Universidad Miguel Hernández de Elche

MÁSTER UNIVERSITARIO EN ROBÓTICA

"Redes neuronales con convoluciones en el dominio de Fourier para la localización de robots móviles"

Trabajo de Fin de Máster

Curso académico 2023-2024

Autor: Marcos Alfaro Pérez

Tutores: Luis Payá Castelló
María Flores Tenza

Agredecimientos

Este trabajo ha sido financiado con la Ayuda para estudios de Máster de la Fundación ValgrAI. Asimismo, este trabajo forma parte del proyecto TED2021-130901B-I00, financiado por MCIN/AEI/10.13039/501100011033 y por la Unión Europea "Next-GenerationEU"/PRTR.

Contenido

Lista de Figuras

Lista de Tablas

1 Introducción

Los robots móviles son vehículos autónomos que se desplazan por un entorno mientras realizan una tarea. Hoy en día, este tipo de robots se utiliza para todo tipo de aplicaciones, como por ejemplo la exploración de entornos hostiles o inaccesibles para el ser humano (Figura **1-1**), la asistencia a personas (Figura **1-2**) o la realización de tareas de vigilancia y seguridad (Figura **1-3**).

Figura 1-1: Robot Curiosity utilizado por la NASA para la exploración del planeta Marte https://www.nasa.gov/.

Figura 1-2: Robot camarero Bellabot diseñado por Pudu Robotics https://www.pudurobotics.com.

Un robot móvil debe llevar a cabo una serie de tareas básicas para poder navegar de forma segura por el entorno que le rodea. En primer lugar, un robot necesita construir un mapa del entorno (mapping). Además, el robot debe estimar su pose dentro de dicho mapa (localización). En la mayoría de situaciones, el robot se encuentra originalmente en un entorno desconocido, por lo que debe realizar ambas tareas de forma simultánea (SLAM: Simultaneous Localization and Mapping). Por último, el robot debe ser capaz de planificar una trayectoria desde su posición actual hasta la posición deseada, determinando cuál es el camino óptimo y evitando los obstáculos presentes en el entorno (path planning).

Para poder extraer información del entorno, un robot debe ir equipado con una

Figura 1-3: Robot móvil S5 Series de SMP Robotics Systems empleado para tareas de vigilancia y seguridad https://smprobotics.com/.

serie de sensores. En primer lugar, los sensores de posicionamiento relativo estiman el desplazamiento del robot a partir del desplazamiento angular de sus ruedas (encoders, tacómetros) o a partir de las fuerzas ejercidas sobre el robot (acelerómetros, giróscopos, IMUs). Estos sensores presentan un error acumulativo que debe ser corregido con el uso de otro tipo de sensores. Mediante los sensores de posicionamiento absoluto, un robot es capaz de estimar su posición absoluta dentro del mapa, a partir de varias medidas de distancia o de ángulo (bearing) respecto a una serie de balizas o satélites (GPS) cuya posición es conocida. Sin embargo, los sensores de balizas suelen ser invasivos y el GPS funciona adecuadamente solo en entornos abiertos. Los sensores de rango o distancia se utilizan para medir la distancia a los objetos presentes en el entorno basándose en el concepto de tiempo de vuelo: miden el tiempo transcurrido desde que se emite una señal ultrasónica (sonar) o electromagnética (lidar) y rebota en un objeto de la escena hasta que incide en el sensor receptor. Estos sensores son muy precisos, pero suelen ser demasiado caros comparado con otros tipos de sensores, como las cámaras.

Los sensores de visión, en cambio, permiten obtener información rica del entorno, como colores, formas o texturas, a un bajo coste. Dentro de este grupo, podemos encontrar diferentes tipos:

- **Cámara estándar**: se trata de un sistema formado por una única cámara. Puede recabar información abundante de la escena, a partir de la cual se pueden abordar tareas como la segmentación o la detección de objetos y personas. Sin embargo, con este tipo de sistemas es muy complicado obtener información tridimensional de la escena.

- **Par estereoscópico**: sistema formado por dos cámaras cuya posición y orientación relativa es conocida. De esta forma, se puede recuperar la información de profundidad de los elementos presentes en la escena mediante el principio básico de triangulación.

- **Cámaras omnidireccionales**: este tipo de cámaras extrae información de la escena con un campo de visión de 360^{o}. Esta propiedad les permite obtener la misma información del entorno independientemente de la orientación del robot. La visión omnidireccional se puede obtener de distintas formas: mediante un sistema multicámara, mediante un sistema catadióptrico o mediante un par de cámaras ojo de pez.

En este trabajo, se ha empleado un sensor de visión debido a su gran versatilidad y su capacidad de obtener información abundante del entorno con un número reducido de imágenes. La opción escogida para abordar el problema de localización de un robot móvil ha sido un sistema de visión catadióptrico con un espejo hiperbólico (Figuras **1-4** y **1-5**).

Figura 1-4: Robot móvil Husky A-200 ClearPath Robotics.

Figura 1-5: Cámara omnidireccional con un sistema catadióptrico.

En cuanto a la descripción de imágenes, existen dos grandes líneas de investigación (véase la Figura **1-6**). Por un lado, la descripción global consiste en la extracción de características a partir de la información general de la imagen. Por otro lado, la descripción basada en puntos característicos se basa en la obtención de información de puntos fácilmente distinguibles en una imagen, como los bordes o las esquinas.

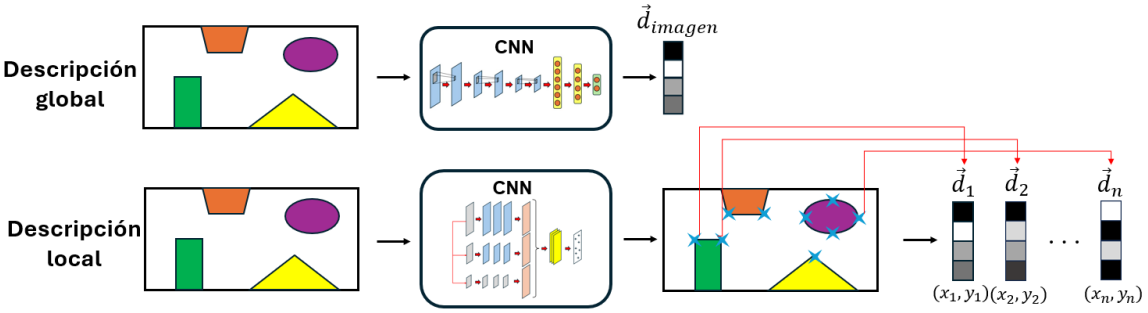


Figura 1-6: Métodos de descripción de imágenes: global y basada en puntos característicos.

Con el aumento de la capacidad de cómputo de los ordenadores actuales, la Inteligencia Artificial (IA) se ha convertido en uno de los campos con mayor potencial de desarrollo. La IA consiste en el diseño de programas de computador utilizados para resolver problemas que normalmente requieren de un cierto grado de inteligencia. En la actualidad, se utiliza para todo tipo de aplicaciones, tales como la detección y el reconocimiento de personas y objetos, la traducción de texto y audio o el diagnóstico prematuro de enfermedades, entre otras.

Dentro de las técnicas de IA, destacan las redes neuronales. Las redes neuronales son estructuras de procesamiento de información que tratan de imitar el funcionamiento del cerebro humano, cuya inteligencia surge de la interacción de millones de neuronas. Para realizar correctamente la tarea deseada, una red neuronal debe ser entrenada. Este proceso consiste en el ajuste de los pesos sinápticos de cada neurona a partir de una secuencia de datos de entrada.

Las redes neuronales unidireccionales o feedforward son las más utilizadas, compuestas por varias capas de neuronas, en las que la información se transmite de atrás hacia adelante (Figura **1-7**). En otras palabras, cada neurona recibe la información de todas las neuronas de la capa anterior y transmite la información a todas las neuronas de la capa siguiente. Este tipo de herramientas se emplean para resolver problemas de clasificación o de regresión de datos.

Sin embargo, las redes unidireccionales no son las más adecuadas cuando los datos de entrada son imágenes, por dos motivos: en primer lugar, si el tamaño de las imágenes es relativamente grande, el número de parámetros a ajustar por la red es demasiado elevado, lo que supone un coste computacional prácticamente inabordable; en segundo lugar, este tipo de redes no conserva la relación espacial entre los píxeles de la imagen. Por tanto, es necesario emplear otras herramientas.

Dentro de la Inteligencia Artificial, las técnicas de Machine Learning se utilizan para realizar tareas de asociación de datos a partir de una serie de características extraídas previamente. En cambio, en el aprendizaje profundo o Deep Learning, se utilizan redes neuronales capaces de extraer características de forma automática a partir de los datos de entrada.

En cuanto al aprendizaje profundo, destacan las redes neuronales convolucionales o CNNs (Figura **1-8**). Las CNNs son capaces de extraer características a partir de imágenes con un nivel de abstracción elevado. Este tipo de redes contiene capas convolucionales, que consisten en filtros basados en la operación de convolución (Figura **1-9**). De esta forma, se reduce significativamente el número de parámetros a ajustar por la red, y además se conservan las relaciones de vecindad entre los píxeles. Por tanto, se trata de herramientas muy útiles para realizar la descripción de imágenes.

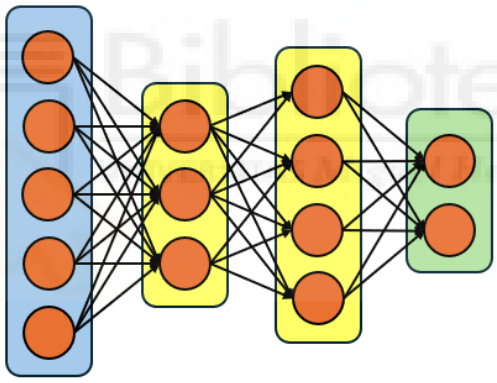


Figura 1-7: Estructura básica de una red neuronal unidireccional (feedforward).

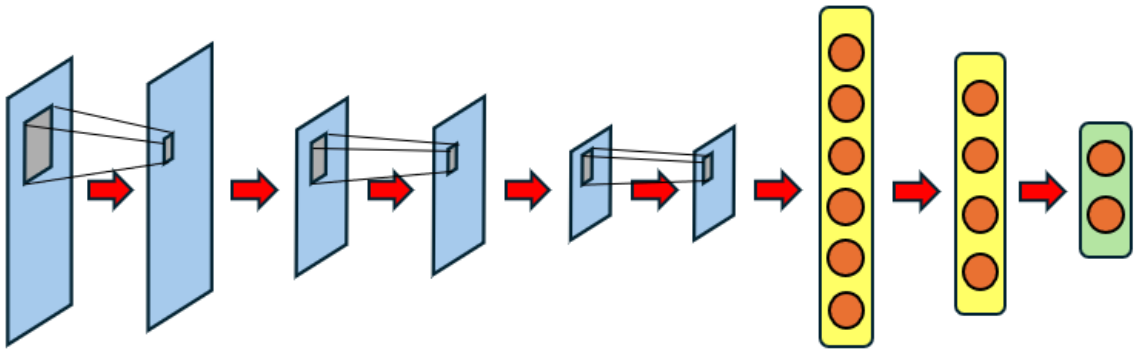


Figura 1-8: Estructura básica de una red neuronal convolucional (CNN).

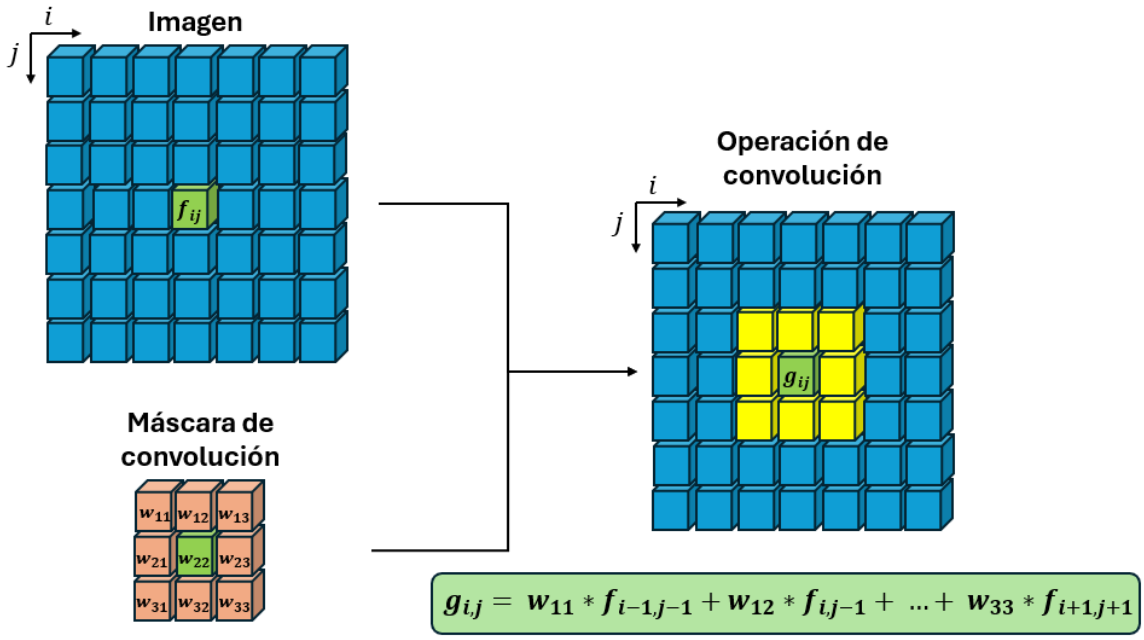


Figura 1-9: Operación de convolución discreta aplicada sobre una imagen.

Respecto a la operación de convolución, puede realizarse en el dominio espacial o en el dominio frecuencial. Las CNNs clásicas están formadas por capas que aplican convoluciones rectangulares sobre la imagen de entrada en el dominio espacial. Sin embargo, en los últimos años se ha empezado a utilizar convoluciones en el dominio de Fourier. Este tipo de convoluciones se aplican tras convertir la imagen al dominio de la frecuencia mediante la Transformada Discreta de Fourier. Como se trata de la misma operación matemática, conserva las mismas propiedades básicas que la convolución rectangular. Sin embargo, la convolución en el dominio de Fourier presenta una serie de ventajas, como una mayor eficiencia o un mayor campo receptivo. Por este motivo, presentan un gran potencial para abordar la extracción de características en imágenes.

Uno de los problemas de la operación de convolución es la pérdida de información en los bordes de la imagen, ya que al situar la máscara de convolución sobre estos bordes, algunos elementos de la máscara caen fuera de la imagen. Por tanto, no se puede realizar la operación de convolución en estos píxeles, y se obtiene una imagen con dimensiones reducidas. Para evitar este problema, es frecuente realizar un padding o relleno en los bordes de la imagen. El tipo de padding más común es el relleno con ceros, pero también existen otros tipos como el circular o el transparente, que pueden ser más adecuados cuando las imágenes cumplen características especiales, como es el caso de las imágenes panorámicas.

Recientemente, otros trabajos han empezado a utilizar arquitecturas compuestas por varias CNNs, como por ejemplo las redes siamesas o tripletas, entre otras. Las redes siamesas están compuestas por dos redes que funcionan en paralelo y son idénticas, es decir, comparten las mismas capas con el mismo número de neuronas y sus respectivos pesos y umbrales, donde cada una de las ramas recibe una imagen de entrada y obtiene su salida correspondiente. Esto permite a este tipo de redes aprender relaciones de similitud o diferencia entre pares de imágenes.

Asimismo, las redes tripletas están formadas por tres CNNs idénticas que funcionan de forma paralela. De la misma forma que las redes siamesas, reciben tres imágenes de entrada, comúnmente llamadas ancla, positiva y negativa, y cada rama obtiene sus respectivas salidas. Estas redes son capaces de aprender de forma simultánea relaciones de similitud entre las imágenes ancla y positiva y relaciones de diferencia entre la imagen negativa y las dos anteriores. En la Figura **1-10** se muestra la estructura básica de una red tripleta.

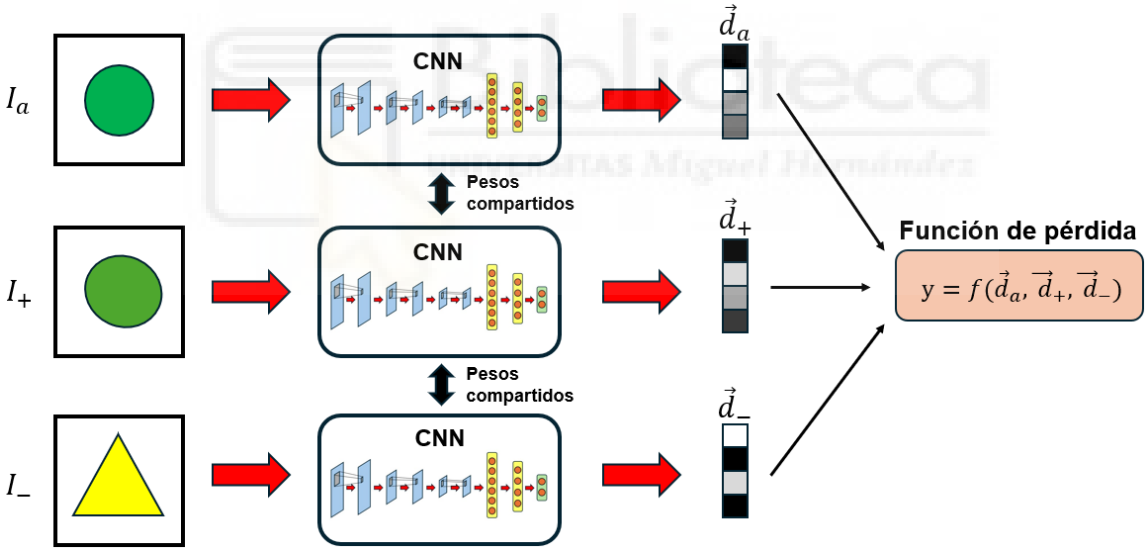


Figura 1-10: Estructura básica de una red tripleta.

Durante el entrenamiento de una red neuronal, la función de pérdida se encarga de estimar el error cometido en las predicciones realizadas por la red. En función del valor devuelto por la función de pérdida, el algoritmo de optimización modificará los pesos de la red en mayor o menor medida, con el fin de corregir el error en las predicciones futuras. Cuando las predicciones son precisas, este valor será cercano a cero. Asimismo, durante el entrenamiento de una red tripleta, la función de pérdida de triplete deberá minimizar su valor cuando las salidas obtenidas a partir de los

datos de entrada ancla y positivo sean similares, y la salida correspondiente al dato negativo sea distinta a las dos anteriores.

En este trabajo se ha propuesto el uso de CNNs que incluyen convoluciones en el dominio de Fourier para abordar diferentes tareas relacionadas con la localización de un robot móvil, como son la clasificación de imágenes en estancias (room retrieval) y el reconocimiento de lugares previos visitados por el robot (place recognition). Para el entrenamiento de las redes propuestas, se ha empleado una arquitectura tripleta. Adicionalmente, se ha explorado el uso de distintos tipos de padding en la operación de convolución.

El resto del trabajo se estructura de la siguiente manera. En la sección 2 se revisan los trabajos relacionados con la temática que se han realizado hasta la fecha. En la sección 3 se describen las herramientas empleadas, mientras que en la sección 4 se detalla el método desarrollado. La sección 5 recoge los experimentos realizados y los resultados obtenidos para cada uno de ellos. Por último, en la sección 6 se comentan las conclusiones extraídas y se proponen posibles líneas de investigación futuras.

2 Estado del arte

Actualmente, los robots móviles se emplean en una gran variedad de aplicaciones, y en consecuencia múltiples trabajos se han centrado en abordar sus tareas fundamentales. Por ejemplo, Román et al. [41] realizaron un mapping jerárquico mediante técnicas de clustering incremental. Asimismo, Balaska et al. [3] llevaron a cabo una localización visual mediante algoritmos de clustering.

Frecuentemente, un robot móvil debe realizar una tarea en un entorno desconocido en el que no conoce su posición. Por este motivo, la literatura se ha centrado en resolver el problema de SLAM (Teed and Deng [48], Zhu et al. [54]). Algunos trabajos, tales como Chen et al. [8], trataron la planificación de trayectorias de vehículos aéreos no tripulados (UAVs).

Para obtener información de la escena, los sensores de visión son una de las opciones más habituales en los robots móviles. Por ejemplo, Xiao et al. [52] emplearon un sistema de visión monocular para abordar el problema de SLAM en entornos dinámicos. Del mismo modo, Schönberger et al. [44] llevaron a cabo una localización visual semántica mediante el uso de un autoencoder.

Como se ha comentado en la sección 1, las cámaras omnidireccionales extraen información abundante del entorno con un FOV (field of view) de 360º, que se puede obtener de distintas formas: mediante un sistema multicámara [20], mediante un sistema catadióptrico [28] o mediante un par de imágenes ojo de pez [15]. El principal inconveniente de este tipo de cámaras es la elevada distorsión que presentan, motivo por el cual algunos autores han tratado de modelar esta distorsión [43].

Una vez que se ha capturado la información del entorno, ésta debe ser procesada para poder ser interpretada correctamente. En lo que respecta a la descripción de imágenes, existen dos enfoques principales en la literatura. Por un lado, algunos trabajos emplean descriptores obtenidos a partir de puntos característicos. Por ejemplo, Luo et al. [30] desarrollaron una red neuronal capaz de obtener descriptores invariantes a rotación y escalado a partir de los puntos característicos de la imagen de entrada. Por otro lado, Payá et al. [36] y Cebollada et al. [7] trataron los problemas de mapping y localización mediante descriptores de apariencia global, respectivamente.

Asimismo, Hausler et al. [19] combinaron ambas técnicas para resolver el problema de localización.

Anteriormente, la obtención de estos descriptores se realizaba mediante técnicas analíticas, tales como la firma de Fourier, HOG o gist [35]. Sin embargo, el aumento de la capacidad de cómputo de los ordenadores actuales ha provocado el auge de las técnicas basadas en aprendizaje profundo, dando lugar a las redes neuronales convolucionales.

Las CNNs fueron propuestas por primera vez por LeCun et al. [25]. Posteriormente, otros trabajos desarrollaron arquitecturas más elaboradas, compuestas por un mayor número de capas, tales como AlexNet [24], GoogLeNet [46] o VGG [45], todas ellas entrenadas para clasificar imágenes entre 1000 objetos distintos con la base de datos ImageNet [11].

También existen redes convolucionales que, en vez de obtener descriptores globales, extraen puntos característicos y el descriptor de cada uno de ellos. Dentro de este grupo, destacan SuperPoint [12] y D2Net [13]. Otras redes, en cambio, fueron entrenadas para realizar la tarea de detección de objetos, una tarea crítica para la navegación de vehículos autónomos. En este campo, la red más destacada es YOLO (You Only Look Once) [40]. No obstante, las CNNs no se emplean únicamente para el procesamiento de imágenes. También existen arquitecturas capaces de procesar nubes de puntos 3D capturadas con sensores lidar, tales como PointNet [39] o MinkUNeXt [4], entre otras.

Debido a su capacidad para procesar imágenes y extraer características a partir de ellas, las CNNs se emplean a menudo para resolver las tareas de creación de mapas y localización visual. Por ejemplo, Foroughi et al. [16] llevaron a cabo una localización en un entorno de interior, mientras que Neubert and Protzel [32] hicieron lo mismo en un entorno de exterior. Alternativamente, Zhang et al. [53] abordaron el problema de Graph-SLAM mediante características extraídas a partir de las imágenes con una CNN.

En cuanto al uso de CNNs con imágenes omnidireccionales, algunos trabajos, tales como Cabrera et al. [5] o Rostkowska and Skrzypczyński [42], realizaron una localización jerárquica en un entorno de interior con imágenes obtenidas con un sistema catadióptrico. Además, Wang et al. [50] desarrollaron una CNN adaptada a la distorsión presente en las imágenes omnidireccionales. En su trabajo, Won et al. [51] emplean una CNN para obtener mapas de profundidad a partir de imágenes ojo de pez. Asimismo, Jayasuriya et al. [22] hacen uso de una CNN capaz de medir la

orientación relativa o bearing respecto a objetos presentes en la escena. Sin embargo, estos últimos enfoques se apoyan en otras técnicas para abordar el problema de localización, como la odometría visual o el filtro de Kalman extendido.

Tradicionalmente, las CNNs se han construido con capas convolucionales que realizan esta operación en el dominio espacial. No obstante, cada vez más estudios utilizan convoluciones en el dominio de Fourier [37] para el diseño de arquitecturas más eficientes destinadas a tareas de clasificación de imágenes. Por un lado, Kent et al. [23] proponen una CNN para procesar datos de electrocardiogramas en el dominio frecuencial con el objetivo de detectar patologías cardiovasculares. Por otro lado, Han and Hong [18] incorporaron este tipo de convoluciones en arquitecturas sencillas de tipo LeNet, mientras que Nair et al. [31] emplearon la transformada rápida de Fourier (FFT) para acelerar el proceso de convolución con una arquitectura de tipo U-Net. Asimismo, Chi et al. [10] proponen una arquitectura de tipo ResNet en la que mezclan convoluciones en los dominios espacial y temporal a distintas escalas. No obstante, este tipo de convoluciones no han sido muy exploradas hasta la fecha para la tarea de localización visual.

En los últimos años, otros trabajos han explorado el uso de arquitecturas más complejas, dando lugar a las redes siamesas [47] y a las redes tripletas [34], entre otras. Las redes siamesas están compuestas por dos CNNs idénticas que funcionan en paralelo, y son capaces de aprender relaciones de similitud o diferencia a partir de las imágenes de entrada. Esta habilidad las convierte en una herramienta muy útil para realizar tareas como la detección de objetos [17], el seguimiento de objetos (visual tracking) [9] o la monitorización de desastres naturales en imágenes satelitales [14]. Asimismo, las redes siamesas se utilizan a menudo para abordar el problema de localización, tanto con imágenes [27] como con nubes de puntos obtenidas con sensores lidar [26].

Sin embargo, las redes tripletas van un paso más allá. Este tipo de redes, compuestas por tres CNNs, pueden aprender relaciones de similitud y diferencia de manera simultánea. Esta propiedad permite que estas redes se ajusten por igual a ejemplos positivos y negativos. Además, el número de combinaciones posibles de los datos de entrada crecen exponencialmente, por lo que un conjunto de imágenes reducido puede ser suficiente para realizar el entrenamiento de la red.

Por estos motivos, las redes tripletas se han convertido en una de las opciones más destacadas a la hora de abordar el problema de localización. Algunos trabajos, como por ejemplo Liu et al. [29] o Arandjelovic et al. [2], se centraron en el desarrollo de nuevas CNNs, las cuales fueron entrenadas con una arquitectura de red tripleta.

En cambio, otros trabajos se dedicaron al análisis de diferentes funciones de pérdida de triplete (Hermans et al. [21], Uy and Lee [49]). En su investigación, Olid et al. [33] realizaron una comparación entre arquitecturas de CNN simple, redes siamesas y tripletas en la tarea de localización, y demostraron las ventajas de emplear arquitecturas de red tripleta.

Por último, Cattaneo et al. [6] llevaron a cabo una localización cruzada, en la cual las combinaciones de datos de entrada empleadas durante el entrenamiento de la red pueden ser de naturaleza distinta (por ejemplo, el dato ancla puede ser una imagen RGB y los datos positivo y negativo pueden ser nubes de puntos obtenidas con un lidar).

En este trabajo se ha optado por el diseño de CNNs que incluyen convoluciones en el dominio de Fourier. Aunque todavía no han sido empleadas para la localización de robots móviles, pueden ser especialmente útiles debido a su capacidad de extracción de características. Para entrenar estas CNNs, se ha empleado una arquitectura de red tripleta, ya que la literatura demuestra que emplear este tipo de arquitecturas permite obtener un mejor desempeño en la tarea de localización.

3 Herramientas utilizadas

3.1. Visión omnidireccional

Como se ha comentado en las secciones anteriores, las cámaras omnidireccionales capturan información de la escena con un campo de visión de 360^{o}. Esta propiedad permite construir mapas del entorno ricos en información con un número de imágenes relativamente bajo. Además, un robot situado sobre un punto concreto del mapa puede extraer la misma información de la escena independientemente de su orientación. Por estos motivos, en este trabajo se han empleado imágenes omnidireccionales para abordar los problemas de localización visual y creación de mapas. Para aprovechar las propiedades espaciales de las convoluciones rectangulares, las imágenes empleadas se han convertido a formato panorámico.

Las vistas omnidireccionales se pueden obtener de distintas formas. En este trabajo se ha optado por un sistema catadióptrico montado en un robot móvil. Este tipo de sistemas están compuestos por una cámara estándar y por un espejo parabólico o hiperbólico, y su funcionamiento se basa en que los rayos de luz, procedentes de todas las direcciones, se reflejan en el espejo e inciden sobre el sensor CCD de la cámara, formando una imagen omnidireccional. Este proceso se muestra de forma resumida en la Figura **3-1**.

3.2. Descriptores de apariencia global

En este trabajo, la descripción de las imágenes panorámicas se ha realizado mediante un enfoque global. Este método consiste en la obtención de un descriptor que contenga las características generales de la imagen. En comparación con la descripción basada en puntos característicos, permite emplear algoritmos más sencillos, con un menor coste computacional, y no requieren la calibración de la cámara.

Los descriptores globales deben cumplir una serie de características. En primer lugar, deben ser únicos, esto es, dos imágenes distintas no pueden tener el mismo descriptor. Además, los descriptores deben ser completos y no presentar ambigüeda-

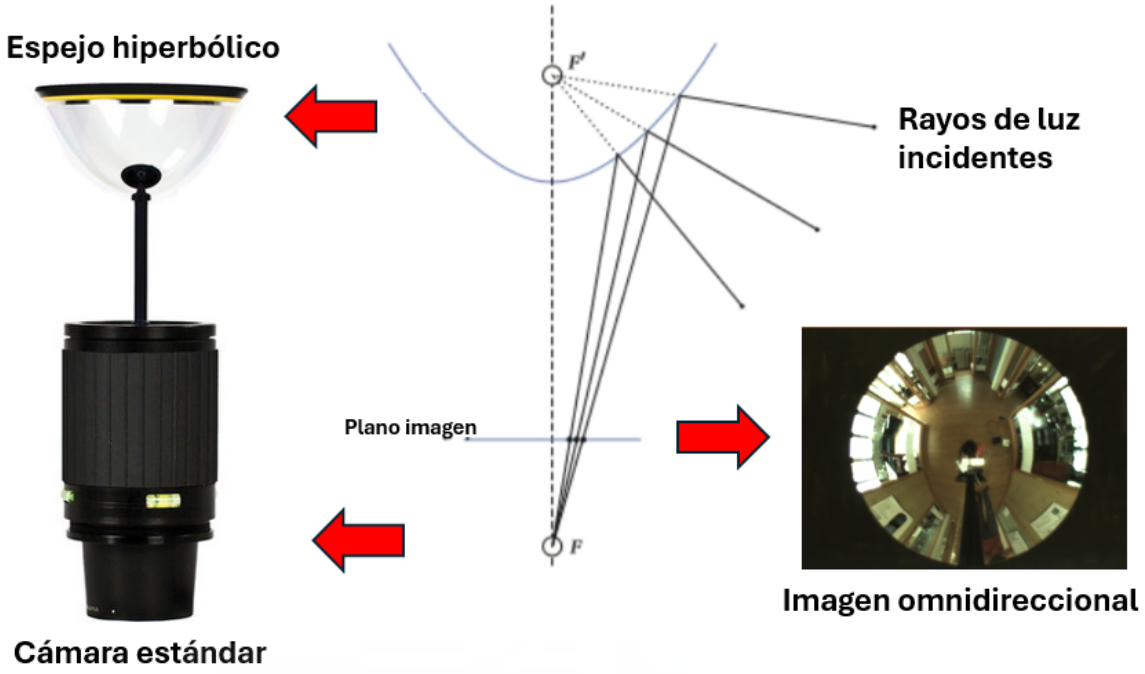


Figura 3-1: Sistema de visión catadióptrico formado por un espejo hiperbólico y una cámara estándar.

des. También deben ser relativamente invariantes a transformaciones geométricas como la rotación o el escalado. Por último, deben ser sensibles a las ligeras variaciones entre imágenes similares.

Los descriptores de apariencia global se pueden obtener mediante técnicas clásicas, entre las cuales destacan la Firma de Fourier, gist, HOG (Histogramas de Orientación del Gradiente) y Bag of Words. Sin embargo, actualmente la obtención de este tipo de descriptores se realiza mediante técnicas de aprendizaje profundo, esto es, mediante redes neuronales convolucionales. Este tipo de redes son capaces de extraer características a partir de las imágenes con un nivel de abstracción elevado. Por este motivo, en este trabajo se ha adaptado y reentrenado un modelo de CNN para obtener descriptores globales a partir de las imágenes panorámicas.

3.3. Redes neuronales tripletas

Para abordar el problema de localización visual, en este trabajo se ha empleado una CNN y se hecho uso de una arquitectura de red tripleta durante su entrenamiento. Las redes tripletas están formadas por tres CNNs idénticas que funcionan

en paralelo y comparten los mismos pesos. Este tipo de redes son entrenadas con combinaciones de tres datos de entrada, comúnmente llamados ancla, positivo y negativo. Durante su entrenamiento, la red debe ajustar sus pesos de tal forma que los descriptores correspondientes a los datos ancla y positivos sean similares, mientras que el descriptor asociado al dato negativo sea diferente a los dos anteriores. En la Figura **3-2** se muestra un esquema explicativo del proceso de entrenamiento de una CNN con una arquitectura de red tripleta.

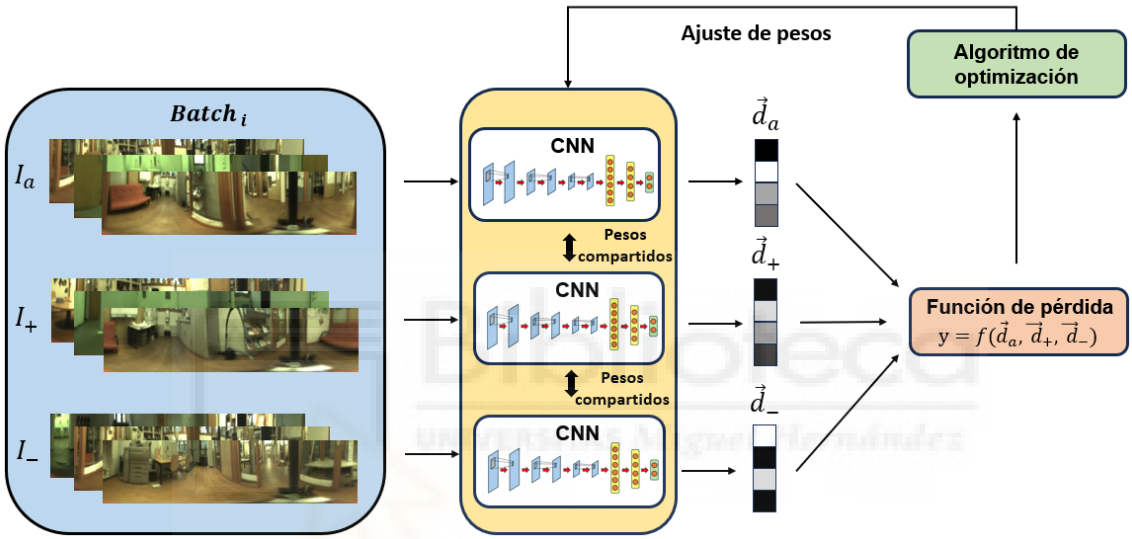


Figura 3-2: Esquema explicativo del proceso de aprendizaje de una CNN con una arquitectura de red tripleta. I_a, I_+ e I_n son las imágenes ancla, positiva y negativa, respectivamente, mientras que $\vec{d}_a$, $\vec{d}_+$ y $\vec{d}_n$ son sus respectivos descriptores.

Las redes tripletas presentan una serie de ventajas respecto a las CNNs simples y las redes siamesas. En primer lugar, el número de posibles combinaciones de datos de entrada aumenta exponencialmente, lo que permite trabajar con un conjunto de datos relativamente pequeño. Además, reciben el mismo número de ejemplos positivos y negativos, por lo que se evitan posibles sesgos durante el entrenamiento de la red. Esto las hace especialmente útiles para resolver tareas de localización y creación de mapas de robots móviles, ya que pueden ser entrenadas con combinaciones de tres imágenes escogidas de tal forma que las imágenes ancla y positiva han sido capturadas en posiciones cercanas, mientras que la imagen negativa ha sido capturada en una posición lejana.

3.4. Funciones de pérdida de triplete

En el proceso de aprendizaje de una red neuronal, la función de pérdida se encarga de estimar el error cometido en las predicciones de la red. Según la magnitud de este error, el algoritmo de optimización actualiza los pesos de la red en mayor o menor medida. Asimismo, cuando se emplea una arquitectura de red tripleta durante el entrenamiento, la función de pérdida de triplete determina el error cometido en las predicciones a partir de las relaciones de similitud y diferencia entre los tres descriptores obtenidos. Una predicción será correcta cuando los descriptores de los datos ancla y positivos sean similares y el descriptor del dato negativo sea diferente a los anteriores. En la Figura **3-3** se muestra un ejemplo de los valores devueltos por la función de pérdida tras una predicción correcta y otra predicción errónea de una red.

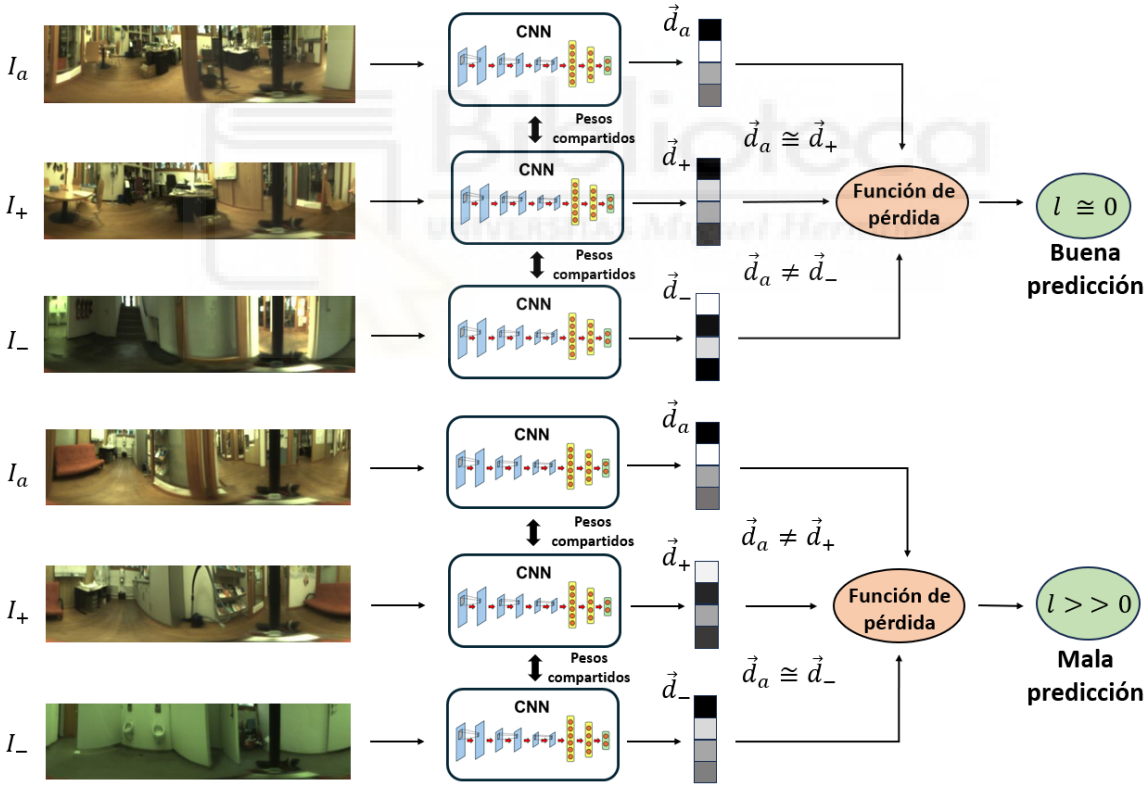


Figura 3-3: Ejemplos de valores devueltos por una función de pérdida de triplete tras una predicción correcta y otra errónea de la red.

Para acelerar el proceso de entrenamiento, es frecuente cargar los patrones de entrada en lotes o batches. La función de pérdida recibe las salidas de varias combinaciones de tres imágenes, esto es, sus descriptores, calcula el error cometido para

cada combinación y devuelve un único valor para todo el batch, que puede ser el promedio, el valor máximo, o una función más compleja. De esta forma, el ajuste de pesos se realiza tras cargar cada lote de imágenes, obtener sus descriptores y calcular el error cometido mediante la función de pérdida.

Escoger una función de pérdida u otra puede determinar el éxito o el fracaso del aprendizaje de una red neuronal. En este trabajo, se ha utilizado la función de pérdida Triplet Margin Loss, la cual obtuvo un buen desempeño en un trabajo previo [1] y ha demostrado ser bastante robusta ante distintas tareas y condiciones de entrenamiento. Esta función de pérdida viene dada por la siguiente expresión:

$$\mathcal{L} = \frac{1}{N}\sum_{i=1}^{N}[D^i_{a,p} - D^i_{a,n} + m]_+$$

donde $D^i_{a,p}$ es la distancia euclídea entre los descriptores ancla y positivo de la i-ésima combinación, $D^i_{a,n}$ es la distancia euclídea entre los descriptores ancla y negativo de la i-ésima combinación, $[...]_+$ es la función ReLU, m es el margen y N es el tamaño del batch.

3.5. Convoluciones en el dominio de Fourier

La principal novedad que aporta este trabajo es el uso de convoluciones en el dominio de Fourier. A diferencia de las convoluciones rectangulares clásicas, las convoluciones de Fourier se aplican en el dominio de la frecuencia. Este tipo de convoluciones no han sido muy exploradas hasta la fecha, pero presentan un gran potencial para realizar la extracción de características en imágenes.

Para implementar este tipo de convoluciones, nos hemos basado en el trabajo realizado por [10], en el cual proponen una CNN de tipo ResNet en la que emplean convoluciones en el dominio de Fourier para realizar distintas tareas, como por ejemplo la clasificación de imágenes y vídeos o la detección de puntos característicos. Asimismo, otros trabajos han empleado bloques de esta arquitectura para abordar otros problemas como el relleno de huecos en imágenes con oclusiones.

Concretamente, en este trabajo hemos utilizado dos bloques que forman parte de su arquitectura, llamados FourierUnit y FFC, los cuales se describen a continuación. Los experimentos realizados en [10] demuestran que al sustituir las capas convolucionales por el bloque FFC se obtienen mejores resultados que con las convoluciones rectangulares clásicas.

3.5.1. Bloque Fourier Unit (FU)

Este bloque trata de extraer las características globales de la imagen de entrada. Para ello, realiza la transformada directa de Fourier sobre la imagen, si se trata de la capa de entrada, o sobre los mapas de características, si se trata de una capa intermedia. Posteriormente, se separa la parte real de la parte imaginaria, y se realiza la operación de convolución por separado. Por último, se junta de nuevo la parte real con la imaginaria y se realiza la transformada de Fourier inversa. Este proceso se muestra detallado en la Figura **3-4**. La terminología empleada en este apartado se muestra en la Figura **3-8**.

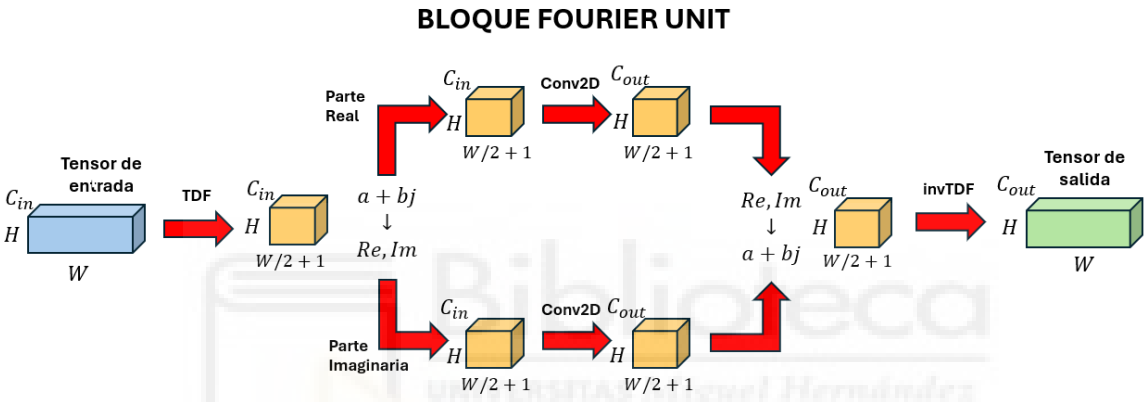


Figura 3-4: Bloque Fourier Unit.

3.5.2. Bloque Fast Fourier Convolution (FFC)

Este bloque presenta una complejidad bastante mayor que el bloque Fourier Unit. Su objetivo consiste en obtener información a partir de la imagen a escala local y a escala global simultáneamente. En la Figura **3-5** se muestra la arquitectura de este bloque.

En primer lugar, se separan los canales del tensor de entrada en función de un parámetro α_{in}, dando lugar a una rama local y a una rama global. En la rama local se realizan dos operaciones de convolución para obtener dos tensores diferentes que tienen en conjunto el número de canales de salida deseados. La proporción de canales de estos dos tensores viene dada por el parámetro α_{out}. De la misma forma, en la rama global se realiza una operación de convolución y se aplica el bloque Spectral Transformer. El resultado de estas cuatro operaciones son dos tensores con $\alpha_{out}C_{out}$ canales y otros dos tensores con $(1-\alpha_{out})C_{out}$ canales. Por último, los tensores de la misma dimensión se suman elemento a elemento, de tal forma que se produce un

intercambio de información entre ambas ramas, y luego se concatenan, para obtener un tensor de salida con las dimensiones deseadas.

El bloque Spectral Transformer forma parte del bloque FFC, y su funcionamiento se describe en la Figura **3-6**. Su objetivo principal consiste en aumentar el campo receptivo de la convolución de manera eficiente. En primer lugar, se realiza la Transformada Discreta de Fourier sobre el tensor de entrada y se divide en dos tensores, uno de ellos con $C_{in}/4$ canales de entrada y otro con el resto. El primero de ellos se introduce en el bloque Local Fourier Unit, mientras que sobre el segundo se realizan dos operaciones: la operación identidad y la realizada por el bloque Fourier Unit, descrito en apartado anterior. Tras realizar estas operaciones se obtienen tres tensores con las mismas dimensiones, los cuales se suman elemento a elemento para obtener un único tensor. Por último, se realiza la Transformada de Fourier Inversa para obtener el tensor de salida deseado.

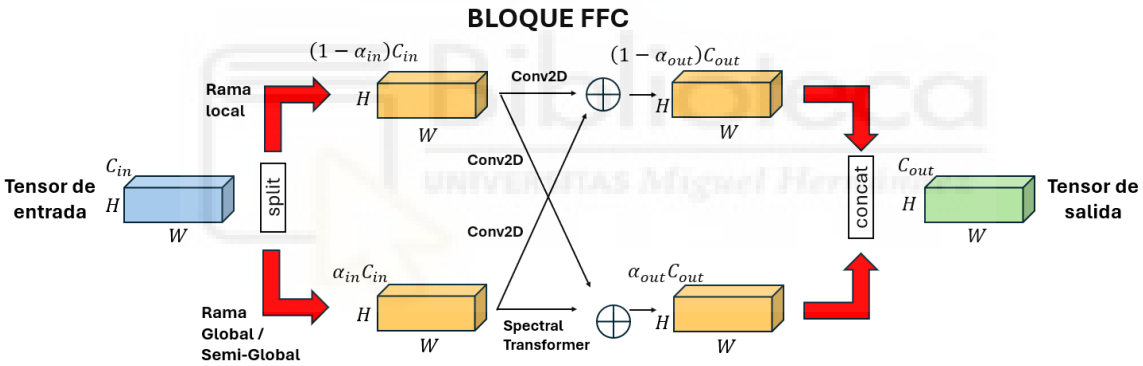


Figura 3-5: Bloque FFC completo.

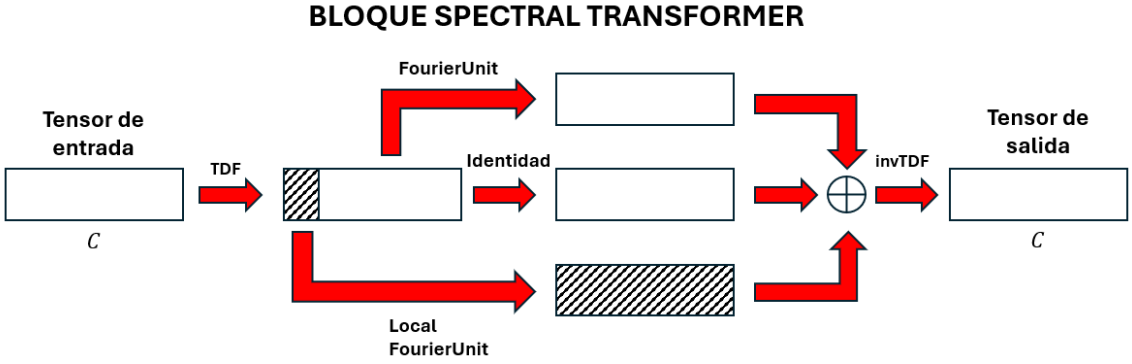


Figura 3-6: Bloque Spectral Transformer.

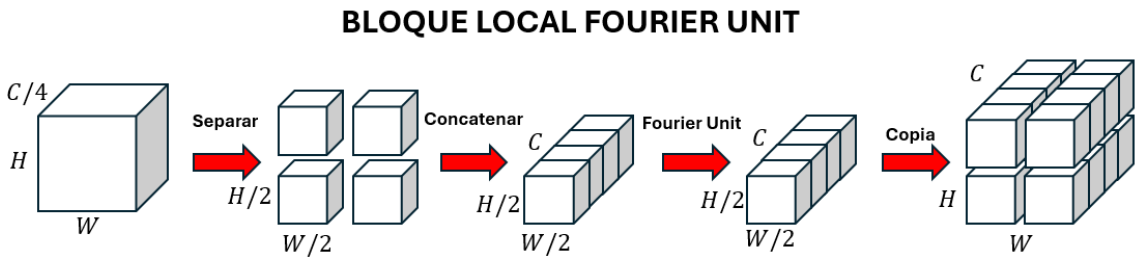


Figura 3-7: Bloque Local Fourier Unit.

Conv2D	Capa convolucional	C_{in}	Canales de entrada
TDF	Transformada discreta de Fourier	C_{out}	Canales de salida
invTDF	Transformada de Fourier inversa	H	Número de filas (alto)
α_{in}	Ratio de canales de entrada	W	Número de columnas (ancho)
α_{out}	Ratio de canales de salida	$a + bj$	Número complejo en forma binomial
$\oplus$	Suma elemento a elemento	Re	Parte real
split	Separación de canales	Im	Parte imaginaria
concat	Concatenación de canales		

Figura 3-8: Símbolos utilizados en las figuras del apartado 3.5.

Mientras que el bloque Fourier Unit trata de extraer las características de la imagen en su conjunto, el bloque Local Fourier Unit se emplea para extraer características a partir de regiones más reducidas de la imagen (fig. **3-7**), de tal forma que obtienen información complementaria. Las operaciones realizadas en este segundo bloque consta de los siguientes pasos. En primer lugar, el tensor se divide en cuatro partes de dimensiones $H/2 * W/2$. A continuación, se concatenan las cuatro partes y el tensor se introduce al bloque Fourier Unit. Por último, el tensor resultante se copia 4 veces para recuperar el tamaño original.

3.6. Padding en las convoluciones

Cuando se realiza la operación de convolución sobre una imagen y el tamaño de la máscara o kernel es superior a 1, se produce una reducción del tamaño de la imagen resultante. Para evitar este problema, se realiza un padding o relleno sobre la imagen inicial que compense las filas y columnas que se pierden con la operación de convolución. De esta forma, la imagen resultante tendrá las mismas dimensiones que la imagen de entrada.

Frecuentemente, este relleno se realiza con ceros, pero no se trata del único modo. En este trabajo, se ha realizado una comparación entre varios tipos de padding y se han propuesto dos métodos adicionales que se adaptan a las propiedades de las imágenes panorámicas. Los distintos tipos que se han empleado se muestran en la Figura **3-9**.

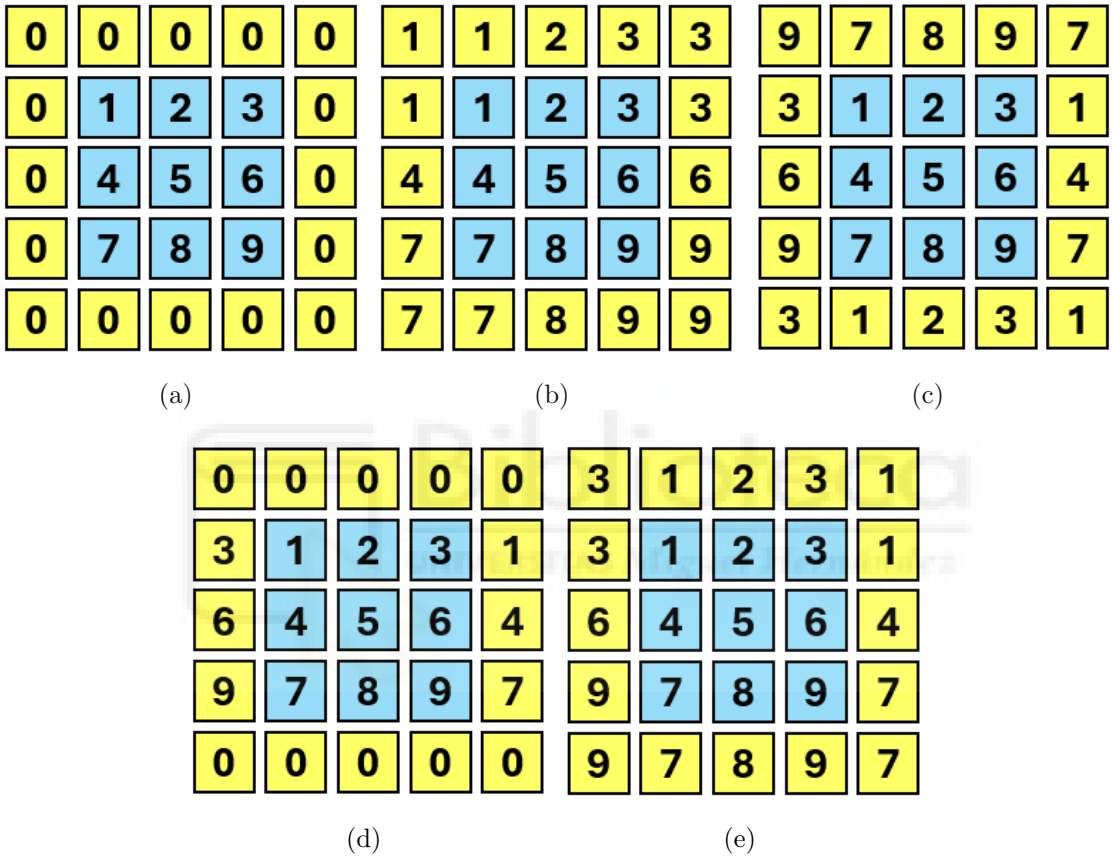


Figura 3-9: Tipos de padding evaluados en este trabajo: (a) Ceros, (b) Transparente, (c) Circular, (d) Propuesto 1 (Columnas: Circular, Filas: Ceros) y (e) Propuesto 2 (Columnas: Circular, Filas: Transparente).

Las imágenes panorámicas presentan la propiedad de que las columnas situadas en sus extremos presentan una continuidad visual. Por este motivo, aplicar un padding circular en las columnas de las imágenes puede ayudar a conservar las relaciones de vecindad entre estas zonas de la imagen. En cambio, esta propiedad no se cumple en las filas, así que tiene más sentido aplicar otros tipos de padding en las filas como el relleno con ceros o el transparente. De esta forma, los tipos de padding propuestos en este trabajo son los siguientes: en primer lugar, se ha realizado un padding circular

sobre las columnas y un padding con ceros sobre las filas; y en segundo lugar, se ha realizado un padding circular sobre las columnas y un padding transparente sobre las filas. Como es lógico, los distintos tipos de padding se han evaluado en convoluciones en el dominio de la imagen, ya que las propiedades de las imágenes panorámicas no se conservan en el dominio frecuencial.

3.7. Arquitectura de red utilizada

En este trabajo, se ha partido del modelo de red VGG16, propuesto por Simonyan and Zisserman [45], el cual se ha adaptado tal y como se muestra en la Figura **3-10**. Se han empleado tres arquitecturas distintas, que han sido comparadas para cada una de las tareas que se han abordado en esta investigación.

La primera de ellas es la arquitectura VGG16 original. Las capas convolucionales, correspondientes a la fase de extracción de características, no se han modificado. La primera capa totalmente conectada, en cambio, sí que ha sido modificada para adaptar la red al tamaño de las imágenes de entrada. Por último, el resto de capas totalmente conectadas también han sido modificadas y se ha eliminado la capa SoftMax, con el fin de obtener un descriptor global de dimensiones 5x1.

Asimismo, se han propuesto dos arquitecturas adicionales, en las cuales se ha sustituido las capas convolucionales de la red VGG16 por convoluciones en el dominio de Fourier (fig. **3-10**). En la arquitectura VGGFU16, las capas convolucionales han sido sustituidas por el bloque Fourier Unit, descrito en el apartado 3.5, mientras que en la arquitectura VGGFFC16, estas capas se han cambiado por el bloque FFC, también descrito en el apartado anterior.

Cada una de las arquitecturas ha sido entrenada de tres formas distintas: a) entrenamiento desde cero sin realizar preentrenamiento, b) entrenamiento desde cero realizando un preentrenamiento, c) entrenamiento partiendo de los pesos del modelo de red VGG16 (transfer learning), entrenada con la base de datos ImageNet [11]. En el capítulo 5 se describen con detalle los experimentos realizados.

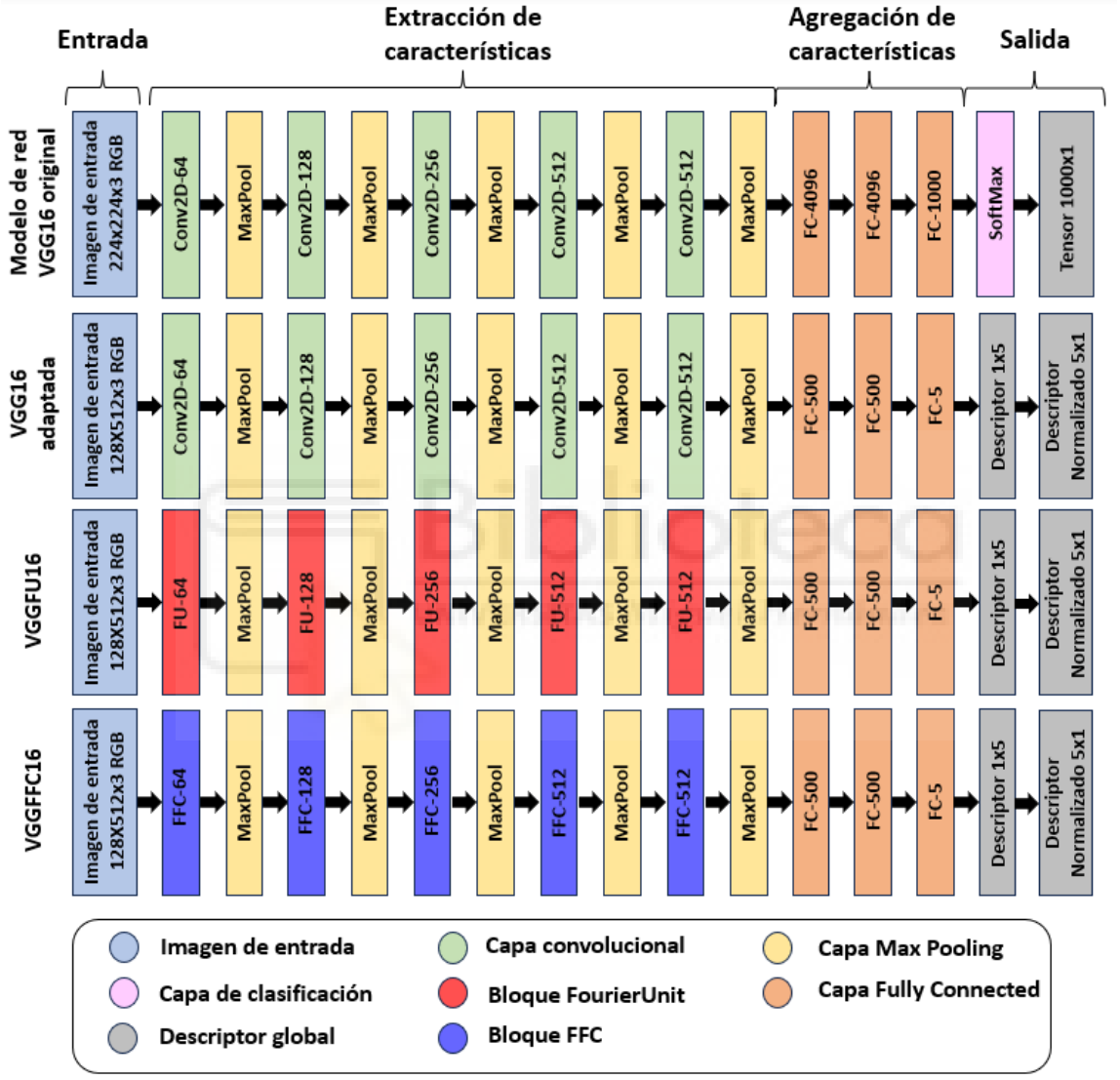


Figura 3-10: Arquitectura del modelo de red VGG16 original, modelo VGG16 adaptado y arquitecturas propuestas en este trabajo: VGGFU16 y VGGFFC16.

4 Localización visual

En este trabajo se ha abordado el problema de localización de un robot móvil mediante redes neuronales convolucionales. Para entrenar a la red, se han utilizado imágenes omnidireccionales, las cuales han sido capturadas con un sistema de visión catadióptrico. Las imágenes han sido capturadas por un robot móvil, que ha seguido diversas trayectorias en un entorno de interior. Estas imágenes se han convertido a formato panorámico, con un tamaño de 128x512 píxeles RGB.

Posteriormente, el conjunto inicial de imágenes se ha dividido en conjuntos de entrenamiento, validación y test. La estancia a la que pertenecen las imágenes, así como las coordenadas del punto de captura de cada imagen, son conocidas. Por tanto, se ha podido llevar a cabo un aprendizaje supervisado.

Para llevar a cabo la localización, se han empleado diferentes arquitecturas de CNNs, descritas en la sección 3.7, que han sido adaptadas a partir del modelo VGG16. Estas arquitecturas están compuestas por capas convolucionales de distinta naturaleza. Algunas de ellas realizan convoluciones rectangulares en el dominio espacial (convoluciones clásicas), mientras que otras realizan la operación de convolución en el dominio de la frecuencia (convoluciones en el dominio de Fourier).

En este trabajo, la arquitecturas propuestas han sido entrenadas para dos tareas distintas: en primer lugar, se ha realizado la clasificación de imágenes en estancias; y en segundo lugar, se ha llevado a cabo una localización global.

4.1. Room retrieval

La tarea de room retrieval consiste en determinar en qué estancia se ha capturado una imagen. Para ello, se ha entrenado una red con combinaciones de tres imágenes, escogidas de tal forma que las imágenes ancla y positiva deben pertenecer a la misma habitación, mientras que la imagen negativa debe haber sido capturada en una habitación distinta (véase la Figura **4-1**).

Para el test de la red, se ha escogido una imagen representativa de cada habitación,

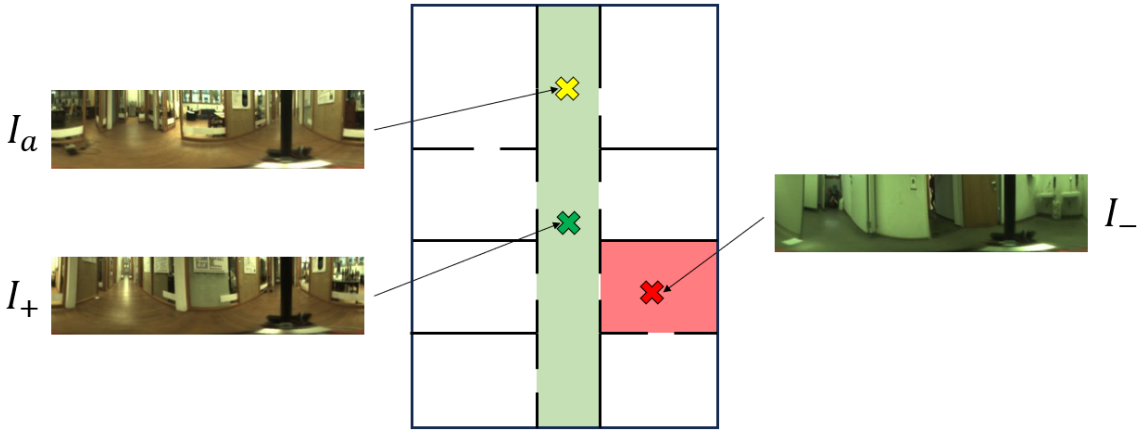


Figura 4-1: Ejemplo de selección de una combinación de tres imágenes en el entrenamiento de la red para la tarea de room retrieval.

que en este caso ha sido la más cercana al centro geométrico de dicha habitación. La estancia predicha por la red se obtiene comparando el descriptor de la imagen de test con los descriptores de las imágenes representativas de cada habitación (véase la Figura **4-2**). El procedimiento seguido en el test consta de los siguientes pasos:

1. El robot captura una imagen I_{test} desde una posición desconocida (x_{test}, y_{test}).
2. La red entrenada comprime la imagen en un descriptor global $\vec{d}_{test} \in \mathbb{R}^{5x1}$.
3. El descriptor $\vec{d}_{test}$ es comparado mediante la distancia euclídea con el descriptor de la imagen representativa de cada una de las m habitaciones $\boldsymbol{D^r} = \left[\vec{d}_1^{\,r}, \vec{d}_2^{\,r}, ..., \vec{d}_m^{\,r}\right]$.
4. La posición estimada del robot al capturar la imagen I_{test} viene dada por las coordenadas del vecino más cercano $(x_{pred}, y_{pred}) = (x_K^r, y_K^r)$.

4.2. Place recognition

Esta tarea consiste en plantear la localización como un problema de image retrieval, esto es, se compara la imagen capturada en el instante actual (imagen de test) con las imágenes de un modelo visual construido previamente. De esta forma, la imagen predicha como la más cercana devolverá la posición del robot.

Para realizar el entrenamiento de la red con una arquitectura de red tripleta, se han utilizado combinaciones de tres imágenes de entrada, que han sido escogidas de

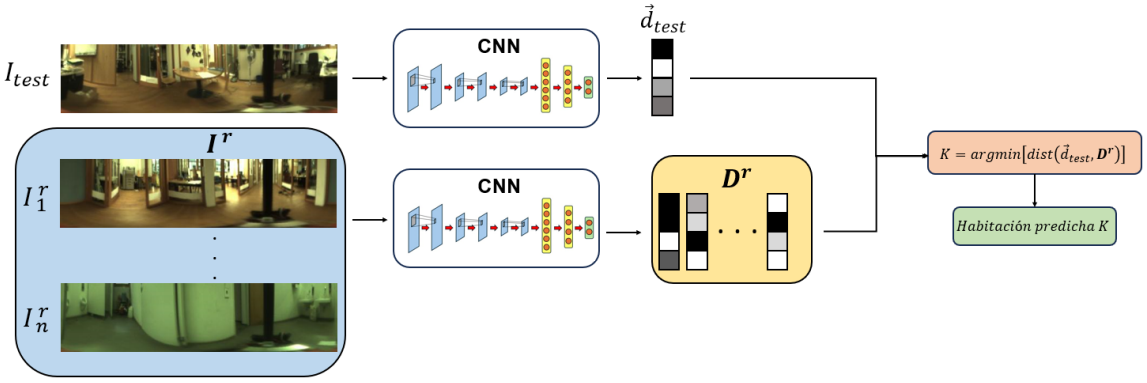


Figura 4-2: Test para la tarea de room retrieval. El descriptor de cada imagen de test $\vec{d}_{test}$ es comparado con los descriptores representativos de cada habitación $\boldsymbol{D^r} = \left[\vec{d}_1^{\,r}, \vec{d}_2^{\,r}, ..., \vec{d}_m^{\,r}\right]$ y el vecino más cercano indica la estancia predicha K.

forma aleatoria, salvo por la restricción de que las imágenes ancla y positiva deben haber sido capturadas a una distancia menor que un umbral r_+, mientras que las imágenes ancla y negativa deben haber sido capturadas a una distancia mayor que un umbral r_- (véase la Figura **4-3**).

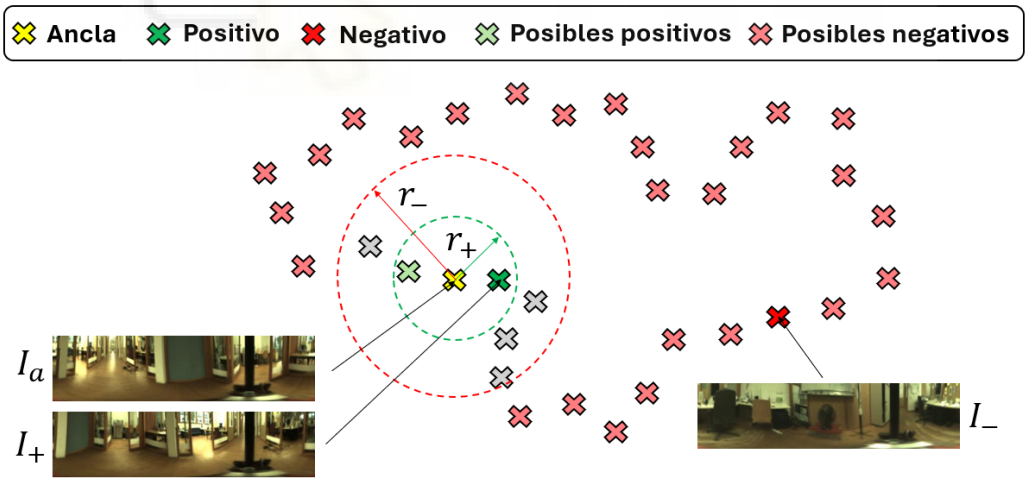


Figura 4-3: Ejemplo de selección de una combinación de tres imágenes en el entrenamiento de la red para la tarea de place recognition.

Para el test de la red en la tarea de place recognition, se ha comparado cada imagen de test con las imágenes del modelo visual, construido a partir de las imágenes utilizadas para el entrenamiento de la red (véase la Figura **4-4**). Para ello, se ha seguido el siguiente procedimiento:

1. El robot captura una imagen I_{test} desde una posición desconocida (x_{test}, y_{test}).
2. La red entrenada comprime la imagen en un descriptor global $\vec{d}_{test} \in \mathbb{R}^{5x1}$.
3. El descriptor $\vec{d}_{test}$ es comparado mediante la distancia euclídea con los descriptores de las imágenes que componen el modelo visual del mapa completo $\boldsymbol{D^{MV}} = \left[\vec{d}_1{}^{MV}, \vec{d}_2{}^{MV}, ..., \vec{d}_n{}^{MV}\right]$.
4. La posición estimada del robot al capturar la imagen I_{test} viene dada por las coordenadas del vecino más cercano $(x_{pred}, y_{pred}) = \left(x_i{}^{MV}, y_i{}^{MV}\right)$.

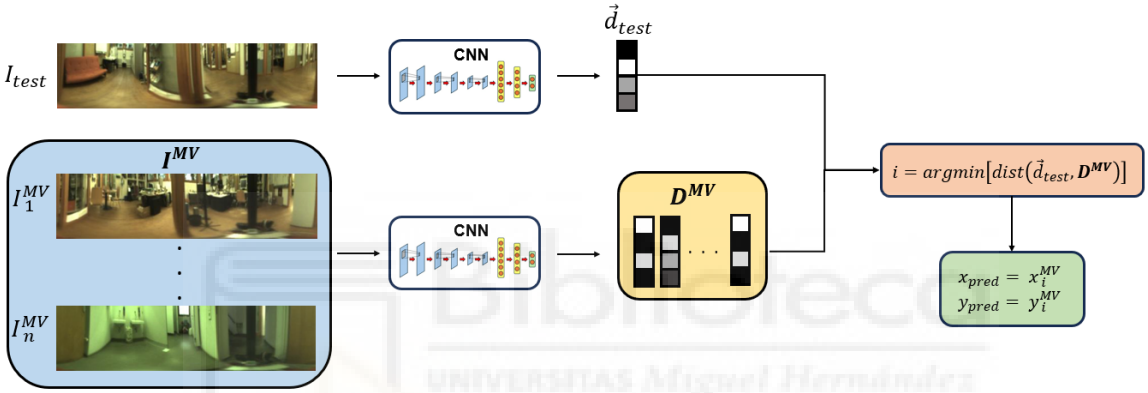


Figura 4-4: Test para la tarea de place recognition. El descriptor de cada imagen de test $\vec{d}_{test}$ es comparado con los descriptores que conforman el modelo visual del mapa completo $\boldsymbol{D^{MV}} = \left[\vec{d}_1{}^{MV}, \vec{d}_2{}^{MV}, ..., \vec{d}_n{}^{MV}\right]$ y el vecino más cercano indica las coordenadas estimadas del robot.

5 Experimentos y resultados

En este trabajo se ha estudiado el uso de las convoluciones en el dominio de Fourier para la tarea de localización de un robot móvil. Para ello, se han realizado dos experimentos distintos. En el experimento 1 se han analizado estas convoluciones para la tarea de room retrieval, mientras que en el experimento 2 se han empleado para la tarea de place recognition. Adicionalmentte, en el experimento 3 se ha realizado una comparación entre distintos tipos de padding empleados en las convoluciones. Antes de entrar en detalle con los experimentos, se describirá la base de datos que se ha utilizado.

5.1. Base de datos COLD

Las imágenes empleadas en este trabajo pertenecen a la base de datos COLD [38], las cuales se pueden descargar a través del siguiente enlace: https://www.cas.kth.se/COLD/. Esta base de datos contiene imágenes omnidireccionales capturadas por un robot móvil que lleva incorporado un sistema catadióptrico.

La base de datos contiene distintas secuencias realizadas por el robot en tres entornos de interior diferentes: Friburgo, Ljubljana y Saarbrücken. En cada secuencia, el robot realiza un recorrido a lo largo de un edificio, atravesando varias estancias de distinta naturaleza: oficinas, laboratorios, pasillos, aseos, etc. Además, se han capturado secuencias en tres condiciones de iluminación distintas: nublado, de noche y soleado. Asimismo, las imágenes incluyen ejemplos difíciles para la red, como por ejemplo cambios en la ubicación del mobiliario o la presencia de personas.

Por todos estos motivos, se trata de una base de datos completa y desafiante para entrenar y validar las distintas arquitecturas propuestas en este trabajo. En la Figura **5-1** se muestran algunos ejemplos de las imágenes que constituyen la base de datos COLD, mientras que en la Figura **5-2** se muestran dos trayectorias diferentes realizadas por el robot en la planta baja del edificio Friburgo.

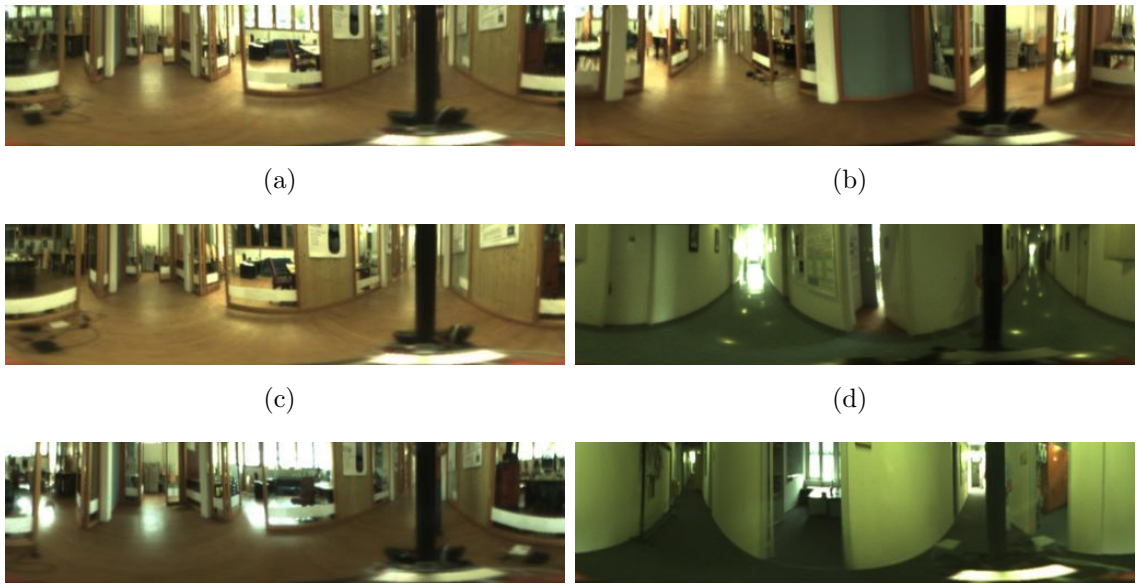
Para realizar los experimentos, se han construido tres conjuntos de imágenes distintos, cuya composición ha sido detallada en las Tablas **5-1**, **5-2** y **5-3**. En primer

lugar, el conjunto 0 se ha utilizado para realizar el preentrenamiento de la red, cuyo objetivo es ajustar los pesos de tal forma que la red se adapte al tipo de imágenes que componen la base de datos empleada. Este conjunto contiene 92075 imágenes, que han sido capturadas bajo las tres condiciones de iluminación y que pertenecen a los tres entornos que componen la base de datos. No obstante, no se han utilizado las imágenes de la parte A del edificio Friburgo, la cual ha sido empleada para realizar el entrenamiento, validación y el test de la red.

Los conjuntos 1 y 2, en cambio, han sido utilizados para entrenar, validar y testear la red para la tarea deseada. Estos conjuntos contienen imágenes que pertenecen a la parte A del entorno de Friburgo. La diferencia entre estos dos conjuntos se debe a la exigencia del proceso de entrenamiento, ya que las condiciones en las que se realiza el test de los modelos entrenados son exactamente las mismas. Por un lado, el conjunto 1 corresponde a un entrenamiento más exigente, que se realiza con un número más reducido de imágenes capturadas únicamente bajo la condición nublado. Por otro lado, el conjunto 2 contiene un número de imágenes mucho mayor, las cuales han sido capturadas bajo las tres condiciones de iluminación. En ambos casos, el test de la red se ha realizado con imágenes que la red no ha visto durante el entrenamiento y la validación.

Conjunto 0. Preentrenamiento			
Entorno	**Parte**	**Número de Imágenes**	**Número de Estancias**
Friburgo	B	11918	5
Ljubljana	A	41018	6
Saarbrücken	A, B	39139	13
TOTAL		92075	24

Tabla 5-1: Número de imágenes de cada entorno que componen el conjunto 0, utilizado para el preentrenamiento de la red.



(a) (b)

(c) (d)

(e) (f)

Figura 5-1: Ejemplos de imágenes capturadas bajo distintas condiciones de iluminación (a) Nublado, (c) Noche, (e) Soleado y ejemplos de imágenes capturadas en entornos distintos (b) Friburgo, (d) Ljubljana, (f) Saarbrücken).

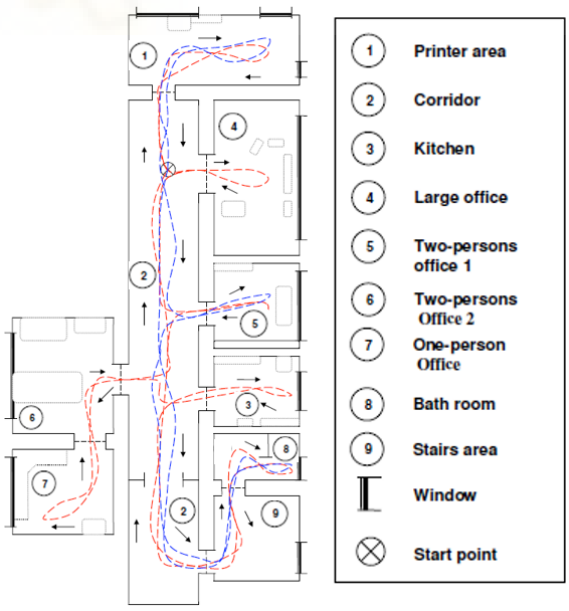


Figura 5-2: Trayectorias realizadas por el robot en la planta baja del edificio Friburgo.

Conjunto 1. Entrenamiento solo con nublado				
Friburgo Parte A	**Nublado**	**Noche**	**Soleado**	**TOTAL**
Entrenamiento / Modelo Visual	588	X	X	588
Validación	586	X	X	586
Test	2595	2707	2114	7416

Tabla 5-2: Número de imágenes que componen el conjunto 1, utilizado para el entrenamiento, la validación y el test de la red.

Conjunto 2. Entrenamiento con las tres condiciones de iluminación				
Friburgo Parte A	**Nublado**	**Noche**	**Soleado**	**TOTAL**
Entrenamiento	2481	2567	1991	7039
Modelo Visual	588	X	X	588
Validación	297	307	238	842
Test	2595	2707	2114	7416

Tabla 5-3: Número de imágenes que componen el conjunto 2, utilizado para el entrenamiento, la validación y el test de la red.

5.2. Experimento 1. Estudio de las convoluciones en el dominio de Fourier para la tarea de room retrieval

En este experimento se ha evaluado el desempeño de tres arquitecturas de CNN distintas en la tarea de clasificación de imágenes en estancias. Las arquitecturas han sido entrenadas de tres modos: en primer lugar, se han entrenado los modelos desde cero, es decir, partiendo de pesos aleatorios; en segundo lugar, se han entrenado desde cero, pero previamente se ha realizado un preentrenamiento con un conjunto de imágenes mucho más extenso (C0); y, en tercer lugar, se han entrenado los modelos mediante la técnica de transfer learning, esto es, se ha partido de los pesos iniciales del modelo VGG16 original en las capas convolucionales de la red. Por tanto, este experimento se compone de tres pruebas. En cada una de estas pruebas, los modelos han sido entrenados con dos conjuntos de imágenes distintos (C1 y C2). La composición de estos conjuntos se ha descrito con detalle en el apartado 5.1. Asimismo, en la Tabla **5-4** se muestran las condiciones en las que se ha realizado el entrenamiento de las distintas arquitecturas en cada una de las pruebas.

Condiciones de Entrenamiento	Prueba 1	Prueba 2	Prueba 3
Nº Épocas	10	10/10	10
Nº Combinaciones por Época	25000	250000/25000	25000
Función de pérdida	Triplet Margin Loss (m=1)		
Optimizador	AdamW	AdamW	SGD
Preentrenamiento	NO	SÍ	NO
Transfer Learning	NO	NO	SÍ

Tabla 5-4: Condiciones de entrenamiento de las distintas pruebas en el experimento 1.

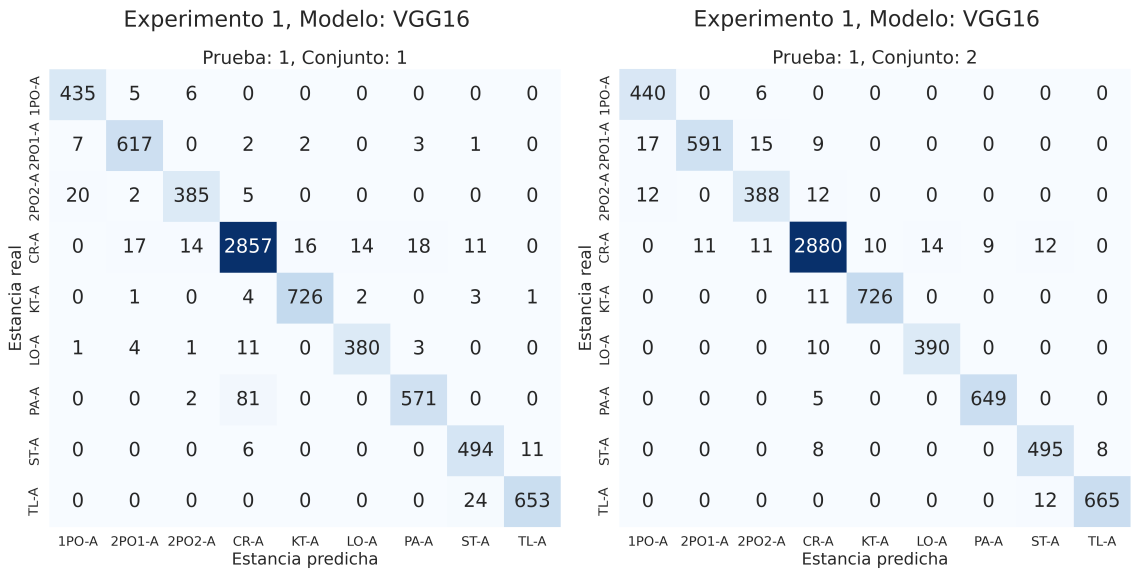
Prueba 1. Red entrenada desde cero sin realizar preentrenamiento

En esta prueba, los distintos modelos propuestos han sido entrenados desde cero para realizar la tarea de clasificación en estancias. La Tabla **5-5** muestra la precisión obtenida con cada modelo bajo cada condición de iluminación y para cada uno de los conjuntos utilizados para el entrenamiento de dichos modelos.

PRUEBA 1	Precisión (%)							
	C1				C2			
Modelo	**Nublado**	**Noche**	**Soleado**	**Medio**	**Nublado**	**Noche**	**Soleado**	**Medio**
VGG16	**99.19**	**97.16**	**90.54**	**95.63**	**99.65**	**97.30**	**94.80**	**97.25**
VGGFU16	48.17	50.83	32.31	43.77	55.34	54.75	43.90	51.33
VGGFFC16	81.27	93.35	48.82	74.48	92.79	92.17	90.54	91.83

Tabla 5-5: Precisión obtenida con cada modelo en la prueba 1 para la tarea de room retrieval.

La Tabla **5-5** muestra que, en esta prueba, el modelo que ha obtenido los mejores resultados ha sido el VGG16. El modelo VGGFFC ha obtenido una precisión media bastante inferior cuando ha sido entrenado con el conjunto de imágenes C1, pero esta diferencia se ha producido sobre todo bajo condiciones soleadas. En cambio, cuando ha sido entrenado con el conjunto C2, que incluye imágenes capturadas bajo las tres condiciones de iluminación, ha obtenido una precisión elevada para las tres condiciones de iluminación. Esto se debe a que, en el primer caso, la red ha experimentado un cierto sobreajuste a la condición de iluminación de entrenamiento (nublado). Por último, el modelo VGGFU16 ha obtenido resultados claramente inferiores a los otros dos modelos. Esto quiere decir que, para esta red, el entrenamiento realizado ha sido insuficiente en términos de cantidad y variedad de imágenes.

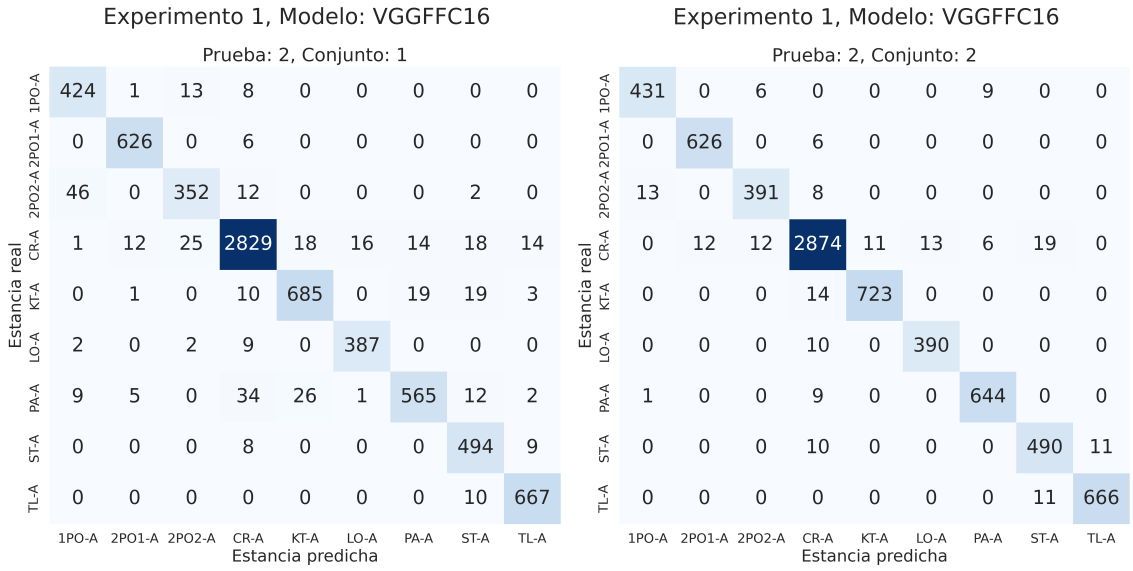


(a) (b)

Figura 5-3: Matrices de confusión obtenidas para la prueba 1 con el modelo de red VGG16, entrenado con los conjuntos (a) C1 y (b) C2.

Las matrices de confusión son herramientas muy útiles para mostrar las predicciones realizadas por una red neuronal en una tarea de clasificación. La Figura **5-3** muestra las matrices de confusión del modelo VGG16 al ser entrenado con los conjuntos C1 y C2, donde cada fila representa las predicciones del modelo para las imágenes de una determinada estancia y cada columna representa a qué estancia real pertenecen las imágenes asociadas por la red a una determinada estancia. Cuanto más se asemejen estas matrices a una matriz diagonal, la precisión del modelo será mayor.

Las matrices de confusión de la Figura **5-3** muestran que, como es lógico, la red ha cometido un menor número de errores cuando ha sido entrenada con el conjunto C2. Además, estas matrices nos muestran las habitaciones en las que la red ha encontrado una mayor dificultad. Por ejemplo, se puede observar claramente que la mayoría de errores se han cometido en el pasillo (CR-A). Esto es lógico, pues se trata del nexo entre el resto de estancias del edificio, y por tanto comparte información visual con casi todas las estancias. Además, el resto de errores se concentra principalmente entre habitaciones conectadas (1PO-A y 2PO2-A, ST-A y TL-A) y entre habitaciones de la misma naturaleza, como es el caso de las oficinas (1PO-A, 2PO1-A y 2PO2-A).



(a) (b)

Figura 5-4: Matrices de confusión obtenidas para la prueba 2 con el modelo de red VGGFFC16, entrenado con los conjuntos (a) C1 y (b) C2.

Prueba 2. Red entrenada tras realizar un preentrenamiento

En esta prueba, los distintos modelos han sido entrenados desde cero en dos etapas: en primer lugar, se ha realizado un preentrenamiento para la tarea de room retrieval con un conjunto de imágenes extenso y variado, con imágenes pertenecientes a tres entornos distintos y capturadas bajo las tres condiciones de iluminación; y en segundo lugar, se ha realizado el entrenamiento para la tarea de room retrieval, partiendo de los pesos de la red guardada tras la primera etapa. La Tabla **5-6** y la Figura **5-4** muestran la precisión obtenida con cada uno de los modelos y las matrices de confusión obtenidas con el modelo VGGFFC16, respectivamente.

PRUEBA 2	**Precisión (%)**							
	C0 + C1				C0 + C2			
Modelo	**Nublado**	**Noche**	**Soleado**	**Medio**	**Nublado**	**Noche**	**Soleado**	**Medio**
VGG16	95.80	**95.94**	62.44	84.73	98.61	**97.19**	94.37	96.72
VGGFU16	85.24	87.00	59.41	77.22	95.45	92.91	86.71	91.69
VGGFFC16	**98.11**	94.79	**90.68**	**94.53**	**99.34**	96.79	**96.36**	**97.50**

Tabla 5-6: Precisión obtenida con cada modelo en la prueba 2 para la tarea de room retrieval.

La Tabla **5-6** y la Figura **5-4** muestran que, en este caso, el modelo de red VGGFFC16 ha sido el que ha tenido un mejor desempeño. Al emplear el conjunto C1 para el entrenamiento, los modelos VGG16 y VGGFU16 presentan un mayor sobreajuste a la condición de entrenamiento, ya que han tenido una precisión elevada en nublado y noche pero no para soleado, mientras que el modelo VGGFFC16 presenta una precisión elevada bajo las tres condiciones de iluminación. Al emplear el conjunto C2 para entrenar los modelos, las arquitecturas VGGFU16 y VGGFFC16 han tenido un rendimiento superior en comparación con la prueba 1. Esto puede ser debido a que estas arquitecturas, que en la prueba 1 habían obtenido peores resultados en comparación con el modelo VGG16, presentan un buen rendimiento cuando son entrenadas con condiciones favorables, es decir, con las tres condiciones de iluminación y tras realizar un preentrenamiento.

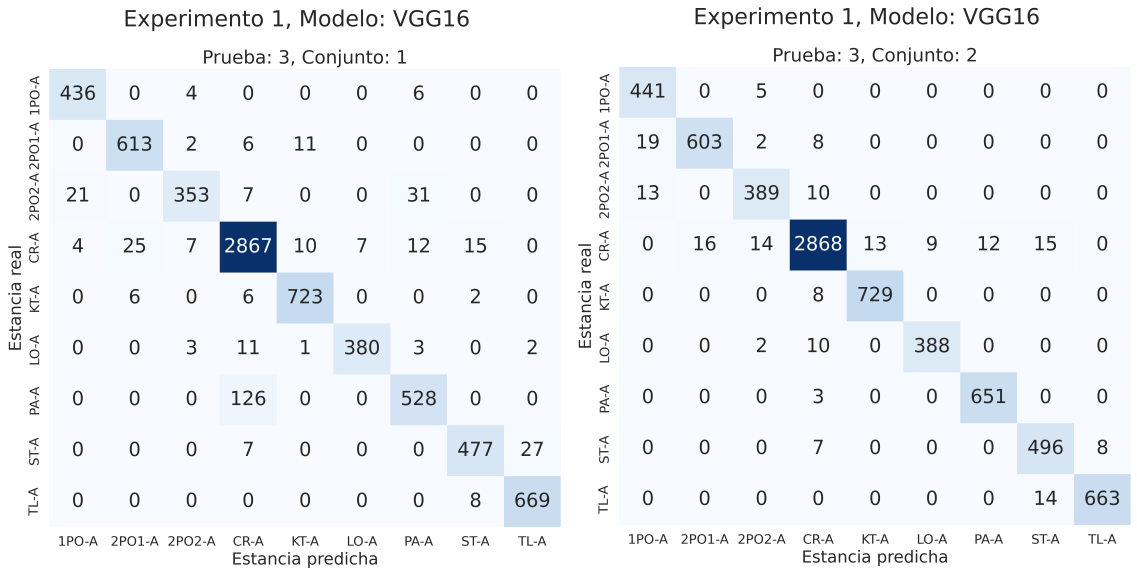
Prueba 3. Red entrenada partiendo de los pesos del modelo VGG16 original

Por último, en la prueba 3 se han entrenado los tres modelos mediante la técnica de transfer learning, es decir, se ha partido de los pesos del modelo VGG16 original, con el fin de aprovechar la capacidad de esta arquitectura en la extracción de características. Los resultados de esta prueba se muestran en la Tabla **5-7** y en la Figura **5-5**.

PRUEBA 3	**Precisión (%)**							
	C1				C2			
Modelo	**Nublado**	**Noche**	**Soleado**	**Medio**	**Nublado**	**Noche**	**Soleado**	**Medio**
VGG16	**98.96**	**96.60**	**88.13**	**94.56**	**99.58**	**97.08**	**95.36**	**97.34**
VGGFU16	77.11	84.60	32.83	64.84	94.84	97.01	92.01	94.62
VGGFFC16	86.36	95.79	60.79	80.98	96.96	95.20	92.62	94.93

Tabla 5-7: Precisión obtenida con cada modelo en la prueba 3 para la tarea de room retrieval.

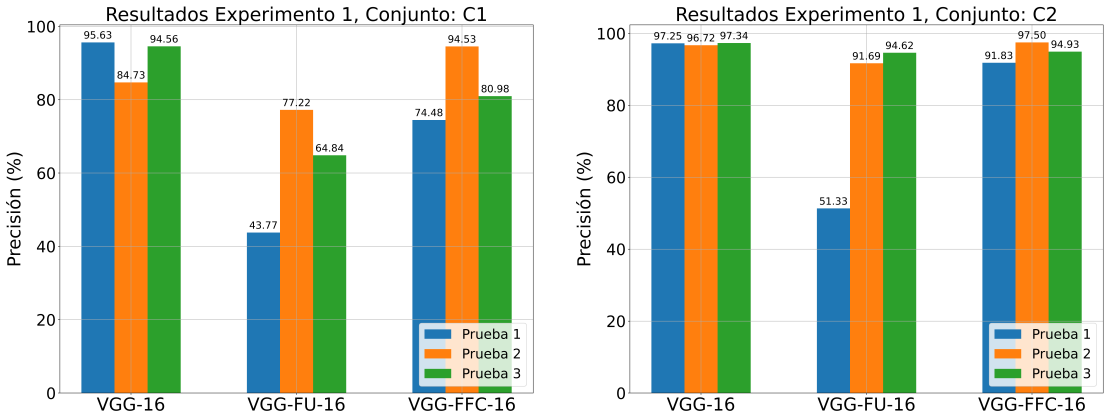
Los resultados obtenidos en la prueba 3 muestran que el modelo que ha obtenido los mejores resultados es el VGG16. Esto es lógico, pues se ha partido de los pesos originales de este modelo, el cual presenta una gran capacidad para extraer características a partir de imágenes. Los modelos VGGFU16 y VGGFFC16 no han obtenido una precisión tan elevada, ya que, aunque se ha partido de los pesos de la red VGG16 en las capas convolucionales, estas convoluciones se realizan en el dominio de la imagen y no en el dominio frecuencial. No obstante, si se comparan los resultados con los obtenidos en la prueba 1, los resultados son claramente mejores,



(a) (b)

Figura 5-5: Matrices de confusión obtenidas para la prueba 3 con el modelo de red VGG16, entrenado con los conjuntos (a) C1 y (b) C2.

por lo que se ha demostrado que emplear la técnica de transfer learning favorece el aprendizaje de las CNNs propuestas.



(a) (b)

Figura 5-6: Comparación entre los resultados obtenidos en el experimento 1 para cada modelo según el conjunto utilizado: (a) C1 y (b) C2.

En la Figura **5-6** se realiza una comparación entre los resultados obtenidos con

cada uno de los modelos en cada prueba. Las barras verticales representan la precisión en la tarea de room retrieval. Esta figura muestra una clara diferencia en el comportamiento de cada modelo: mientras que el modelo VGG16 obtiene los peores resultados en la prueba 2, los otros dos modelos han mostrado un rendimiento superior en la prueba 1. Esto puede ser debido a que el preentrenamiento ha funcionado mejor para los modelos propuestos. En líneas generales, se ha obtenido un rendimiento similar o incluso superior en la prueba 3 respecto a la prueba 1, lo que justifica el uso de la técnica de transfer learning. Finalmente, se han obtenido mejores resultados con el conjunto C2, como es lógico, ya que se trata de un conjunto de imágenes más completo.

5.3. Experimento 2. Estudio de las convoluciones en el dominio de Fourier para la tarea de place recognition

En este experimento, las distintas arquitecturas han sido evaluadas en la tarea de place recognition. La organización de las distintas pruebas ha sido la misma que en el experimento 1: se ha entrenado cada modelo de tres formas distintas, y en cada prueba se han empleado dos conjuntos de entrenamiento distintos (C1 y C2). Las condiciones en las que se ha realizado cada entrenamiento han sido las mismas que en el experimento 1 (véase la Tabla **5-4**). En cuanto a la selección de las tripletas de entrenamiento, los umbrales han tomado los siguientes valores: $r_+ = 0,5m$ y $r_- = 0,5m$ (la explicación del significado de estos umbrales se encuentra detallada en el apartado 4.2).

Prueba 1. Red entrenada desde cero sin realizar preentrenamiento

En esta prueba, se han entrenado los distintos modelos desde cero para la tarea de place recognition. La Tabla **5-8** muestra el error de localización medio obtenido para cada modelo y bajo cada condición de iluminación con cada uno de los conjuntos de entrenamiento. El error de localización se obtiene a partir de la distancia geométrica entre las coordenadas del punto de captura de cada imagen de test (ground truth) y las coordenadas de la imagen del modelo visual predicha como la más cercana. Por la distribución en el plano del suelo de las imágenes que componen el modelo visual y las secuencias de test, este error nunca podrá ser cero. En la Tabla **5-9** se muestra el error mínimo que se puede alcanzar bajo cada condición de iluminación, que sería el que se obtendría si, para todas las imágenes de test, la imagen predicha como la más cercana coincidiera con la imagen más cercana a la imagen de test.

PRUEBA 1	Error de localización (m)							
	C1				C2			
Modelo	**Nublado**	**Noche**	**Soleado**	**Medio**	**Nublado**	**Noche**	**Soleado**	**Medio**
VGG16	**0.337**	**0.333**	**0.719**	**0.463**	**0.293**	**0.304**	**0.450**	**0.349**
VGGFU16	3.635	2.520	6.494	4.216	3.389	2.841	5.552	3.927
VGGFFC16	2.232	0.852	6.155	3.080	0.403	0.478	1.021	0.634

Tabla 5-8: Error de localización cometido por cada modelo en la tarea de place recognition en la prueba 1.

	Error mínimo alcanzable (m)			
Condición de Iluminación	**Nublado**	**Noche**	**Soleado**	**Promedio**
	0.127	0.126	0.119	0.124

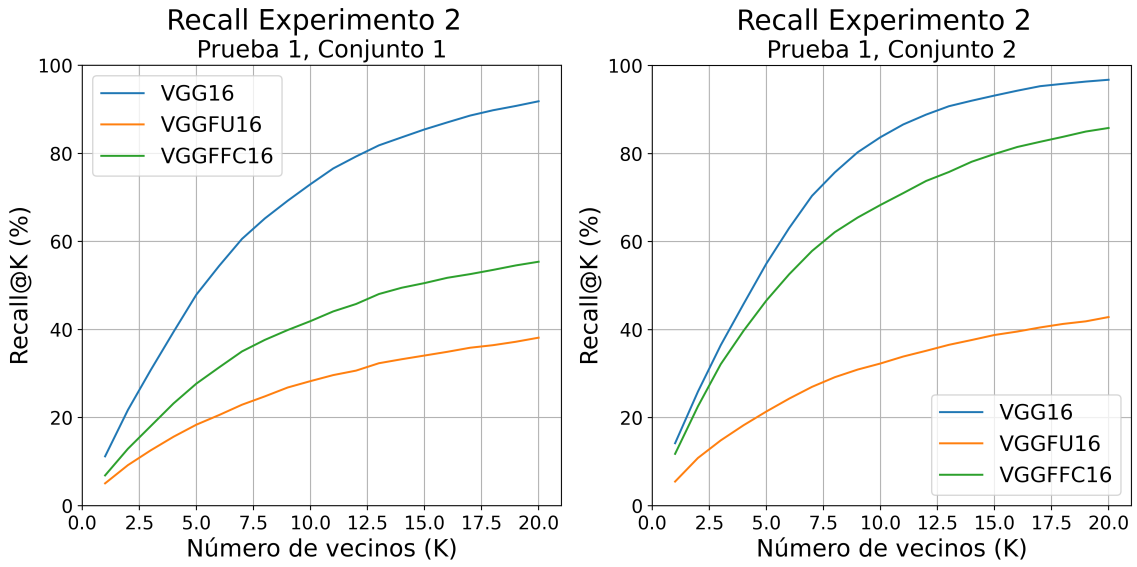
Tabla 5-9: Error mínimo alcanzable considerando la distribución de las secuencias de entrenamiento y test en el plano del suelo.

La Figura **5-7** muestra el Recall@K obtenido con cada uno de los modelos en la prueba 1. El Recall@K se define como la proporción de imágenes de test que son localizadas correctamente dentro de los K vecinos más cercanos. Por ejemplo, un Recall@1 igual al 100 % significa que la imagen del modelo visual predicha como la más cercana coincide con la imagen más cercana real en todos los casos, por lo que el desempeño del modelo puede considerarse como perfecto.

La Tabla **5-8** y las gráficas de la Figura **5-7** muestran que el modelo que ha obtenido los mejores resultados ha sido el VGG16. Si se comparan los resultados con los obtenidos en la prueba 1 del experimento 1, podemos observar que el comportamiento es similar. Cuando se realiza el entrenamiento con el conjunto C1, el error cometido por el modelo VGGFFC16 es bastante elevado, mientras que con el conjunto C2 el error se reduce considerablemente, aunque continúa siendo ligeramente superior al error cometido por el modelo VGG16. De la misma forma, el modelo VGGFU16 devuelve un error bastante elevado en ambos casos.

Prueba 2. Red entrenada tras realizar un preentrenamiento

En esta prueba, los modelos han sido entrenados desde cero en dos etapas: en primer lugar, se ha realizado un preentrenamiento para la tarea de room retrieval con el conjunto C0; y en segundo lugar, se han entrenado los modelos de red guardados para la tarea de place recognition con los conjuntos C1 y C2. El preentrenamiento no se ha realizado para la tarea de place recognition, por dos motivos. Por un lado, en



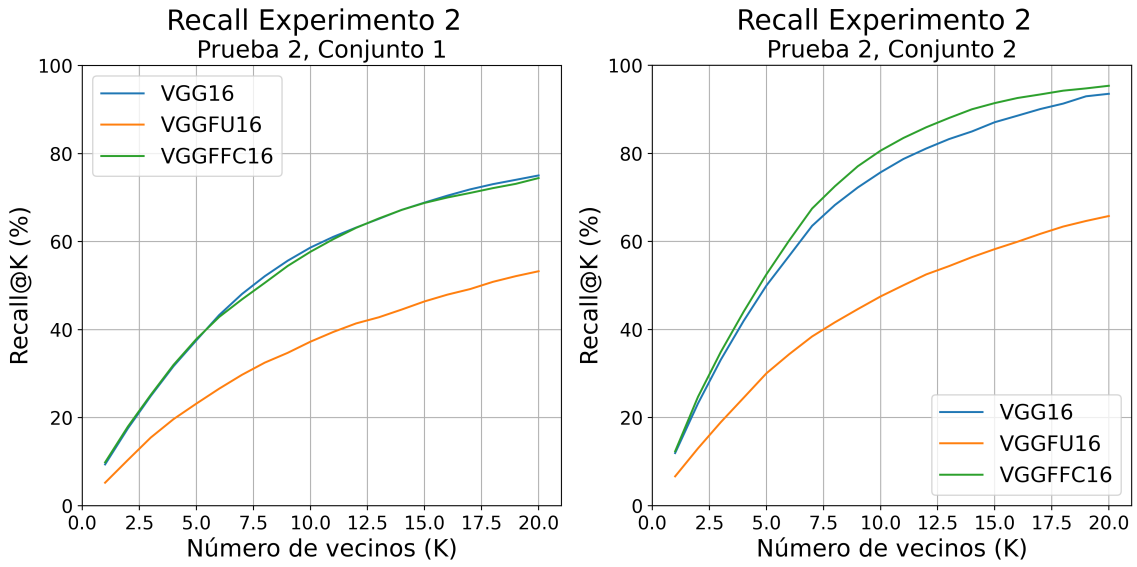
(a) (b)

Figura 5-7: Recall@K obtenido para la prueba 1 con cada arquitectura, entrenadas con los conjuntos (a) C1 y (b) C2.

algunas de las secuencias utilizadas para el preentrenamiento de la red, las coordenadas de las imágenes (ground truth) contienen un error bastante elevado debido a un mal funcionamiento de los sensores odométricos. Por otro lado, dado que el preentrenamiento es exactamente el mismo que el que se ha realizado en el experimento 1, se han reutilizado los modelos guardados tras realizar el preentrenamiento en dicho experimento, y así se ha evitado el tener que realizar de nuevo el preentrenamiento, el cual conlleva bastante tiempo. La Tabla **5-10** y la Figura **5-8** recogen los resultados obtenidos para esta prueba.

PRUEBA 2	**Error de localización (m)**							
	C1				C2			
Modelo	**Nublado**	**Noche**	**Soleado**	**Medio**	**Nublado**	**Noche**	**Soleado**	**Medio**
VGG16	0.679	**0.397**	2.553	1.210	0.362	**0.311**	0.671	0.448
VGGFU16	1.297	1.556	3.732	2.195	1.381	1.527	1.862	1.590
VGGFFC16	**0.526**	0.445	**1.312**	**0.761**	**0.315**	0.327	**0.431**	**0.358**

Tabla 5-10: Error de localización cometido por cada modelo en la tarea de place recognition en la prueba 2.



(a) (b)

Figura 5-8: Recall@K obtenido para la prueba 2 con cada arquitectura, entrenadas con los conjuntos (a) C1 y (b) C2.

La Tabla **5-10** y la Figura **5-8** muestran que, tal como sucede en el experimento 1, cuando se realiza un preentrenamiento extenso y variado, el modelo VGGFFC16 tiene un mejor desempeño. Asimismo, el error cometido por el modelo VGGFU16 también se reduce significativamente, especialmente al ser entrenado con el conjunto C2, pero sigue siendo mayor que el error cometido por los otros dos modelos.

Prueba 3. Red entrenada partiendo de los pesos del modelo VGG16 original (transfer learning)

En esta prueba, los modelos han sido entrenados empleando la técnica de transfer learning. La Tabla **5-11** muestra el error geométrico cometido por cada modelo y la Figura **5-9** muestra el Recall@K.

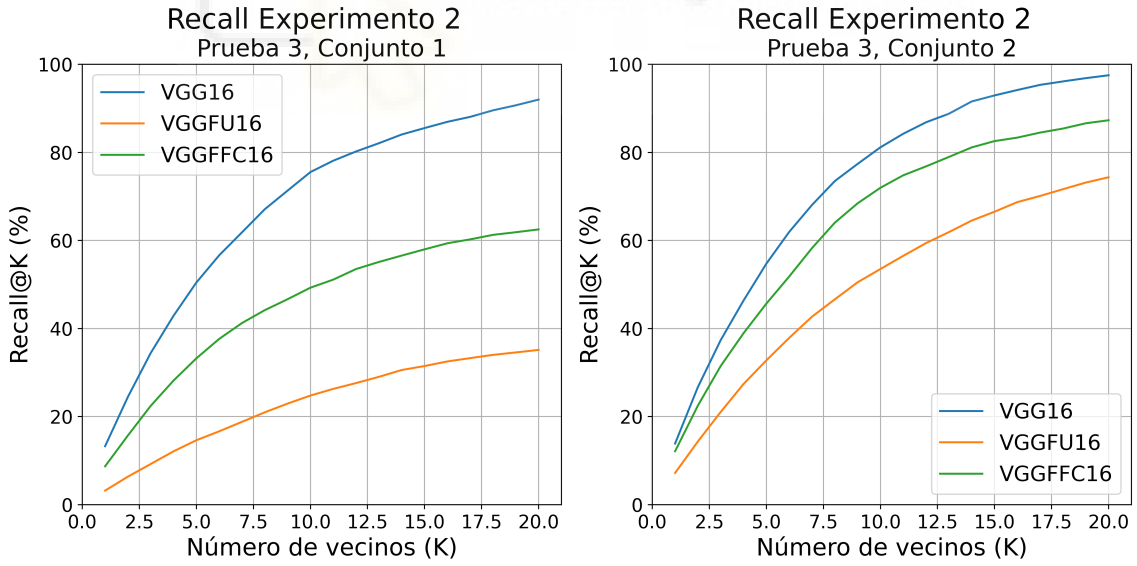
Los resultados de la prueba 3 muestran que el modelo que ha tenido un error más reducido es el VGG16. En este caso, los tres modelos también han presentado mejores resultados que en la prueba 1, lo que justifica el uso del transfer learning para el entrenamiento de los modelos. Además, el modelo VGGFU16 ha presentado mejores resultados que en la prueba 2, lo que quiere decir que el uso del transfer learning ha sido más efectivo que la realización del preeentrenamiento.

La Figura **5-10** resume los resultados obtenidos en el experimento 2 con cada modelo en cada una de las pruebas, donde las barras verticales representan el error de

PRUEBA 3	**Error de localización (m)**							
	C1				C2			
Modelo	**Nublado**	**Noche**	**Soleado**	**Medio**	**Nublado**	**Noche**	**Soleado**	**Medio**
VGG16	**0.306**	**0.321**	**0.657**	**0.428**	**0.296**	**0.310**	**0.411**	**0.339**
VGGFU16	3.167	2.821	6.582	4.190	0.994	0.689	1.294	0.992
VGGFFC16	0.800	0.496	5.066	2.121	0.534	0.407	0.823	0.588

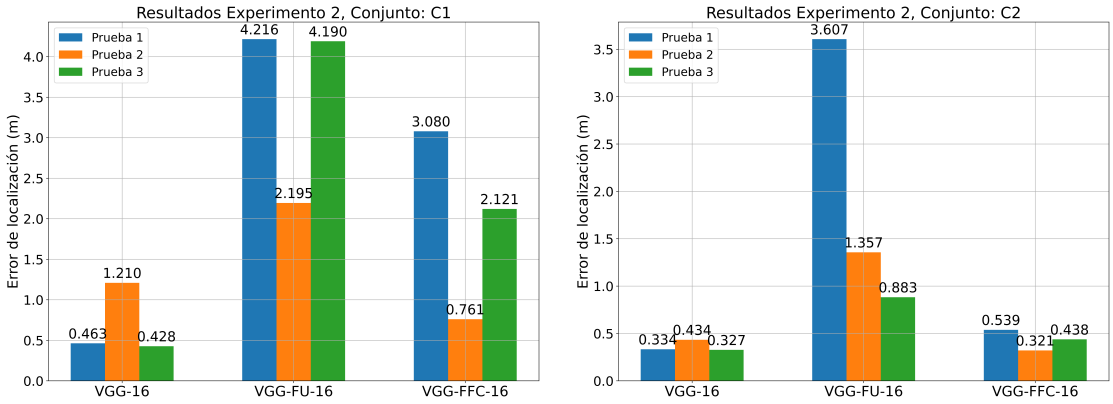
Tabla 5-11: Error de localización cometido por cada modelo en la tarea de place recognition en la prueba 3.

localización global. A partir de esta figura se pueden extraer conclusiones similares a las del experimento 1. Por un lado, los modelos VGGFU16 y VGGFFC16 han obtenido los mejores resultados cuando se realiza un preeentrenamiento, mientras que el modelo VGG16 ha tenido un error mayor en esta prueba. En cuanto al uso de transfer learning, en este caso también se ha producido una mejora para los tres modelos. Por último, al emplear el conjunto C2 también se reduce el error, especialmente bajo condiciones soleadas.



(a) (b)

Figura 5-9: Recall@K obtenido para la prueba 3 con cada arquitectura, entrenadas con los conjuntos (a) C1 y (b) C2.



(a) (b)

Figura 5-10: Comparación entre los resultados obtenidos en el experimento 2 para cada modelo según los conjuntos utilizados: (a) C1 y (b) C2.

5.4. Experimento 3. Análisis del tipo de padding en las convoluciones

Para terminar, en este trabajo se ha explorado el uso de otros tipos de padding en las capas convolucionales que componen una CNN, con el objetivo de aprovechar las propiedades de las imágenes panorámicas. Para ello, el modelo VGG16 adaptado se ha entrenado para las tareas de room retrieval y place recognition, y se han empleado cinco tipos de padding distintos (fig. **3-9**): con ceros (el que realiza por defecto el modelo VGG16), circular, transparente y dos modos adicionales que se han propuesto en este trabajo. El primero de ellos consiste en aplicar un padding circular en las columnas de la imagen y un padding con ceros en las filas, mientras que el segundo de ellos consiste en un padding circular en las columnas y un padding transparente en las filas. Las condiciones de los entrenamientos realizados coinciden con las de la prueba 3 de los experimentos 1 y 2, respectivamente (véase la Tabla **5-4**). Sin embargo, se ha empleado únicamente el conjunto de imágenes C1 para realizar el entrenamiento, la validación y el test de la red.

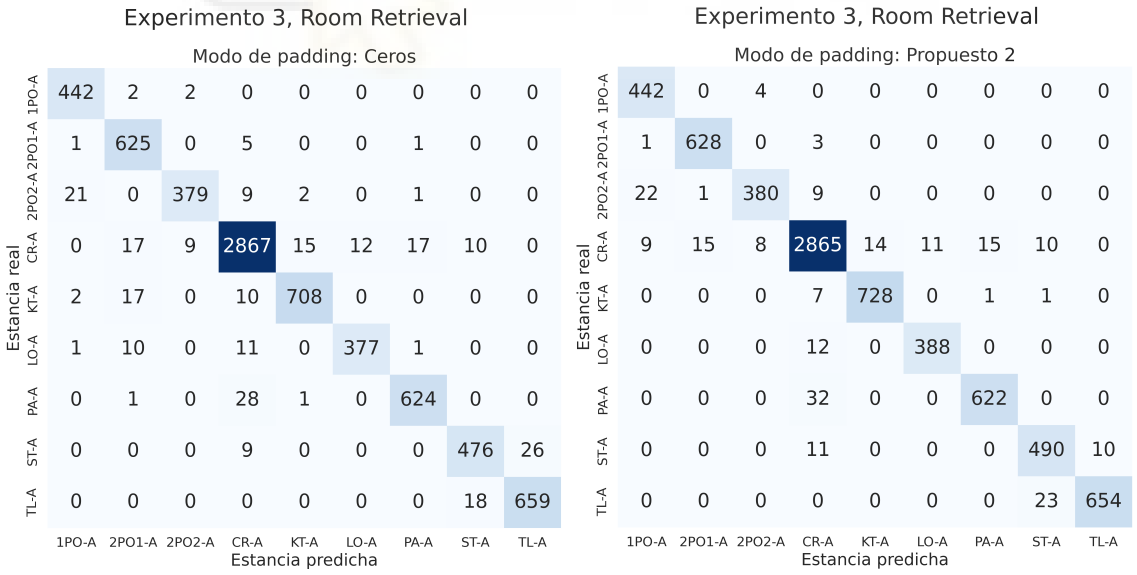
Prueba 1. Tarea de room retrieval

En primer lugar, se ha entrenado el modelo VGG16 con cada uno de los tipos de padding para la tarea de discriminación entre estancias. La Tabla **5-12** muestra la precisión obtenida con cada uno de los tipos. Asimismo, la Figura **5-11** compara las matrices de confusión obtenidas con los dos métodos que han devuelto los mejores resultados.

PRUEBA 1. Room retrieval	**Precisión (%)**			
Modo de padding	**Nublado**	**Noche**	**Soleado**	**Promedio**
Ceros (VGG16)	98.96	97.04	92.81	96.27
Circular	97.46	97.01	84.44	92.97
Transparente	98.81	96.31	87.94	94.35
Propuesto 1	99.00	97.04	91.34	95.80
Propuesto 2	**99.11**	**97.23**	**94.28**	**96.87**

Tabla 5-12: Precisión obtenida con cada modo de padding en la tarea de room retrieval.

La Tabla **5-12** muestra que el modo de padding que ha devuelto los mejores resultados ha sido el segundo método propuesto en este trabajo, que consiste en aplicar un padding circular en las columnas y un padding transparente en las filas. Esto es lógico, pues las columnas situadas en los extremos de las imágenes panorámicas se encuentran visualmente conectadas, y emplear un padding circular en las columnas hace que no se pierda esa continuidad.



(a) (b)

Figura 5-11: Matrices de confusión obtenidas para el experimento 3 con los modos de padding (a) Ceros y (b) Propuesto 2 (circular en columnas y transparente en filas).

Las matrices de confusión obtenidas para los modos de padding Ceros y Propuesto 2, recogidas en la Figura **5-11**, reflejan que el número de errores cometido en la tarea de room retrieval es bastante reducido en ambos casos. No obstante, si se consideran únicamente los errores cometidos entre las habitaciones que no están conectadas en el mapa, esto es, que no comparten información visual, en el caso del padding con ceros se han cometido 58 errores y en el caso del segundo método propuesto se han cometido 21 errores. Por lo tanto, se puede afirmar que con el método Propuesto 2 se obtiene un menor número de errores y de menor magnitud.

Prueba 2. Tarea de place recognition

A continuación, se ha realizado la misma prueba para la tarea de place recognition. La Tabla **5-13** recoge el error geométrico cometido con cada tipo de padding, mientras que la Figura **5-12** muestra el Recall@K.

En este caso, se puede observar como el segundo método propuesto en este trabajo también ha obtenido el menor error para esta tarea. En cuanto al resto de métodos, el padding con ceros y el padding transparente también han obtenido un error bastante pequeño. El modo circular puro ha presentado los peores resultados, lo cual se puede explicar porque no existe ninguna continuidad entre las filas superior e inferior de las imágenes panorámicas, y está provocando la aparición de características ficticias en los mapas de activación resultantes.

PRUEBA 2. Place recognition	**Error de localización (m)**			
Modo de padding	**Nublado**	**Noche**	**Soleado**	**Promedio**
Ceros (VGG16)	0.305	0.345	0.678	0.443
Circular	0.358	**0.312**	1.028	0.566
Transparente	0.315	0.346	0.682	0.448
Propuesto 1	**0.301**	0.316	0.994	0.537
Propuesto 2	0.317	0.354	**0.644**	**0.438**

Tabla 5-13: Error de localización cometido con cada modo de padding en la tarea de place recognition.

En la Figura **5-13** se muestran unos mapas que recogen las predicciones realizadas por la red en la tarea de place recognition. Los puntos azules representan los puntos de captura de las imágenes que componen el modelo visual, mientras que el resto de puntos representan los puntos de captura de las imágenes de test. Las imágenes de test se han representado de distinto color en función de la calidad de la predicción de la red. Si la red es capaz de localizar la imagen de test dentro de los K=1 vecinos

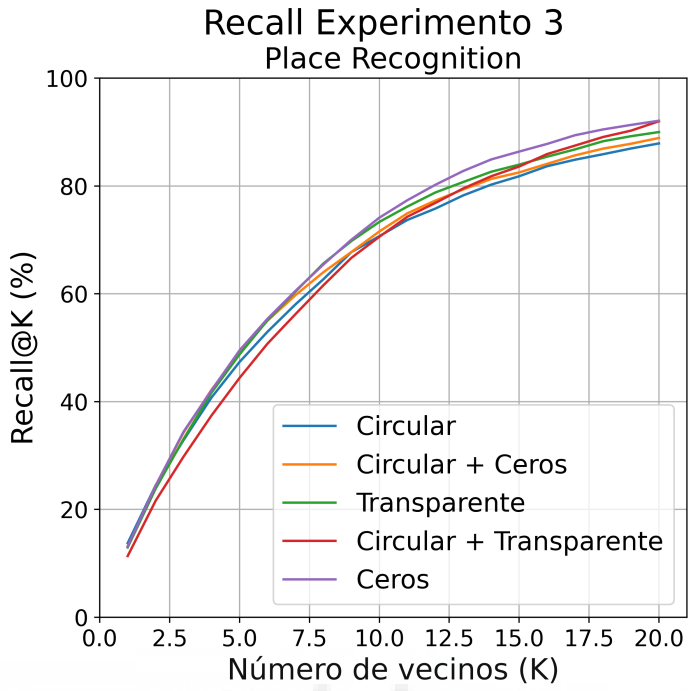


Figura 5-12: Recall@K obtenido en la tarea de place recognition con cada modo de padding.

más cercanos, el punto se pintará de color verde, mientras que si la red no es capaz de localizar la imagen dentro de los K=20 vecinos, el punto tomará un color rojo. Para valores intermedios, los puntos se representarán con tonos amarillentos y anaranjados. Además, para cada imagen de test se ha dibujado una línea que conecta este punto con las coordenadas de la imagen del modelo visual predicha como la más cercana. Por lo tanto, estas líneas indican la magnitud del error de localización para cada imagen de test.

Las gráficas de la Figura **5-13** muestran de una forma visual e intuitiva las predicciones de la red, así como las zonas del mapa en las que el robot ha tenido mayores dificultades para localizarse. Por ejemplo, se puede observar de manera clara que el error cometido bajo condiciones nubladas es inferior al error bajo condiciones nocturnas y significativamente inferior respecto a soleado. Bajo condiciones nubladas, se ha cometido tan solo un único error entre habitaciones no conectadas. Bajo condiciones nocturnas, se producen varios errores entre estancias no conectadas, pero se trata de dos estancias de la misma naturaleza, por lo que se ha producido un problema de visual aliasing. Por último, bajo condiciones soleadas se ha producido un mayor número entre estancias no conectadas. En líneas generales, el grueso de los errores se ha producido en zonas de transición entre habitaciones y en zonas en las

que gira el robot. Esto puede ser debido a que los descriptores obtenidos por la red no son totalmente invariantes a rotación. Sin embargo, el error cometido en todas las condiciones de iluminación es bastante bajo considerando la dificultad de la tarea.

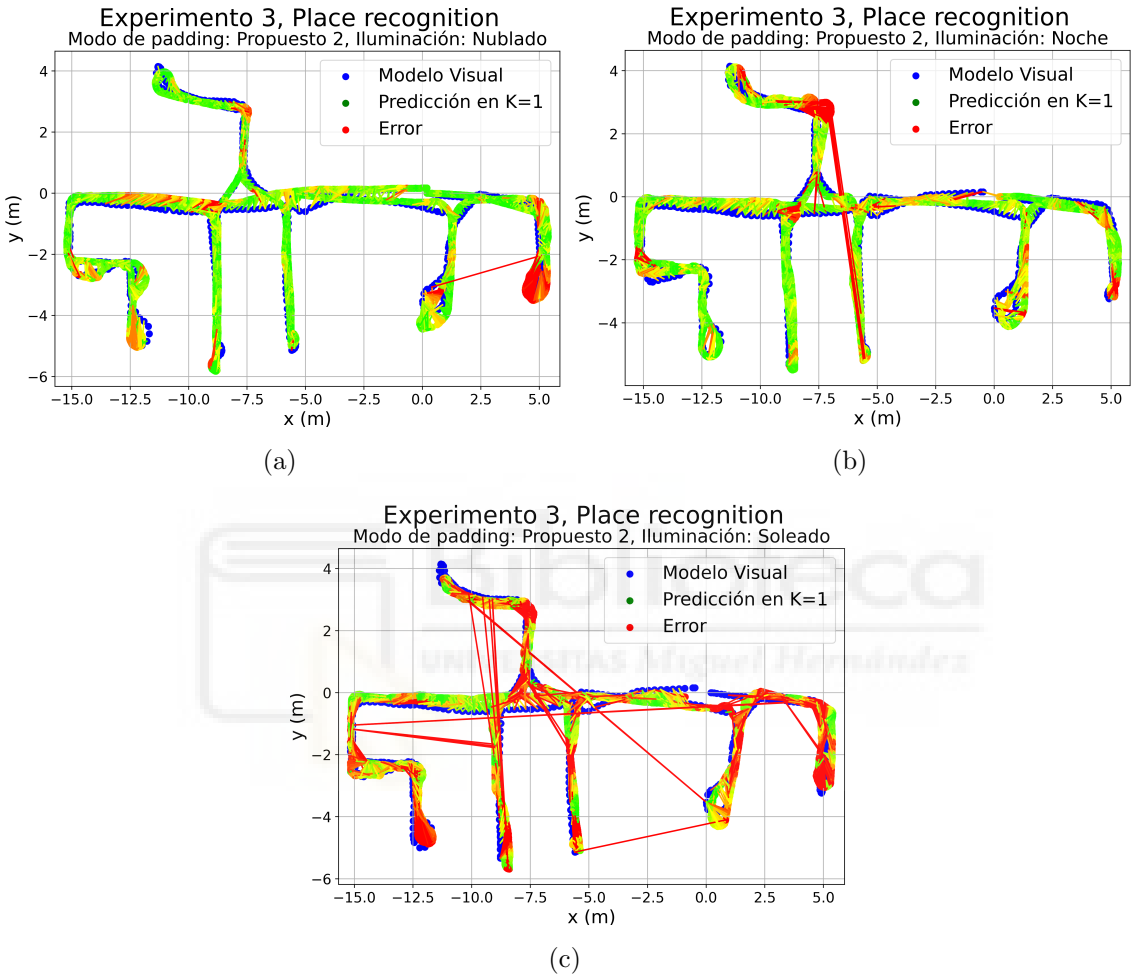


Figura 5-13: Mapas con las predicciones de la red obtenidos con el modo de padding Propuesto 2 (circular en columnas y transparente en filas) para (a) Nublado, (b) Noche y (c) Soleado.

6 Conclusiones y trabajos futuros

En este trabajo se ha propuesto el uso de CNNs que incluyen convoluciones en el dominio de Fourier para abordar el problema de localización visual de robots móviles. Para ello, se ha empleado la arquitectura de la red VGG16 y se han sustituido las capas convolucionales por bloques que incluyen convoluciones en el dominio frecuencial.

Para entrenar y validar estas arquitecturas, se han empleado imágenes omnidireccionales, capturadas con un sistema de visión catadióptico montado en un robot móvil, que se han convertido a formato panorámico. Durante el entrenamiento de la red, se ha empleado una arquitectura de red tripleta, ya que este tipo de arquitecturas han demostrado tener un impacto positivo en el proceso de aprendizaje de las CNNs.

Los experimentos demuestran que, cuando se realiza un entrenamiento breve y con un conjunto de imágenes reducido, el modelo VGG16 original presenta los mejores resultados. No obstante, una de las arquitecturas propuestas, el modelo VGGFFC16, mejora los resultados obtenidos con el modelo VGG16, cuando se realiza un preentrenamiento extenso y variado antes de entrenar la red para la tarea deseada.

Asimismo, se ha explorado el uso de distintos tipos de padding en las capas convolucionales de la red, como son el padding con ceros, circular y transparente, además de dos métodos propuestos en este trabajo. Los resultados indican que uno de los modos de padding propuestos supera los resultados obtenidos con el modo de padding con ceros, empleado en el modelo VGG16.

En cuanto a posibles trabajos futuros, se plantea el uso de imágenes omnidireccionales, sin convertirlas a formato panorámico, para entrenar y validar las arquitecturas propuestas que incluyen convoluciones en el dominio de Fourier. Además, se explorará el uso de otros tipos de convoluciones. Concretamente, se tratará de diseñar máscaras de convolución adaptadas a la distorsión de las imágenes panorámicas.

Bibliografía

[1] M. Alfaro, J. J. Cabrera, L. M. Jiménez, Óscar Reinoso, and L. Payá. Hierarchical localization with panoramic views and triplet loss functions, 2024.

[2] R. Arandjelovic, P. Gronat, A. Torii, T. Pajdla, and J. Sivic. Netvlad: Cnn architecture for weakly supervised place recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5297–5307, 2016. DOI: 10.48550/arXiv.1511.07247.

[3] V. Balaska, L. Bampis, M. Boudourides, and A. Gasteratos. Unsupervised semantic clustering and localization for mobile robotics tasks. *Robotics and Autonomous Systems*, 131:103567, 2020. DOI: 10.1016/j.robot.2020.103567.

[4] J. Cabrera, A. Santo, A. Gil, C. Viegas, and L. Payá. Minkunext: Point cloud-based large-scale place recognition using 3d sparse convolutions. *arXiv preprint arXiv:2403.07593*, 2024. DOI: 10.48550/arXiv.2403.07593.

[5] J. J. Cabrera, S. Cebollada, M. Flores, Ó. Reinoso, and L. Payá. Training, optimization and validation of a cnn for room retrieval and description of omnidirectional images. *SN Computer Science*, 3(4):271, 2022. DOI: 10.1007/s42979-022-01127-8.

[6] D. Cattaneo, M. Vaghi, S. Fontana, A. L. Ballardini, and D. G. Sorrenti. Global visual localization in lidar-maps through shared 2d-3d embedding space. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4365–4371. IEEE, 2020. DOI: 10.1109/ICRA40945.2020.9196859.

[7] S. Cebollada, L. Payá, A. Peidró, W. Mayol, and Ó. Reinoso. Environment modeling and localization from datasets of omnidirectional scenes using machine learning techniques. *Neural Computing and Applications*, 35(22):16487–16508, 2023. DOI: 10.1007/s00521-023-08515-y.

[8] J. Chen, C. Du, Y. Zhang, P. Han, and W. Wei. A clustering-based coverage path planning method for autonomous heterogeneous uavs. *IEEE Transactions on Intelligent Transportation Systems*, 23(12):25546–25556, 2021. DOI: 10.1109/TITS.2021.3066240.

[9] Z. Chen, B. Zhong, G. Li, S. Zhang, and R. Ji. Siamese box adaptive network for visual tracking. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 6668–6677, 2020. DOI: 10.48550/arXiv.2003.06761.

[10] L. Chi, B. Jiang, and Y. Mu. Fast fourier convolution. *Advances in Neural Information Processing Systems*, 33:4479–4488, 2020.

[11] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, 2009. DOI: 10.1109/CVPR.2009.5206848.

[12] D. DeTone, T. Malisiewicz, and A. Rabinovich. Superpoint: Self-supervised interest point detection and description. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pages 224–236, 2018. DOI: 10.48550/arXiv.1712.07629.

[13] M. Dusmanu, I. Rocco, T. Pajdla, M. Pollefeys, J. Sivic, A. Torii, and T. Sattler. D2-net: A trainable cnn for joint detection and description of local features. *arXiv preprint arXiv:1905.03561*, 2019. DOI: 10.48550/arXiv.1905.03561.

[14] S. Fang, K. Li, J. Shao, and Z. Li. Snunet-cd: A densely connected siamese network for change detection of vhr images. *IEEE Geoscience and Remote Sensing Letters*, 19:1–5, 2021. DOI: 10.1109/LGRS.2021.3056416.

[15] M. Flores, D. Valiente, A. Gil, O. Reinoso, and L. Paya. Efficient probability-oriented feature matching using wide field-of-view imaging. *Engineering Applications of Artificial Intelligence*, 107:104539, 2022. DOI: 10.1016/j.engappai.2021.104539.

[16] F. Foroughi, Z. Chen, and J. Wang. A cnn-based system for mobile robot navigation in indoor environments via visual localization with a small dataset. *World Electric Vehicle Journal*, 12(3):134, 2021. DOI: 10.3390/wevj12030134.

[17] K. Fu, D.-P. Fan, G.-P. Ji, Q. Zhao, J. Shen, and C. Zhu. Siamese network for rgb-d salient object detection and beyond. *IEEE transactions on pattern analysis and machine intelligence*, 44(9):5541–5559, 2021. DOI: 10.1109/TPAMI.2021.3073689.

[18] Y. Han and B.-W. Hong. Deep learning based on fourier convolutional neural network incorporating random kernels. *Electronics*, 10(16):2004, 2021. DOI: 10.3390/electronics10162004.

[19] S. Hausler, S. Garg, M. Xu, M. Milford, and T. Fischer. Patch-netvlad: Multi-scale fusion of locally-global descriptors for place recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14141–14152, 2021. DOI: 10.48550/arXiv.2103.01486.

[20] L. Heng, B. Choi, Z. Cui, M. Geppert, S. Hu, B. Kuan, P. Liu, R. Nguyen, Y. C. Yeo, A. Geiger, et al. Project autovision: Localization and 3d scene perception for an autonomous vehicle with a multi-camera system. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 4695–4702. IEEE, 2019. DOI: 10.1109/ICRA.2019.8793949.

[21] A. Hermans, L. Beyer, and B. Leibe. In defense of the triplet loss for person re-identification. *arXiv preprint arXiv:1703.07737*, 2017. DOI: 10.48550/arXiv.1703.07737.

[22] M. Jayasuriya, R. Ranasinghe, and G. Dissanayake. Active perception for outdoor localisation with an omnidirectional camera. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4567–4574, 2020. DOI: 10.1109/IROS45743.2020.9340974.

[23] M. Kent, L. Vasconcelos, S. Ansari, H. Ghanbari, and I. Nenadic. Fourier space approach for convolutional neural network (cnn) electrocardiogram (ecg) classification: A proof-of-concept study. *Journal of Electrocardiology*, 80:24–33, 2023. DOI: 10.1016/j.jelectrocard.2023.04.004.

[24] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. *Commun. ACM*, 60(6):84–90, may 2017. ISSN 0001-0782. DOI: 10.1145/3065386.

[25] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998. DOI: 10.1109/5.726791.

[26] M. Leyva-Vallina, N. Strisciuglio, and N. Petkov. Generalized contrastive optimization of siamese networks for place recognition. *arXiv preprint arXiv:2103.06638*, 2021. DOI: 10.48550/arXiv.2103.06638.

[27] L. Li, X. Kong, X. Zhao, T. Huang, W. Li, F. Wen, H. Zhang, and Y. Liu. Rinet: Efficient 3d lidar-based place recognition using rotation invariant neural network. *IEEE Robotics and Automation Letters*, 7(2):4321–4328, 2022. DOI: 10.1109/LRA.2022.3150499.

[28] H.-Y. Lin, Y.-C. Chung, and M.-L. Wang. Self-localization of mobile robots using a single catadioptric camera with line feature extraction. *Sensors*, 21(14): 4719, 2021. DOI: 10.3390/s21144719.

[29] D. Liu, Y. Cui, L. Yan, C. Mousas, B. Yang, and Y. Chen. Densernet: Weakly supervised visual localization using multi-scale feature aggregation. In *Proceedings of the AAAI conference on artificial intelligence*, volume 35, pages 6101–6109, 2021. DOI: 10.1609/aaai.v35i7.16760.

[30] Z. Luo, L. Zhou, X. Bai, H. Chen, J. Zhang, Y. Yao, S. Li, T. Fang, and L. Quan. Aslfeat: Learning local features of accurate shape and localization. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 6589–6598, 2020. DOI: 10.48550/arXiv.2003.10071.

[31] V. Nair, M. Chatterjee, N. Tavakoli, A. S. Namin, and C. Snoeyink. Optimizing cnn using fast fourier transformation for object recognition. In *2020 19th IEEE International Conference on Machine Learning and Applications (ICMLA)*, pages 234–239, 2020. DOI: 10.1109/ICMLA51294.2020.00046.

[32] P. Neubert and P. Protzel. Local region detector+ cnn based landmarks for practical place recognition in changing environments. In *2015 European Conference on Mobile Robots (ECMR)*, pages 1–6. IEEE, 2015. DOI: 10.1109/ECMR.2015.7324051.

[33] D. Olid, J. M. Fácil, and J. Civera. Single-view place recognition under seasonal changes. *arXiv preprint arXiv:1808.06516*, 2018. DOI: 10.48550/arXiv.1808.06516.

[34] M. Orouskhani, C. Zhu, S. Rostamian, F. S. Zadeh, M. Shafiei, and Y. Orouskhani. Alzheimer's disease detection from structural mri using conditional deep triplet network. *Neuroscience Informatics*, 2(4):100066, 2022. DOI: 10.1016/j.neuri.2022.100066.

[35] L. Payá, F. Amorós, L. Fernández, and Ó. Reinoso. Performance of global-appearance descriptors in map building and localization using omnidirectional vision. *Sensors*, 14(2):3033–3064, 2014. DOI: 10.3390/s140203033.

[36] L. Payá, A. Peidró, F. Amorós, D. Valiente, and O. Reinoso. Modeling environments hierarchically with omnidirectional imaging and global-appearance descriptors. *Remote Sensing*, 10(4), 2018. ISSN 2072-4292. DOI: 10.3390/rs10040522.

[37] H. Pratt, B. Williams, F. Coenen, and Y. Zheng. Fcnn: Fourier convolutional neural networks. In *Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD 2017, Skopje, Macedonia, September 18–22, 2017, Proceedings, Part I 17*, pages 786–798. Springer, 2017. DOI: 10.1007/978-3-319-71249-9_47.

[38] A. Pronobis and B. Caputo. Cold: The cosy localization database. *The International Journal of Robotics Research*, 28(5):588–594, 2009. DOI: 10.1177/0278364909103912.

[39] C. R. Qi, H. Su, K. Mo, and L. J. Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 652–660, 2017. DOI: 10.48550/arXiv.1612.00593.

[40] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 779–788, 2016. DOI: 10.48550/arXiv.1506.02640.

[41] V. Román, L. Paya, S. Cebollada, and O. Reinoso. Creating incremental models of indoor environments through omnidirectional imaging. *Applied Sciences*, 10 (18):6480, 2020. DOI: 10.3390/app10186480.

[42] M. Rostkowska and P. Skrzypczyński. Optimizing appearance-based localization with catadioptric cameras: Small-footprint models for real-time inference on edge devices. *Sensors*, 23(14):6485, 2023. DOI: 10.3390/s23146485.

[43] D. Scaramuzza, A. Martinelli, and R. Siegwart. A flexible technique for accurate omnidirectional camera calibration and structure from motion. In *Fourth IEEE International Conference on Computer Vision Systems (ICVS'06)*, pages 45–45. IEEE, 2006. DOI: 10.1109/ICVS.2006.3.

[44] J. L. Schönberger, M. Pollefeys, A. Geiger, and T. Sattler. Semantic visual localization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[45] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. DOI: 10.48550/arXiv.1409.1556.

[46] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9, 2015. DOI: 10.48550/arXiv.1409.4842.

[47] X. Tao, D. Zhang, W. Ma, Z. Hou, Z. Lu, and C. Adak. Unsupervised anomaly detection for surface defects with dual-siamese network. *IEEE Transactions on Industrial Informatics*, 18(11):7707–7717, 2022. DOI: 10.1109/TII.2022.3142326.

[48] Z. Teed and J. Deng. Droid-slam: Deep visual slam for monocular, stereo, and rgb-d cameras. *Advances in neural information processing systems*, 34:16558–16569, 2021. DOI: 10.48550/arXiv.2108.10869.

[49] M. A. Uy and G. H. Lee. Pointnetvlad: Deep point cloud based retrieval for large-scale place recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018. DOI: 10.48550/arXiv.1804.03492.

[50] T.-H. Wang, H.-J. Huang, J.-T. Lin, C.-W. Hu, K.-H. Zeng, and M. Sun. Omnidirectional cnn for visual place recognition and navigation. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2341–2348. IEEE, 2018. DOI: 10.1109/ICRA.2018.8463173.

[51] C. Won, H. Seok, Z. Cui, M. Pollefeys, and J. Lim. Omnislam: Omnidirectional localization and dense mapping for wide-baseline multi-camera systems. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 559–566, 2020. DOI: 10.1109/ICRA40945.2020.9196695.

[52] L. Xiao, J. Wang, X. Qiu, Z. Rong, and X. Zou. Dynamic-slam: Semantic monocular visual localization and mapping based on deep learning in dynamic environment. *Robotics and Autonomous Systems*, 117:1–16, 2019. DOI: 10.1016/j.robot.2019.03.012.

[53] X. Zhang, L. Wang, Y. Zhao, and Y. Su. Graph-based place recognition in image sequences with cnn features. *Journal of Intelligent & Robotic Systems*, 95:389–403, 2019. DOI: 10.1007/S10846-018-0917-2.

[54] Z. Zhu, S. Peng, V. Larsson, W. Xu, H. Bao, Z. Cui, M. R. Oswald, and M. Pollefeys. Nice-slam: Neural implicit scalable encoding for slam. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12786–12796, 2022. DOI: 10.48550/arXiv.2112.12130.