0	141	37	0	0	0	37	0	0	141	0	0
31	0	0	58	0	0	82	82	0	0	0	58	0	0	0	0	0	0	82	0	0	0	0	0	0	0	0	0	0	0	0		0	0	0	0	0	0	0	0	58	0	0	0	0	0
32	32	0	32	0	1	23	0	0	32	1	23	1	23	0	0	0	0	0	0	23	0	0	23	0	0	32	0	0	0	23	0		32	32	32	23	0	0	0	0	0	32	0	23	0
33	45	126	45	93	0	126	0	126	45	0	0	0	126	0	126	126	0	0	0	0	0	0	0	0	0	0	93	93	126	0	0	45		0	0	0	0	93	93	0	0	45	126	0	0
34	68	20	20	68	107	0	26	0	20	0	0	0	26	20	68	0	68	0	0	0	0	0	0	68	0	20	107	0	68	0	0	20	0		107	68	0	107	20	0	68	107	0	0	20
35	0	128	18	91	128	0	0	0	0	18	0	91	128	0	0	128	18	128	0	0	0	0	0	18	18	0	18	95	91	0	128	0	18	0	95		91	0	95	0	0	0	128	128	91	18
36	70	0	0	103	36	0	0	0	0	0	0	103	36	0	0	0	0	0	36	36	0	0	36	70	0	70	0	103	103	36	0	36	0	70	103		70	0	103	36	0	0	103	103	0
37	0	55	135	0	0	0	55	0	0	0	0	0	0	135	55	0	0	0	0	0	135	0	0	74	74	74	0	0	0	0	0	0	0	0	74		135	0	135	74	74	0	0	0
38	0	0	148	0	106	0	0	143	0	0	143	0	106	0	0	0	0	0	0	0	0	0	0	0	0	0	106	0	0	0	0	0	106	106	106	0	148		106	148	0	106	143	0	0
39	133	21	21	105	0	0	0	0	21	0	21	133	105	21	116	133	0	0	0	21	133	133	21	0	116	0	105	105	105	0	0	0	105	21	0	116	0	105		0	0	105	116	116	0
40	85	0	149	0	49	0	0	0	0	0	0	0	0	0	0	85	0	85	49	49	0	0	0	0	85	0	0	85	0	49	0	0	0	0	0	49	149	149	0		0	0	0	0	0
41	0	0	62	62	62	151	0	151	0	0	62	151	0	0	72	0	62	0	151	0	0	62	0	72	0	0	0	0	62	0	62	0	0	72	0	0	72	0	0	0		0	0	0	151
42	0	0	46	0	120	0	0	0	46	0	0	0	120	0	0	120	75	0	0	0	75	46	0	0	75	75	0	0	0	0	0	46	46	120	120	0	75	120	120	0	0		120	0	0
43	131	0	0	0	129	0	0	131	0	0	131	131	0	0	117	0	0	129	0	0	0	0	0	0	117	0	0	117	117	129	0	0	117	0	129	117	0	131	117	0	0	129		117	0
44	0	0	0	0	0	10	57	0	57	0	10	102	10	0	57	0	0	0	0	0	0	0	10	0	0	0	0	0	0	0	0	10	0	0	102	102	0	0	102	0	0	0	102		0
45	0	0	0	111	0	150	0	0	111	0	0	0	0	6	0	111	0	111	150	111	0	0	6	0	0	6	0	0	0	0	0	0	0	6	6	0	0	0	0	150	0	0	0	

Tabla 8. *Base de datos. Características de las diferentes ubicaciones y luminarias.*

Direccion IP	PWM-1 o Seno-0	Numero de Frecuencias	Frecuencia 1	Frecuencia 2	Frecuencia 3	%PWM	ID	Coordenada X	Coordenada Y	Planta (Baja-0 Primera-1)	Zona (Este-0 Oeste-1)	LANZAMIENTO
-1	0	0	0	0	0	0	0	1	1	0	0	FALSO
-1	0	0	0	0	0	0	0	1	2	0	0	FALSO
-1	0	0	0	0	0	0	0	1	3	0	0	FALSO
-1	0	0	0	0	0	0	0	1	4	0	0	FALSO
-1	0	0	0	0	0	0	0	1	5	0	0	FALSO
-1	0	0	0	0	0	0	0	1	6	0	0	FALSO
-1	0	0	0	0	0	0	0	1	7	0	0	FALSO
-1	0	0	0	0	0	0	0	1	8	0	0	FALSO
-1	0	0	0	0	0	0	0	1	9	0	0	FALSO
-1	0	0	0	0	0	0	0	1	10	0	0	FALSO
-1	0	0	0	0	0	0	0	1	11	0	0	FALSO
-1	0	0	0	0	0	0	0	1	12	0	0	FALSO
-1	0	0	0	0	0	0	0	1	13	0	0	FALSO
-1	0	0	0	0	0	0	0	1	14	0	0	FALSO
-1	0	0	0	0	0	0	0	1	15	0	0	FALSO
-1	0	0	0	0	0	0	0	1	16	0	0	FALSO
0	0	2	3000	9000	0	50	23	2	1	0	0	FALSO
-1	0	0	0	0	0	0	0	2	2	0	0	FALSO
1	0	2	5000	7000	0	50	32	2	3	0	0	FALSO
2	0	2	1000	6000	0	50	5	2	4	0	0	FALSO
3	0	2	4000	7000	0	50	27	2	5	0	0	FALSO
4	0	2	2000	10000	0	50	17	2	6	0	0	FALSO
5	0	2	3000	9000	0	50	23	2	7	0	0	FALSO
6	0	2	9000	10000	0	50	45	2	8	0	0	FALSO
7	0	2	2000	7000	0	50	14	2	9	0	0	FALSO
8	0	2	1000	3000	0	50	2	2	10	0	0	FALSO
9	0	2	2000	4000	0	50	11	2	11	0	0	FALSO
10	0	2	8000	10000	0	50	44	2	12	0	0	FALSO
11	0	2	1000	7000	0	50	6	2	13	0	0	FALSO
12	0	2	1000	3000	0	50	2	2	14	0	0	FALSO
-1	0	0	0	0	0	0	0	2	15	0	0	FALSO
-1	0	0	0	0	0	0	0	2	16	0	0	FALSO
13	0	2	1000	8000	0	50	7	3	1	0	0	FALSO
-1	0	0	0	0	0	0	0	3	2	0	0	FALSO
14	0	2	2000	3000	0	50	10	3	3	0	0	FALSO
15	0	2	2000	5000	0	50	12	3	4	0	0	FALSO
16	0	2	4000	8000	0	50	28	3	5	0	0	FALSO
17	0	2	3000	10000	0	50	24	3	6	0	0	FALSO
18	0	2	5000	10000	0	50	35	3	7	0	0	FALSO
19	0	2	4000	6000	0	50	26	3	8	0	0	FALSO
20	0	2	5000	9000	0	50	34	3	9	0	0	FALSO
21	0	2	6000	10000	0	50	39	3	10	0	0	FALSO
22	0	2	3000	9000	0	50	23	3	11	0	0	FALSO
23	0	2	5000	7000	0	50	32	3	12	0	0	FALSO
24	0	2	2000	6000	0	50	13	3	13	0	0	FALSO
25	0	2	2000	3000	0	50	10	3	14	0	0	FALSO
-1	0	0	0	0	0	0	0	3	15	0	0	FALSO
-1	0	0	0	0	0	0	0	3	16	0	0	FALSO
26	0	2	5000	9000	0	50	34	4	1	0	0	FALSO
-1	0	0	0	0	0	0	0	4	2	0	0	FALSO
27	0	2	1000	5000	0	50	4	4	3	0	0	FALSO
28	0	2	4000	6000	0	50	26	4	4	0	0	FALSO
29	0	2	3000	6000	0	50	20	4	5	0	0	FALSO
30	0	2	2000	3000	0	50	10	4	6	0	0	FALSO
31	0	2	1000	4000	0	50	3	4	7	0	0	FALSO
32	0	2	5000	7000	0	50	32	4	8	0	0	FALSO
33	0	2	1000	10000	0	50	9	4	9	0	0	FALSO
34	0	2	1000	4000	0	50	3	4	10	0	0	FALSO
35	0	2	3000	6000	0	50	20	4	11	0	0	FALSO

De los campos de la Tabla 8 citados anteriormente, los últimos cinco son meramente de información. Los primeros son los que vamos a utilizar para la configuración de las lámparas inteligentes. Dichos dispositivos van a estar preparados para recibir una cadena de caracteres y, de esta manera, puedan configurarse automáticamente a través de una red Wi-Fi con una conexión TCP que MATLAB es capaz de generar con cada foco por separado.

Lo trataremos como un modelo Cliente-Servidor tal y como podemos apreciar en la Figura 16, en el que el Cliente realiza las peticiones y el Servidor es el que las recibe, por tanto, el PC que estemos usando para lanzar los códigos actuará como cliente mientras que las lámparas inteligentes actuarán como servidores.

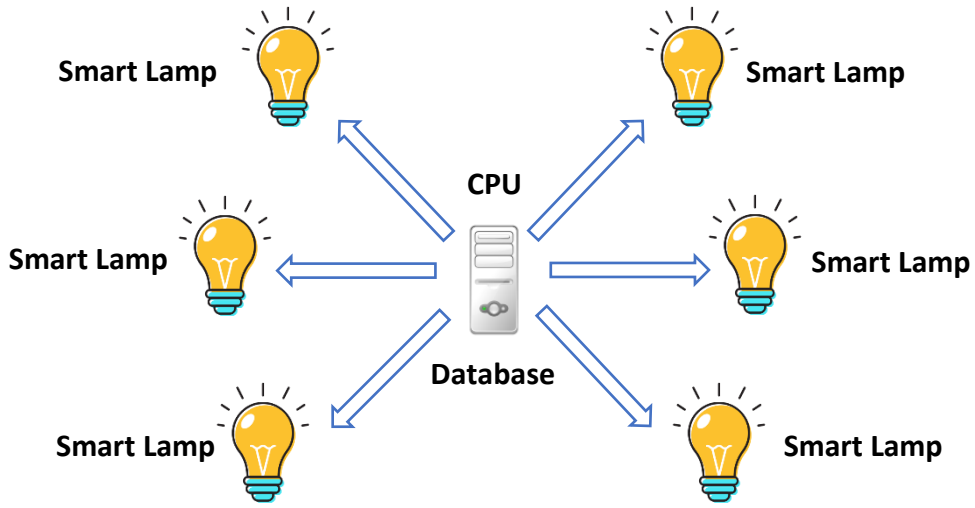

Figura 16. *Modelo Cliente-Servidores del sistema.*

La cadena de caracteres que vamos a enviar hacia las lámparas inteligentes será de longitud fija, y se compondrá de siete campos, separados entre sí por comas ",". Cada cadena de caracteres será diferente, pero mantendrá la estructura que se presenta en la Figura 17.

Figura 17. *Esquema de la cadena de caracteres a enviar hacia las lámparas inteligentes.*

El primer campo con la dirección IP contará con siete caracteres: la parte de red (en principio, la vamos a dividir en cuatro para las cuatro zonas del edificio piloto y así poder tener un control más exhaustivo), un punto separador y el dispositivo o foco. Debemos configurar una máscara de subred del tipo 255.255.252.0 teniendo la

posibilidad de configurar 1024 direcciones IP, y si en algún momento queremos añadir focos a nuestra plantilla, lo podremos hacer sin sobrepasar el direccionamiento máximo. Para las primeras direcciones, ya que la cadena de caracteres tiene que ser fija, pondremos X o XX al final de la dirección, para poder detectarlo de manera más sencilla. Por ejemplo, la dirección IP que termina con 137.1, en la cadena de caracteres se completará con *137.1XX*, al igual que, por ejemplo, la dirección IP que termina con 137.25, se completará con *137.25X*.

Seguidamente, enviaremos un campo de un solo carácter para definir el tipo de señal que vamos a utilizar, una modulación PWM o Seno.

A continuación, los cuatro campos que definirán las frecuencias del foco. El número de frecuencias que estamos utilizando en el algoritmo (un solo carácter) y las frecuencias a utilizar, todas ellas en Hercios (Hz). En principio con las frecuencias que se van a utilizar actualmente hubiera bastado con cuatro caracteres, pero se ponen 5 para ampliaciones futuras. En nuestro caso las frecuencias de 4 dígitos que colocaremos llevarán un 0 delante, por el mismo motivo que hemos explicado.

Por último, escribiremos el porcentaje de la señal PWM que solo tendremos en cuenta si la señal que estamos enviando es del propio tipo PWM. Tendrá dos caracteres representando el porcentaje del ciclo de trabajo.

4.2 COMUNICACIÓN CON LAS LUMINARIAS

TCP Server es una aplicación para Android, que nos permite realizar pruebas configurando conexiones TCP. Utilizando esta aplicación podemos crear un servidor TCP que se comunique con varios clientes. Esta aplicación detecta directamente la dirección IP a la que está asociado el smartphone, y solamente tendremos que asignar un puerto para crear el servidor.

Seguidamente, a través de MATLAB podemos ser el cliente que complete la conexión que necesitamos. Por tanto, el PC, que tiene instalado MATLAB con la base de datos, se comportará como cliente y el smartphone con la aplicación TCP Server hará de servidor, emulando en las primeras pruebas a una lámpara inteligente.

Tabla 9. *Función de MATLAB, tcpclient()*

```
t = tcpclient(address,port) %Función para abrir una conexión TCP
```

Una vez establecida la conexión TCP a través de la red Wi-Fi, es el momento de probar el envío de información, en concreto las cadenas de caracteres. Mostramos un fragmento del código que hemos utilizado en MATLAB para explicar el procedimiento del envío de los códigos.

Como se puede apreciar, realizamos un bucle con todas las direcciones IP posibles de las diferentes plantillas. Una para cada zona del museo. Seguidamente, mediante una condición, descartamos los casos que no queremos configurar a través de la variable *lanzar* y los casos en los que no exista identificador. Después, concatenamos los parámetros que necesitamos en la cadena de caracteres, abrimos la conexión TCP con el servidor, enviamos la cadena y finalmente cerramos la conexión.

Tabla 10. *Script de MATLAB. Lanzamiento de las cadenas de caracteres hacia los focos inteligentes.*

```
red = 137;

for ind=1:1728

      if lanzar(ind) == 1 && IP_matrix(ind,8) ~= 0

      if IP_matrix(ind, 1)<10
            cadena_a_enviar=red+"."+ IP(ind, 1) +"XX,"+PWM_Seno(ind,
            1)+","+Num_frecs(ind, 1)+","+Frec1(ind,
            1)+","+Frec2(ind, 1)+","+Frec3(ind,
            1)+","+PWM_percent(ind, 1);
            t=tcpclient("192.168."+red+"."+IP(ind,1),3333,"ConnectTi
            meout",10);
            writeline(t, cadena_a_enviar)
            clear t
      elseif IP_matrix(ind, 1)<100 && IP_matrix(ind, 1)>=10
            cadena_a_enviar=red+"."+ IP(ind, 1) +"X,"+PWM_Seno(ind,
            1)+","+Num_frecs(ind, 1)+","+Frec1(ind,
            1)+","+Frec2(ind, 1)+","+Frec3(ind,
            1)+","+PWM_percent(ind, 1);
            t=tcpclient("192.168."+red+"."+IP(ind,1),3333,"ConnectTi
            meout",10);
            writeline(t, cadena_a_enviar)
            clear t
      else
            cadena_a_enviar=red+"."+ IP(ind, 1) +","+PWM_Seno(ind,
            1)+","+Num_frecs(ind, 1)+","+Frec1(ind,
            1)+","+Frec2(ind, 1)+","+Frec3(ind,
            1)+","+PWM_percent(ind, 1);
            t=tcpclient("192.168."+red+"."+IP(ind,1),3333,"ConnectTi
            meout",10);
            writeline(t, cadena_a_enviar)
            clear t
      end
```

Para diferenciar las distintas zonas del museo hemos desarrollado una parte del código en la que a cada zona se le asigna un rango de direcciones concreto (256), a saber .137, .138, .139 y .140. De esta forma estará todo más ordenado a la hora de poder cambiar la configuración sin tener que desconfigurar el resto de los focos inteligentes.

Tabla 11. *Script de MATLAB. Diferenciación de las zonas del museo con las direcciones IP.*

```
switch IP_matrix(ind, 11)
            case 0 %Planta Baja
                switch IP_matrix(ind, 12)
                    case 0 %Zona Este
                        red = 137;
                    case 1 %Zona Oeste
                        red = 138;
                end
            case 1 %Planta Baja
                switch IP_matrix(ind, 12)
                    case 0 %Zona Este
                        red = 139;
                    case 1 %Zona Oeste
                        red = 140;
                end
        end
        end
end
```

En la Figura 18, podemos observar la pantalla principal de la aplicación con el servidor configurado. En la parte superior vemos la dirección IP 192.168.137.187 asociada al servidor y el número de puerto, en este caso 3333. A la derecha vemos un globo rojo con un "1". Es la notificación de que existe un cliente que ha abierto una sesión TCP con el servidor.

En el panel principal, vemos un mensaje con la siguiente cadena de caracteres:

137.3XX,0,2,04000,07000,00000,50

Este mensaje lo ha enviado el PC con fecha 7 de Julio de 2021 a las 18:54:28. Se trata de una cadena de caracteres del tipo que hemos explicado antes, con 7 campos. El primero se trata de la parte final de la dirección IP que corresponde al dispositivo, 137.3. El segundo campo indica que queremos una señal del tipo PWM. A continuación, describimos que necesitamos dos frecuencias, y el valor de ellas, en este caso 4000Hz y 7000Hz. También tenemos el campo preparado para tres frecuencias. Por último, el valor del porcentaje de la señal PWM, que en este ejemplo tiene el valor del 50%.

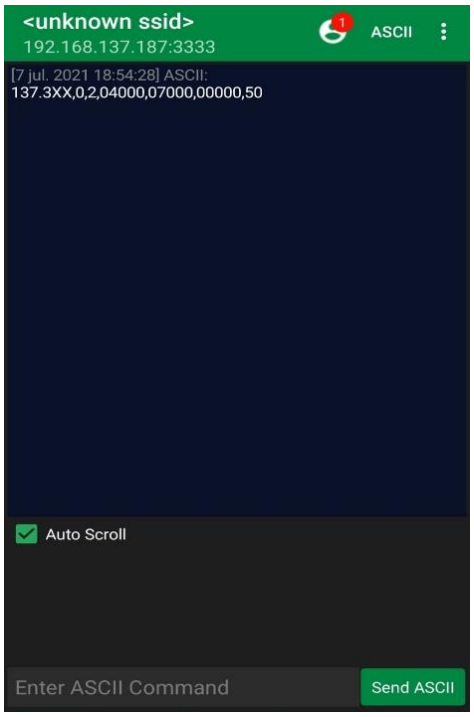

Figura 18. *Pantalla principal de la aplicación TCP Server.*

5. INTERFAZ GRÁFICA

A la hora de ejecutar todo el proceso, resulta muy engorroso tener que utilizar MATLAB para ejecutar todos los apartados por separado. Por ello, hemos decido implementar una pequeña interfaz gráfica que nos ayude a automatizar procesos con la herramienta App Designer que incorpora el propio MATLAB. App Designer es un entorno de desarrollo interactivo para diseñar una aplicación y programar su comportamiento. Contiene unos procedimientos interactivos para colocar a gusto del usuario los elementos visuales. Por detrás lleva un código de MATLAB que se ejecuta cuando iniciamos la aplicación que hemos realizado. Cada elemento tiene una función *callback* asociada para que nuestra aplicación funcione como nosotros queremos. Solamente tendremos que escribir lo que queremos hacer con cada uno de los elementos.

La interfaz gráfica que vamos a diseñar contendrá, inicialmente, los siguientes elementos interactivos:

- ***Un botón para ejecutar el algoritmo de distribución.*** Será muy sencillo hacer click, encadenar un bucle para completar el algoritmo y mostrarlo en una tabla de manera ordenada.
- ***Una caja de texto.*** Nos será útil para saber en qué situación se encuentra el algoritmo una vez lanzado el algoritmo de distribución. Por ejemplo, antes de realizar cualquier proceso nos informará de que debemos tocar algún botón disponible para realizar alguna acción. Además, mientras se ejecuta el algoritmo de distribución obtendremos información del estado en el que se encuentra.
- ***Una tabla con toda la información de todos los focos.*** Será una tabla editable, donde podremos cambiar a nuestro gusto todos los campos del algoritmo desarrollado automáticamente. En la tabla, una de las columnas contendrá checkboxes. Un checkbox o casilla de verificación es un pequeño cuadro al lado de la opción a seleccionar que permite a este hacer selecciones múltiples de un conjunto de opciones. Estos nos servirán para poder elegir las filas que contienen la información necesaria y enviarlas hacia las lámparas inteligentes para configurarlas.
- ***Un checkbox para marcar de manera automática todas las filas de la tabla.*** Así nos ahorraremos el procedimiento de ir marcando una a una las filas con los datos que queremos enviar hacia las lámparas.
- ***Un botón para guardar.*** Necesario para guardar la base de datos en un archivo .xls de Microsoft Excel.
- ***Un botón para lanzar los códigos hacia las lámparas inteligentes.*** Una vez tenemos nuestra tabla guardada podremos ejecutar el script que nos permita configurar todas las lámparas deseadas en un solo click.
- ***Un botón para escoger los archivos .png deseados.*** Con una sencilla función de MATLAB, podremos abrir el explorador de archivos y guardar en la variable correspondiente el archivo .png que necesitamos, junto con el tamaño de celda deseado en el cuadro numérico *Grid [dcm]*.

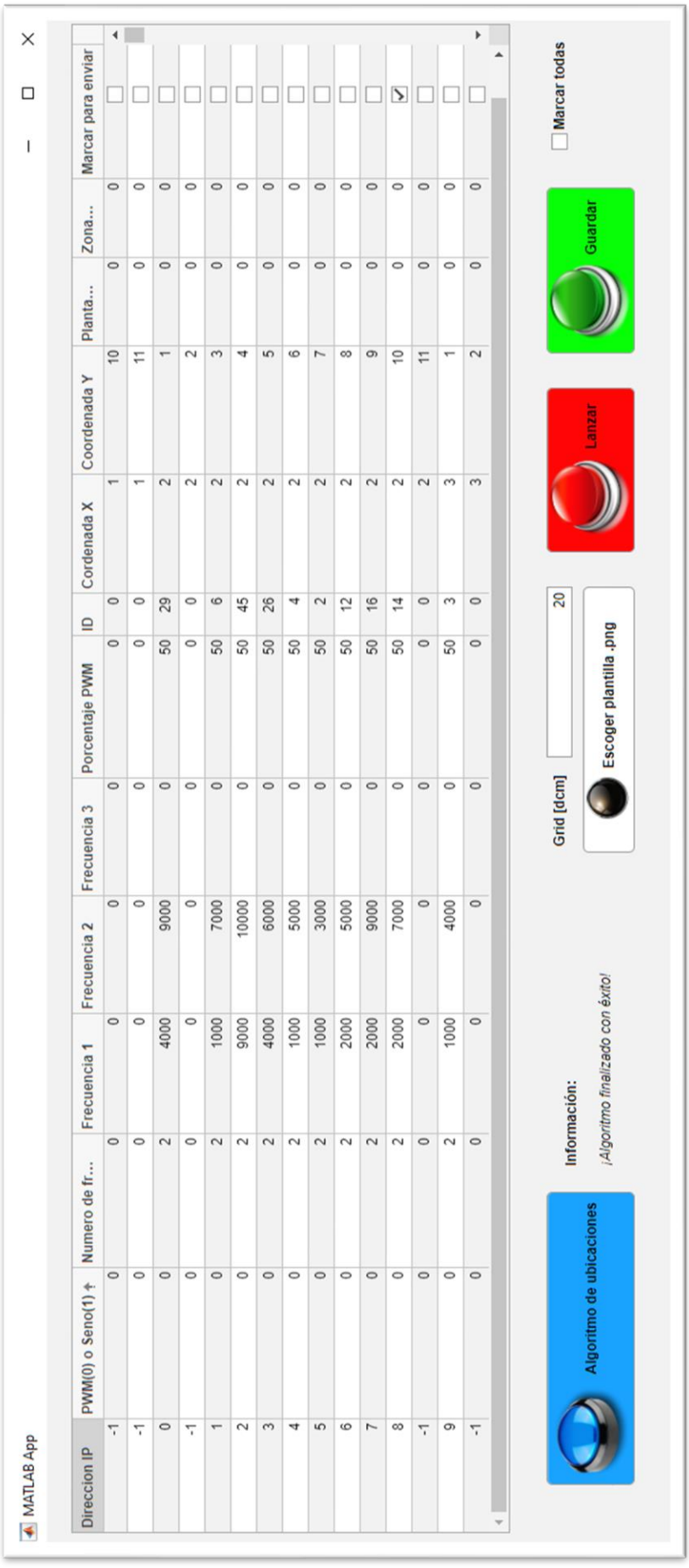

Figura 19. *Interfaz gráfica.*

Podremos ir añadiendo más elementos a la interfaz gráfica según los vayamos necesitando en un futuro. Por el momento se trata de que sea fácil y muy intuitiva de utilizar para que cualquiera que no conozca el funcionamiento interno del desarrollo sea capaz de configurar los focos a su libre elección, o de forma automática.

Al igual que hemos hecho con anterioridad, vamos a explicar cómo se ejecuta una función *callback*. De esta forma quedará más claro el por qué necesitamos una interfaz gráfica en este proyecto.

Este fragmento de código utilizado en MATLAB nos muestra la lógica de decisión del checkbox para marcar de manera automática todas las filas de la tabla. Tras haber hecho click en el checkbox, automáticamente saltamos a la función y se ejecuta. En primer lugar, leemos la matriz obtenida, y guardada en el documento Excel, en la variable *tablon*.

Tabla 12. *Script de MATLAB. Funcion callback del checkbox para marcar todas las casillas.*

```
function ckeckbox_MarcarTodasValueChanged(app, event)

        value = app.ckeckbox_MarcarTodas.Value;
        tablon = readtable('IPs_BBDD.xls');

        if value==1
                for indx=1:1728
                        if tablon.ID(indx)~=0
                                tablon.LANZAMIENTO(indx) = 1;
                                app.tabla.Data = tablon;
                        end
                end
        else
                for indx=1:1728
                        if tablon.ID(indx)~=0
                                tablon.LANZAMIENTO(indx) = 0;
                                app.tabla.Data = tablon;
                        end
                end
        end
end
```

A continuación, debemos diferenciar si hemos marcado o desmarcado la casilla de checkbox. En caso de haberla marcado, todas las filas se marcarán (1). En caso contrario, se desmarcarán (0). Actualizamos la tabla que se muestra en la interfaz gráfica y se termina la ejecución de la función *callback*.

De esta forma, ya estaríamos listos para enviar todos los códigos hacia sus respectivas direcciones IP y, por tanto, poder configurar los focos inteligentes de manera rápida y eficiente.

Por otro lado, tras pulsar el botón para seleccionar la plantilla *png* y llamar a la función *callback asociada* para escoger los archivos, utilizamos la siguiente función de MATLAB:

Tabla 13. *Función de MATLAB, uigetfile().*

```
[filename, pathname] = uigetfile('*.png', 'Abre una imagen .png');
```

En la variable *filename* guardamos el nombre del archivo con la extensión .png que elegimos a través de la ventana que se nos abre a continuación. Por otra parte, en la variable *pathname* se guarda una cadena de caracteres que aparece en la ventana al abrir el explorador de archivos *("Abre una imagen .png").*

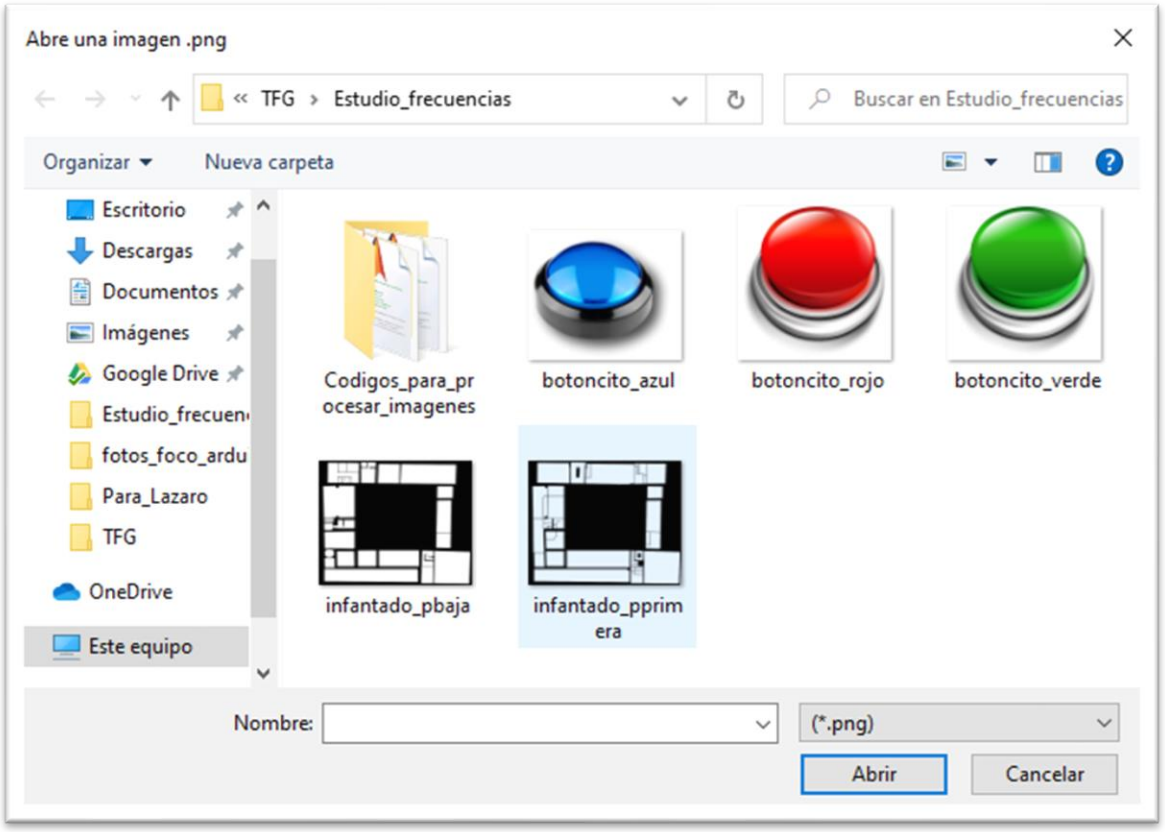

Figura 20. *Ventana emergente para escoger archivo .png.*

Acto seguido, podremos lanzar el algoritmo de distribución a través del botón azul en la interfaz gráfica y continuar con el proceso de envío de códigos hacia las lámparas inteligentes.

6. RESULTADOS

En este capítulo vamos a explicar los resultados obtenidos para el caso concreto del Museo de Guadalajara. Explicaremos cómo se desarrolla el algoritmo en este escenario tras conseguir los archivos necesarios en InkScape.

En primer lugar, para que podamos lanzar el algoritmo de ubicaciones explicado en el capítulo 4 necesitamos una matriz objetivo. En este caso serán unas plantillas del Museo de Guadalajara, ya que tenemos dos plantas, una por cada planta del edificio. El siguiente paso era conseguir el plano del edificio en cuestión. El propio museo nos proporcionó unos planos con medidas de dimensiones concretas, y se utilizó Google Maps para realizar mediciones en el espacio real.

A continuación, entramos en InkScape. Este software nos permitió elaborar nuestra plantilla en base a los planos. De manera manual fuimos descartando las zonas no iluminadas insertando encima zonas coloreándolas de negro, dejando visibles las zonas que sí llevan iluminación artificial con códigos. Existen varios métodos para colorear, pero el más útil fue elegir los vértices de las salas, pasillos y escaleras que no íbamos a utilizar y colorear el interior de las formas obtenidas a través de las aristas y vértices escogidos.

El siguiente paso fue decidir las dimensiones de la imagen para exportarla a MATLAB en formato .png, para la consecución de un mapa de bits. Debido a las dimensiones reales del edificio y a que tendríamos una lámpara inteligente cada 2 metros (esto debe ser un parámetro para introducir en MATLAB para que establezca el tamaño de las celdas), las dimensiones en pixeles elegidas fueron 542x669, que a su vez son, aproximadamente, las medidas del edificio en decímetros [dm]. Esta decisión fue tomada teniendo en cuenta las dimensiones de superficie reales del museo.

***Figura 21**. Archivo .png de la Planta Baja dividida por zonas iluminadas y no iluminadas.*

La matriz que exportamos a MATLAB contiene valores entre 0 y 255, equivalentes a la gama de colores puesto que existen zonas donde hay grises que no podemos apreciar. Puede dar la sensación de que es todo negro o blanco tanto en la Figura 21 como en la Figura 22, pero en realidad, al pasar al formato .png, las transiciones de blanco a negro contienen grises en su interior.

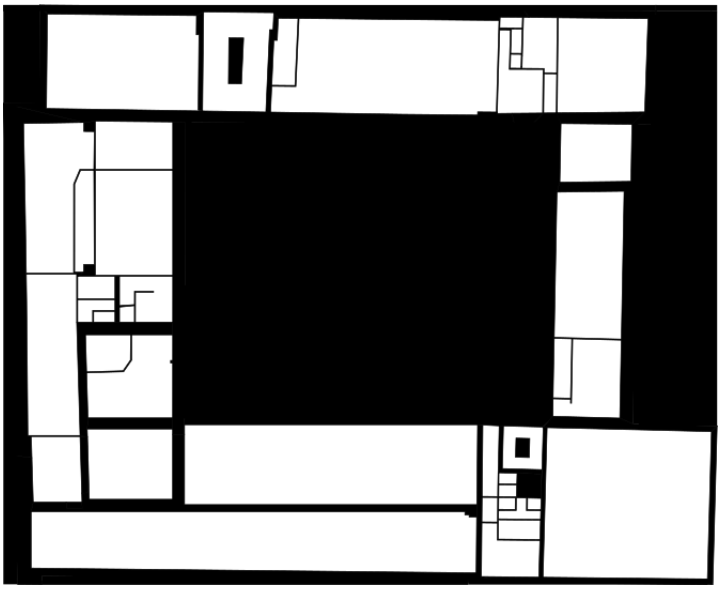

Figura 22. *Archivo .png de la Planta Primera dividida por zonas iluminadas y no iluminadas.*

Después de la consecución de la plantilla y de la matriz de pixeles, el mapa seria acotado, a través de los scripts de MATLAB explicados con anterioridad, para colocar nuestras frecuencias en una matriz de 27x33 (que es a su vez dependiente de la distancia que deseemos entre luminarias, y que la calculará Matlab). El objetivo de dicho script es reducir el archivo .png con una consigna clara: ir recorriendo la matriz en sub matrices de 20x20y tras decidir un determinado porcentaje de blanco mínimo escoger el nuevo tipo de celda, 0 o -1 (blanco o negro). Decidimos esta reducción porque, como anteriormente se ha explicado, en esta primera tentativa tendremos un foco inteligente cada 2 metros (20x20 dm^2). Por tanto, como nuestra matriz y plantilla está en decímetros, conseguiremos tener una plantilla en la que cada celda ya es definitiva, y tendremos que colocar una de las dos opciones de tipo de celda. Así conseguiremos una matriz, por ejemplo, la Figura 23 o la Figura 24, más reducida, más compacta y muy fiable para aplicar el algoritmo principal de relleno de cuadricula.

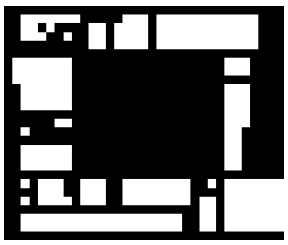

Figura 23. *Matriz objetivo definitiva de la Planta Baja.*

La razón por la cual desarrollamos este método para conseguir una matriz más pequeña es porque la matriz origen tiene por ejemplo muros muy estrechos que podemos obviar, incluso salas subdivididas que no merece la pena dividir claramente. O, por el contrario, muros muy anchos y zonas como el patio que son tan extensas en las que no merece la pena tener tantos posibles lugares para asignar pares de frecuencias.

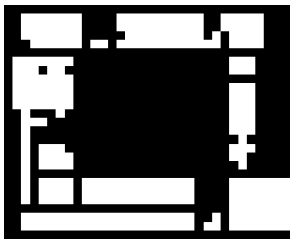

Figura 24. *Matriz objetivo definitiva de la Planta Primera.*

Una vez obtenidas las matrices objetivo tanto de la primera planta como de la planta baja del museo, es hora de decidir cuál es el número de frecuencias que vamos a poder utilizar en cada foco. Si utilizamos tres frecuencias nuestras posibilidades aumentan considerablemente y no sería necesario dividir el museo en zonas. Por el contrario, utilizando dos frecuencias, que es nuestro propósito original, no conseguiríamos rellenar por completo la matriz objetivo de una tirada. Luego, la mejor decisión es dividir en dos zonas cada planta, Este y Oeste, y ayudarnos de la tecnología GPS para un posicionamiento inicial *groso modo*.

Aun así, vamos a estudiar y explicar ambos casos, tanto con ternas como con duplas de frecuencias. Empezaremos por el óptimo, teniendo 3 frecuencias posibles en un mismo foco. Primeramente, podemos observar en las Figuras 25 y 26 que la plantilla se ha rellenado completamente para las plantas baja y primera, y en segundo lugar vemos que tenemos identificadores del 1 al 120.

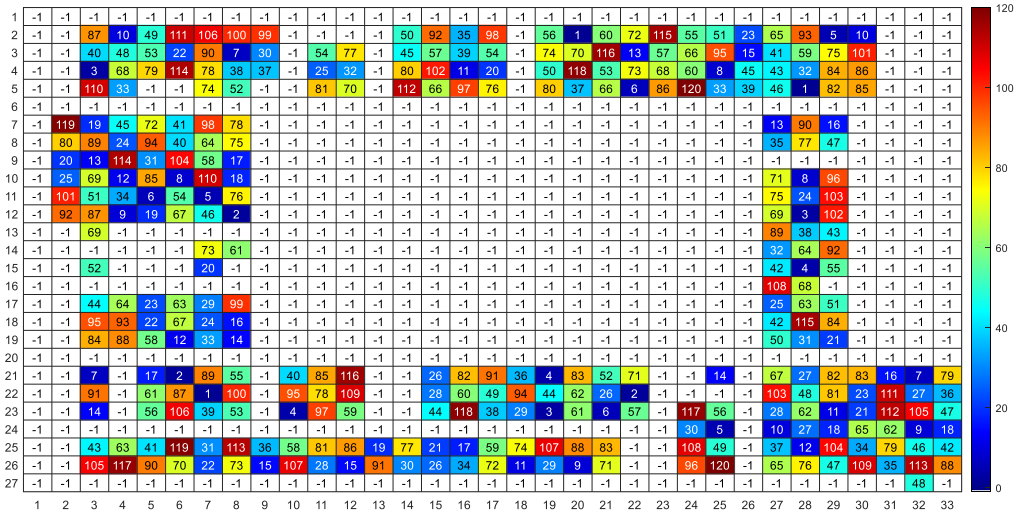

Figura 25. *Planta Baja con 120 posibles ternas de frecuencias.*

A modo de ejemplo, explicaremos el caso concreto de un identificador de la planta primera y cómo se desarrolla el algoritmo. En la Figura 26, en la zona superior izquierda (fila 2 columna 4), se coloca el identificador número 45, que tiene por vecinos los siguientes ocho identificadores: 113, 103, 64, 90, 68, 117, 94 y 11. Este identificador también aparece un poco más abajo (fila 10 columna 4), en el borde de la zona izquierda, esta vez con los siguientes cinco vecinos: 55, 108, 32, 62, 23. Hemos reutilizado el mismo identificador porque los vecinos de ambos son totalmente diferentes. Y es posible que existan más a lo largo y ancho de la plantilla. No necesariamente ha de tener 8 vecinos, es decir, puede tener un muro cerca, una sala no iluminada marcada con -1 o un límite del plano.

En consecuencia, el plano se va rellenando de manera aleatoria pero ordenadamente, es decir, los identificadores se van escribiendo en lugares aleatorios de la plantilla, pero siempre desde el identificador 1 hasta el 120. Si en algún momento uno de los identificadores no se puede escribir, saltamos al siguiente.

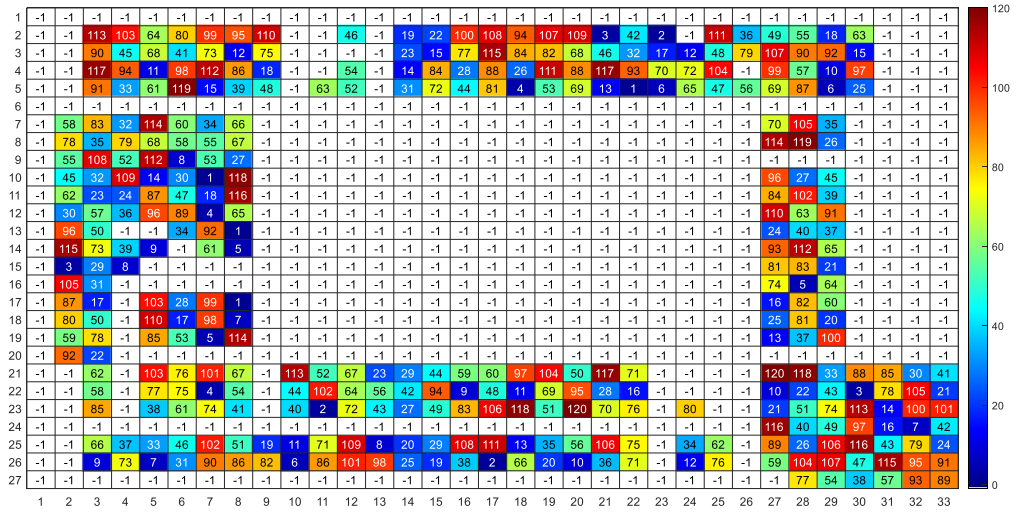

Figura 26. *Planta Primera con 120 posibles ternas de frecuencias.*

La lógica es clara y concisa, en ningún momento nos vamos a encontrar el mismo identificador en dos zonas distintas con otro identificador vecino coincidente.

A continuación, procedemos a explicar el caso más viable pero menos fácil, ya que escogiendo duplas de frecuencias en lugar de ternas las posibilidades bajan de manera considerable. Es por ello que debemos dividir la plantilla en dos zonas. En esta ocasión, tanto la Figura 27 como la Figura 28 muestran las zonas Oeste y Este de la Planta Baja, respectivamente. Se ha desarrollado el algoritmo para el caso de dos frecuencias y se han rellenado perfectamente cada uno por separado, esta vez con dos matrices de 27x16, para que ambas tengan las mismas dimensiones. Como hemos explicado anteriormente, con ayuda del GPS no tendremos problemas a la hora de ubicarnos inicialmente en una de las dos partes del plano. Nótese que una vez ubicado en un punto, salvo que se pierda la ubicación, ya se tendría información para no necesitar el uso orientativo de GPS

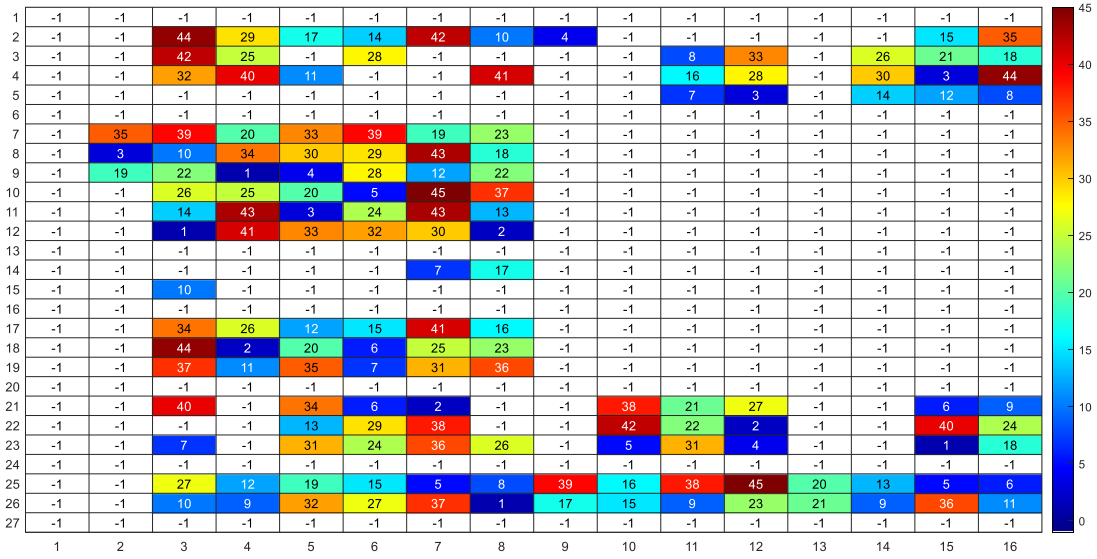

Figura 27. *Zona Oeste de la Planta Baja con 45 posibles duplas de frecuencias.*

Al igual que en el caso con tres frecuencias, el algoritmo funciona de la misma manera. Simplemente tendremos menos identificadores disponibles y por tanto nuestra matriz debe ser más pequeña para poder rellenarla con garantías.

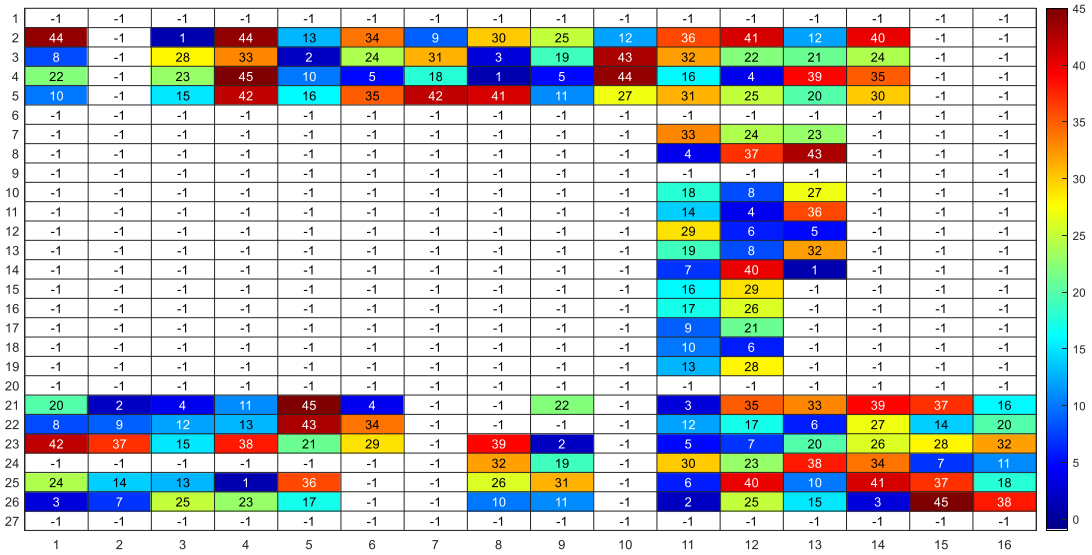

Figura 28. *Zona Este de la Planta Baja con 45 posibles duplas de frecuencias.*

En cuanto al porcentaje de blanco, es una variable que nos ayuda a perfeccionar la matriz definitiva. Nos ayuda a decidir si las casillas deben ser desde el inicio del algoritmo 0 o -1. Si esas matrices 20x20 que vamos recorriendo contienen un 90% de blanco o más (recuérdese que está en escala de grises) las propondremos para que sean 0, es decir, zonas donde habrá focos inteligentes. En caso contrario, serán casillas que irán marcadas con -1. Debemos asegurarnos de que la plantilla sea lo más real posible respecto al edificio.

Finalmente, con el objetivo de que sea fácil de aplicar una vez ha sido desarrollado todo el grueso del algoritmo, realizaremos una función para que realice hasta que genere una solución buena y fiable. No será más que un bucle de repetición hasta que encuentre una solución posible. Comenzaremos aplicándolo a una de las zonas de la Planta Baja hasta terminar por la última de la Primera Planta. De esta manera conseguiremos desarrollar la base de datos que necesitamos.

7. CONCLUSIONES Y TRABAJOS FUTUROS

Dado que hemos realizado un arduo trabajo, tanto con la fase del estudio teórico al inicio de este proyecto, como con las distintas fases del desarrollo de este, daremos por finalizadas las fases de desarrollo y resultados, y pasaremos a la fase de reportes. ¿Qué conclusiones se pueden extraer del estudio realizado? ¿Qué cosas se han hecho bien y cuáles se podrían mejorar en el futuro? ¿Qué líneas futuras se podrían seguir para enriquecer este estudio y darle más profundidad?

7.1 CONCLUSIONES

En el apartado teórico, se ha estudiado en profundidad la viabilidad de la idea principal del proyecto. Detectar las frecuencias que emite un foco con la cámara de un dispositivo smartphone no ha sido una tarea fácil. Es por ello por lo que damos un gran valor a este apartado y damos por buenos los resultados experimentales obtenidos. La utilización de MATLAB ha sido un acierto, porque que es una herramienta muy versátil con la que podemos realizar muchos cálculos y estudios exhaustivos de casi cualquier cosa que nos imaginemos. En cuanto a los resultados obtenidos en el banco de pruebas han sido como esperábamos, la mayoría de ellos han sido satisfactorios, salvo unos pocos pares de frecuencias. Además, hemos añadido gráficas explicativas de lo que estaba sucediendo a nivel frecuencial y de potencia lumínica, que nos han ayudado a comprender mejor todo este apartado tan importante del proyecto.

En lo que respecta al desarrollo del algoritmo de distribución de ubicaciones fue complejo llegar a depurarlo y que funcionase como necesitábamos. Sin la ayuda de algunos profesores del departamento de Electrónica no habría sido posible. Es un algoritmo eficiente, rápido y de bajo coste computacional. Simplemente necesitamos comparar la lámpara inteligente actual con una de las lámparas vecinas, y no habrá posibilidad de equivocación, pues el algoritmo funciona a la perfección. Es cierto que tiene limitaciones, no es válida cualquier plantilla de cualquier dimensión, porque dependemos del número de identificadores posibles del que dispongamos. Pero al haber encontrado una solución alternativa, dividiendo las plantillas y orientándonos *grosso modo* con GPS, permite que esto sea posible incluso con plantillas más extensas.

El desarrollo de la base de datos fue más sencillo, ya que solamente debíamos guardar de manera ordenada los datos que nos proporciona el algoritmo de ubicaciones. Usar Microsoft Excel fue un acierto, ya que se permite con facilidad realizar lecturas y escrituras directas desde MATLAB hacia un programa ya conocido por todos globalmente.

El envío de datos vía Wi-Fi con una conexión basada en el protocolo TCP/IP fue una idea conjunta entre el otro trabajo que acompaña a nuestro proyecto y este. Como se requería una estructura unívoca y específica para configurar las lámparas inteligentes, tuvimos que ponernos de acuerdo, establecer las dimensiones de las cadenas de caracteres que necesitábamos enviar y decidir qué parámetros debíamos proporcionar.

El resultado es óptimo, ya que no se pierden datos tras la utilización de TCP, una de las características más importantes de este protocolo. Además, unificando la lógica de envío y recepción todo es más sencillo a la hora de compenetrarse y estructurar los sistemas.

Finalmente, respecto a la realización de la interfaz gráfica, no estaba claro cómo la íbamos a desarrollar. Debíamos estructurar un panel con lo básico para que cualquier persona que no entienda sobre el mecanismo interno del código, supiera lanzar el algoritmo de ubicaciones y poder configurar las lámparas de manera fácil, rápida y sin problemas. Pensamos que, ya que estábamos usando MATLAB como herramienta principal, lo más lógico era seguir utilizándolo. Y desde luego, ha sido una decisión acertada, debido a compatibilidad y unificación de los scripts que ya habíamos desarrollado con anterioridad.

7.2 TRABAJOS FUTUROS

Si bien este estudio ha orbitado alrededor de la gestión telemática de lámparas inteligentes al final hay un gran pilar que sustenta todo el estudio: el desarrollo del algoritmo de ubicaciones.

En cuanto a la mejora general del proyecto, va encaminada a la posibilidad de tener más de dos frecuencias en un mismo foco, porque ampliaría muchísimo las posibilidades de ejecutar el algoritmo en entornos más extensos y evitar la utilización de GPS dividiendo las plantillas. Con 10 frecuencias en total, podríamos generar hasta 5 en un mismo foco y lograríamos obtener hasta 252 identificadores distintos, siguiendo la fórmula de combinaciones sin repetición de m elementos tomados de n en n:

$$N_{combinaciones} = \frac{10!}{5! \cdot (10-5)!} = 252$$

Sin embargo, si pasamos de 5 no podremos aumentar el número de identificadores, pues estaríamos disminuyendo este número de posibilidades. No obstante, se considera que pasar de 3 será muy difícil por la resolución temporal disponible para la detección de las frecuencias a partir del rolling shutter de la cámara

En cuanto a la base de datos, la idea principal de trabajo futuro es tenerla en un repositorio online. El objetivo es que el usuario que accede al museo pueda acceder a esa base de datos a través de la conexión Wi-Fi que se le proporciona al entrar (en realidad el acceso no lo realiza el usuario, si no, automáticamente, la App instalada en su móvil). De esta forma se podrán comparar automáticamente los focos desde el smartphone y poder posicionarse en el mapa generado en la aplicación.

Finalmente, podremos mejorar los scripts de MATLAB, depurándolos hasta convertirlos en más rápidos y eficientes, así como la interfaz gráfica a la que podremos añadir tantos elementos como requiramos y sean de gran ayuda y utilidad.

8. BIBLIOGRAFÍA

[1] L. Yicheny, Z. Chenqian, L. Bo y M. Teng, "Intelligent Frequency Assignment Algorithm Based on Hybrid Genetic Algorithm", en *International Conference of Computer Vision, Image and Deep Learning*, 2020 [En línea]. Disponible en: https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=9270430&tag=1.

[2] Kin. K. Leung y Byoung-Jo "J" Kim, "Frequency Assignment for IEEE 802.11 Wireless Networks", [En línea]. Disponible en: https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=1285259

[3] Yusuf Said Eroglu *et al*, "Multi-Element Transmitter Design and Performance Evaluation for Visible Light Communication", [En línea]. Disponible en: https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=7414124&tag=1

[4] J. C Valencia-Estrada, B. Béchadergue y J. Gracía-Márquez, "Full Field Radiant Flux Distribution of Multiple Tilted Flat Lambertian Light Sources", [En línea]. Disponible en: https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=9139466

[5] K. Saxena, R. Ra y A. Dixit, "A Novel Optimization Approach for Transmitter Semi-Angle and Multiple Transmitter Configurations in Indoor Visible Light Communication Links", [En línea]. Disponible en: https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=8493666

[6] D. Wu *et al*, "Power Distribution and Q-factor Analysis of Diffuse Cellular Indoor Visible Light Communication Systems", [En línea]. Disponible en: https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=5996230

[7] K. Cui *et al*, "Line-of-sight Visible Light Communication System Design and Demonstration", [En línea]. Disponible en: https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=5580360

[8] D. Deqiang, K. Xizheng y X. Linpeng, "An Optimal Lights Layout Scheme for Visible-Light Communication System", [En línea]. Disponible en: https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=4350650

[9] B. Ghimire, J. Seitz y C. Mutschler, "Indoor Positioning using OFDM-based Visible Light Communication System", [En línea]. Disponible en: https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=8533834

[10] D. Nguyem, D. Le, T. Tran y V. Bao, "A Prototype of FPGA-based Centralized Multiple Transmitters for Visible Light Communications", [En línea]. Disponible en: https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=9023844

[11] E. Aparicio, A. Hernández y J. Ureña, "Visible Light Positioning System Based on a Quadrant Photodiode and Encoding Techniques", [En línea]. Disponible en: https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=8946554

[12] S. Cincotta, C. He, A. Neild y J. Armstrong, "QADA-PLUS: a novel two-stage receiver for visible light positioning", [En línea]. Disponible en: https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=8533733

[13] «Combinatoria - Wikipedia» [En línea]. Disponible en: https://es.wikipedia.org/wiki/Combinatoria#Combinaciones

[14] «Estadística para todos» [En línea]. Disponible en: https://www.estadisticaparatodos.es/software/misjavascript/javascript_combinatorio2.html

[15] «Source Education Group Franhofer – Visual Light Communication» [En línea]. Disponible en: https://www.youtube.com/watch?v=C_Vk04ShvDY&ab_channel=fug

[16] T. Komine y M. Nakagawa, «Fundamental Analysis for Visible-Light Communication System using LED Lights» [En línea]. Disponible en: https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=1277847

[17] «Rolling shutter - Wikipedia» [En línea]. Disponible en: https://es.wikipedia.org/wiki/Rolling_shutter

[18] «Protocolo de control de transmisión - Wikipedia» [En línea]. Disponible en: https://es.wikipedia.org/wiki/Protocolo_de_control_de_transmisi%C3%B3n

[19] «¿Qué es una dirección IP? ¿Cómo puedo saber mi IP? » [En línea]. Disponible en: https://raiolanetworks.es/blog/que-es-una-direccion-ip/

[20] «TCP Server – Google Play » [En línea]. Disponible en: https://play.google.com/store/apps/details?id=com.mightyIT.gops.tcpserver&hl=es_NI

[21] «InkScape» [En línea]. Disponible en: https://inkscape.org/es/

[22] «Measuring rolling shutter with a strobing LED» https://joancharmant.com/blog/measuring-rolling-shutter-with-a-strobing-led/

Universidad de Alcalá

Escuela Politécnica Superior

ESCUELA POLITECNICA SUPERIOR