UNIVERSITAT POLITÈCNICA DE VALÈNCIA

UNIVERSITAT POLITÈCNICA DE VALÈNCIA
CAMPUS D'ALCOI

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escuela Politécnica Superior de Alcoy

Estudio, propuesta y prototipado de solución para la geolocalización y orientación en interiores para personas con discapacidad

Trabajo Fin de Grado

Grado en Ingeniería Informática

AUTOR/A: Jorda Juan, Joan

Tutor/a: Pérez Llorens, Rubén

CURSO ACADÉMICO: 2022/2023

Estudio, propuesta y prototipado de solución para la geolocalización y orientación en interiores para personas con discapacidad

El objetivo de este trabajo consistirá en tras realizar el pertinente estudio, proponer un prototipo de dispositivo o conjunto de dispositivos que sea capaz de orientar a una persona en espacios de interior. El trabajo se realizará juntamente con Daniel Lopez, estudiante del Grado en Ingeniería en Diseño Industrial y Desarrollo de Productos.

Estudi, proposta i prototipat de solució per a la geolocalització i orientació en interiors per a persones amb discapacitat

L'objectiu d'aquest treball consistirà en després de realitzar el pertinent estudi, proposar un prototip de dispositiu o conjunt de dispositius que siga capaç d'orientar a una persona en espais d'interior. El treball es realitzarà conjuntament amb Daniel Lopez, estudiant del Grau en Enginyeria en Disseny Industrial i Desenvolupament de Productes.

Study, solution proposal and prototyping of a solution for geolocation and orientation indoors for people with disabilities

The objective of this work will consist of, after carrying out the pertinent study, proposing a prototype device or set of devices that will be capable of guiding a person in interior spaces. The work will be carried out jointly with Daniel Lopez, student of the Degree in Engineering in Industrial Design and Product Development.

Keywords: Indoor geolocation, solution proposal, Guidance

Agradecimientos

En primer lugar, me gustaría agradecer a mi tutor, Rubén Perez Llorens, por haber tutorizado este proyecto y haberme guiado en su elaboración hasta el final.

Gracias a l'Universitat Politècnica de València y muy especialmente a mi Campus de Alcoy por los conocimientos y la guía adquiridos durante el curso de la titulación. Gracias, como no puede ser de otra forma al conjunto de la Comunidad Universitaria, PDI, PAS, compañeros estudiantes y personal externo. Todos en mayor o menor medida han sido clave para la superación de mis estudios. Ha sido un placer y un honor cursar mis estudios y mi formación aquí.

Quiero, además, reconocer la obra de aquellos que han publicado sus trabajos y los han hecho abiertos al público. Gracias a ello me ha sido posible tumbar las barreras que he tenido en la realización del proyecto y llevarlo a buen puerto.

Me gustaría también dedicar unas líneas al Colegio San Roque de Alcoy, donde empecé a cursar el Ciclo Formativo de Grado Medio de Sistemas Microinformáticos y Redes. Fue aquí donde empecé el camino que me ha llevado hasta aquí y quiero agradecerles el buen trabajo que hicieron en inculcarme la pasión por estos estudios y esta profesión.

Por último, pero no por ello menos importante, quiero agradecer a mi familia, amigos y compañeros de trabajo que han supuesto un apoyo moral muy importante y que su ayuda a sido clave para superar los obstáculos psicológicos y tecnológicos de un proyecto como este. Muy especialmente a mis padres, abuelos, a mi tío Pepe, *de la llibreria* y a Aranza.

Abuelo, siempre me llamaste inventor, espero que estés donde estés veas este invento.

"Scientia potentia este, sed parva; quia scientia egregia rara este, nec proinde apparens nisi paucissimis, te in paucis rebus. Scientiae enim and natura este, ut esse intelligi non possit, nisi ab illis quién sunt scientia praediti". (Thomas Hobbes, De Homine)

Tabla de contenido

Introducción

El presente trabajo pretende proponer una propuesta de solución para contribuir a la accesibilidad y a la integración de las personas con discapacidad visual en los espacios de interior.

Hoy en día aplicaciones como Google Maps ya permiten localizar la ubicación mediante el uso de tecnologías de visión por computador en espacios de exterior. En algún momento, si se ha hecho uso de dicha aplicación se habrá pedido al usuario que use la cámara de su dispositivo para poder mejorar la orientación del dispositivo.

Este trabajo, de forma similar pretende diseñar un prototipo que permita localizar al usuario mediante el uso de un dispositivo *wearable* y guiarlo hasta su destino.

Planificación

El desarrollo del presente trabajo se ha desarrollado en tres etapas bien definidas, una primera etapa de investigación y planteamiento de soluciones con Daniel López para la delimitación de las especificaciones y los requerimientos tanto en lo que sería el diseño del dispositivo final como las limitaciones para el presente prototipo. Una segunda etapa de desarrollo del prototipo y una tercera parte de testeo, verificación de funcionamiento y documentación.

Aunque en un primer momento me propuse tratar de realizar la defensa en diciembre de 2022, la complejidad del tema en cuestión, mis responsabilidades y otras circunstancias me hicieron replantearme esta fecha. Finalmente, y tras consultarlo con mi tutor, me propuse realizar la defensa a partir de marzo de 2023. Aunque no se ha seguido unas fechas estrictas para cada etapa, si se han tenido ciertas fechas como referencia para un correcto desarrollo del trabajo y una correcta distribución de carga dada mis responsabilidades laborales durante el desarrollo de este.

La fecha de inicio del proyecto fue el 1 de septiembre de 2022 y la primera propuesta de solución fue realizada el 3 de septiembre de 2022 tras una reunión mantenida con Daniel López donde definimos los objetivos del prototipo. Esta primera propuesta corresponde al restante apartado de la introducción, con algunas modificaciones y adiciones según se ha ido avanzando sobre el proyecto y se han descartado o se han incorporado ideas o planteamientos al trabajo. En este primer borrador decidí acertadamente hacer uso de unas placas ESP32, aunque como se desarrolla con posterioridad en el documento las placas ESP32-Cam que propuse en su momento no era la mejor opción. Me propuse hasta noviembre para la investigación y búsqueda de estudios relacionados con la temática para de esta forma aprovechar el periodo de navidades y poder trabajar de forma intensa en la implementación de la solución para llegar a enero con el prototipo terminado o al menos con la mayor de funcionalidades implementadas y dedicar el tiempo restante a terminar la documentación y realizar las pruebas finales.

Considero que esta planificación, junto al hecho de haber estado trabajando la documentación al mismo tiempo que desarrollaba el proyecto, ha ayudado a tener los objetivos en mente y una línea clara de actuación que me ha permitido afrontar mejor el trabajo.

Aproximaciones

Según han avanzado los años las soluciones respecto a la geolocalización han variado. Uno de los principales problemas con respecto a la geolocalización en el interior de edificios mediante

el uso de elementos no estáticos, es la homogeneidad de los espacios. Siendo difícil diferenciar, por ejemplo, una planta de otra.

Existen varias agrupaciones de soluciones para la solución de este problema. Siguiendo la clasificación de (Yassin, 2017), encontramos las siguientes aproximaciones de acuerdo con la posición del usuario:

1. Triangulation
    a. Lateration
        i. TOA: Time of Arrival
        ii. TDOA: Time Difference of Arrival
        iii. RSS based Fingerprint
        iv. RTOF: Rount-trip Time of Flight
    b. Angulation
        i. AOA Angle of Arrival
        ii. AOD Angle of Depart
2. Scene Analysis
    a. Fingerprinting-based
3. Proximity Detection:
    a. Cell-ID
    b. RFID

Además, con la entrada de los algoritmos de Visión por computador e Inteligencia Artificial, nuevas metodologías se han ido desarrollando. Se acuerdo con la clasificación realizada por (Alkhawaja, 2019), encontramos:

1. Local Infraestructure Dependent Techiques
    a. Ultra-Wideband (UWB)
    b. Wireless Beacons
2. Local Infrastructure Independent Techniques
    a. Ultrasound
    b. Assisted-GNSS (AwGNSS)
    c. Magnetic Localization
    d. Infrared Localization
3. Visual/Depth sensors
    a. Structured Light Technology
    b. Pulsed Light Technology
    c. Stereo Camera

En otro artículo realizado por (Mendoza-Silva GM, 2019), se mencionan las tecnologías anteriormente mencionadas y se hace hincapié en la diferenciación de soluciones basadas en dispositivos y soluciones libres. Cabe destacar la mención a la Simultaneous localization and mapping (SLAM).

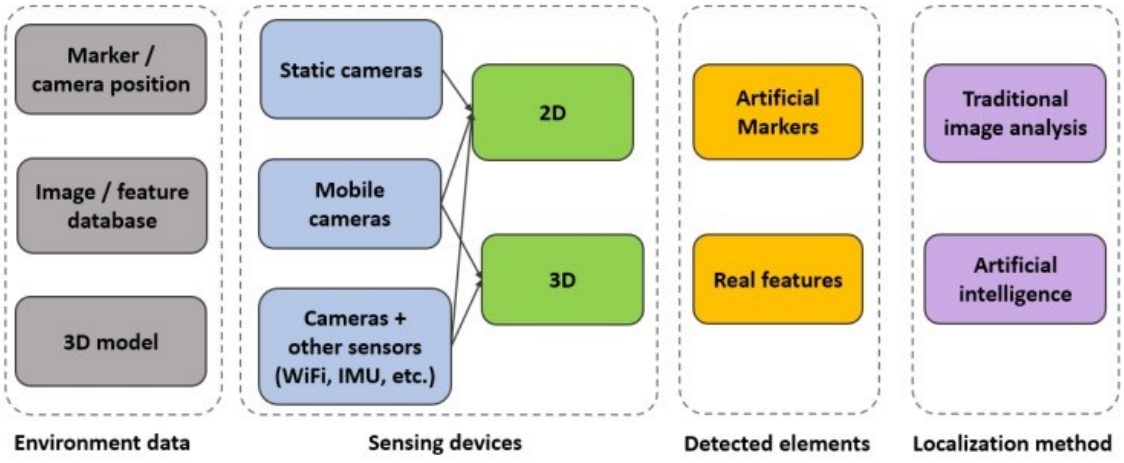

Figura 1 Clasificación propuesta por (Morar A, 2020) de las aproximaciones a la Geolocalización mediante técnicas de Visión por Computador.

Es importante destacar que (Morar A, 2020) en su clasificación especifica la combinación de varias tecnologías para una solución más certera en el cálculo de la posición relativa del dispositivo. En el futuro se discutirá la clasificación para métodos de localización mediante el uso de eventos/streaming, en la aplicación de tratamiento de imágenes o de algoritmos de inteligencia artificial.

El uso de dispositivos *wearables* para la geolocalización de personas en interior no es algo nuevo y la combinación de tecnologías para una mayor precisión en la localización, tampoco. Un ejemplo lo tenemos en el estudio llevado a cabo por (Sottile, 2017) en el que plantean una propuesta mediante el uso combinado de una banda Ultra ancha (UWB) y unidades de medición inercial (IMU) para el cálculo de la ubicación relativa del usuario. Otro ejemplo podría ser el uso de la realidad aumentada como el que se usó en el trabajo "Trajectes augmentats per a persones amb mobilitat reduïda" donde de forma similar al objetivo de este trabajo se pretendía guiar a usuarios vulnerables en la red de transporte público de Barcelona. (Catasús Llena, 2022)

La gran ventaja añadida del uso de tecnologías de visión por computador para este tipo de soluciones es que no solo permiten obtener una estimación relativa de la ubicación del usuario, sino que también permiten obtener información clave del espacio en el que se encuentra el usuario. Por ejemplo, identificar peligros inherentes a su ubicación. En el contexto de este trabajo y aunque por limitaciones temporales y de objetivos del proyecto no se ha implementado, se podría haber diseñado otro modelo de aprendizaje automático para la identificación de objetos en la escena que pudieran ser peligrosos para el usuario y, mediante un sensor de distancia como podría ser un sensor de ultrasonidos, poder calcular la distancia a este e informar al usuario.

Limitaciones técnicas

Para la propuesta de la solución deberán tenerse en cuenta varios aspectos que limitan la aproximación final.

Dado que el dispositivo debe poder ser usado sin que este resulte en molestias para el usuario, el tamaño y el peso son factores clave. Esto implica que la capacidad del prototipo estará limitada en cuanto a potencia y batería. Además, su uso no debe resultar en un potencial peligro para el usuario, por lo que las baterías a usar también deberán ser estudiadas.

Tomaremos en consideración que el producto final debe respetar unas medidas de 37,65 de ancho, 42mm de alto y 10,44mm de profundidad. Sin embargo y dado que esto limitaría excesivamente el prototipado, tomaremos como máximo 10cm de alto, 10 cm de ancho y 5 cm de profundidad.

Consideraciones legales y éticas

En la implementación de la solución hay que realizar una reflexión respecto al tratamiento de los datos dado que este tipo de sistema va a ser diseñado por el uso de usuarios con una discapacidad y, por lo tanto, es información de salud y si en condiciones normales hay que respetar la privacidad de los usuarios. Hay que ser más prudentes aún.

En este sentido se debe seguir la regulación 2016/679 OF THE EUROPEAN PARLIAMENT AND OF THE COUNCIL on the protection of natural persons with regard to the processing of personal data and on the free movement of such data, and repealing Directive 95/46/EC (General Data Protection Regulation) y la Ley Orgánica 3/2018, de 5 de diciembre, de Protección de Datos Personales y garantía de los derechos digitales.

De acuerdo con esta legislación estos son los principios sobre los que se debe basar la solución final:

1. En este sentido, se deben tratar los datos con un propósito delimitado y solo aquellos que sean necesarios para el propósito de la acción.
2. Los datos deben ser precisos y mantenidos actualizados.
3. Los datos no deben mantenerse más tiempo del estrictamente necesario.
4. Los datos se deben procesar de forma que se asegure la integridad y la confidencialidad.
5. Se deben diseñar protocolos que permitan mantener seguros los datos y las comunicaciones.
6. El acceso debe ser identificado, de forma que un usuario solo pueda ver sus propios datos.

Para el desarrollo del prototipo no se seguirán en principio estas directrices ya que el objetivo es implementar una primera versión y comprobar las funciones principales. Sin embargo, cualquier desarrollo posterior durante el presente proyecto que vaya más allá de los objetivos iniciales establecidos deberá tener en cuenta estas directrices.

Solución para el prototipado

El prototipado deberá ser basado en una propuesta de dispositivo IoT que sea capaz de enviar y recibir información para ser procesada en un servidor externo. Este servidor (centralizado o descentralizado) procesará la información y la devolverá al dispositivo IoT que realizará la acción pertinente, ya sea mediante señal física, acústica o visual.

La información que enviar dependerá en última instancia de la solución a adoptar, es por esto por lo que tomaremos en cuenta varios dispositivos antes de realizar la selección final, en cualquier caso, una conexión WIFI es necesaria para la trasmisión de datos en tiempo real.

1. Arduino Nano 33 IoT – 15,24mm x 43,16mm * 17,77mm Aprox.
2. Familia ESP32 (Serie ESP32-Cam posee cámara incorporada) 55,5mm x 27mm x 10mm Aprox.
3. Modelos Raspberry Pi 85mm x 53mm Aprox.

Deberán tenerse en cuenta elementos adicionales:

1. Cámara – si se usara
2. Micrófono
3. Altavoz
4. Batería

Como podemos observar en la página de Intel real sense, la conectividad de cámaras de profundidad como la D435 es plausible con elementos como un Arduino Micro

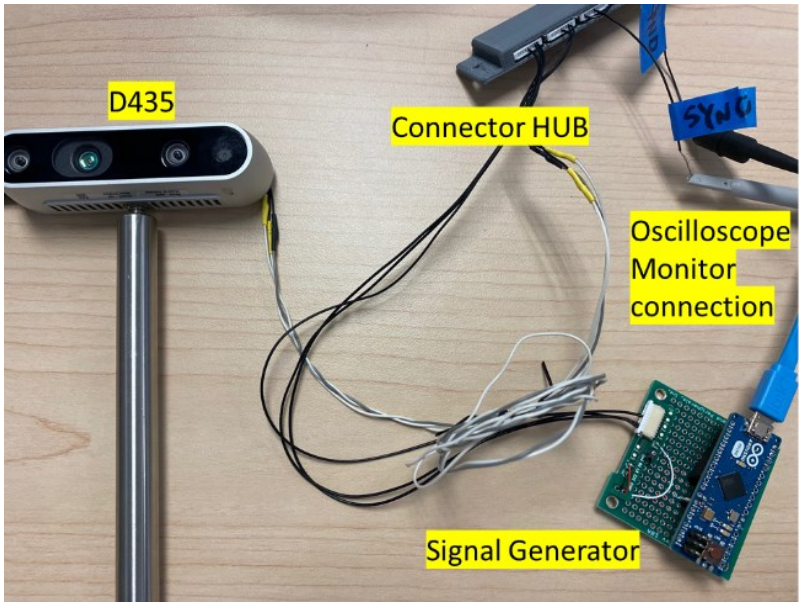

Figura 2 Ejemplo de combinación de los elementos y espacio a usar para el prototipo (Anders Grunnet-Jepsen)

El impedimento de este tipo de cámaras es el precio, pudiendo superar los 300 euros.

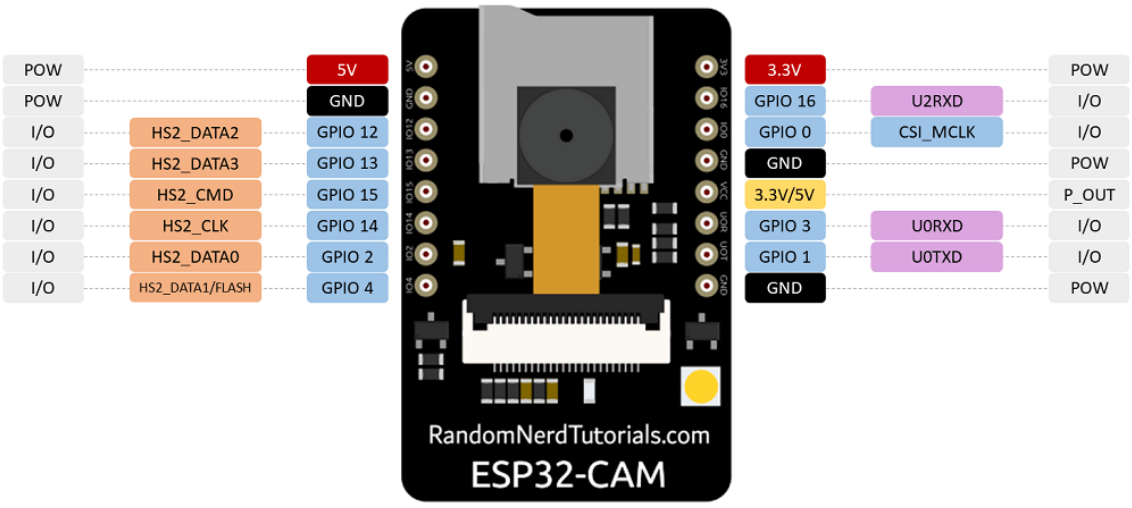

Figura 3 Esquema ESP32-CAM

Un dispositivo como el ESP32-CAM tiene una cámara integrada que puede ser intercambiada por varios modelos desde los 0,3MP (640x480) hasta los 5MP (2592x1944) OV5640. Su precio ronda los 20 euros, lo que lo convierte en un modelo mucho más asequible de cara al prototipado. Además, posee una tarjeta WIFI/Bluetooth integrada y mediante un módulo

MAX98537, sería posible reproducir sonido en unos altavoces externos. Además, su tamaño encajaría con el producto final, si se consiguiera realizar una localización mediante visión por computador y otras técnicas de localización para mejorar la precisión.

Elección final

Para el prototipado se han seleccionado los siguientes componentes:

1. Dos placas ESP32-WROVER-DEV
2. Un PCM5102A para la reproducción de sonido
3. Un speaker de 8 ohms
4. Una Jetson Nano en su versión de 2GB
5. Un dispositivo inteligente con micrófono incorporado

La cámara incorporada por las placas ESP32 utiliza la mayor parte de los pines GPIO, haciendo inviable tanto el uso de una única placa ESP32 WROVER como el uso planteado al inicio del proyecto de la placa ESP32-CAM con muchos menos pines que la versión WROVER. Por lo cual decidí dividir la solución en cuatro dispositivos:

1. Placa ESP32 - A
    a. Recogida de peticiones POST y GET http mediante una API REST.
    b. Reproducción de sonido.
    c. Gestión y sincronización del prototipado.
2. Placa ESP32 - B
    a. Recogida y envío de imágenes.
3. Jetson nano:
    a. Servidor para la predicción de escenas, cálculo de camino más corto y generación de ficheros de audio.
4. Dispositivo inteligente con micrófono integrado. Ej: Smartphone.
    a. Envío de comandos mediante Google Assistant.

Pese a que la intención inicial era la recogida de datos mediante un micrófono para el reconocimiento de voz y el inicio de las instrucciones, la problemática de los pines sumada a la complejidad que adquiere la solución hace que haya optado algo más funcional para el prototipado. En su lugar, las órdenes son recogidas mediante el Asistente de Google de cualquier dispositivo móvil sincronizado con el proyecto y enviadas a un applet de IFTTT. Este applet hace una petición de tipo HTTP - POST a una de las placas ESP32, que recoge el comando e inicia el proceso de detección y posterior comunicación con la otra placa ESP32 y la JETSON nano.

Para el cálculo del camino más corto entre un punto A y un punto B se ha decidido hacer uso del algoritmo Dijkstra dado que es uno de los más conocidos y utilizados en un grafo ponderado. Aunque no siempre es el más eficiente. Dado que estamos en espacios de interior y el número de nodos está acotado al número de habitaciones y el número de aristas al número de puertas. Durante la realización del prototipo no considero que la complejidad de los espacios que van a ser testados justifique un análisis exhaustivo del algoritmo a utilizar. Sin embargo, si la complejidad del problema lo justificara, el uso de algoritmos como el de A* permite reducir el espacio de búsqueda y mejorar los tiempos de ejecución.

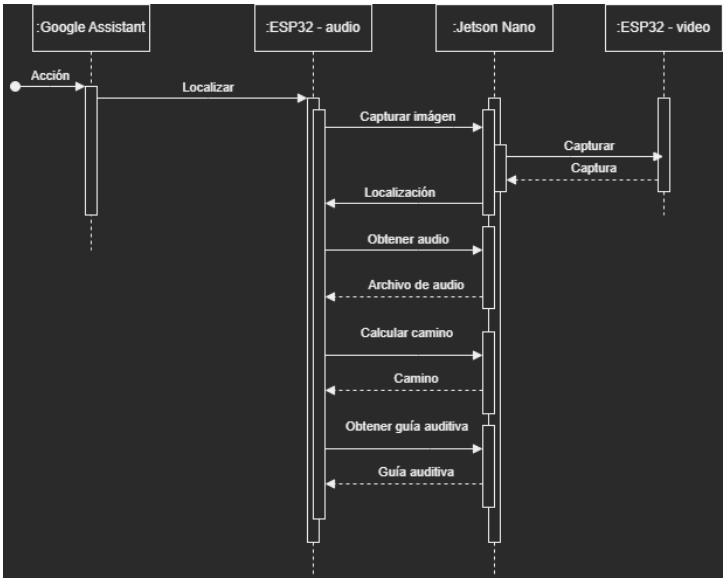

Figura 4 Diagrama de comunicación entre componentes

Objetivos

Objetivos funcionales del proyecto:

1. Elaborar un prototipo capaz de orientar en el espacio a una persona con discapacidad visual.
2. Identificar la posición del usuario mediante el uso de Visión por Computador y modelos de aprendizaje automático.
3. Guiar al usuario hasta su destino mediante indicaciones auditivas.

Objetivos de Desarrollo Sostenible:

1. En pro del Objetivo de Desarrollo Sostenible 9 – Impulsar el desarrollo de investigaciones y propuestas innovadoras en torno a la discapacidad visual.
2. En pro del Objetivo de Desarrollo Sostenible 10 – Reducción de las desigualdades – contribuir a la independencia de las personas con discapacidad visual.
3. En pro del Objetivo de Desarrollo Sostenible 11 – Apoyo a que los espacios comunitarios sean más accesibles, mejorando la comunicación y contribuyendo a la inclusión social.

Desarrollo del trabajo

La fase de desarrollo del proyecto ha tenido varias partes bien diferenciadas:

1. Configuración por etapas:
    a. Servidor WEB para el envío y retransmisión de peticiones POST y GET.
    b. Reproducción de sonidos.
    c. Servidor WEB para la retransmisión de vídeo
    d. Entrenamiento de modelo de ML para la clasificación de espacios
    e. Aplicación web desarrollada en Python para la clasificación y guía en espacios.
    f. Configuración de IFTTT en un dispositivo inteligente.
2. Pruebas unitarias:
    a. Verificación servidor web y obtención de imágenes.

   b. Verificación del servidor web y de la REST API para el envío y recepción de peticiones.
3. Pruebas de integración:
   a. Comunicación entre los dos microcontroladores ESP32 y la Jetson Nano.
4. Pruebas funcionales:
   a. Enviar petición mediante comando de voz y obtener las instrucciones auditivas correspondientes a la localización y el recorrido más corto.

Dada la complejidad del proyecto en cuanto a tecnologías, sistemas y conexión entre varios dispositivos se empezó por las funciones básicas del sistema. Estas son la emisión de imágenes mediante un servidor web y la reproducción de sonido haciendo uso de un altavoz, un conversor digital a analógico y la conexión a un streaming de audio.

Configuración por etapas:

Programación y configuración de los módulos ESP32

El proyecto se ha realizado mediante la combinación de distintas herramientas y lenguajes. En un primer lugar se consideró la programación de los microcontroladores ESP32 mediante el lenguaje de programación MicroPython. La programación de las placas ESP32 mediante MicroPython es muy versátil, dado que permite enviar comandos e interactuar con la placa sin necesidad de compilar previamente. Al no ser necesario compilar el programa y subirlo cada vez, lo que reduce considerablemente el tiempo de programación y de testeo. Este coste, sin embargo, es repercutido posteriormente al necesitar ser traducidas las instrucciones de alto nivel a bajo nivel en tiempo real.

No obstante, y pese a que fue posible establecer un servidor web para la retransmisión de vídeo, no se consiguió hacer funcionar la liberaría I2S – Inter-IC Sound bus protocol para la reproducción de audio. En su lugar se decidió proceder con la programación típica de sistemas Arduino mediante C/C++.

Como ha sido mencionado en el apartado *Elección final* se compraron dos placas ESP32 WROVER. Una en el paquete “Freenove Ultimate Starter Kit for ESP32-WROVER” que contenía un conversor y amplificador de audio PCM5102A, además de la propia placa. La idea inicial era la de usar una única placa, al considerar en su momento que los pines de la placan eran independientes de los pines usados por el BUS de la cámara web, al no ser así, fue imprescindible la adquisición de otra placa.

Los dispositivos de Freenove vienen con algunos ejemplos de uso para el uso de un servidor de vídeo web y la reproducción de sonido. Se ha realizado una adaptación de estos ejemplos, además de ejemplos de uso de la librería “ESP32-audioI2S”.

Servidor WEB para el envío y retransmisión de peticiones POST y GET.

Mediante las librerías Wifi, HTTPClient y ESPAsyncWebServer y AsyncTCP realizaremos la conexión y comunicación entre los distintos componentes. ESPAsynWebServer es un WebSocket desarrollado para placas ESP8266 y ESP32. Además de la compatibilidad con las placas ESP32, la principal razón del uso de esta librería fue la posibilidad de tener distintas instancias asíncronas esperando recibir peticiones de conexión mediante HTTP. Esto me permite, bajo el mismo puerto poder enviar y recibir peticiones sin tener que gestionar manualmente la disponibilidad de recursos de CPU o memoria.

Dentro del fichero cabe destacar:

```
37
38  // Set web server port number to 32769
39  AsyncWebServer server(32769);
40
```

Figura 5 Servidor web asíncrono

Iniciamos el servidor AsyncWebServer en el puerto 32769, esto es importante ya que el router de mi vivienda para conexiones a exterior obliga a usar puertos por encima de 32768 para conexiones con el exterior.

Sobre este puerto se realizarán todas las peticiones HTTP y es sobre el que las instancias asíncronas del servidor escucharán el tráfico entrante.

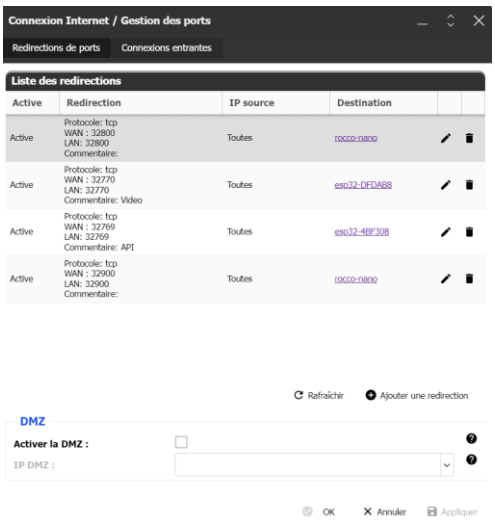

Figura 6 gestión de puertos

Dentro de la función Setup de nuestro programa Arduino se destacan las siguientes funciones:

```
56
57  const char* ssid     = "********";   //input your wifi name
58  char* password = "********";   //input your wifi passwords
59
60
61  WiFi.begin(ssid, password);
62
63  while (WiFi.status() != WL_CONNECTED) {
64    delay(1000);
65    Serial.println("Connecting to WiFi..");
66  }
67
```

Figura 7 Conexión WIFI

La figura 5 ilustra la configuración de la red WIFI se realiza indicando el SSID o nombre de la red y la contraseña en el método begin de la clase WiFi.

```
});

AsyncCallbackJsonWebHandler *handler = new AsyncCallbackJsonWebHandler("/post-message", [](AsyncWebServerRequest *request, JsonVariant &json) {
  JsonObject jsonObj = json.as<JsonObject>();

  endPoint = jsonObj["endpoint"].as<String>();
  Serial.println(endPoint);
```

Figura 8 handler para la recepción de peticiones POST en /post-message

En la figura 8 definimos una instancia asíncrona del servidor para la gestión de peticiones POST en la dirección URL relativa /post-message que debe incorporar un elemento JSON en la misma. La gestión de elementos JSON se realiza mediante la librería "AsyncJson". Un elemento JSON (JavaScript Object Notation) en un formato de texto para representar estructuras de datos de tipo clave – valor. Como puede verse en la figura 9 en el *Body* de la petición HTTP tenemos una representación de datos con un elemento entrecomillado seguido de dos puntos y otro elemento entrecomillado siguiendo el convenio definido por JSON. Mediante esta librería y la petición, facilitamos la obtención del dato. El dato es almacenado en la variable endPoint. La comunicación se ha realizado mediante la herramienta Postman, una plataforma API que permite hacer uso de otras API para hacer pruebas y verificar el funcionamiento.

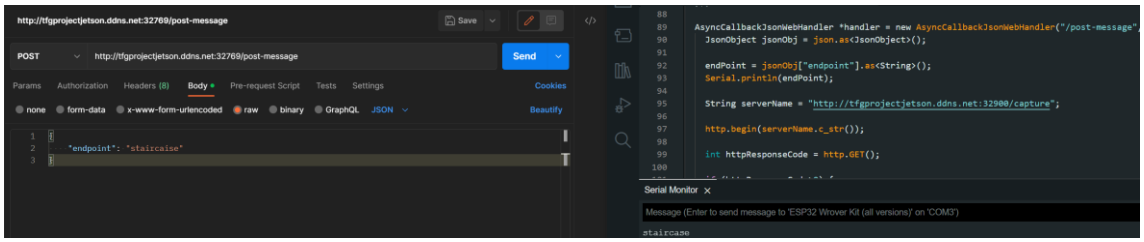

Figura 9 Comunicación mediante la instancia asíncrona haciendo uso de Postman

Reproducción de sonido.

Habiendo habilitado el servidor web para la emisión y recepción de peticiones HTTP, el siguiente paso fue conseguir reproducir archivos de sonido mediante un altavoz y el microcontrolador. Como se ha explicado anteriormente para la reducción del sonido se contaba con un dispositivo PCM5102A

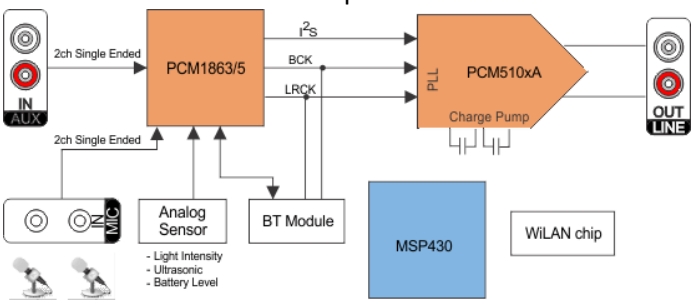

Figura 10 Esquema PCM5102A

Este circuito integrado, permite conectar dos dispositivos de salida estéreo y la reproducción de sonido mediante el estándar I2S para la transmisión de datos. El uso del estándar I2S permite separar las señales de datos y la de reloj, permitiendo reducir las fluctuaciones.

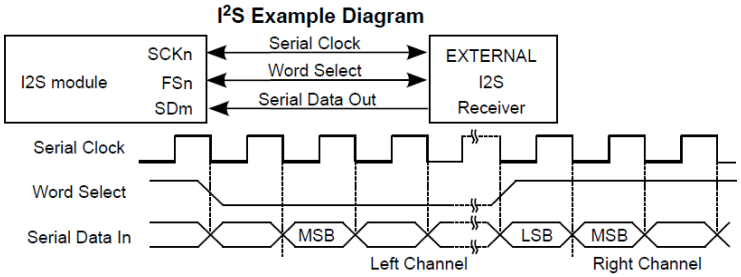

Figura 11 Ejemplo diagrama I2S (Microchip Technology, Inc., 2021)

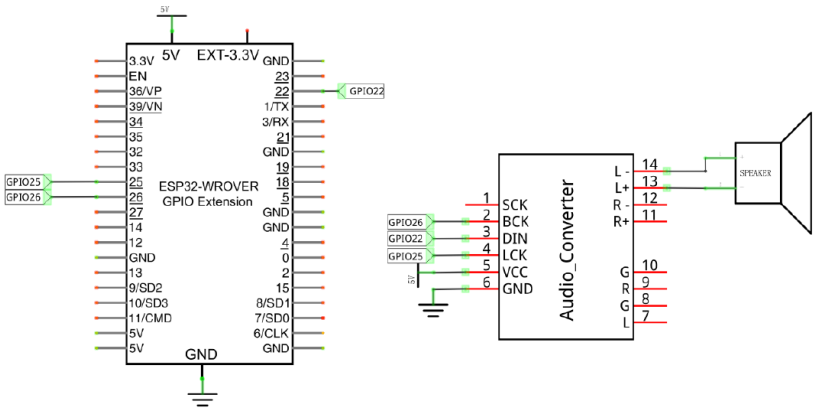

Figura 12 Diagrama esquemático de conexión PM5105A y ESP32

Al necesitar reproducir archivos de audio mediane streaming decidí hacer uso de la librería ESP32-audioI2S. Esta librería incluye ejemplos de uso y permite la reproducción de ficheros de audio tanto mediante una URL como a través de la tarjeta SD.

```
111
112     String audioName = obj["location"];
113     startPoint = obj["startPoint"].as<String>();
114     String url = "http://tfgprojectjetson.ddns.net:32900";
115     String urlRequest = url + audioName;
116     audio.connecttohost(urlRequest.c_str());
117     flagLocation = true;
118     flagGuide = true;
119     request->send(200, "application/js  on", payload);
```

Figura 13 Uso de librería para conexión y reproducción por streaming

Mediante el método connecttohost de la clase audio nos conectamos a un servidor web para la obtención del fichero.

```
167
168 void loop(){
169   if(flagLocation == true) {
170     audio.loop();
171   }
172 }
```

Figura 14 reproducción de audio

Mediante el método loop de la clase audio, reproducimos el fichero de audio de la fuente seleccionada.

Servidor WEB para la retransmisión de vídeo

Para la retransmisión de vídeo WEB solo se ha realizado la modificación del puerto sobre el que se realiza la conexión en el software original de freenove Sketch_32.2_As_VideoWebServer.ino para posibilitar el envío de la información fuera de la red privada.

Entrenamiento de modelo de ML para la clasificación de espacios

El Instituto Tecnológico de Massachusetts ha realizado varios estudios para la clasificación y reconocimiento espacios exteriores e interiores. El artículo “Indoor Space Recognition using Deep Convolutional Neural Network: A Case Study at MIT Campus” describe como desarrollaron una red neuronal convulacional para la clasificación de espacios interiores en el propio Campus. Este estudio se aproxima mucho a las necesidades del presente estudio, pero debido a las carencias en términos de material y de privacidad, decidí hacer uso de un set de datos del MIT llamado places365standard_easyformat para el entrenamiento de una red neuronal.

Debido a las limitaciones de recursos hardware hice uso de la herrmienta web gratuita de Google, Google colab.

Uno de los primeros problemas que me encontré al tratar de entrenar la red neuronal mediante los ficheros proporcionados por el departamento del MIT CSAIL Computr Vision, es que se hacía uso de funciones y de librerías obsoletas, por lo que fue necesaria una adaptación del código.

Otro aspecto a tener en cuenta es que de las 365 clases (exteriores e interiores) en el set de datos, había que seleccionar solo aquellas que nos interesaran para el entrenamiento. Es por ello por lo que solo se seleccionaron las siguientes 15 clases:

['corridor', 'waiting_room', 'shower', 'basement', 'dining_hall', 'alcove', 'computer_room', 'bathroom', 'living_room', 'dining_room', 'kitchen', 'entrance_hall', 'staircase', 'dorm_room', 'bedroom']

Una vez seleccionadas las clases que nos interesaban para el entrenamiento había que prepara el set de datos de entrenamiento y de validación:

```
trainFile = open('places365standard_easyformat/places365_standard/train.txt', 'r')
Lines = trainFile.readlines()

if not os.path.exists("customPlaces"):
  os.mkdir("customPlaces")

if not os.path.exists("customPlaces/train"):
  os.mkdir("customPlaces/train")

for element in goodElements:
  if not os.path.exists("customPlaces/train/" + element):
    os.mkdir("customPlaces/train/" + element)

customTrainFile = open("customPlaces/train.txt", "a")

for line in Lines:
    line = line.strip()
    if os.path.exists("places365standard_easyformat/places365_standard/" + line):
      if os.path.basename(os.path.dirname(line)) in goodElements:
        customTrainFile.write(line + "\n")
        shutil.copyfile("places365standard_easyformat/places365_standard/" + line, "customPlaces/" + line)

customTrainFile.close()
```

```
[ ] valFile = open('places365standard_easyformat/places365_standard/val.txt', 'r')
    Lines = valFile.readlines()

    if not os.path.exists("customPlaces"):
      os.mkdir("customPlaces")

    if not os.path.exists("customPlaces/val"):
      os.mkdir("customPlaces/val")

    for element in goodElements:
      if not os.path.exists("customPlaces/val/" + element):
        os.mkdir("customPlaces/val/" + element)

    customValFile = open("customPlaces/val.txt", "a")

    for line in Lines:
        line = line.strip()
        if os.path.exists("places365standard_easyformat/places365_standard/" + line):
          if os.path.basename(os.path.dirname(line)) in goodElements:
            customValFile.write(line + "\n")
            shutil.copyfile("places365standard_easyformat/places365_standard/" + line, "customPlaces/" + line)

    customTrainFile.close()
```

Figura 15 preparación de set de datos de entrenamiento y de validación

Una vez ejecutado el código obtenemos una estructura como la siguiente:

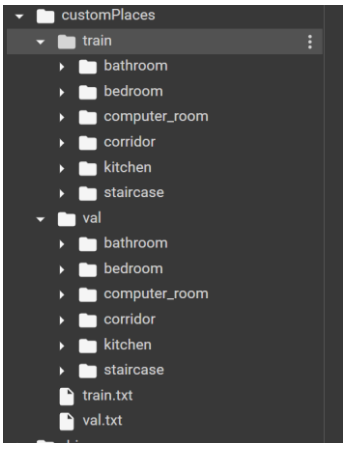

Figura 16 Estructura del set de datos

Si usamos el dataset proporcionado por places365 dispondremos de unas 1000 imágenes para entrenamiento en cada carpeta y unas 360 imágenes de validación.

	Validation Set of Places365		Test Set of Places365	
	Top-1 acc.	Top-5 acc.	Top-1 acc.	Top-5 acc.
Places365-AlexNet	53.17%	82.89%	53.31%	82.75%
Places365-GoogLeNet	53.63%	83.88%	53.59%	84.01%
Places365-VGG	**55.24%**	84.91%	**55.19%**	85.01%
Places365-ResNet	54.74%	**85.08%**	54.65%	**85.07%**

Figura 17 Tabla comparativa de precisión de distintos modelos de redes neuronales entrenadas mediante el set de datos Places365

Tras la adaptación del código train_placesCNN.py para hacerlo compatible con las versiones actuales de las librerías, se decidió hacer uso de la red neuronal ResNet por su mayor precisión en sus 5 primeras respuestas.

```
!python3 places365/train_placesCNN.py -a resnet18 customPlaces --epochs 1

Namespace(arch='resnet18', batch_size=256, data='customPlaces', dataset='places365', epochs=1, eva
=> creating model 'resnet18'
DataParallel(
```

Figura 18 fase de entrenamiento del modelo de ML

```
Epoch: [0][780/816]     Time 0.726 (0.735)      Data 0.000 (0.094)      Loss 2.5482 (2.9627)    Prec@1 26.172 (20.707)
Epoch: [0][790/816]     Time 0.724 (0.735)      Data 0.000 (0.094)      Loss 2.5591 (2.9578)    Prec@1 26.562 (20.817)
Epoch: [0][800/816]     Time 0.727 (0.735)      Data 0.000 (0.094)      Loss 2.6202 (2.9533)    Prec@1 31.641 (20.909)
Epoch: [0][810/816]     Time 0.723 (0.735)      Data 0.000 (0.094)      Loss 2.5424 (2.9487)    Prec@1 31.641 (21.016)
Test: [0/17]    Time 2.494 (2.494)      Loss 2.9630 (2.9630)    Prec@1 23.828 (23.828)  Prec@5 57.422 (57.422)
Test: [10/17]   Time 0.014 (0.475)      Loss 2.5359 (2.6018)    Prec@1 34.375 (28.693)  Prec@5 60.547 (63.636)
 * Prec@1 29.930 Prec@5 64.930
```

Figura 19 precisión en fase de validación del modelo de ML

La precisión final, sin embargo, al haber limitado tanto el número de clases fue muy inferior a la de los modelos entrenados con las 365 clases.

Al ser unos resultados pobres para una clasificación, se decidió reducir el número de clases y la ratio de aprendizaje para de esta forma evitar similitudes entre clases que puedan llevar a una incorrecta clasificación y para permitir a la red neuronal aprender más de las características de cada clase, pese al riesgo de *overfitting*.

```
Epoch: [74][0/112]      Time 1.498 (1.498)      Data 1.466 (1.466)      Loss 0.4093 (0.4093)    Prec@1 85.547 (85.547)  Prec@5 100.000 (100.000)
Epoch: [74][10/112]     Time 0.097 (0.266)      Data 0.000 (0.193)      Loss 0.4227 (0.4187)    Prec@1 82.812 (84.730)  Prec@5 100.000 (99.680)
Epoch: [74][20/112]     Time 0.097 (0.221)      Data 0.000 (0.154)      Loss 0.5079 (0.4161)    Prec@1 83.594 (85.026)  Prec@5 99.609 (99.647)
Epoch: [74][30/112]     Time 0.548 (0.205)      Data 0.521 (0.139)      Loss 0.5110 (0.4229)    Prec@1 83.203 (84.854)  Prec@5 100.000 (99.647)
Epoch: [74][40/112]     Time 0.096 (0.186)      Data 0.000 (0.117)      Loss 0.3870 (0.4212)    Prec@1 87.891 (85.013)  Prec@5 99.609 (99.667)
Epoch: [74][50/112]     Time 0.097 (0.179)      Data 0.000 (0.110)      Loss 0.3813 (0.4227)    Prec@1 87.891 (85.011)  Prec@5 99.609 (99.694)
Epoch: [74][60/112]     Time 0.638 (0.178)      Data 0.610 (0.109)      Loss 0.4465 (0.4233)    Prec@1 82.812 (84.964)  Prec@5 100.000 (99.705)
Epoch: [74][70/112]     Time 0.097 (0.170)      Data 0.000 (0.100)      Loss 0.4455 (0.4272)    Prec@1 81.250 (84.782)  Prec@5 100.000 (99.675)
Epoch: [74][80/112]     Time 0.097 (0.168)      Data 0.000 (0.098)      Loss 0.3347 (0.4244)    Prec@1 88.672 (84.939)  Prec@5 99.219 (99.667)
Epoch: [74][90/112]     Time 0.374 (0.166)      Data 0.347 (0.096)      Loss 0.5734 (0.4262)    Prec@1 79.297 (84.912)  Prec@5 99.219 (99.639)
Epoch: [74][100/112]    Time 0.097 (0.163)      Data 0.000 (0.092)      Loss 0.3809 (0.4231)    Prec@1 86.719 (85.017)  Prec@5 100.000 (99.640)
Epoch: [74][110/112]    Time 0.096 (0.161)      Data 0.000 (0.090)      Loss 0.4371 (0.4232)    Prec@1 86.719 (85.051)  Prec@5 100.000 (99.641)
Test: [0/3]     Time 1.004 (1.004)      Loss 0.3867 (0.3867)    Prec@1 87.500 (87.500)  Prec@5 100.000 (100.000)
 * Prec@1 89.333 Prec@5 99.833
```

Figura 20 Precisión final en fase de validación del modelo de ML

En la etapa 75 fue donde se consiguió la mayor precisión, con un 89,33% de probabilidad de clasificar la imagen en el primer resultado y un 99.83% de probabilidad de acierto entre las cinco primeras opciones. Sin embargo, este último valor no es estadísticamente significativo al tener el modelo únicamente 6 posibles clases:

['corridor', 'computer_room', 'bathroom', 'kitchen', 'staircase', 'bedroom']

El programa Python usado para el entrenamiento del modelo y su salvaguardado, guarda el mejor modelo hasta el momento, así como el último modelo generado.

En cuanto a los cambios en el código de train_placesCNN.py para su funcionamiento, están reflejados en el fichero adjunto con dicho nombre.

Por último, para la clasificación de los resultados es necesario disponer de un archivo con los nombres de las clases, así como su posición con la siguiente estructura:

/b/bathroom 0
/b/bedroom 1
/c/computer_room 2
/c/corridor 3
/k/kitchen 4
/s/staircase 5

Aplicación web desarrollada en Python para la clasificación y guía en espacios.

Para la creación de un servidor web que se encargue no solo de recibir y enviar peticiones HTTP, sino también de realizar la clasificación de imágenes y generación de archivos de audio, decidí usar el framework Flask de Python que permite crear aplicaciones web rápidamente bajo este lenguaje de programación. El uso de Python era imprescindible para el uso del modelo de Machine Learning, si bien es cierto hay otros lenguajes que permiten su uso Python tiene una menor complejidad en su uso y una mayor versatilidad de librerías. Además, el proyecto places365 de CSAILVision tiene un archivo Python denominado run_placesCNN_unified que contenía un código sobre el que probar la librería.

En un primer lugar y para poder programar la aplicación encargada de clasificar y generar los archivos de audio, fue necesario seguir la guía de NVIDIA para la configuración y puesta en marcha de la Jetson NANO (NVIDIA, n.d.). Posteriormente, aunque podría haber hecho uso del proyecto Docker para hacer uso de los contenedores que incluyen por defecto PyTorch, TorchVision y otras librerías, decidí hacer uso de la guía (Q-engineering, 2022) para su instalación con los paquetes 1.8.0 para PyTorch y 0.11.0 para TorchVision dada la versión de Ubuntu instalada en la Jetson Nano 18.04.

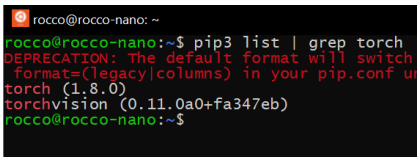

Figura 21 librerías instaladas de torch

Una vez finalizada la instalación se probó el programa run_placesCNN_unified para comprobar la correcta ejecución.

```
2023-02-12 17:12:57 (8,85 MB/s) - 'wideresnet.py.1' saved [6691/6691]

RESULT ON http://places.csail.mit.edu/demo/6.jpg
--TYPE OF ENVIRONMENT: indoor
--SCENE CATEGORIES:
0.512 -> food_court
0.084 -> fastfood_restaurant
0.083 -> cafeteria
0.040 -> dining_hall
0.021 -> flea_market/indoor
--SCENE ATTRIBUTES:
no horizon, enclosed area, man-made, socializing, indoor lighting, cloth, congregating, eating, working
Class activation map is saved as cam.jpg
rocco@rocco-nano:~$
```

Figura 22 prueba de funcionamiento del modelo de ML

Figura 23 Mapa de calor generado con las areas usadas por el modelo de ML para la clasificación

El Framework de Flask permitió que con la adición de las funciones y etiquetas para procesar las peticiones HTTP se pudiera usar el fichero run_placesCNN_unified modificando solo las rutas a las categorías y al modelo de aprendizaje automático exportado. Concretamente se realizaron los siguientes cambios:

Para poder iniciar el servidor flask.

```
app = Flask(__name__)
```

Obtención de una captura de imagen de la retransmisión de video llevada a cabo por el servidor web, además de su transformación a una imagen PIL para poder ser procesada, además de la ejecución del resto de código original de run_placesCNN_unified y la generación de un archivo de audio con la ubicación mediante el comando espeak:

```
@app.get("/capture")
def get_capture():
    cam = cv2.VideoCapture("http://tfgprojectjetson.ddns.net:32770/stream")

    img_counter = 0

    ret, frame = cam.read()
    img_name = ""
    if not ret:
        img_name = "failed to grab frame"
    else:
        img_name = "opencv_frame_{}.png".format(img_counter)
        cv2.imwrite(img_name, frame)
        app.logger.warning("{} written!".format(img_name))
        img_counter += 1

    cam.release()
    #Converting to PIL imaged (added by Joan)
    img = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)
    img = Image.fromarray(img)

(..) #Codigo de run_placesCNN_unified
    os.system(f'espeak -v en-gb "With a probability of {round(probs[0]*100)} you are located in
the {classes[idx[0]]}" -w audios/command.wav')
```

```
    os.system('lame audios/command.wav audios/command.mp3')

    return {"startPoint": classes[idx[0]], "location":f"/audios?audio=command.mp3"}, 200
```

Obtención de la ruta del archivo de audio a reproducir con la ubicación.

```
@app.get("/audios")
def getAudio():
    audio = request.args.get('audio')
    def generate():
        with open(f"audios/{audio}", "rb") as fwav:
            data = fwav.read(1024)
            while data:
                yield data
                data = fwav.read(1024)
    return Response(generate(), mimetype="audio/mp3")
```

Obtención del archivo de audio con la ruta guiada

```
@app.get("/audioGuide")
def getGuide():
    audio = request.args.get('audio')
    def generate():
        with open(f"audios/{audio}", "rb") as fwav:
            data = fwav.read(1024)
            while data:
                yield data
                data = fwav.read(1024)
    return Response(generate(), mimetype="audio/mp3")
```

Mediante la function getPath() se calcula el camino más corto entre un punto A y un punto B mediante el algoritmo Dijkstra:

```
@app.get("/guide")
def getPath():
    startPoint = request.args.get('startPoint')
    endPoint = request.args.get('endPoint')
```

Para la generación de la matriz de adyacencia en los espacios, se ha tenido en consideración que la distancia entre un punto A y su punto adyacente B es de 1 para mayor simplicidad. No obstante, lo ideal sería obtener la distancia entre la ubicación del usuario y el punto A para calcular la distancia real del usuario y los distintos puntos. Una de las ideas iniciales para ello fue hacer uso de la fuerza de la señal WIFI en puntos de acceso para de esta forma obtener la distancia relativa en unidades métricas a través de los decibelios. Algunos puntos de acceso como los de la marca MicroTik permiten obtener fácilmente estas medidas y podría realizarse un mapeado del espacio para obtener las medidas genéricas de la señal para cada punto de acceso en cada ubicación. Esto, al final, permitiría triangulizar la ubicación del usuario y calcular la distancia.

El método diseñado para exportar un espacio a un fichero de texto ha consistido en mediante un fichero yaml, especificar las ubicaciones indicando los lugares adyacentes al mismo. Este fichero yaml es exportado y procesado para transformarlo en un grafo no dirigido.

```
Map:
 corridor:
 - dining_room
 - Cocina
 - staircase
 - staircase-Bajar
 - bathroom
 dining_room:
 - corridor
 Cocina:
 - corridor
 - terrace
```

Para ello se ha creado una librería denominada “path”, en ella se incorporan todas las funciones y clases para la generación e impresión de los grafos, así como el cálculo del camino más corto. En el desarrollo del programa me he basado en una implementación encontrada en la web (Miles, 2018)

En el siguiente código se puede apreciar como se realiza la generación del grafo a partir del fichero yaml.

```
   class Map:
     def __init__(self, fileName):
        self.visited = []
        with open(fileName, 'r') as file:
            self.data = json.loads(json.dumps(yaml.safe_load(file)))

        self.elements = {}
        self.n = 0
        self.edges = []
        self.dict = {}
        for node in self.data["Map"].keys():
            if node not in self.elements.keys():
                self.elements[node] = self.n
                self.n += 1
            self.dict[self.elements[node]] = set()
            for puerta in self.data["Map"][node]:
                print(node + " -> " + puerta)
                if puerta not in self.elements.keys():
                    self.elements[puerta] = self.n
                    self.n += 1
                element = (self.elements[node], self.elements[puerta])
                self.edges.append(element)
                self.dict[self.elements[node]].add((self.elements[puerta], 1))
```

En el código que sigue a continuación se puede apreciar cómo, una vez obtenidos el punto de inicio y el punto de destino mediante una petición POST en la url http://servidor/guide, realizamos la generación del grafo instanciando la clase Map de indoormap y, a partir del mapa devuelto por la instancia realizamos la llamada a las funciones Dijkstra y make_path para el cómputo y retorno del camino.

```
@app.get("/guide")
def getPath():
    startPoint = request.args.get('startPoint')
    endPoint = request.args.get('endPoint')

    graph = indoormap.Map("path/map.yml")
    graph.printGraph(graph)
```

```
    key_list = list(graph.elements.keys())
    val_list = list(graph.elements.values())

    app.logger.warning(startPoint)
    app.logger.warning(endPoint)


    parent = graph.dijkstra(graph.dict, key_list.index(startPoint), key_list.index(endPoint))
    finalPath = graph.make_path(parent, key_list.index(endPoint))

    # print key with val 100
       print("The shortest path from: " + key_list[val_list.index(0)] + " To: " +
key_list[val_list.index(10)] + " Is: ")
    camino = []
    comando = f"In order to go from {startPoint} to {endPoint}, you need to go to the following
places: "
    for i in finalPath:
        print(key_list[val_list.index(i)])
        camino.append(key_list[val_list.index(i)])
        comando = comando + key_list[val_list.index(i)] + ". "

    os.system(f'espeak -v en-gb "{comando}" -w audios/guide.wav')
    os.system('lame audios/guide.wav audios/guide.mp3')

    return {"guide":"/audioGuide?audio=guide.mp3"}, 200
```

Configuración de IFTTT en un dispositivo inteligente

IFTTT, “IF This, Then That” es un sitio web que permite crear y programar servicios web. Estos servicios realizan acciones que atienden a la lógica definida por el usuario. Entre otras funciones, permite la comunicación entre distintos servicios web o en mi caso, lo usaré para conectar el asistente de Google con mi prototipo.

Dentro de IFTTT tenemos un espacio de trabajo con los “Applets” que creamos para la conexión. Un Applet, será una aplicación programada de forma gráfica.

Para el propósito del prototipo se ha creado un único applet, se podrían crear hasta 5 de forma gratuita y 20 en la versión Pro. No obstante, el uso de IFTTT ha sido únicamente con la intención de acelerar el desarrollo del prototipo. Idealmente en la versión final, el dispositivo incorporaría su propio micrófono y el servidor realizaría la conversión de voz a texto. Posteriormente, podrían utilizarse tecnologías como chatGPT para transformar dicha oración de texto en una instrucción o comando similar a la que puede apreciarse en el applet de la Figura 24.

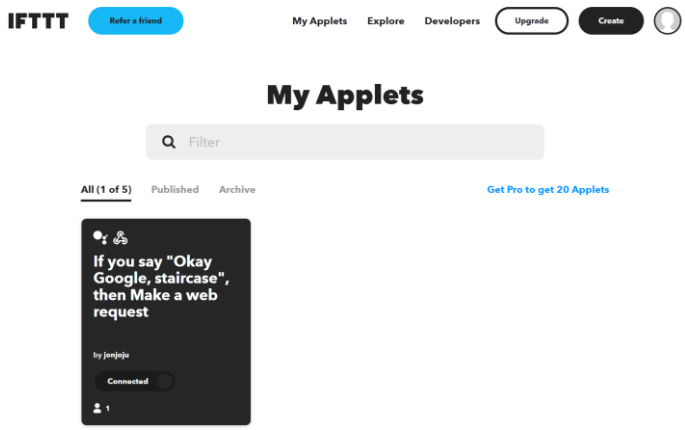

Figura 24 Espacio de trabajo

La estructura del Applet como puede apreciarse en la Figura 25, consta de una escena del asistente de Google que al activarse realiza una petición HTTP. Todas las peticiones HTTP realizadas en el flujo del programa Arduino de los microcontroladores ESP32 podrían haberse incorporado en este flujo. No obstante, dada la flexibilidad, la rapidez y el control que permite tener sobre el código he preferido realizar la programación del flujo en los propios microcontroladores. Además, como he indicado el uso de IFTTT esta es una solución temporal.

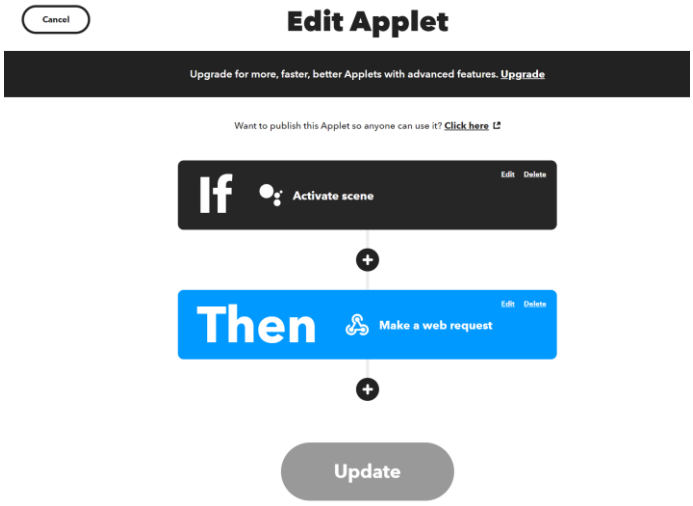

Figura 25 estructura del Applet

La escena de Google, contiene únicamente el nombre de la escena. En este caso, al indicarle a nuestro smartphone “Ok, Google. Activate staircase” iniciará la aplicación.

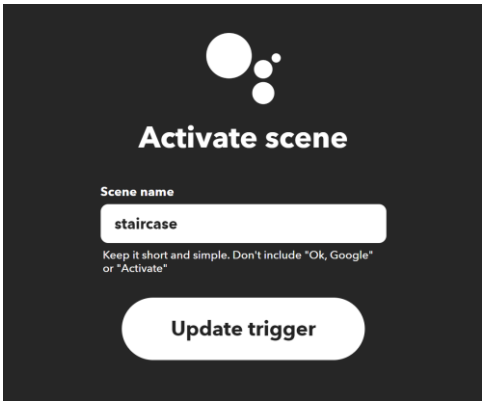

Figura 26 Escena Google

Esta escena inicia al activarse una petición web que inicia el flujo del prototipo dirigida al microcontrolador ESP32 para la localización de nuestro punto actual, indicando donde queremos dirigirnos. Siendo donde queremos dirigirnos el nombre de la escena. De esta forma si queremos dirigirnos al comedor, crearíamos el applet con la escena comedor y le hablaríamos a nuestro smartphone con el siguiente comando: “Ok, Google. Activa comedor”.

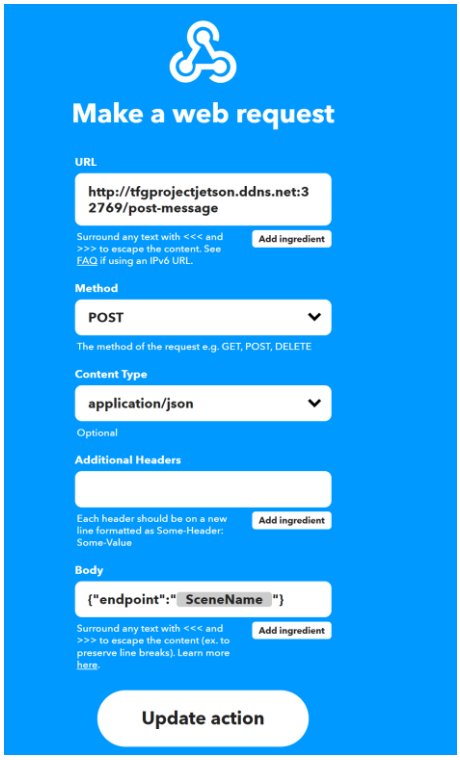

Figura 27 Petición web

Por último, realicé la configuración de mi dispositivo móvil para conectar mi asistente de Google con IFTTT. En primer lugar, tuve que realizar la instalación de la aplicación Home de Google y crear un espacio al que denominé TFG, posteriormente tuve que enlazar mi cuenta de IFTTT con Home tal y como puede verse en la Figura 28. Añadiendo un nuevo dispositivo que funciona con Google, seleccionando IFTTT e introduciendo nuestro usuario y contraseña para vincularlo.

Figura 28 Configuración de la cuenta IFTTT dentro de Home

Tras realizar esta configuración ya pude empezar a emitir comandos de voz al prototipo.

Pruebas unitarias:

Verificación servidor web y obtención de imágenes.

Para verificar la correcta obtención de imágenes mediante las peticiones HTTP, se ha hecho uso de la herramienta Postman para realizar una petición HTTP a la URL: http://tfgprojectjetson.ddns.net:32900/capture

Esta petición ha sido recibida por el servidor FLASK y ejecutado la porción de código etiquetada en la ruta /capture correspondiente al método get_capture(). Este método realiza otra petición HTTP mediante la función VideoCapture de OpenCV y capturamos una única imagen mediante el método read(). Esta imagen es transformada a formato PIL – Python Image Library – para poder ser utilizada por el modelo de ML y poder realizar las modificaciones de la imagen como por ejemplo la máscara de calor.

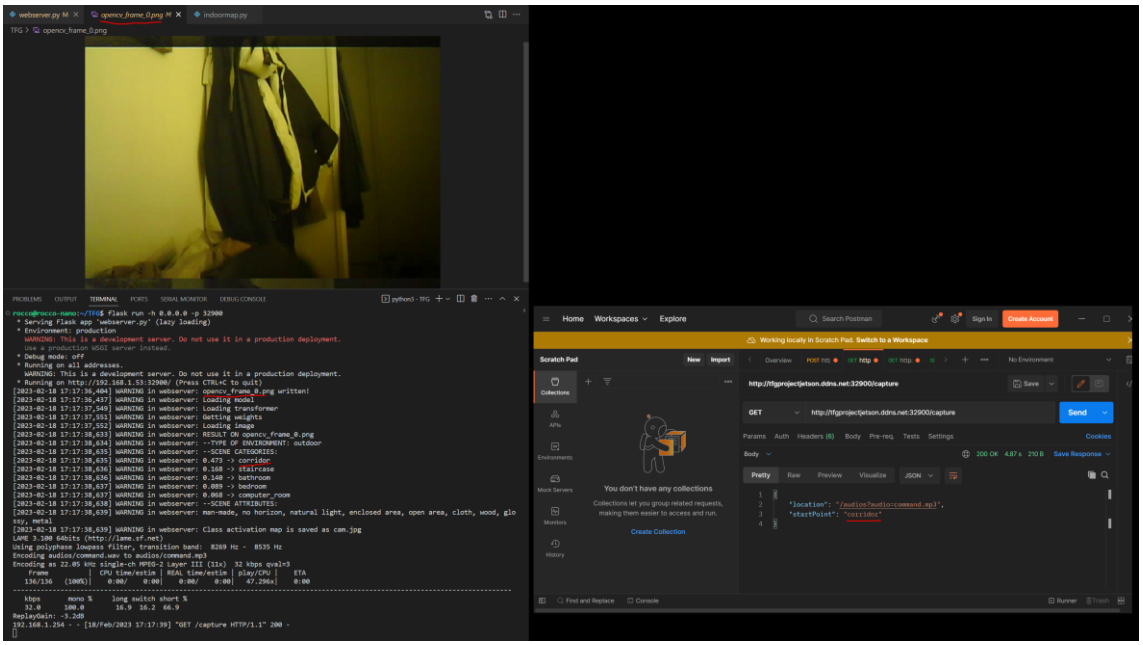

Figura 29 Obtención de imágenes del servidor web

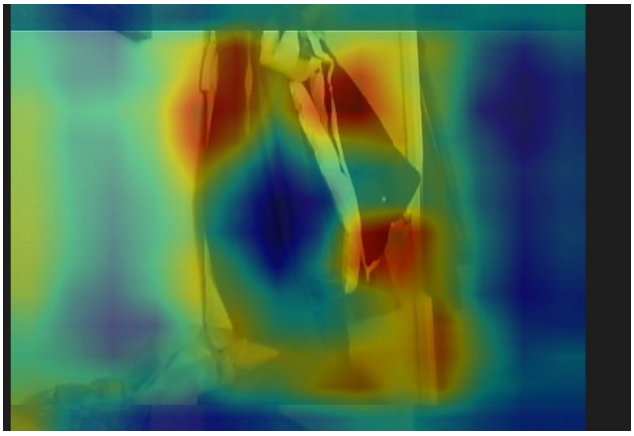

Figura 30 Máscara de calor aplicada a la imagen

Verificación del servidor web principal para el envío y recepción de peticiones.

El servidor encargado de recoger las instrucciones del usuario y comunicarse con el resto de las componentes se encuentra en una de las placas ESP32. Sobre este servidor realizaremos la comprobación de las siguientes peticiones HTTP:

Petición a la placa ESP32 indicando nuestro destino:

Mediante la herramienta Postman simulé la petición que se realiza desde el Asistente de Google a través del portal IFTTT a la URL http://tfgprojectjetson.ddns.net:32769/post-message. Como puede observarse en la Figura 31, en el cuerpo de la petición Post enviamos un diccionario JSON con la clave endpoint y el valor staircase. Indicando a la placa ESP32 que nuestro destino son las escaleras. Esta petición puede observarse en el Monitor Serie en la sexta línea donde se indica “staircase”. Esta prueba unitaria ha sido superada con éxito.

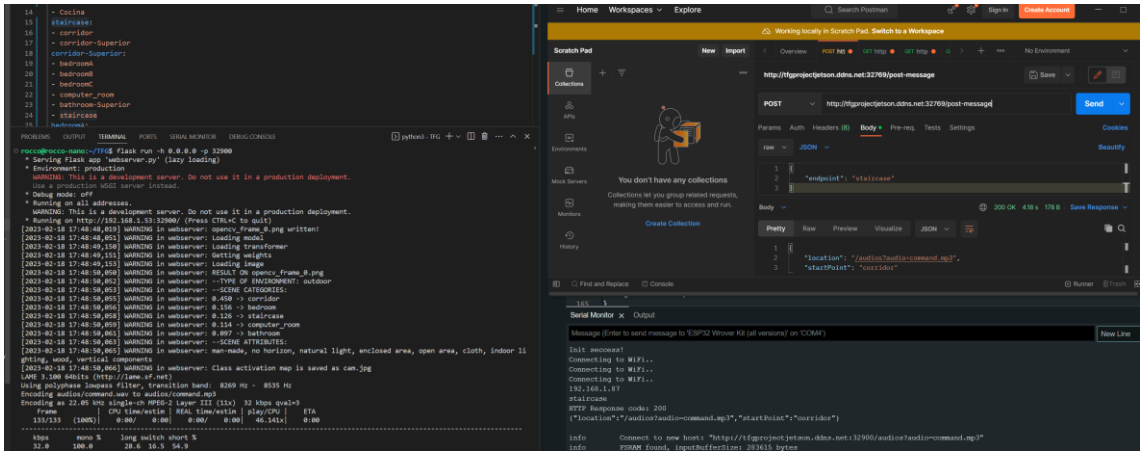

Figura 31 Petición /post-message

Petición desde la placa ESP32 al dispositivo Jetson Nano pidiendo que calcule nuestra ubicación:

Mediante la herramienta Postman y la petición HTTP a la dirección http://tfgprojectjetson.ddns.net:32900/capture se verifica el correcto functionamiento de la conexión recibiendo como resultado un diccionario JSON con con las claves “location”y “startpoint”, tomando como valores y siguiendo el mismo orden: la petición HTTP para localizar

el fichero de audio generado y el lugar predicho en el que nos encontramos a raíz de la imagen capturada desde el servidor de vídeo por streaming. Esta prueba unitaria ha sido superada con éxito.

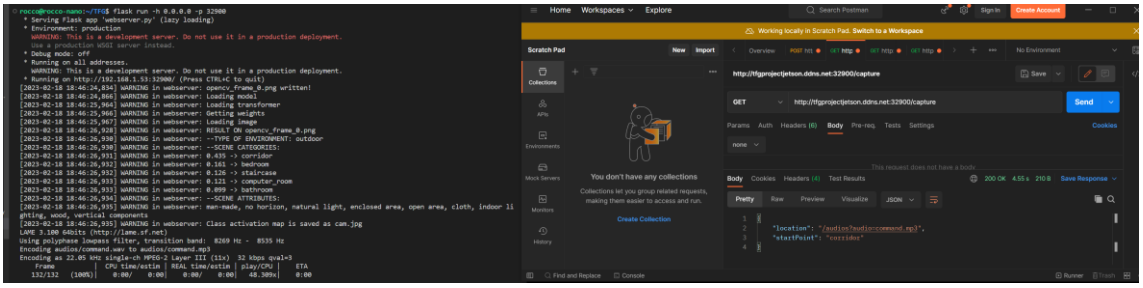

Figura 32 petición /capture

Petición desde la placa ESP32 al dispositivo Jetson Nano pidiendo que devuelva el audio con nuestra ubicación:

Mediante la herramienta Postman y la petición HTTP a la dirección http://tfgprojectjetson.ddns.net:32900/audios?audio=command.mp3 se verifica el correcto functionamiento de la conexión recibiendo como resultado un fichero de audio en formato mp3. Este fichero de audio puede también ser accedido y reproducido desde cualquier navegador web. Esta prueba unitaria ha sido superada con éxito.

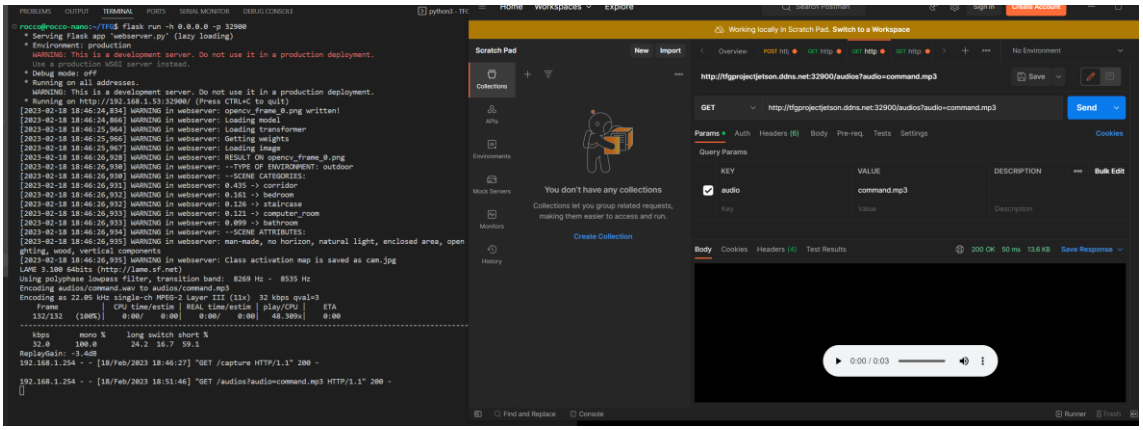

Figura 33 petición/audios?audio=[name]

Petición desde la placa ESP32 al dispositivo Jetson Nano pidiendo que calcule nuestra ruta una vez calculada nuestra ubicación:

Mediante la herramienta Postman y la petición HTTP a la dirección http://tfgprojectjetson.ddns.net:32900/guide?startPoint=corridor&endPoint=staircase se verifica el correcto functionamiento de la conexión recibiendo como resultado un diccionario JSON con la clave “guide” y como valor la dirección relativa sobre la que debe hacerse una petición HTTP para obtener el fichero de audio con la ruta guiada. Se comprueba, además, que el algoritmo Dijkstra ha calculado el camino más corto. Esta prueba unitaria ha sido superada con éxito.

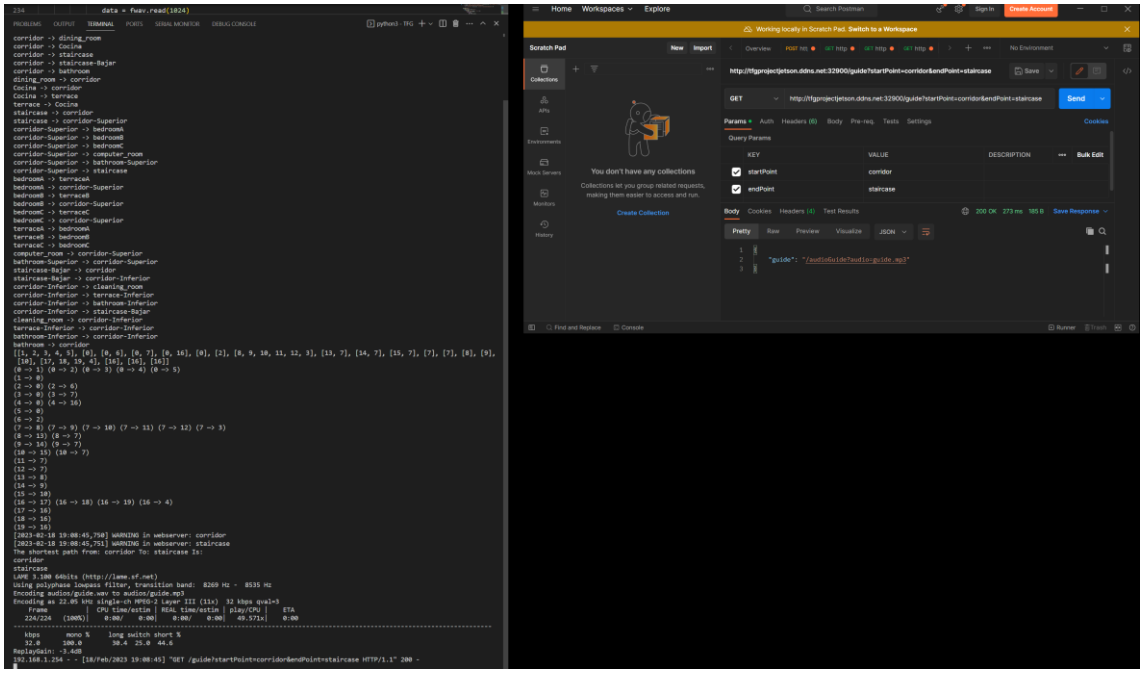

Figura 34 Cálculo ruta guíada

Petición desde la placa ESP32 al dispositivo Jetson Nano pidiendo que genere el audio con nuestra ruta guiada:

Mediante la herramienta Postman y la petición HTTP a la dirección http://tfgprojectjetson.ddns.net:32900/audioGuide?audio=guide.mp3 se verifica el correcto functionamiento de la conexión recibiendo como resultado un archivo de audio que contiene la ruta guiada desde nuestra ubicación hasta nuestro destino, este fichero puede reproducirse desde postman y puede también ser reproducido desde cualquier navegador web. Esta prueba unitaria ha sido superada con éxito.

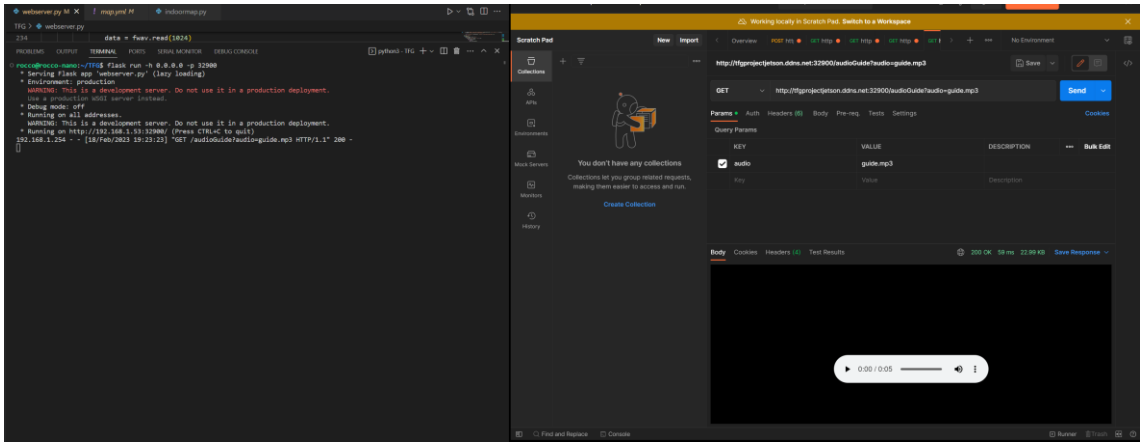

Figura 35 Petición /audioGuide?audio

Pruebas de integración:

Comunicación entre los dos microcontroladores ESP32 y la Jetson Nano.

Como puede observarse en la Figura 35, se realiza la petición http://tfgprojectjetson.ddns.net:32900/capture desde la placa ESP32 y se recibe en el

dispositivo Jetson Nano. En el Monitor serie podemos ver la respuesta HTTP Response Code: 200 y una cadena de texto correspondiente a un tipo de dato JSON con las claves “location”y “startpoint”, tomando como valores y siguiendo el mismo orden: la petición HTTP para localizar el fichero de audio generado y el punto predicho en el que nos encontramos.

Este dato es devuelto por la función tal como puede observarse en la siguiente línea de código. El diccionario seguido por el código de respuesta HTTP 200 (OK). Esta prueba de integración ha sido superada con éxito.

```
return {"startPoint": classes[idx[0]], "location":f"/audios?audio=command.mp3"}, 200
```

Como puede observarse en la Figura 36, se realiza la petición http://tfgprojectjetson.ddns.net:32900 + audioName, donde audioName toma el valor “/audios?audio=command.mp3”. De esta forma la petición se realiza sobre la URL http://tfgprojectjetson.ddns.net:32900/audios?audio=command.mp3 en forma de conexión al servidor streaming de audio.

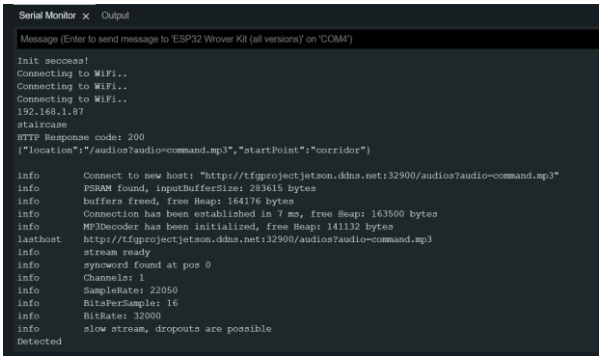

Figura 36 Conexión al servidor streaming de audio

La figura 18 ilustra la conexión y la correcta devolución del audio en streaming. Esta prueba de integración ha sido superada con éxito.

Pruebas funcionales:

Enviar petición mediante comando de voz y obtener las instrucciones auditivas correspondientes a la localización y el recorrido más corto.

Tal y como puede ser visualizado en el archivo adjunto al presente trabajo “testFuncional.mp4” se ha realizado una última prueba para verificar el correcto funcionamiento del prototipo. En esta prueba se ha iniciado el comando de voz “Ok, Google. Activate staircaise” dirigido al asistente de Google de mi teléfono inteligente, el asistente ha activado la aplicación con nombre de escena staircaise y dicha aplicación ha realizado una petición HTTP de tipo POST a la microcontroladora ESP32 encargada de comunicarse con la Jetson Nano. Esta, a su vez ha realizado la captura de la imagen una vez emitida la comunicación por parte de la primera placa ESP32 desde el servidor web de vídeo por streaming localizado en la segunda placa ESP32 y, una vez en posesión de la imagen ha realizado una conversión de la imagen a formato PIL para su clasificación por el modelo de Machine Learning y su manipulación para generar una nueva imagen con el mapa de calor. Posteriormente, se ha devuelto la respuesta a la primera placa ESP32 con la ubicación, se ha trazado la guía auditiva para llegar al destino y se han reproducido los distintos audios generados por ordenador.

Asignaturas relacionadas con el presente trabajo

Dada la cantidad de conocimientos adquiridos a partir de asignaturas de la titulación he considerado oportuno incluir un espacio en específico para poner en valor lo aprendido a lo largo de la titulación, algo que ha sido clave para la realización de este trabajo y memoria.

Fundamentos de organización de empresa, Gestión de proyectos

La planificación del tiempo, el análisis del proyecto, así como de los riesgos y soluciones, la delimitación del proyecto y la puesta en común con mi compañero de la titulación del Grado en Ingeniería en Diseño Industrial y Desarrollo de Productos.

Deontología y profesionalismo y el NICCoLLa project

Tanto la asignatura de Deontología y profesionalismo como la participación en el NICCoLLa Project han sido claves para caer en la relevancia de los aspectos legales y éticos en la implementación de soluciones que pueden poner en peligro los datos y las vidas de los usuarios, además de tratar de generar proyectos que ayuden tanto a las personas como a poner en valor la función de nuestra profesión a la hora de mejorar la salud y el bienestar.

Matemáticas discretas, análisis matemático y Teoría de autómatas y lenguajes formales

El poder hacer uso de las matrices de adyacencia, la interacción con grafos o la elección del algoritmo Dijkstra para el cálculo del camino más corto. Sin ambas asignaturas no habría tenido los conocimientos necesarios para poder llevar a cabo una parte importante de este trabajo.

Sistemas inteligentes, Machine Learning en entornos industriales, Tecnologías para sistemas inteligentes

El uso de herramientas como Google Colab, conocer las librerías para poder utilizar y desarrollar modelos de aprendizaje automático, hasta la elección de la Jetson Nano para el prototipado ha sido en gran medida debido a estas asignaturas.

Automatización y robótica para la industria digital, Internet de las cosas (IoT), Diseño, configuración y evaluación de los sistemas informáticos

No limitarse a dispositivos Arduino, sino poder hacer uso de otro tipo de microcontroladores, conocer el funcionamiento de los conversores digital a analógico o analógico a digital e implementarlos para la reproducción de sonido. Conexiones I2S o I2C. Las problemáticas y limitaciones encontradas durante el desarrollo del proyecto han sido solucionadas y afrontadas con éxito gracias a los conocimientos adquiridos en estas asignaturas.

Redes de computadores, Tecnología de sistemas informáticos en red, Cloud computing, Redes en la Industria 4.0, Seguridad en los sistemas informáticos, Sistemas de almacenamiento y procesado distribuido, Tecnología de redes

La gestión de las redes, la asignación de un DNS dinámico (DDNS) para solventar el problema de la conexión desde el exterior, la administración de puertos. El tunelado SSH de las conexiones para poder trabajar desde España cuando el servidor de la Jetson Nano estaba en Francia y no había forma alguna de acceder al puerto 32900. La propuesta de hacer uso de routers Microtik para la triangulación de la señal dado que son capaces de indicarte los DB en su servidor web. El uso de REST API para la conexión de los componentes o incluso la posibilidad de gestionar las

peticiones como eventos mediante topics de KAFKA o colas de Rabbit MQ, entre otros y ser enviados a un servidor de procesamiento por streaming como Apache Spark Streaming para realizar las operaciones.

Visión por computador

Gran parte del trabajo abordado en este proyecto ha sido gracias a los conocimientos adquiridos en esta asignatura. Como el uso de la librería OpenCV y el tratamiento de imágenes.

Introducción a la informática y a la programación, Programación, Fundamentos de computadores, Tecnología de computadores, Concurrencia y sistemas distribuidos, Estructura de Computadores, Arquitectura e ingeniería de computadores, Estructura de Datos y Algoritmos, Lenguajes, tecnologías y paradigmas de la programación, Fundamentos de Sistemas operativos y Computación Paralela.

De forma general en menor o mayor medida los conocimientos obtenidos en el resto de las asignaturas han sido importantes para el desarrollo del trabajo.

Conclusiones

Pese a las limitaciones técnicas, económicas, espaciales y temporales, considero que he cumplido todos los objetivos propuestos en el trabajo y que he desarrollado un prototipo funcional que puede ser tomado como base para desarrollar una industrialización de la solución y hacerla totalmente funcional para el usuario.

La solución ha sido diseñada para funcionar con o sin el prototipo y todos los elementos en el flujo son perfectamente intercambiables gracias al uso de las peticiones HTTP para realizar la interconexión entre ellos.

Este proyecto me ha servido no solo para asentar los conocimientos adquiridos durante el curso de la titulación, sino también para ponerlos en práctica por mi mismo con la ayuda de mi tutor del TFG.

Aunque creo que el prototipo cumple los objetivos, considero que la solución dista mucho de ser la más óptima o la mejor. Cumples los objetivos y su propósito, pero es dependiendiente de aplicaciones de terceros para su funcionamiento y no permite el uso simultaneo de sus servicios por más de un usuario. En este sentido y como propuesta de mejor habría que plantear un servidor multi hilo o incluso el procesamiento por streaming. Además, cabe la posibilidad de implementar otro modelo de aprendizaje automático para la detección y clasificación de objetos dentro de la escena que puedan suponer un peligro para el usuario o que impidan el correcto seguimiento de la guía, por ejemplo unas escaleras bloqueadas.

Hay que indicar además que durante el primer planteamiento del proyecto quise hacer uso de una solución combinada para el cálculo de la ubicación relativa del usuario. Debido a las limitaciones anteriormente mencionadas no he podido realizarlo, sin embargo, considero no solo que es posible, sino que con los recursos suficientes y con el tiempo debido podría haberse implementado una triangulación de la posición del usuario mediante el uso de redes WIFi o podría haberse implementado la solución propuesta por (Sottile, F) junto a la solución propuesta en el presente proyecto.

Por último, aunque no menos importante, considero que este proyecto cumple los Objetivos de Desarrollo Sostenible 9, al Impulsar el desarrollo de investigaciones y propuestas innovadoras en torno a la discapacidad visual; 10, al contribuir a la independencia de las personas con discapacidad visual y 11, al prestar apoyo para que los espacios comunitarios sean accesibles en igualdad de oportunidades.

Bibliografía

Alkhawaja, F. &. (2019). *Techniques of Indoor Positioning Systems (IPS).* 10.1109/ICASET.2019.8714291.

Anders Grunnet-Jepsen, A. T. (n.d.). External Synchronization of Intel® RealSense™ Depth cameras. *Intel RealSense*.

Catasús Llena, O. (2022). Trajectes augmentats per a persones amb mobilitat reduïda. *UPCommons*.

Mendoza-Silva GM, T.-S. J. (2019). A Meta-Review of Indoor Positioning Systems. *Sensors*, 19(20):4507.

Microchip Technology, Inc. (2021). *microchipdeveloper.* Retrieved from https://microchipdeveloper.com/32arm:samd21-i2s-overview

Miles, L. H. (2018, 12 03). *Dijkst*. Retrieved from https://gist.github.com/qpwo/cda55deee291de31b50d408c1a7c8515

Morar A, M. A. (2020). A Comprehensive Survey of Indoor Localization Methods Based on Computer Vision. *Sensors* , 6;20(9):2641.

NVIDIA. (n.d.). *Getting Started with Jetson Nano Developer Kit*. Retrieved from Getting Started with Jetson Nano Developer Kit

Q-engineering. (2022, 12 21). *Install PyTorch on Jetson Nano.* Retrieved from https://qengineering.eu/install-pytorch-on-jetson-nano.html

Sottile, F. &. (2017). A Hybrid Localization Algorithm for Wearable Safety Devices. *ACM*, 10.4108/eai.15-12-2016.2267783.

Yassin, A. N.-D. (2017). *Recent Advances in Indoor Localization: A Survey on Theoretical Approaches and Applications.* IEEE Communications Surveys & Tutorials.