COMILLAS
UNIVERSIDAD PONTIFICIA
ICAI

GRADO EN INGENIERÍA EN TECNOLOGÍAS DE TELECOMUNICACIÓN

TRABAJO FIN DE GRADO

DISEÑO DE UN SISTEMA DE IDENTIFICACIÓN DE PLAZAS EN PARKINGS DE BICICLETAS BASADOS EN SISTEMAS DE RECONOCIMIENTO DE IMÁGENES CON ALGORITMOS DE INTELIGENCIA ARTIFICIAL

Autor: Rodríguez Monfort, Javier

Director: Rosa Casado, Pablo

Co-Director: Teijeiro Bello, Jesús

Madrid

Junio 2026

Firmado por ROSA CASADO PABLO - ***8736** el día 27/06/2026 con un certificado emitido por AC FNMT Usuarios

Declaración de originalidad

Declaro bajo mi responsabilidad que el Proyecto presentado con el título **DISEÑO DE UN SISTEMA DE IDENTIFICACIÓN DE PLAZAS EN PARKINGS DE BICICLETAS BASADOS EN SISTEMAS DE RECONOCIMIENTO DE IMÁGENES CON ALGORITMOS DE INTELIGENCIA ARTIFICIAL** de la ETS de Ingeniería – ICAI de la Universidad Pontificia Comillas en el curso académico **2025/2026** es de mi autoría y no ha sido presentado con anterioridad a otros efectos. El Proyecto no es plagio de otro, ni total ni parcialmente y la información que ha sido tomada de otros documentos está debidamente referenciada.

Uso de Inteligencia Artificial[1]

Declaro bajo mi responsabilidad que (indicar la opción correcta):

☐ No he utilizado Inteligencia Artificial en la elaboración del presente documento.

☒ He utilizado Inteligencia Artificial en la elaboración del presente documento y/o del Anexo B siempre en las condiciones permitidas por la Universidad Pontificia Comillas, es decir, aplicando el Nivel 2 de la Escala de Evaluación de Perkins et al. (2024): "*La IA puede utilizarse para actividades previas a la tarea, como la lluvia de ideas, la descripción y la investigación inicial. Este nivel se centra en el uso de la IA para la planificación, las síntesis y la generación de ideas, pero las evaluaciones deben hacer hincapié en la capacidad de desarrollar y refinar estas ideas de forma independiente*". En concreto, las Inteligencia Artificial ha sido empleada para:

La Inteligencia Artificial ha sido empleada como apoyo en las siguientes tareas: (1) generación de ideas y estructuración inicial del proyecto (2) revisión y mejora estilística de textos redactados previamente por el autor (3) identificación inicial de referencias posteriormente contrastadas y validadas por el autor En todos los casos, el desarrollo conceptual, la toma de decisiones técnicas y la elaboración final del contenido han sido realizados por el autor.

RODRIGUEZ MONFORT JAVIER - 39466634Z Digitally signed by RODRIGUEZ MONFORT JAVIER - 39466634Z Date: 2026.06.28 11:37:13 +02'00'
Firmado (alumno): Rodríguez Monfort, Javier
Fecha: 25 de junio de 2026

[1] Esta declaración se refiere al uso de la Inteligencia Artificial generativa para realizar los documentos del Proyecto (Anexo B y Memoria). No aplica a Proyectos donde, por su naturaleza, deban emplear inteligencia artificial como parte de los mismos (aplicación de técnicas de aprendizaje automático, redes neuronales, análisis de datos…)

Autorización para la entrega del Proyecto

El Director del Proyecto	El co-Director del Proyecto (si aplica)
	33325309B JESUS TEIJEIRO BELLO (R: B81823890) Digitally signed by 33325309B JESUS TEIJEIRO BELLO (R: B81823890) Date: 2026.06.27 15:01:23 +02'00'
Fdo: Rosa Casado, Pablo	Fdo: Teijeiro Bello, Jesús
Fecha: 25/06/2026	Fecha: 25/06/2026

GRADO EN INGENIERÍA EN TECNOLOGÍAS DE TELECOMUNICACIÓN

TRABAJO FIN DE GRADO

DISEÑO DE UN SISTEMA DE IDENTIFICACIÓN DE PLAZAS EN PARKINGS DE BICICLETAS BASADOS EN SISTEMAS DE RECONOCIMIENTO DE IMÁGENES CON ALGORITMOS DE INTELIGENCIA ARTIFICIAL

Autor: Rodríguez Monfort, Javier

Director: Rosa Casado, Pablo

Co-Director: Teijeiro Bello, Jesús

Madrid

Junio 2026

Agradecimientos

Muchas gracias a todos los que me han apoyado en la elaboración de este proyecto.

DISEÑO DE UN SISTEMA DE IDENTIFICACIÓN DE PLAZAS EN PARKINGS DE BICICLETAS BASADOS EN SISTEMAS DE RECONOCIMIENTO DE IMÁGENES CON ALGORITMOS DE INTELIGENCIA ARTIFICIAL

Autor: Rodríguez Monfort, Javier

Director: Rosa Casado, Pablo

Co-Director: Teijeiro Bello, Jesús

Entidad Colaboradora: Don Cicleto

RESUMEN DEL PROYECTO

Este proyecto desarrolla e implementa un sistema inteligente para la detección automática de ocupación de plazas de aparcamiento para bicicletas, utilizando técnicas de visión artificial e inteligencia artificial embebida.

Los resultados obtenidos demuestran la viabilidad técnica de emplear plataformas embebidas de bajo consumo para ejecutar algoritmos de detección de objetos en tiempo real, proporcionando información útil sobre la disponibilidad de plazas mediante una interfaz web accesible en remoto.

Palabras clave: bicicleta, inteligencia artificial embebida, detección de objetos, Yolo, Edge AI y estacionamiento.

1. Introducción

Durante los últimos años, la bicicleta ha experimentado un crecimiento significativo como medio de transporte urbano debido a sus beneficios medioambientales, económicos y sociales. La reducción de emisiones contaminantes, la disminución de la congestión del tráfico y la promoción de hábitos de vida saludables han impulsado el desarrollo de políticas orientadas a fomentar la movilidad ciclista en numerosas ciudades.

Este incremento en el uso de la bicicleta ha provocado, de forma paralela, una creciente necesidad de infraestructuras adecuadas para su estacionamiento. Entre estas infraestructuras destacan los denominados *bici hangares* o aparcamientos colectivos protegidos, diseñados para proporcionar mayor seguridad frente a robos y actos vandálicos. Sin embargo, uno de los principales inconvenientes asociados a este tipo de instalaciones es la dificultad de conocer previamente la disponibilidad de plazas libres antes de desplazarse hasta ellas. Esta situación puede generar pérdidas de tiempo, reducir la comodidad del servicio e incluso desincentivar el uso de la bicicleta como medio de transporte habitual.

Paralelamente, el avance de las tecnologías de visión artificial, inteligencia artificial y sistemas embebidos ha permitido el desarrollo de soluciones capaces de interpretar automáticamente información visual y monitorizar entornos físicos en tiempo real. En particular, los modelos de detección de objetos basados en redes neuronales [6] y [7]

profundas han demostrado un elevado potencial para aplicaciones de supervisión automática en ámbitos como la videovigilancia inteligente, la robótica o las infraestructuras urbanas inteligentes.

En este contexto, el presente Trabajo Fin de Grado plantea el desarrollo de un sistema inteligente capaz de determinar automáticamente el estado de ocupación de un aparcamiento de bicicletas mediante el análisis de imágenes capturadas por una cámara convencional. El sistema desarrollado combina técnicas de visión artificial, inteligencia artificial embebida y comunicaciones de red, ejecutándose sobre una plataforma hardware de bajo consumo basada en el procesador Renesas RZ/V2L.

El objetivo principal del Proyecto consiste en demostrar la viabilidad de desplegar algoritmos de detección de objetos en tiempo real sobre plataformas embebidas con recursos limitados, proporcionando simultáneamente información útil a los usuarios a través de una interfaz web accesible remotamente. De este modo, el trabajo contribuye tanto al desarrollo tecnológico en el ámbito de las ciudades inteligentes como a la promoción de modelos de movilidad urbana más sostenibles.

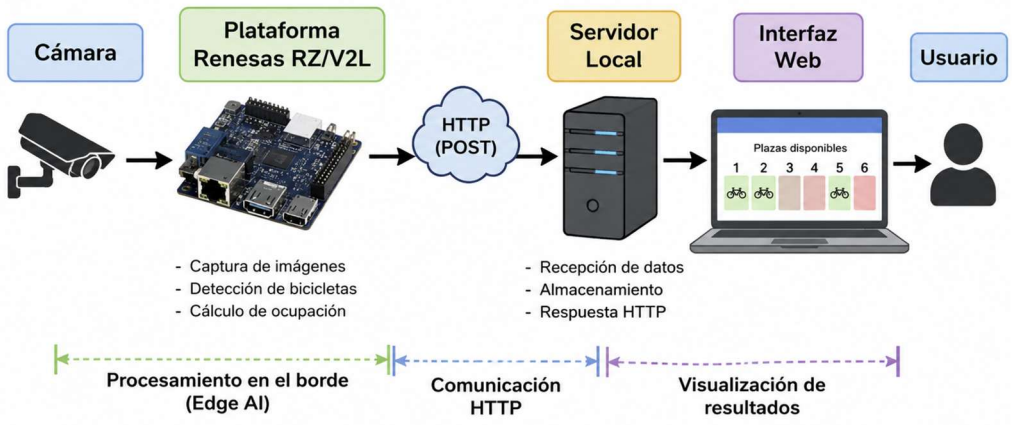


Ilustración 1 Flujo general de comunicación utilizado

2. Estado del arte y fundamentos tecnológicos

La monitorización automática de plazas de aparcamiento ha sido ampliamente estudiada en el ámbito de los vehículos automóviles. Sin embargo, la aplicación de estas tecnologías a aparcamientos de bicicletas continúa siendo limitada y presenta características particulares derivadas del menor tamaño de los objetos, la mayor variabilidad de posiciones y la necesidad de minimizar costes de instalación.

Tradicionalmente, la detección de ocupación se ha realizado mediante sensores físicos individuales, tales como sensores ultrasónicos, sensores infrarrojos, lazos inductivos o sensores de presión. Aunque estas soluciones pueden ofrecer buenos resultados en determinados entornos, presentan inconvenientes relacionados con la complejidad de instalación, el coste asociado al despliegue de múltiples sensores y la necesidad de mantenimiento periódico. Además, algunos de estos sistemas presentan limitaciones cuando las bicicletas no se encuentran correctamente alineadas o cuando las condiciones ambientales son desfavorables.

Como alternativa, la visión artificial permite supervisar simultáneamente múltiples plazas utilizando una única cámara, reduciendo significativamente la complejidad del sistema y mejorando su escalabilidad. El uso combinado de cámaras y algoritmos de inteligencia artificial posibilita además la adaptación del sistema a diferentes configuraciones de aparcamiento sin necesidad de modificaciones hardware significativas.

Dentro del ámbito de la detección automática de objetos, la familia de modelos YOLO (*You Only Look Once*) se ha consolidado como una de las soluciones de referencia debido a su capacidad para combinar elevados niveles de precisión con tiempos de inferencia reducidos. A diferencia de arquitecturas tradicionales como Faster R-CNN, que realizan la detección en múltiples etapas, YOLO aborda simultáneamente la localización y clasificación de objetos mediante una única pasada sobre la imagen de entrada.

Esta característica resulta especialmente relevante en aplicaciones que requieren procesamiento en tiempo real y funcionamiento sobre dispositivos embebidos con recursos computacionales limitados. Gracias a ello, YOLO ha sido ampliamente utilizado en ámbitos como la videovigilancia inteligente, los vehículos autónomos, la robótica móvil y numerosas aplicaciones industriales.

Por este motivo, se seleccionó un modelo basado en la arquitectura YOLO para realizar la detección automática de bicicletas dentro de las imágenes capturadas por la cámara. La elección se fundamentó principalmente en su elevada velocidad de inferencia, su capacidad para ejecutarse sobre plataformas embebidas y su amplio uso en aplicaciones reales de detección de objetos en tiempo real.

Asimismo, el sistema desarrollado hace uso de la biblioteca OpenCV, una de las herramientas de código abierto más utilizadas en el ámbito de la visión artificial. Esta biblioteca proporciona un amplio conjunto de funciones para la adquisición, procesamiento y análisis de imágenes, facilitando la integración entre la cámara y el modelo de inteligencia artificial.

3. Diseño e implementación del sistema

La solución desarrollada se ha concebido como una prueba de concepto funcional orientada a validar la integración de tecnologías de visión artificial, inteligencia artificial embebida y comunicaciones dentro de una aplicación real de movilidad sostenible.

Desde el punto de vista hardware, el sistema se basa en una plataforma embebida Renesas RZ/V2L, seleccionada debido a sus capacidades específicas para la ejecución de algoritmos de inteligencia artificial en el borde (*Edge AI*). Este dispositivo ejecuta un sistema operativo Linux, proporcionando un entorno flexible para el desarrollo y despliegue de aplicaciones de visión artificial.

La arquitectura implementada se compone de diversos elementos principales. En primer lugar, una cámara es la encargada de capturar imágenes del aparcamiento monitorizado. Estas imágenes son enviadas a la plataforma embebida, donde se ejecuta el modelo de inteligencia artificial basado en YOLO. Posteriormente, la información generada es procesada para determinar el estado de ocupación y transmitida a una interfaz web accesible a través de la red local.

El flujo de funcionamiento comienza con la adquisición periódica de imágenes desde la cámara. Cada imagen es procesada por el modelo de detección, que identifica automáticamente las bicicletas presentes en la escena y genera las correspondientes regiones delimitadoras o *bounding boxes*, junto con un nivel de confianza asociado a cada detección.

A partir del número total de bicicletas detectadas, el sistema estima automáticamente el número de plazas ocupadas y disponibles. Esta información se almacena temporalmente y se transmite posteriormente a una interfaz web sencilla que permite consultar remotamente el estado actual del aparcamiento.

Durante la fase experimental se optó por utilizar un servidor web local accesible desde la misma red local utilizada en las pruebas. Esta aproximación permitió simplificar el desarrollo del sistema, evitando la necesidad de desplegar infraestructuras públicas accesibles desde Internet. La interfaz implementada posee un carácter demostrativo y está orientada principalmente a validar la viabilidad técnica de la solución propuesta. No obstante, permite mostrar en tiempo real la información esencial necesaria para la gestión del aparcamiento.

4. Validación experimental y resultados obtenidos

Una vez implementado el sistema completo, se realizaron diferentes pruebas experimentales destinadas a verificar el correcto funcionamiento de todos los módulos desarrollados.

En primer lugar, se validó el funcionamiento de la plataforma embebida y la correcta ejecución del modelo de inteligencia artificial sobre el procesador Renesas RZ/V2L. Las pruebas realizadas demostraron que la plataforma es capaz de ejecutar satisfactoriamente el modelo de detección y procesar las imágenes capturadas en tiempos compatibles con aplicaciones de monitorización en tiempo real.

Posteriormente, se comprobó el correcto funcionamiento del flujo completo de comunicaciones entre la plataforma embebida y el servidor web local. Los resultados obtenidos confirmaron que la información generada por el sistema puede transmitirse y visualizarse remotamente de forma fiable.

Desde el punto de vista funcional, el sistema fue capaz de detectar automáticamente la presencia de bicicletas en las imágenes capturadas y estimar correctamente el número de plazas ocupadas y disponibles en la mayoría de las situaciones analizadas. Aunque la solución desarrollada presenta un carácter demostrativo, los resultados obtenidos permiten considerar alcanzados satisfactoriamente los objetivos inicialmente planteados.

Asimismo, se comprobó que la utilización de técnicas de detección de objetos basadas en inteligencia artificial constituye una alternativa flexible frente a soluciones tradicionales basadas en sensores físicos individuales instalados en cada plaza. Mediante una única cámara es posible supervisar simultáneamente múltiples plazas de estacionamiento, reduciendo significativamente la complejidad de instalación y favoreciendo la escalabilidad del sistema.

Finalmente, el desarrollo del proyecto ha puesto de manifiesto la importancia de la integración entre hardware, software y comunicaciones dentro del diseño de sistemas IoT

inteligentes. El trabajo realizado no se limita exclusivamente a la aplicación de algoritmos de inteligencia artificial, sino que aborda el despliegue completo de una solución funcional que integra procesamiento embebido, comunicaciones en red y visualización remota de la información generada.

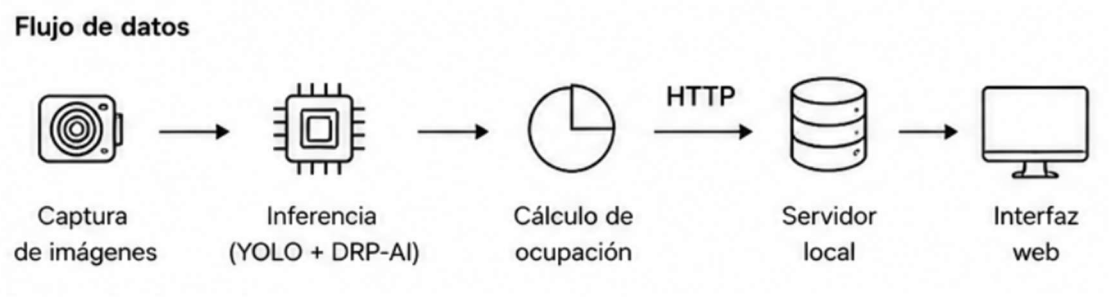


Ilustración 2 Diagrama del Sistema Propuesto

5. Conclusiones

En este Trabajo Fin de Grado se ha diseñado e implementado un sistema inteligente basado en visión artificial para monitorizar, de forma automática, la ocupación de aparcamientos de bicicletas.

El proyecto demuestra que las tecnologías de inteligencia artificial embebida y de bajo consumo son totalmente capaces de procesar imágenes en tiempo real, generando datos de disponibilidad, útiles que se pueden consultar mediante una aplicación web.

Además, el Proyecto contribuye a la promoción de modelos de movilidad sostenible mediante la incorporación de tecnologías inteligentes en infraestructuras urbanas destinadas al uso de la bicicleta.

En conjunto, este trabajo establece una base sólida para futuros desarrollos orientados al despliegue de soluciones inteligentes dentro del ámbito de las ciudades inteligentes y la gestión avanzada de infraestructuras urbanas sostenibles.

ABSTRACT

Intelligent System for the Automatic Detection of Bicycle Parking Occupancy Using Computer Vision and Embedded Artificial Intelligence

Project Summary

This project develops and implements an intelligent system for the automatic detection of bicycle parking occupancy using computer vision and embedded artificial intelligence techniques.

The results obtained demonstrate the technical feasibility of using low-power embedded platforms to execute real-time object detection algorithms, providing useful information about parking availability through a remotely accessible web interface.

Keywords: bicycle, embedded artificial intelligence, object detection, YOLO, Edge AI, and parking.

1. Introduction

In recent years, bicycles have experienced significant growth as a means of urban transportation due to their environmental, economic, and social benefits. The reduction of pollutant emissions, the decrease in traffic congestion, and the promotion of healthier lifestyles have encouraged the development of policies aimed at fostering cycling mobility in many cities.

This increase in bicycle usage has simultaneously generated a growing need for suitable parking infrastructures. Among these infrastructures, bicycle hangars or protected collective parking facilities stand out, as they are designed to provide greater security against theft and vandalism. However, one of the main drawbacks associated with these facilities is the difficulty of knowing in advance whether parking spaces are available before travelling to them. This situation may lead to time losses, reduce user convenience, and even discourage the use of bicycles as a regular means of transportation.

At the same time, advances in computer vision, artificial intelligence, and embedded systems have enabled the development of solutions capable of automatically interpreting visual information and monitoring physical environments in real time. In particular, object detection models based on deep neural networks have shown great potential for automatic monitoring applications in fields such as intelligent video surveillance, robotics, and smart urban infrastructures.

Within this context, this Final Degree Project proposes the development of an intelligent system capable of automatically determining the occupancy status of a bicycle parking facility through the analysis of images captured by a conventional camera. The developed system combines computer vision techniques, embedded artificial intelligence, and network communications, running on a low-power hardware platform based on the Renesas RZ/V2L processor.

The main objective of the project is to demonstrate the feasibility of deploying real-time object detection algorithms on embedded platforms with limited computational resources

while simultaneously providing useful information to users through a remotely accessible web interface. In this way, the project contributes both to technological development in the field of smart cities and to the promotion of more sustainable urban mobility models.

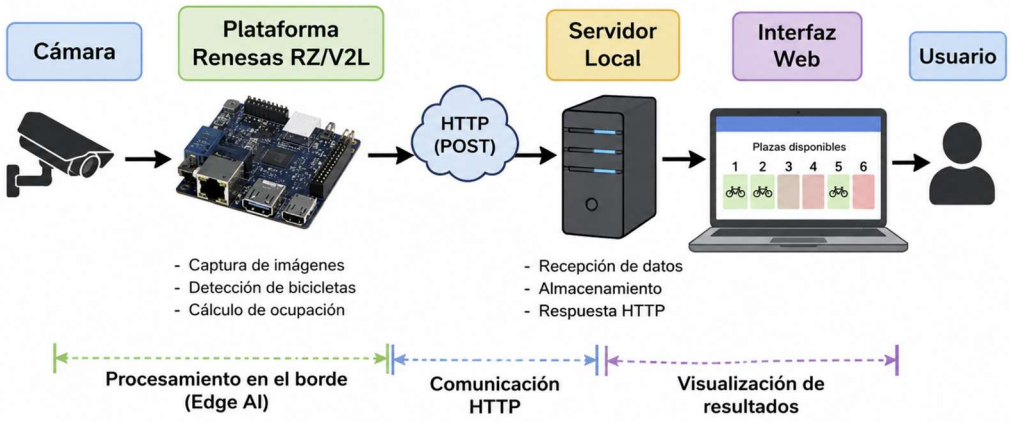


Ilustración 2 Flujo general de comunicación utilizado

2. State of the Art and Technological Foundations

Automatic parking occupancy monitoring has been widely studied in the context of motor vehicles. However, the application of these technologies to bicycle parking facilities remains limited and presents particular challenges due to the smaller size of the objects, the greater variability in bicycle positions, and the need to minimize installation costs.

Traditionally, occupancy detection has been performed using individual physical sensors, such as ultrasonic sensors, infrared sensors, inductive loops, or pressure sensors. Although these solutions can provide good results in certain environments, they present drawbacks related to installation complexity, the cost associated with deploying multiple sensors, and the need for periodic maintenance. Furthermore, some of these systems exhibit limitations when bicycles are not properly aligned or when environmental conditions are unfavourable.

As an alternative, computer vision makes it possible to monitor multiple parking spaces simultaneously using a single camera, significantly reducing system complexity and improving scalability. The combined use of cameras and artificial intelligence algorithms also enables the adaptation of the system to different parking configurations without requiring significant hardware modifications.

Within the field of automatic object detection, the YOLO (*You Only Look Once*) family of models has become one of the reference solutions due to its ability to combine high levels of accuracy with reduced inference times. Unlike traditional architectures such as Faster R-CNN, which perform detection in multiple stages, YOLO simultaneously addresses object localization and classification through a single pass over the input image.

This characteristic is particularly relevant in applications requiring real-time processing and operation on embedded devices with limited computational resources. As a result, YOLO has been widely adopted in areas such as intelligent video surveillance, autonomous vehicles, mobile robotics, and numerous industrial applications.

For this reason, a model based on the YOLO architecture was selected to perform the automatic detection of bicycles in the images captured by the camera. This choice was mainly motivated by its high inference speed, its capability to run on embedded platforms, and its extensive use in real-world real-time object detection applications.

Additionally, the developed system makes use of the OpenCV library, one of the most widely used open-source tools in the field of computer vision. This library provides a comprehensive set of functions for image acquisition, processing, and analysis, facilitating the integration between the camera and the artificial intelligence model.

3. System Design and Implementation

The developed solution was conceived as a functional proof of concept aimed at validating the integration of computer vision, embedded artificial intelligence, and communication technologies within a real sustainable mobility application.

From a hardware perspective, the system is based on the Renesas RZ/V2L embedded platform, selected due to its specific capabilities for running artificial intelligence algorithms at the edge (*Edge AI*). This device runs a Linux operating system, providing a flexible environment for the development and deployment of computer vision applications.

The implemented architecture consists of several main components. First, a camera is responsible for capturing images of the monitored parking facility. These images are sent to the embedded platform, where the YOLO-based artificial intelligence model is executed. Subsequently, the generated information is processed to determine the occupancy status and transmitted to a web interface accessible through the local network.

The operating workflow begins with the periodic acquisition of images from the camera. Each image is processed by the detection model, which automatically identifies the bicycles present in the scene and generates the corresponding *bounding boxes*, together with a confidence level associated with each detection.

Based on the total number of detected bicycles, the system automatically estimates the number of occupied and available parking spaces. This information is temporarily stored and subsequently transmitted to a web interface that allows users to remotely check the current status of the parking facility.

During the experimental phase, a local web server accessible through the same home network used during testing was employed. This approach simplified system development by avoiding the need to deploy public Internet-accessible infrastructures. The implemented interface has a demonstrative nature and is primarily intended to validate the technical feasibility of the proposed solution. Nevertheless, it allows the real-time visualization of the essential information required for parking management.

4. Experimental Validation and Results

Once the complete system had been implemented, several experimental tests were conducted to verify the correct operation of all the developed modules.

First, the operation of the embedded platform and the correct execution of the artificial intelligence model on the Renesas RZ/V2L processor were validated. The tests demonstrated that the platform is capable of successfully running the detection model and processing the captured images within time constraints compatible with real-time monitoring applications.

Subsequently, the correct operation of the complete communication flow between the embedded platform and the local web server was verified. The obtained results confirmed that the information generated by the system can be reliably transmitted and remotely displayed.

From a functional perspective, the system was able to automatically detect the presence of bicycles in the captured images and correctly estimate the number of occupied and available parking spaces in most of the analyzed situations. Although the developed solution has a demonstrative nature, the obtained results indicate that the initially established objectives have been satisfactorily achieved.

Furthermore, it was verified that the use of artificial intelligence-based object detection techniques constitutes a flexible alternative to traditional solutions based on individual physical sensors installed in each parking space. By using a single camera, it is possible to simultaneously monitor multiple parking spaces, significantly reducing installation complexity and improving system scalability.

Finally, the development of the project highlighted the importance of integrating hardware, software, and communications in the design of intelligent IoT systems. The work carried out is not limited solely to the application of artificial intelligence algorithms but also addresses the complete deployment of a functional solution integrating embedded processing, network communications, and remote visualization of the generated information.

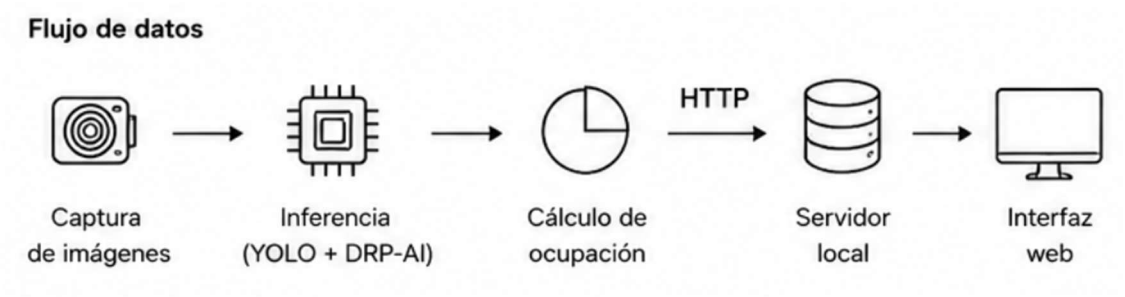


Ilustración 2 Diagrama del Sistema Propuesto

5. Conclusions

This Final Degree Project has designed and implemented an intelligent computer vision-based system to automatically monitor bicycle parking occupancy.

The project demonstrates that low-power embedded artificial intelligence technologies are fully capable of processing images in real time and generating useful availability information that can be accessed through a web application.

Furthermore, the project contributes to the promotion of sustainable mobility models through the incorporation of intelligent technologies into urban infrastructures intended for bicycle use.

Overall, this work establishes a solid foundation for future developments aimed at deploying intelligent solutions within the field of smart cities and the advanced management of sustainable urban infrastructures.

ÍNDICE DE LA MEMORIA

CAPÍTULO 1. INTRODUCCIÓN

La movilidad urbana sostenible se ha convertido en uno de los principales retos de las ciudades modernas debido al crecimiento de la población, la congestión del tráfico y la necesidad de reducir las emisiones contaminantes asociadas al transporte. En este contexto, el uso de la bicicleta como medio de transporte alternativo ha experimentado un crecimiento significativo durante los últimos años, impulsado por las políticas de movilidad sostenible desarrolladas por administraciones públicas y organismos internacionales. La bicicleta presenta numerosas ventajas frente a los medios de transporte motorizados, entre las que destacan la reducción de emisiones de gases de efecto invernadero, la disminución del ruido urbano, la mejora de la salud de los usuarios y la reducción de la congestión del tráfico.

El incremento del uso de la bicicleta ha provocado una creciente demanda de infraestructuras específicas destinadas a su estacionamiento. Sin embargo, en numerosas ocasiones los usuarios encuentran dificultades para conocer la disponibilidad de plazas libres en los aparcamientos de bicicletas antes de desplazarse hasta ellos. Esta situación genera pérdidas de tiempo, reduce la comodidad del servicio y puede desincentivar el uso de este medio de transporte.

Paralelamente, el desarrollo de las tecnologías asociadas a la visión artificial, la inteligencia artificial y el Internet de las Cosas (IoT) ha permitido la creación de sistemas inteligentes capaces de monitorizar entornos físicos de forma automática y en tiempo real. En particular, los avances experimentados en el campo de las redes neuronales [6] y [7] convolucionales y de la detección automática de objetos han abierto la posibilidad de desarrollar soluciones de bajo coste capaces de identificar elementos presentes en una escena mediante el análisis de imágenes capturadas por cámaras convencionales.

Dentro de este ámbito, los modelos de detección de objetos basados en la familia YOLO (*You Only Look Once*) han demostrado un elevado rendimiento en aplicaciones que requieren procesamiento en tiempo real, combinando una alta precisión con tiempos de inferencia reducidos. Estas características hacen que este tipo de modelos resulten especialmente adecuados para su despliegue en dispositivos embebidos con recursos computacionales limitados.

Este Trabajo Fin de Grado (en adelante, el TFG o el Proyecto) aborda el desarrollo de un sistema inteligente para la detección automática de ocupación de plazas de aparcamiento para bicicletas mediante técnicas de visión artificial e inteligencia artificial. El sistema propuesto utiliza una cámara para capturar imágenes de un aparcamiento de bicicletas y un modelo de detección de objetos basado en YOLO para identificar la presencia de bicicletas en cada una de las plazas monitorizadas. La información obtenida se procesa localmente en una plataforma embebida basada en el procesador Renesas RZ/V2L, permitiendo determinar en tiempo real el número de plazas ocupadas y disponibles.

Además de la detección automática, el sistema incorpora un mecanismo de comunicación que permite transmitir la información obtenida a una interfaz web funcional, desde la que un usuario puede consultar de forma remota el estado de ocupación del aparcamiento. De esta manera, se demuestra la viabilidad de integrar tecnologías de inteligencia artificial, sistemas embebidos y comunicaciones en una aplicación orientada a la mejora de la movilidad sostenible.

El Proyecto se plantea como una prueba de concepto funcional cuyo objetivo principal es validar la capacidad de una plataforma embebida de bajo consumo para ejecutar algoritmos de visión artificial en tiempo real y proporcionar información útil a los usuarios de un aparcamiento de bicicletas. Asimismo, el trabajo permite analizar las limitaciones y desafíos asociados al despliegue de este tipo de soluciones en entornos reales, sirviendo como punto de partida para futuros desarrollos en el ámbito de las ciudades inteligentes y la gestión inteligente de infraestructuras urbanas

CAPÍTULO 2. FUNDAMENTOS TEÓRICOS Y TECNOLÓGICOS

2.1 Introducción

El desarrollo de sistemas capaces de interpretar automáticamente información visual constituye uno de los campos de mayor crecimiento dentro de la ingeniería informática y de telecomunicación. Los avances producidos durante los últimos años en visión artificial, aprendizaje profundo y sistemas embebidos han permitido la creación de soluciones capaces de analizar imágenes en tiempo real y extraer información relevante del entorno sin necesidad de intervención humana [1], [2].

Estas tecnologías han encontrado aplicación en numerosos ámbitos, entre los que destacan la automatización industrial, los sistemas de videovigilancia inteligente, la robótica, la conducción autónoma y las infraestructuras urbanas inteligentes. En todos estos casos, la capacidad para detectar y reconocer objetos de forma automática resulta fundamental para la toma de decisiones y la monitorización de los entornos observados [1].

El Proyecto se centra en el desarrollo de un sistema capaz de detectar automáticamente la ocupación de plazas de aparcamiento para bicicletas mediante técnicas de visión artificial e inteligencia artificial. Para ello se emplea una cámara encargada de capturar imágenes del aparcamiento y un modelo de detección de objetos basado en redes neuronales profundas que permite identificar la presencia de bicicletas en cada una de las plazas monitorizadas.

La ejecución del modelo de inteligencia artificial se realiza sobre una plataforma embebida basada en el procesador Renesas RZ/V2L, diseñado específicamente para aplicaciones de visión artificial en el borde de la red (*Edge AI*). De este modo, el procesamiento de las imágenes se realiza localmente, reduciendo la dependencia de servidores externos y permitiendo una respuesta en tiempo real.

Con el objetivo de comprender adecuadamente las tecnologías empleadas durante el desarrollo del Proyecto, en este capítulo se presentan los fundamentos teóricos relacionados con la visión artificial, el aprendizaje profundo, las redes neuronales convolucionales, la detección de objetos y las plataformas de inteligencia artificial embebida utilizadas en la implementación final del sistema.

2.2 Visión Artificial

La visión artificial es una disciplina de la informática cuyo objetivo consiste en permitir que los sistemas computacionales obtengan información útil a partir de imágenes y secuencias de vídeo. Para ello se emplean algoritmos capaces de procesar los datos visuales capturados por cámaras u otros sensores ópticos, transformándolos en información interpretable para la realización de tareas específicas [1].

Los primeros sistemas de visión artificial se basaban principalmente en técnicas clásicas de procesamiento digital de imágenes. Entre las operaciones más habituales se encontraban la detección de bordes, la segmentación de regiones, la identificación de formas geométricas y la extracción manual de características. Aunque estos métodos siguen utilizándose en determinadas aplicaciones, presentan limitaciones importantes cuando las condiciones de iluminación, perspectiva o apariencia de los objetos varían significativamente.

La evolución de la capacidad de procesamiento de los sistemas informáticos y el desarrollo de nuevas técnicas basadas en inteligencia artificial han impulsado un crecimiento notable de la visión artificial durante las últimas décadas. Actualmente, los sistemas modernos son capaces de identificar, clasificar y localizar objetos de manera automática con elevados niveles de precisión, incluso en entornos complejos y dinámicos [2].

Las aplicaciones de la visión artificial abarcan numerosos ámbitos tecnológicos. Entre ellos destacan la automatización industrial, los sistemas de videovigilancia inteligente, la robótica, los vehículos autónomos, la medicina y las infraestructuras inteligentes. En todos estos casos, la capacidad de interpretar automáticamente la información contenida en una imagen permite mejorar la eficiencia de los procesos y reducir la necesidad de intervención humana.

Dentro del ámbito de las ciudades inteligentes, la visión artificial ha adquirido una relevancia creciente debido a su capacidad para monitorizar infraestructuras urbanas y analizar el comportamiento de los usuarios en tiempo real. Gracias a ello, resulta posible desarrollar sistemas capaces de controlar el tráfico, gestionar plazas de aparcamiento o supervisar espacios públicos mediante el uso de cámaras convencionales y algoritmos de análisis de imagen.

En el Proyecto, la visión artificial constituye la tecnología fundamental sobre la que se desarrolla la solución propuesta. Mediante el análisis de las imágenes capturadas por una cámara instalada frente a un aparcamiento de bicicletas, el sistema es capaz de detectar automáticamente la presencia de bicicletas y determinar el estado de ocupación de las distintas plazas monitorizadas.

2.3 Inteligencia Artificial y Aprendizaje Profundo

La inteligencia artificial (IA) puede definirse como el conjunto de técnicas y métodos destinados a desarrollar sistemas capaces de realizar tareas que tradicionalmente requieren inteligencia humana. Entre estas tareas se encuentran el reconocimiento de patrones, la clasificación de información, la toma de decisiones o la resolución de problemas complejos [3].

Dentro de la inteligencia artificial, una de las áreas que ha experimentado un mayor crecimiento durante las últimas décadas es el aprendizaje automático o *Machine Learning*. Este enfoque permite que los sistemas aprendan automáticamente a partir de datos de entrenamiento, identificando relaciones y patrones sin necesidad de programar explícitamente todas las reglas que describen el problema [4].

A partir del desarrollo del aprendizaje automático surgió el aprendizaje profundo o *Deep Learning*, una rama basada en el empleo de redes neuronales artificiales formadas por múltiples capas de procesamiento. Estas arquitecturas permiten aprender representaciones jerárquicas de los datos, de forma que las capas iniciales extraen características simples mientras que las capas más profundas son capaces de identificar patrones progresivamente más complejos [5].

El crecimiento del aprendizaje profundo ha estado estrechamente relacionado con el aumento de la capacidad computacional disponible y con la aparición de grandes conjuntos de datos para entrenamiento. Gracias a ello, las redes neuronales profundas han alcanzado niveles de rendimiento superiores a los obtenidos mediante técnicas tradicionales en tareas relacionadas con la visión artificial, el reconocimiento de voz y el procesamiento del lenguaje natural.

En el ámbito de la visión artificial, el aprendizaje profundo ha supuesto un cambio significativo respecto a los métodos clásicos basados en extracción manual de características. Mientras que en los enfoques tradicionales era necesario diseñar explícitamente los atributos que describían un objeto, los modelos actuales son capaces de aprender automáticamente dichas características directamente a partir de las imágenes de entrenamiento [5].

Esta capacidad de aprendizaje automático resulta especialmente útil en aplicaciones reales, donde las condiciones de iluminación, perspectiva o apariencia de los objetos pueden variar considerablemente. Gracias a ello, los sistemas basados en aprendizaje profundo ofrecen una mayor robustez y capacidad de generalización frente a escenarios complejos.

La mayoría de los sistemas modernos de detección de objetos emplean actualmente modelos basados en aprendizaje profundo. Entre ellos destaca la familia de modelos YOLO (“*You Only Look Once”*), utilizada en el Proyecto para la detección automática de bicicletas en las imágenes capturadas por la cámara.

2.4 Detección de Objetos

La detección de objetos es una de las tareas más importantes dentro del ámbito de la visión artificial. Su objetivo consiste en identificar la presencia de determinados objetos dentro de una imagen y determinar simultáneamente su localización espacial. A diferencia de los problemas de clasificación de imágenes, donde únicamente se determina la categoría principal representada en una escena, la detección de objetos permite identificar múltiples elementos presentes en una misma imagen y conocer su posición exacta [8].

Para representar la ubicación de los objetos detectados se utilizan habitualmente rectángulos delimitadores o *bounding boxes*. Cada uno de estos rectángulos encierra un objeto identificado por el sistema y va acompañado de una etiqueta que indica su clase, así como de una puntuación de confianza asociada a la detección realizada. De esta forma, el resultado final de un detector de objetos no solo informa sobre qué elementos aparecen en la imagen, sino también sobre dónde se encuentran.

Los primeros sistemas de detección de objetos se basaban en técnicas clásicas de procesamiento de imágenes y extracción manual de características. Sin embargo, estos métodos presentaban limitaciones significativas cuando los objetos aparecían bajo diferentes condiciones de iluminación, escala o perspectiva. La aparición de las redes neuronales profundas permitió superar gran parte de estas limitaciones, dando lugar a una nueva generación de detectores con niveles de precisión significativamente superiores.

Actualmente existen diversas arquitecturas de detección de objetos basadas en aprendizaje profundo. Entre las más conocidas se encuentran Faster R-CNN, SSD (*Single Shot Detector*) y la familia de modelos YOLO (*You Only Look Once*) [9]. Cada una de estas soluciones presenta diferentes compromisos entre precisión, velocidad de inferencia y requisitos computacionales.

En aplicaciones donde resulta necesario procesar imágenes en tiempo real, la velocidad de inferencia constituye un factor especialmente importante. Este requisito es habitual en sistemas de vigilancia, vehículos autónomos, robótica móvil o infraestructuras inteligentes, donde la información obtenida debe estar disponible con una latencia reducida. En este contexto, las arquitecturas de detección de una sola etapa (*single-stage detectors*) han adquirido una relevancia creciente debido a su capacidad para ofrecer tiempos de procesamiento muy reducidos manteniendo niveles de precisión adecuados.

El sistema desarrollado en el TFG emplea técnicas de detección de objetos para identificar automáticamente las bicicletas presentes en las imágenes capturadas por la cámara. A partir de las detecciones obtenidas es posible determinar el estado de ocupación de cada una de las plazas monitorizadas y calcular el número de plazas disponibles en tiempo real.

La arquitectura seleccionada para llevar a cabo esta tarea pertenece a la familia YOLO, ampliamente utilizada en aplicaciones de visión artificial en tiempo real debido a su equilibrio entre precisión y velocidad de procesamiento. Sus características y funcionamiento se describen con mayor detalle en el apartado siguiente.

2.5 Arquitectura YOLO

YOLO (*You Only Look Once*) es una familia de modelos de detección de objetos basada en redes neuronales convolucionales diseñada para realizar tareas de localización y clasificación de objetos en tiempo real [10]. Desde la publicación de la primera versión en 2016, YOLO se ha convertido en una de las arquitecturas más utilizadas en aplicaciones de visión artificial debido a su capacidad para combinar una elevada velocidad de procesamiento con niveles de precisión competitivos frente a otros detectores [11].

La principal diferencia entre YOLO y otros métodos de detección radica en su filosofía de funcionamiento. Mientras que arquitecturas tradicionales como Faster R-CNN dividen el problema en varias etapas independientes, YOLO aborda la detección como un único problema de regresión. Esto permite que la red procese la imagen completa en una sola pasada (*single forward pass*), obteniendo simultáneamente la localización de los objetos y su clasificación [10].

Durante el proceso de inferencia, la imagen de entrada es analizada por la red neuronal, que genera una serie de predicciones compuestas por coordenadas espaciales,

probabilidades de pertenencia a distintas clases y niveles de confianza asociados a cada detección. Posteriormente, se aplican algoritmos de filtrado para eliminar detecciones redundantes y conservar únicamente aquellas que presentan una mayor probabilidad de corresponder a objetos reales.

Una de las características más relevantes de YOLO es su capacidad para procesar imágenes a gran velocidad. Esta propiedad resulta especialmente importante en aplicaciones que requieren funcionamiento en tiempo real, donde la latencia del sistema constituye un factor crítico. Gracias a ello, YOLO ha sido ampliamente utilizado en ámbitos como la videovigilancia inteligente, los vehículos autónomos, la robótica móvil, los sistemas industriales y las aplicaciones de inteligencia artificial embebida.

A lo largo de los años han aparecido diferentes versiones de la arquitectura, cada una de ellas incorporando mejoras destinadas a incrementar la precisión, reducir los requisitos computacionales o facilitar el entrenamiento de los modelos. Estas evoluciones han permitido ampliar significativamente el ámbito de aplicación de la familia YOLO, consolidándola como una de las soluciones de referencia dentro del campo de la detección de objetos.

En el Proyecto se emplea un modelo basado en la arquitectura YOLO para detectar bicicletas dentro de las imágenes capturadas por la cámara. La elección de esta familia de modelos se fundamenta principalmente en tres factores: su elevada velocidad de inferencia, su capacidad para ejecutarse sobre plataformas embebidas con recursos limitados y su amplio uso en aplicaciones de detección de objetos en tiempo real [11].

El resultado generado por el modelo consiste en un conjunto de detecciones asociadas a bicicletas presentes en la escena. Cada detección incorpora la posición del objeto dentro de la imagen mediante un rectángulo delimitador (*bounding box*) y un nivel de confianza que indica la probabilidad de que dicha detección corresponda efectivamente a una bicicleta. A partir de esta información es posible determinar automáticamente la ocupación de las distintas plazas monitorizadas por el sistema.

La utilización de YOLO constituye uno de los elementos fundamentales de la solución desarrollada, ya que permite combinar precisión, velocidad de procesamiento y compatibilidad con sistemas embebidos, requisitos esenciales para la implementación propuesta en este proyecto.

2.6 OpenCV

OpenCV (*Open Source Computer Vision Library*) es una biblioteca de código abierto especializada en visión artificial y procesamiento de imágenes. Desde su lanzamiento inicial, se ha convertido en una de las herramientas más utilizadas tanto en entornos académicos como industriales gracias a la gran cantidad de funcionalidades que ofrece y a su compatibilidad con múltiples plataformas y lenguajes de programación [12] [13]

La biblioteca proporciona un amplio conjunto de algoritmos destinados a la adquisición, procesamiento y análisis de imágenes y vídeo. Entre sus funcionalidades destacan la captura de imágenes desde cámaras, el procesamiento digital de imágenes, la detección de características visuales, el seguimiento de objetos, la calibración de cámaras y la

integración con modelos de aprendizaje profundo. Estas capacidades permiten desarrollar sistemas completos de visión artificial utilizando una única plataforma software.

Una de las principales ventajas de OpenCV es su eficiencia computacional. Muchas de sus funciones han sido optimizadas para ejecutarse sobre diferentes arquitecturas hardware, permitiendo procesar imágenes en tiempo real incluso en sistemas con recursos limitados. Esta característica ha favorecido su adopción en aplicaciones embebidas, robótica, automatización industrial y sistemas inteligentes de monitorización.

La biblioteca también incorpora mecanismos que facilitan la integración de modelos de inteligencia artificial desarrollados mediante diferentes *frameworks* de aprendizaje profundo. Gracias a ello, OpenCV puede utilizarse como interfaz entre los algoritmos de detección y los dispositivos encargados de capturar las imágenes, simplificando considerablemente el desarrollo de aplicaciones de visión artificial.

En el Proyecto, OpenCV se emplea como biblioteca de soporte para la gestión y procesamiento de imágenes. Concretamente, se utiliza para la adquisición de imágenes procedentes de la cámara, la manipulación de los datos visuales y determinadas operaciones auxiliares necesarias para la ejecución de la aplicación. Asimismo, permite realizar tareas de visualización de resultados, como la representación de las detecciones obtenidas por el modelo mediante rectángulos delimitadores y etiquetas asociadas a cada bicicleta identificada por el sistema.

La integración de OpenCV simplifica el desarrollo de la aplicación al proporcionar una interfaz unificada para el tratamiento de imágenes y la interacción con los dispositivos de captura utilizados durante el proyecto.

La amplia documentación disponible, su carácter multiplataforma y su integración con tecnologías de inteligencia artificial han convertido a OpenCV en una de las herramientas más utilizadas en proyectos de visión artificial.

2.7 Inteligencia Artificial en el Borde (*Edge AI*)

La Inteligencia Artificial en el Borde, conocida habitualmente como *Edge AI*, consiste en ejecutar algoritmos de inteligencia artificial directamente en dispositivos próximos a la fuente de datos, en lugar de enviar toda la información a servidores remotos o plataformas en la nube para su procesamiento [14]. Este enfoque resulta especialmente relevante en aplicaciones donde se trabaja con sensores, cámaras o dispositivos conectados que generan información de forma continua.

En un sistema basado exclusivamente en computación en la nube, las imágenes capturadas por una cámara tendrían que transmitirse a un servidor externo para ser analizadas. Aunque este modelo puede ofrecer una gran capacidad de cálculo, también introduce ciertas limitaciones, como una mayor latencia, dependencia de la conectividad, consumo de ancho de banda y posibles problemas relacionados con la privacidad de los datos [14].

Frente a este planteamiento, el *Edge AI* permite analizar los datos localmente y enviar únicamente los resultados obtenidos. En el caso de un sistema de detección de ocupación,

esto significa que no es necesario transmitir continuamente imágenes completas del aparcamiento, sino únicamente información procesada, como el número de plazas ocupadas y libres.

La reducción de la latencia es una de las principales ventajas de *Edge AI*. Al ejecutar el modelo de inteligencia artificial en el propio dispositivo embebido, el sistema puede obtener resultados en tiempo real sin depender de la respuesta de un servidor externo. Esta característica resulta especialmente importante en aplicaciones de monitorización automática, donde la información debe actualizarse con rapidez [15].

Otra ventaja importante es la disminución del tráfico de datos. En aplicaciones basadas en visión artificial, las imágenes y secuencias de vídeo pueden generar un volumen elevado de información. Si todo ese contenido se envía a la nube, la red puede convertirse en un cuello de botella. El procesamiento local reduce esta carga al transmitir únicamente los datos finales necesarios para la visualización o almacenamiento [15].

La creciente adopción de *Edge AI* ha impulsado el desarrollo de plataformas hardware específicamente diseñadas para ejecutar modelos de inteligencia artificial de forma eficiente en dispositivos embebidos. Estas plataformas incorporan aceleradores especializados capaces de realizar operaciones de inferencia con un consumo energético reducido y manteniendo tiempos de respuesta compatibles con aplicaciones en tiempo real.

Entre las aplicaciones más habituales de *Edge AI* destacan los sistemas de videovigilancia inteligente, la automatización industrial, los vehículos autónomos, la monitorización de infraestructuras y los sistemas de gestión urbana inteligente. En todos estos casos, la capacidad de procesar la información localmente permite mejorar la eficiencia del sistema y reducir la dependencia de servicios externos.

En el ámbito de la visión artificial, el paradigma *Edge AI* resulta especialmente adecuado para tareas de detección y clasificación de objetos. Al ejecutar los modelos de inteligencia artificial directamente sobre el dispositivo de procesamiento, es posible analizar imágenes en tiempo real y transmitir únicamente la información relevante obtenida tras la inferencia.

El sistema desarrollado en el Proyecto sigue esta filosofía de funcionamiento. La detección de bicicletas se realiza localmente sobre una plataforma embebida Renesas RZ/V2L, evitando el envío continuo de imágenes a servidores externos. Como resultado, únicamente se transmiten los datos necesarios para informar del estado de ocupación del aparcamiento, reduciendo el tráfico de red y permitiendo una respuesta prácticamente inmediata ante cambios en la disponibilidad de plazas.

Por este motivo, la elección de una plataforma compatible con aplicaciones *Edge AI* constituye un aspecto fundamental dentro de la arquitectura propuesta. En el siguiente apartado se describen las principales características de la plataforma Renesas RZ/V2L utilizada durante el desarrollo del Proyecto.

Tradicionalmente, muchos sistemas de inteligencia artificial se han basado en arquitecturas *cloud*, donde los datos generados por los dispositivos son enviados a servidores remotos para su procesamiento. En este modelo, la capacidad de cálculo se concentra en centros de datos externos, mientras que los dispositivos desplegados en campo actúan principalmente como elementos de captura de información.

Por el contrario, en una arquitectura *Edge AI* el procesamiento se realiza directamente sobre el dispositivo que genera o recibe los datos. De este modo, únicamente es necesario transmitir la información resultante del análisis, reduciendo significativamente el volumen de comunicaciones requerido.

2.8 Plataforma Renesas RZ/V2L

La plataforma Renesas RZ/V2L (en adelante, la *Plataforma*) es un procesador diseñado específicamente para aplicaciones de visión artificial e inteligencia artificial en sistemas embebidos. Pertenece a la familia RZ/V de *Renesas Electronics*, una serie de dispositivos orientados a la ejecución eficiente de algoritmos de inferencia en el borde (*Edge AI*), permitiendo el procesamiento local de imágenes y datos sin necesidad de recurrir continuamente a servicios externos de computación en la nube [16].

Una de las principales características de la Plataforma es la incorporación de un acelerador hardware especializado denominado DRP-AI (*Dynamically Reconfigurable Processor for Artificial Intelligence*). Este acelerador ha sido desarrollado por *Renesas Electronics* con el objetivo de optimizar la ejecución de modelos de inteligencia artificial manteniendo un consumo energético reducido. Gracias a esta arquitectura, es posible ejecutar tareas de detección y clasificación de objetos en tiempo real sobre dispositivos embebidos con recursos limitados [17].

Además de las capacidades de aceleración para inteligencia artificial, la Plataforma integra una arquitectura basada en procesadores ARM Cortex-A55 de 64 bits capaces de ejecutar sistemas operativos Linux embebidos. Esta combinación permite desarrollar aplicaciones complejas que integran procesamiento de imágenes, comunicaciones de red, interfaces gráficas y algoritmos de inteligencia artificial dentro de un único dispositivo [16].

La plataforma dispone también de múltiples interfaces de comunicación y periféricos orientados a aplicaciones industriales y sistemas inteligentes. Entre ellos destacan las interfaces Ethernet, USB, HDMI y las conexiones destinadas a cámaras, lo que facilita su integración en proyectos de visión artificial y monitorización automática.

2.8.1 Acelerador DRP-AI

El acelerador DRP-AI constituye uno de los elementos diferenciales de la familia RZ/V. Su función principal consiste en ejecutar operaciones asociadas a redes neuronales de forma más eficiente que una implementación basada exclusivamente en CPU.

La utilización de hardware especializado permite reducir significativamente los tiempos de inferencia y el consumo energético del sistema. Como consecuencia, resulta posible desplegar modelos de inteligencia artificial directamente sobre dispositivos embebidos manteniendo un rendimiento adecuado para aplicaciones en tiempo real [17].

Esta capacidad resulta especialmente relevante en sistemas de visión artificial, donde cada imagen debe ser procesada con rapidez para proporcionar información actualizada sobre el entorno observado.

2.8.2 Aplicaciones de la Plataforma

Las capacidades de procesamiento e inteligencia artificial de la Plataforma han favorecido su utilización en una amplia variedad de aplicaciones. Entre las más habituales se encuentran los sistemas de videovigilancia inteligente, la automatización industrial, los sistemas de transporte inteligente, la robótica y las aplicaciones de ciudades inteligentes (*Smart Cities*).

En todos estos casos, la posibilidad de ejecutar algoritmos de inteligencia artificial directamente sobre el dispositivo permite reducir la latencia y mejorar la autonomía del sistema, siguiendo la filosofía *Edge AI* descrita en el apartado anterior.

2.8.3 Utilización de la Plataforma en el Proyecto

La Plataforma constituye el núcleo del sistema desarrollado. Sobre ella se ejecuta la aplicación encargada de procesar las imágenes capturadas por la cámara y realizar la detección automática de bicicletas mediante técnicas de visión artificial aceleradas por DRP-AI.

Una vez detectadas las bicicletas, la propia plataforma calcula el estado de ocupación de las plazas monitorizadas y genera la información que posteriormente será enviada a un servidor local mediante comunicaciones HTTP. De esta forma, la Plataforma concentra las funciones de captura de datos, procesamiento de inteligencia artificial y generación de resultados, actuando como elemento central de la arquitectura propuesta.

2.9 Comunicaciones HTTP y Visualización Web

La comunicación entre dispositivos constituye un elemento fundamental en numerosos sistemas embebidos y aplicaciones de Internet de las Cosas (*IoT*). Una vez procesada la información obtenida por los sensores o sistemas de visión artificial, resulta necesario transmitir los resultados a otros dispositivos para su almacenamiento, visualización o utilización en procesos posteriores.

Entre los diferentes protocolos disponibles, HTTP (*HyperText Transfer Protocol*) [18][19] se ha consolidado como uno de los estándares más utilizados para el intercambio de información entre sistemas conectados a una red. Su amplia compatibilidad con diferentes plataformas y lenguajes de programación ha favorecido su adopción tanto en aplicaciones web tradicionales como en sistemas embebidos modernos.

HTTP se basa en una arquitectura cliente-servidor en la que un dispositivo cliente envía una petición a un servidor, el cual procesa la solicitud y devuelve una respuesta. Este modelo permite separar claramente las tareas de generación y presentación de la información, facilitando el desarrollo y mantenimiento de sistemas distribuidos.

Las peticiones HTTP pueden utilizar diferentes métodos dependiendo de la operación que se desee realizar. Entre los más habituales destacan:

- **GET**, utilizado para solicitar información almacenada en un servidor.
- **POST**, empleado para enviar información desde un cliente hacia un servidor.
- **PUT**, destinado a actualizar información existente.
- **DELETE**, utilizado para eliminar recursos almacenados.

En aplicaciones de monitorización y sistemas *IoT*, el método POST es especialmente útil para transmitir datos generados por dispositivos embebidos hacia servidores encargados de almacenar o visualizar dicha información.

Por otra parte, los servidores web permiten publicar información accesible mediante navegadores convencionales. Gracias a ellos es posible desarrollar interfaces de visualización que muestran en tiempo real el estado de un sistema sin necesidad de instalar aplicaciones específicas en los dispositivos de los usuarios.

La combinación de protocolos HTTP y servidores web constituye una solución sencilla y ampliamente utilizada para la implementación de sistemas de supervisión remota. Esta arquitectura permite desacoplar el procesamiento de los datos de su representación visual, favoreciendo la modularidad y escalabilidad de la solución desarrollada.

2.9.1 Aplicación en el Sistema Desarrollado

En el Proyecto se ha empleado una arquitectura cliente-servidor basada en HTTP para comunicar la plataforma de procesamiento con el sistema de visualización.

La Plataforma actúa como cliente HTTP, enviando periódicamente información relativa al estado de ocupación del aparcamiento a un servidor local conectado a la misma red. Esta información incluye los resultados obtenidos tras el proceso de detección de bicicletas y el cálculo de plazas ocupadas y disponibles.

El servidor local recibe las peticiones generadas por la plataforma embebida y actualiza la información mostrada en una interfaz web accesible mediante un navegador. De esta forma, los usuarios pueden consultar el estado del aparcamiento sin necesidad de interactuar directamente con el sistema de procesamiento.

La utilización de un servidor local simplifica el desarrollo del prototipo y permite validar el funcionamiento completo del sistema sin depender de servicios externos o infraestructuras *cloud*. Asimismo, esta arquitectura facilita futuras ampliaciones orientadas a la incorporación de nuevos servicios o mecanismos de acceso remoto.

La Figura 1 muestra el flujo general de comunicación utilizado en la solución desarrollada.

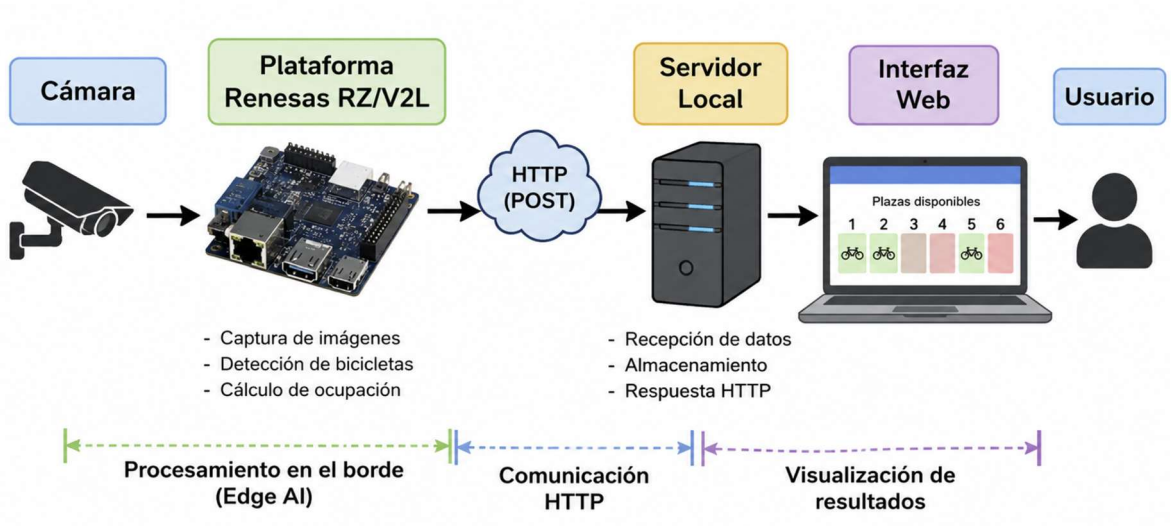


Ilustración 3 Flujo general de comunicación utilizado

CAPÍTULO 3. ESTADO DEL ARTE

3.1 Introducción

Durante los últimos años, el desarrollo de infraestructuras urbanas inteligentes ha impulsado la aparición de numerosos sistemas destinados a monitorizar el uso de espacios públicos de forma automática [20]. Entre estas aplicaciones destacan los sistemas de gestión de aparcamientos, cuyo objetivo consiste en proporcionar información actualizada sobre la disponibilidad de plazas y facilitar la utilización eficiente de los recursos disponibles.

La detección automática de ocupación puede abordarse mediante diferentes tecnologías, cada una de ellas con ventajas e inconvenientes específicos. Tradicionalmente, las soluciones más extendidas se han basado en sensores físicos instalados en cada plaza de estacionamiento. Sin embargo, el avance de la visión artificial y de las técnicas de aprendizaje profundo ha favorecido la aparición de sistemas capaces de realizar la misma tarea utilizando únicamente cámaras y algoritmos de procesamiento de imagen [21].

La creciente disponibilidad de plataformas embebidas con capacidades de inteligencia artificial ha permitido además trasladar parte del procesamiento a dispositivos situados junto a la fuente de datos, reduciendo la dependencia de servidores externos y facilitando el despliegue de soluciones de bajo coste. Como consecuencia, los sistemas basados en visión artificial se han convertido en una alternativa cada vez más atractiva para aplicaciones de monitorización urbana [21].

En este capítulo se analizan las principales tecnologías utilizadas actualmente para la detección automática de ocupación, prestando especial atención a las soluciones basadas en visión artificial y a las arquitecturas de detección de objetos empleadas en aplicaciones de monitorización inteligente. El objetivo es identificar las características, ventajas y limitaciones de las distintas alternativas existentes y justificar la elección de la solución desarrollada en el Proyecto.

3.2 Sistemas de Detección Basados en Sensores

Las primeras soluciones desarrolladas para la monitorización automática de plazas de aparcamiento se basaban en la instalación de sensores físicos capaces de detectar la presencia o ausencia de vehículos en una posición determinada. Estos sistemas continúan utilizándose ampliamente en aparcamientos públicos y privados debido a su simplicidad conceptual y a la fiabilidad que ofrecen cuando se encuentran correctamente instalados y calibrados.

Entre las tecnologías más empleadas destacan los sensores magnéticos, los sensores ultrasónicos y los sensores infrarrojos. Los sensores magnéticos detectan variaciones en el campo magnético provocadas por la presencia de elementos metálicos, mientras que los sensores ultrasónicos utilizan ondas acústicas para medir la distancia entre el sensor y el objeto situado sobre la plaza. Por su parte, los sensores infrarrojos permiten detectar la ocupación mediante la interrupción o reflexión de un haz de luz [22].

La principal ventaja de estos sistemas es que proporcionan una detección directa e independiente para cada plaza monitorizada. Además, suelen presentar un comportamiento relativamente estable frente a cambios de iluminación o condiciones meteorológicas, factores que pueden afectar a otros métodos de detección.

Sin embargo, las soluciones basadas en sensores también presentan diversas limitaciones. En primer lugar, requieren la instalación de un dispositivo físico en cada plaza de aparcamiento, lo que incrementa significativamente el coste de despliegue cuando el número de plazas es elevado. Asimismo, la instalación y mantenimiento de estos dispositivos puede resultar compleja, especialmente en entornos exteriores donde los sensores están expuestos a condiciones ambientales adversas.

Otra limitación importante es la escalabilidad del sistema. A medida que aumenta el número de plazas monitorizadas, también aumenta el número de sensores necesarios, así como el cableado, las tareas de mantenimiento y la infraestructura asociada. Como consecuencia, el coste total de la solución crece de forma aproximadamente proporcional al tamaño del aparcamiento.

Frente a estas limitaciones, los sistemas basados en visión artificial permiten supervisar múltiples plazas utilizando una única cámara. Esta característica reduce considerablemente la cantidad de hardware necesario y proporciona una mayor flexibilidad para adaptar el sistema a diferentes configuraciones de aparcamiento. Gracias a los avances recientes en inteligencia artificial y detección de objetos, estas soluciones han alcanzado niveles de precisión que las convierten en una alternativa viable a los sistemas tradicionales basados exclusivamente en sensores [21].

Por estos motivos, en los últimos años se ha observado un creciente interés por el desarrollo de sistemas de monitorización basados en visión artificial, especialmente en aplicaciones relacionadas con las ciudades inteligentes y la gestión eficiente de infraestructuras urbanas.

3.3 Sistemas de Detección Basados en Visión Artificial

La evolución de la visión artificial y de las técnicas de aprendizaje profundo ha favorecido la aparición de sistemas capaces de monitorizar automáticamente espacios de aparcamiento utilizando únicamente cámaras y algoritmos de procesamiento de imagen [21]. Frente a los enfoques tradicionales basados en sensores físicos, estas soluciones permiten supervisar simultáneamente múltiples plazas mediante un único dispositivo de captura, reduciendo significativamente la cantidad de hardware necesaria para el despliegue del sistema.

Los primeros sistemas de visión artificial para monitorización de aparcamientos se apoyaban en técnicas clásicas de procesamiento de imágenes. Estos métodos analizaban características como el color, la textura o la diferencia entre imágenes consecutivas para determinar si una plaza se encontraba ocupada o libre. Aunque podían ofrecer resultados aceptables en entornos controlados, su rendimiento tendía a degradarse cuando se producían cambios de iluminación, sombras, condiciones meteorológicas adversas o modificaciones en la posición de la cámara.

La aparición de las redes neuronales convolucionales [6] y [7] y del aprendizaje profundo permitió superar gran parte de estas limitaciones. Los modelos modernos son capaces de aprender automáticamente las características visuales asociadas a los objetos presentes en una escena, ofreciendo una mayor capacidad de generalización frente a variaciones del entorno [5]. Como consecuencia, la precisión de los sistemas de detección basados en visión artificial ha mejorado notablemente durante los últimos años.

Actualmente, una gran parte de las soluciones de monitorización inteligente utilizan detectores de objetos basados en aprendizaje profundo para identificar automáticamente vehículos, bicicletas, peatones u otros elementos de interés dentro de una imagen. Estos sistemas no solo permiten determinar la ocupación de una plaza, sino que también pueden proporcionar información adicional relacionada con el tipo de objeto detectado o su posición dentro de la escena.

Otra ventaja importante de la visión artificial es su flexibilidad. Mientras que los sistemas basados en sensores requieren la instalación de un dispositivo específico para cada plaza monitorizada, una única cámara puede supervisar simultáneamente un conjunto amplio de plazas. Esta característica reduce los costes de instalación y mantenimiento, especialmente en aplicaciones donde el número de plazas es elevado o la configuración del aparcamiento puede modificarse con el tiempo.

No obstante, los sistemas basados en visión artificial también presentan ciertos desafíos. La calidad de las detecciones puede verse afectada por factores como la iluminación, las condiciones meteorológicas, las oclusiones parciales o la presencia de elementos que dificulten la visibilidad de los objetos. Por este motivo, la selección de algoritmos robustos y plataformas de procesamiento adecuadas resulta fundamental para garantizar un funcionamiento fiable en entornos reales.

Debido a estas ventajas, la visión artificial se ha consolidado como una de las tecnologías más prometedoras para el desarrollo de sistemas inteligentes de monitorización. Esta tendencia ha impulsado la adopción de arquitecturas de detección de objetos como YOLO, ampliamente utilizadas en aplicaciones que requieren análisis visual en tiempo real y que constituyen la base tecnológica de la solución desarrollada en el Proyecto.

3.4 Uso de YOLO en Sistemas de Monitorización Inteligente

La evolución de los modelos de detección de objetos ha favorecido la aparición de nuevas aplicaciones de monitorización capaces de analizar automáticamente información visual en tiempo real. Entre las diferentes arquitecturas desarrolladas durante los últimos años, la familia YOLO (*You Only Look Once*) se ha consolidado como una de las soluciones más utilizadas debido a su capacidad para combinar velocidad de inferencia y precisión de detección [10][11].

Una de las principales ventajas de YOLO es su capacidad para procesar imágenes completas mediante una única pasada a través de la red neuronal. Esta característica reduce significativamente el tiempo necesario para realizar una detección y permite su utilización en aplicaciones donde la información debe actualizarse de forma continua. Como consecuencia, YOLO ha encontrado aplicación en ámbitos como la

videovigilancia, la robótica, los vehículos autónomos y los sistemas de monitorización basados en visión artificial [10].

La disponibilidad de versiones cada vez más eficientes ha favorecido además su integración en plataformas embebidas con recursos limitados. Gracias a ello, resulta posible ejecutar modelos de detección de objetos directamente sobre dispositivos *Edge AI* sin necesidad de recurrir a servidores externos para realizar las inferencias. Esta capacidad resulta especialmente interesante en aplicaciones donde la latencia, la privacidad de los datos o la disponibilidad de la conexión constituyen factores relevantes [15].

Otra característica destacable de YOLO es su flexibilidad para detectar diferentes categorías de objetos dentro de una misma imagen. Dependiendo del modelo utilizado y de los datos empleados durante el entrenamiento, el sistema puede identificar personas, vehículos, bicicletas y numerosos elementos adicionales. Esta versatilidad ha contribuido a su adopción en proyectos relacionados con la gestión inteligente de infraestructuras y la monitorización automática de espacios públicos [11].

En el contexto del Proyecto, la elección de YOLO se fundamenta en su compatibilidad con la Plataforma y en su capacidad para proporcionar detecciones en tiempo real sobre hardware embebido. Estas características permiten desarrollar una solución capaz de identificar bicicletas y determinar automáticamente el estado de ocupación de las plazas monitorizadas sin necesidad de utilizar infraestructuras de procesamiento externas.

3.5 Comparativa de Soluciones Existentes

Las diferentes tecnologías utilizadas para la detección automática de ocupación presentan ventajas e inconvenientes que condicionan su aplicación en función de los requisitos del sistema. La elección de una solución determinada depende de factores como el coste de despliegue, la precisión requerida, la facilidad de instalación, la escalabilidad y los recursos hardware disponibles.

Los sistemas basados en sensores físicos destacan por su simplicidad conceptual y por ofrecer una detección directa de la ocupación de cada plaza. Sin embargo, requieren la instalación de un dispositivo específico para cada posición monitorizada, lo que incrementa los costes de despliegue y mantenimiento a medida que aumenta el número de plazas [22].

Las soluciones basadas en visión artificial clásica reducen considerablemente la cantidad de hardware necesario, ya que una única cámara puede supervisar múltiples plazas simultáneamente. No obstante, estos sistemas suelen depender de algoritmos basados en reglas o características definidas manualmente, por lo que pueden presentar dificultades para adaptarse a cambios de iluminación, sombras o variaciones en el entorno observado [21].

Por su parte, los sistemas basados en aprendizaje profundo combinan las ventajas de la visión artificial con la capacidad de aprendizaje proporcionada por las redes neuronales. Gracias a ello, ofrecen una mayor robustez frente a variaciones del entorno y permiten detectar automáticamente diferentes tipos de objetos sin necesidad de diseñar manualmente las características utilizadas durante el proceso de reconocimiento [5].

Dentro de este grupo, las arquitecturas de detección de objetos en tiempo real, como YOLO, han adquirido una gran relevancia debido a su capacidad para proporcionar resultados precisos con tiempos de inferencia reducidos [10]. Esta característica facilita su integración en plataformas embebidas y aplicaciones de monitorización donde la información debe actualizarse de forma continua.

Considerando las características analizadas, los sistemas basados en visión artificial e inteligencia artificial representan actualmente una de las alternativas más flexibles para la monitorización de aparcamientos y espacios urbanos. La posibilidad de supervisar múltiples plazas mediante una única cámara, junto con la capacidad de adaptación proporcionada por los modelos de aprendizaje profundo, ha favorecido su adopción en un número creciente de aplicaciones de ciudades inteligentes.

3.6 Posicionamiento de la Solución Propuesta

A partir del análisis realizado en los apartados anteriores, puede observarse que las soluciones basadas en visión artificial e inteligencia artificial ofrecen una combinación especialmente atractiva de flexibilidad, escalabilidad y capacidad de adaptación frente a otras alternativas de monitorización [21]. La posibilidad de supervisar múltiples plazas mediante una única cámara permite reducir significativamente la cantidad de hardware necesario, simplificando tanto la instalación como el mantenimiento del sistema.

Por otra parte, la utilización de modelos de detección de objetos basados en aprendizaje profundo proporciona una mayor robustez frente a variaciones del entorno en comparación con los métodos tradicionales basados exclusivamente en reglas o características definidas manualmente [5]. Esta capacidad resulta especialmente relevante en aplicaciones exteriores, donde las condiciones de iluminación y la apariencia de la escena pueden variar a lo largo del tiempo.

La solución desarrollada en el Proyecto se sitúa dentro de esta línea de trabajo, combinando técnicas de visión artificial, detección de objetos e inteligencia artificial embebida. Concretamente, el sistema utiliza una cámara para capturar imágenes del aparcamiento de bicicletas y un modelo de detección de objetos ejecutado sobre la Plataforma para identificar automáticamente las bicicletas presentes en la escena.

A diferencia de otras soluciones basadas en sensores físicos distribuidos, la propuesta desarrollada permite supervisar múltiples plazas utilizando un único punto de captura de información. Asimismo, el procesamiento se realiza localmente sobre la propia plataforma embebida, siguiendo el paradigma *Edge AI*, lo que reduce la dependencia de infraestructuras externas y permite obtener resultados en tiempo real.

Para la implementación del sistema se ha partido de una aplicación de referencia proporcionada por *Renesas Electronics* para la ejecución de modelos de detección de objetos mediante DRP-AI. Sobre dicha aplicación se han realizado las modificaciones necesarias para adaptarla al problema específico de la detección de bicicletas y al cálculo de ocupación de plazas de aparcamiento. Adicionalmente, se ha incorporado un mecanismo de comunicación basado en HTTP que permite transmitir los resultados obtenidos a un servidor local encargado de su visualización mediante una interfaz web.

La combinación de estas tecnologías permite desarrollar una solución capaz de cubrir el ciclo completo de monitorización, desde la adquisición de imágenes hasta la publicación de la información para el usuario final. De este modo, el sistema desarrollado constituye una prueba de concepto funcional orientada a evaluar la viabilidad de emplear plataformas embebidas con capacidades de inteligencia artificial para la gestión inteligente de aparcamientos de bicicletas.

CAPÍTULO 4. DISEÑO DEL SISTEMA

4.1 Introducción

Una vez analizados los fundamentos teóricos, las tecnologías y el estado del arte relacionados con la detección automática de ocupación mediante visión artificial, este capítulo describe el diseño de la solución desarrollada en el Proyecto.

El objetivo principal del sistema consiste en determinar automáticamente el estado de ocupación de un conjunto de plazas de aparcamiento para bicicletas mediante el análisis de imágenes capturadas por una cámara. Para ello se emplea una plataforma embebida capaz de ejecutar algoritmos de detección de objetos en tiempo real y transmitir los resultados obtenidos a una interfaz web destinada a la visualización de la información.

Durante el diseño de la solución se han tenido en cuenta criterios como la simplicidad de implementación, la facilidad de integración de los diferentes componentes y la posibilidad de ejecutar el procesamiento de forma local sobre hardware embebido. Como resultado, se ha desarrollado una arquitectura capaz de cubrir el flujo completo de funcionamiento, desde la captura de imágenes hasta la presentación de los resultados al usuario final.

En las siguientes secciones se describen los requisitos considerados durante el diseño, la arquitectura general del sistema y los principales elementos hardware y software que forman parte de la solución implementada.

4.2 Requisitos del Sistema

Con el fin de desarrollar una solución capaz de monitorizar automáticamente la ocupación de un aparcamiento para bicicletas, se definieron una serie de requisitos funcionales y no funcionales que guiaron el diseño del sistema. Estos requisitos permitieron establecer las capacidades mínimas que debía ofrecer la solución desarrollada y las restricciones asociadas a su implementación.

4.2.1 Requisitos Funcionales

Los requisitos funcionales describen las tareas que debe ser capaz de realizar el sistema para satisfacer los objetivos planteados en el proyecto.

RF-1. Captura de imágenes

El sistema debe ser capaz de adquirir imágenes del aparcamiento mediante una cámara conectada a la plataforma de procesamiento.

RF-2. Detección de bicicletas

El sistema debe identificar automáticamente las bicicletas presentes en las imágenes capturadas utilizando técnicas de visión e inteligencia artificiales.

RF-3. Determinación de ocupación

A partir de las detecciones realizadas, el sistema debe determinar el estado de ocupación de las plazas monitorizadas y calcular el número de plazas ocupadas y disponibles.

RF-4. Comunicación de resultados

El sistema debe transmitir mediante peticiones HTTP la información relativa al estado de ocupación a un servidor local encargado de almacenar y publicar los resultados.

RF-5. Visualización de información

El sistema debe proporcionar una interfaz web alojada en el servidor local que permita consultar de forma sencilla el estado de ocupación del aparcamiento.

4.2.2 Requisitos No Funcionales

Los requisitos no funcionales establecen las restricciones y criterios de calidad que debe cumplir la solución desarrollada.

RNF-1. Procesamiento local

La detección de bicicletas debe ejecutarse localmente sobre la plataforma embebida, evitando la necesidad de enviar las imágenes a servidores externos para realizar las inferencias.

RNF-2. Funcionamiento en tiempo real

El sistema debe proporcionar información actualizada con una latencia reducida, permitiendo reflejar de forma rápida los cambios producidos en la ocupación de las plazas.

RNF-3. Bajo coste de despliegue

La solución debe minimizar la cantidad de hardware necesario para la monitorización, evitando la instalación de sensores individuales en cada plaza de aparcamiento.

RNF-4. Facilidad de integración

La arquitectura debe facilitar la integración de los distintos componentes hardware y software utilizados durante el desarrollo del proyecto.

RNF-5. Escalabilidad

La solución debe permitir la ampliación futura del número de plazas monitorizadas y de las funcionalidades ofrecidas sin requerir modificaciones significativas en la arquitectura general.

4.2.3 Resumen de Requisitos

Los requisitos definidos establecen las capacidades y restricciones fundamentales del sistema desarrollado. A partir de ellos se ha diseñado una solución capaz de capturar

imágenes, detectar bicicletas, determinar la ocupación de las plazas y publicar la información obtenida mediante una interfaz web accesible a través de un servidor local.

4.3 Arquitectura General del Sistema

La solución desarrollada se ha diseñado siguiendo una arquitectura distribuida compuesta por varios módulos encargados de realizar tareas específicas dentro del proceso de monitorización. El objetivo principal de esta arquitectura es transformar las imágenes capturadas por una cámara en información útil para el usuario final, permitiendo conocer en tiempo real el estado de ocupación de las plazas de aparcamiento para bicicletas.

El sistema está formado por cuatro bloques principales: el sistema de captura de imágenes, la plataforma de procesamiento basada en Renesas RZ/V2L, el servidor local encargado de recibir la información generada y la interfaz web utilizada para visualizar los resultados.

El proceso de funcionamiento comienza con la captura de imágenes del aparcamiento mediante una cámara conectada a la plataforma embebida. Estas imágenes son procesadas localmente por la aplicación ejecutada sobre la Plataforma, donde se realiza la inferencia del modelo de detección de objetos acelerado mediante DRP-AI. Como resultado de este proceso se obtiene la localización de las bicicletas presentes en la escena.

Una vez obtenidas las detecciones, la aplicación analiza la información generada por el modelo para determinar el estado de ocupación de las plazas monitorizadas. A partir de esta información se calcula el número de plazas ocupadas y disponibles, generando un conjunto reducido de datos que representan el estado actual del aparcamiento.

Posteriormente, los resultados son transmitidos mediante peticiones HTTP a un servidor local. Este servidor recibe la información enviada por la plataforma embebida y la almacena temporalmente para permitir su posterior consulta desde una interfaz web.

Finalmente, la interfaz web obtiene los datos almacenados en el servidor y los presenta al usuario de forma clara e intuitiva. De este modo, el sistema proporciona una visión actualizada del estado de ocupación del aparcamiento sin necesidad de acceder directamente a la plataforma de procesamiento.

La Figura 2 muestra el esquema general de funcionamiento de la solución desarrollada:

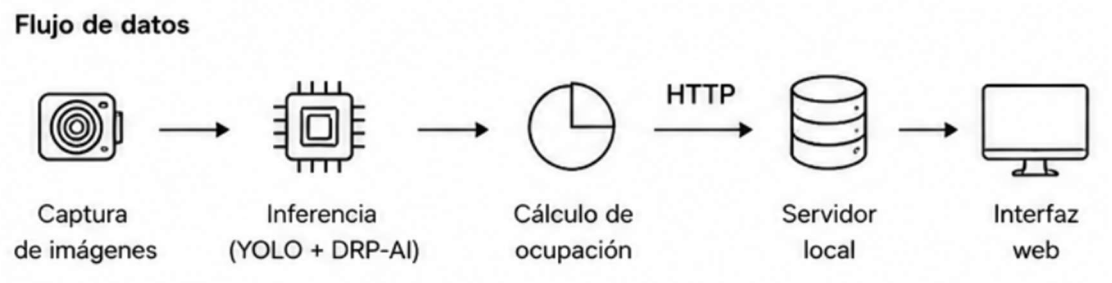


Ilustración 2 Diagrama del Sistema Propuesto

Esta arquitectura permite separar claramente las tareas de adquisición, procesamiento y visualización de la información, facilitando el desarrollo del sistema y permitiendo futuras ampliaciones sin necesidad de modificar significativamente la estructura general de la solución.

4.4 Diseño Hardware

El sistema desarrollado se apoya en una arquitectura hardware sencilla cuyo objetivo es permitir la adquisición de imágenes, el procesamiento mediante inteligencia artificial y la visualización de los resultados obtenidos. Para ello se emplean cuatro elementos principales: una cámara de captura de imágenes, una plataforma de procesamiento Renesas RZ/V2L, una infraestructura de red local y un servidor encargado de alojar la interfaz web.

Ilustración 3 Placa Renesas con cables de conexión Ethernet y de alimentación

4.4.1 Plataforma de Procesamiento

La unidad principal de procesamiento utilizada en el sistema es una plataforma basada en el procesador Renesas RZ/V2L. Esta plataforma incorpora capacidades específicas para aplicaciones de visión artificial e inteligencia artificial embebida gracias a la inclusión del acelerador hardware DRP-AI.

La placa ejecuta un sistema operativo Linux y constituye el núcleo del sistema desarrollado. Sobre ella se ejecutan tanto el modelo de detección de objetos como la lógica encargada de determinar el estado de ocupación de las plazas monitorizadas.

Además de realizar el procesamiento de las imágenes capturadas, la Plataforma es responsable de generar la información final y transmitirla al servidor local mediante peticiones HTTP.

4.4.2 Sistema de Captura de Imágenes

La captura de información visual se realiza mediante una cámara conectada a la Plataforma. La cámara proporciona de forma continua las imágenes necesarias para ejecutar el proceso de detección.

Durante el funcionamiento normal del sistema, las imágenes capturadas son enviadas directamente a la Plataforma, donde son procesadas por el modelo de inteligencia artificial. El uso de una única cámara permite supervisar simultáneamente varias plazas de aparcamiento, reduciendo significativamente la cantidad de hardware necesaria en comparación con soluciones basadas en sensores individuales.

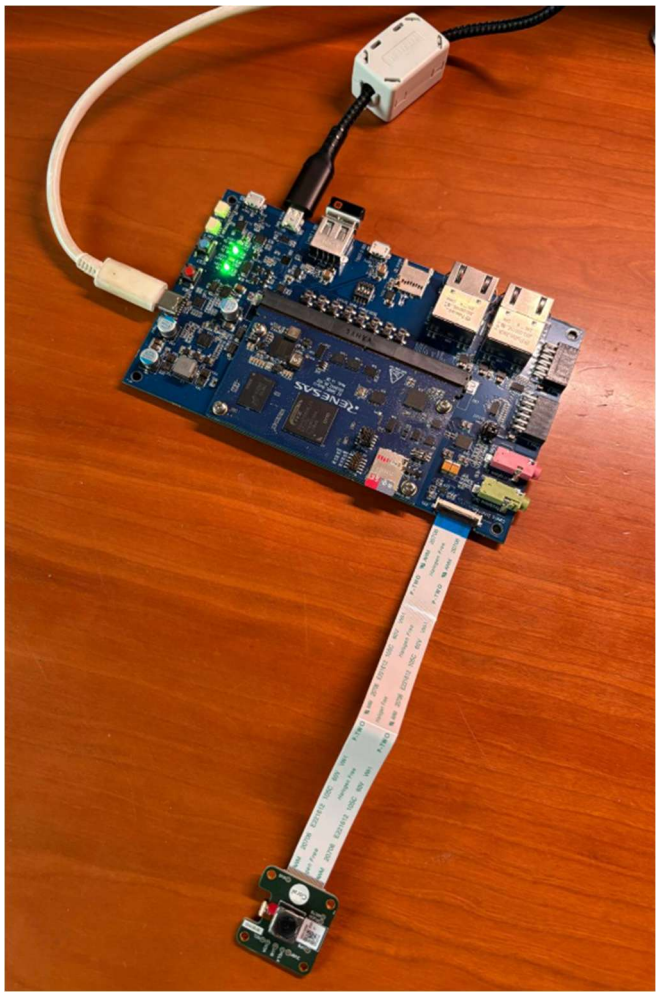


Ilustración 4 Placa Renesas con la cámara

4.4.3 Infraestructura de Red

La comunicación entre los distintos componentes del sistema se realiza a través de una red local basada en tecnología Ethernet.

La Plataforma se conecta a la red mediante su interfaz Ethernet integrada, permitiendo el intercambio de información con el servidor encargado de la visualización de resultados. Durante las pruebas realizadas se empleó un router convencional para proporcionar conectividad entre los dispositivos participantes en la arquitectura.

La utilización de una red local simplifica el despliegue del sistema y permite transmitir la información obtenida con una latencia reducida.

4.4.4 Interconexión de los Componentes Hardware

Los diferentes elementos hardware que forman parte del sistema se encuentran interconectados para permitir el flujo continuo de información desde la captura de imágenes hasta la visualización de los resultados obtenidos.

La cámara se conecta directamente a la Plataforma, proporcionando las imágenes necesarias para la ejecución del proceso de detección. Una vez procesadas las imágenes y calculado el estado de ocupación de las plazas, la información generada es transmitida a través de la red local mediante la interfaz Ethernet integrada en la plataforma.

El intercambio de datos se realiza utilizando un router como elemento de interconexión entre la plataforma embebida y el servidor encargado de la visualización. Esta configuración permite que ambos dispositivos compartan la misma red y puedan intercambiar información utilizando protocolos estándar de comunicación.

Por su parte, el servidor local recibe las peticiones enviadas por la Plataforma y actualiza la información mostrada en la interfaz web. De este modo, cualquier dispositivo conectado a la misma red puede consultar el estado de ocupación del aparcamiento mediante una interfaz web.

CAPÍTULO 5. SISTEMA/MODELO DESARROLLADO

5.1 Análisis del Sistema

El sistema desarrollado en el Proyecto tiene como objetivo principal detectar automáticamente el estado de ocupación de las plazas de un aparcamiento de bicicletas mediante técnicas de visión artificial e inteligencia artificial embebida. Para ello, se ha diseñado una arquitectura compuesta por una cámara, una plataforma de procesamiento Edge basada en la placa Renesas RZ/V2L, un modelo de detección de objetos y un sistema de comunicación capaz de transmitir la información de ocupación a una aplicación web.

En esta sección se describen los requisitos funcionales y no funcionales del sistema, los actores involucrados y los principales casos de uso definidos durante el desarrollo.

5.1.1 Requisitos funcionales

Los requisitos funcionales describen las funcionalidades que el sistema debe proporcionar para cumplir los objetivos establecidos.

- **Captura de imágenes:** el sistema debe capturar imágenes del aparcamiento de bicicletas mediante una cámara conectada a la Plataforma.
- **Detección de bicicletas:** el sistema debe identificar automáticamente la presencia de bicicletas utilizando un modelo de inteligencia artificial basado en redes neuronales convolucionales.
- **Identificación de plazas:** el sistema debe determinar qué plazas del aparcamiento están ocupadas y cuáles permanecen libres.
- **Cálculo de ocupación:** el sistema debe calcular el número total de plazas ocupadas y libres en cada momento.
- **Actualización periódica:** el sistema debe realizar el proceso de detección de forma periódica sin intervención del usuario.
- **Transmisión de datos:** el sistema debe enviar la información de ocupación a un servidor web mediante una conexión de red.
- **Visualización remota:** el sistema debe permitir visualizar el estado del aparcamiento desde una página web accesible a través de la red local.

5.1.2 Requisitos no funcionales

Los requisitos no funcionales establecen las restricciones y características de calidad que debe satisfacer el sistema.

- **Tiempo real:** el sistema debe proporcionar información actualizada con una latencia reducida.
- **Precisión:** el sistema debe ofrecer una tasa de detección suficientemente elevada para resultar útil como demostrador tecnológico.
- **Robustez:** el sistema debe ser capaz de funcionar de manera continuada en condiciones exteriores con variaciones moderadas de iluminación.
- **Bajo consumo energético:** el sistema debe aprovechar las capacidades de computación Edge de la plataforma para minimizar la dependencia de recursos externos.
- **Escalabilidad:** la arquitectura debe permitir ampliar el número de plazas monitorizadas o incorporar nuevas cámaras en trabajos futuros.
- **Mantenibilidad:** el software debe estar estructurado en módulos independientes que faciliten futuras modificaciones.
- **Compatibilidad:** el sistema debe ser compatible con protocolos de comunicación estándar basados en TCP/IP.

5.1.3 Actores del sistema

Se identifican los siguientes actores principales:

- **Usuario final:** persona encargada de consultar el estado del aparcamiento a través de la interfaz web.
- **Administrador del sistema:** responsable de la configuración, mantenimiento y actualización del sistema.
- **Sistema de visión artificial:** subsistema encargado de capturar imágenes y ejecutar el modelo de inteligencia artificial.
- **Servidor web:** componente encargado de recibir y mostrar la información generada por la Plataforma.

5.1.4 Casos de uso

Los principales casos de uso identificados son:

1. Consultar plazas libres.
2. Capturar imagen del aparcamiento.

3. Detectar bicicletas presentes.
4. Determinar estado de cada plaza.
5. Enviar información de ocupación al servidor.
6. Mostrar estado actualizado en el entorno web.
7. Gestionar errores de comunicación.

5.2 Diseño del Sistema

5.2.1 Arquitectura General del Sistema

La arquitectura desarrollada para el sistema de monitorización de plazas de aparcamiento para bicicletas se ha diseñado siguiendo una filosofía modular, de forma que cada componente desempeña una función específica dentro del proceso global de adquisición, procesamiento y visualización de la información.

El sistema está compuesto por cuatro bloques funcionales principales: el subsistema de captura de imágenes, la plataforma de procesamiento basada en la placa Renesas RZ/V2L, el subsistema de comunicaciones y el servidor local encargado de la visualización de la información.

En primer lugar, la cámara captura imágenes del aparcamiento de bicicletas y las envía a la Plataforma. A continuación, la aplicación de visión artificial ejecutada sobre la Plataforma procesa dichas imágenes mediante técnicas de detección de objetos aceleradas mediante DRP-AI. Una vez obtenidas las detecciones, el sistema calcula el estado de ocupación del aparcamiento y transmite la información resultante a un servidor local utilizando comunicaciones HTTP.

Finalmente, el servidor actualiza una interfaz web accesible desde la red local, permitiendo al usuario acceder a los datos.

5.2.2 Flujo de Funcionamiento

El funcionamiento general del sistema se divide en una serie de etapas secuenciales que permiten transformar las imágenes capturadas en información útil para el usuario.

Inicialmente, la cámara adquiere imágenes del aparcamiento y las envía a la plataforma de procesamiento. Posteriormente, la aplicación ejecuta el modelo de detección de objetos con el objetivo de identificar las bicicletas presentes en la escena.

Una vez finalizado el proceso de inferencia, el sistema determina el número de bicicletas detectadas y calcula el número de plazas libres y ocupadas. Posteriormente, esta información se encapsula y se envía al servidor local mediante peticiones HTTP.

5.2.3 Diseño Modular del Software

Con objeto de simplificar el desarrollo y facilitar futuras modificaciones, el software se ha estructurado de forma modular.

Los módulos principales implementados son:

- **Módulo de adquisición de imágenes**, encargado de obtener las imágenes procedentes de la cámara.
- **Módulo de inferencia**, responsable de ejecutar el modelo de detección de objetos acelerado mediante DRP-AI.
- **Módulo de cálculo de ocupación**, encargado de determinar el número de plazas libres y ocupadas a partir de las detecciones obtenidas.
- **Módulo de comunicaciones**, responsable del envío de información al servidor local mediante peticiones HTTP.
- **Módulo de visualización**, implementado en el servidor local y encargado de actualizar la información mostrada en la interfaz web.

Esta división modular facilita la reutilización del código y permite modificar o sustituir determinados componentes sin afectar significativamente al funcionamiento global del sistema.

5.3 Implementación Hardware

5.3.1 Plataforma de Procesamiento

La plataforma seleccionada para el desarrollo del sistema ha sido la placa Renesas RZ/V2L, un dispositivo orientado específicamente a aplicaciones de visión artificial e inteligencia artificial en el borde. La elección de esta plataforma se fundamenta en la disponibilidad del acelerador hardware DRP-AI, capaz de ejecutar algoritmos de inferencia de forma eficiente en sistemas embebidos.

Además de las capacidades de procesamiento, la plataforma dispone de diferentes interfaces de comunicación, entre las que destacan Ethernet, USB y HDMI, permitiendo tanto la conexión de periféricos como la integración con otros dispositivos de la arquitectura desarrollada.

Durante el desarrollo del proyecto, la placa ha actuado como elemento central del sistema, siendo responsable de la adquisición y procesamiento de las imágenes, la ejecución del modelo de detección y la transmisión de la información al servidor local.

5.3.2 Sistema de Captura de Imágenes

La adquisición de imágenes se realiza mediante una cámara conectada directamente a la Plataforma. Este dispositivo proporciona el flujo continuo de imágenes necesario para la ejecución del algoritmo de detección.

La posición de la cámara se ha seleccionado de manera que permita supervisar simultáneamente todas las plazas de aparcamiento incluidas en el área de estudio. De este modo, es posible monitorizar el estado completo del aparcamiento utilizando un único dispositivo de captura, simplificando considerablemente la arquitectura hardware.

Las imágenes obtenidas son enviadas directamente a la aplicación de visión artificial ejecutada sobre la Plataforma.

5.3.3 Infraestructura de Comunicaciones

Con el objetivo de permitir el intercambio de información entre la Plataforma y el sistema de visualización, se ha implementado una red local basada en tecnología Ethernet.

La Plataforma se conecta mediante cable Ethernet a un router, encargado de proporcionar conectividad entre los distintos dispositivos presentes en la arquitectura. El servidor local utilizado para la visualización se encuentra conectado a la misma red, permitiendo el intercambio de información mediante protocolos estándar TCP/IP.

La utilización de una red local simplifica el despliegue experimental y permite validar la arquitectura completa sin necesidad de recurrir a infraestructuras cloud externas.

5.3.4 Servidor Local

Para la recepción y visualización de la información generada por el sistema se ha utilizado un ordenador con sistema operativo Ubuntu configurado como servidor local.

Este servidor ejecuta una aplicación web ligera encargada de recibir las peticiones HTTP enviadas por la Plataforma y actualizar la información mostrada al usuario. La utilización de un servidor local permite demostrar el funcionamiento completo del sistema manteniendo una arquitectura sencilla y fácilmente reproducible.

Adicionalmente, el uso de un entorno Ubuntu ha facilitado el desarrollo y depuración del sistema gracias a la disponibilidad de herramientas de programación, monitorización de red y gestión de servicios.

Esta configuración permite validar el funcionamiento integral del sistema, desde la captura de imágenes hasta la publicación de la información obtenida mediante la interfaz web y el guardado de la misma en una base de datos utilizando SQLite.

5.3.5 Montaje Experimental

La Figura 4 muestra el montaje completo utilizado durante la fase experimental del proyecto. El sistema está formado por la Plataforma, la cámara de captura, el router encargado de la interconexión de red y el servidor local utilizado para la visualización de resultados.

Esta configuración permite validar el funcionamiento integral del sistema, desde la captura de imágenes hasta la publicación de la información obtenida mediante la interfaz web.

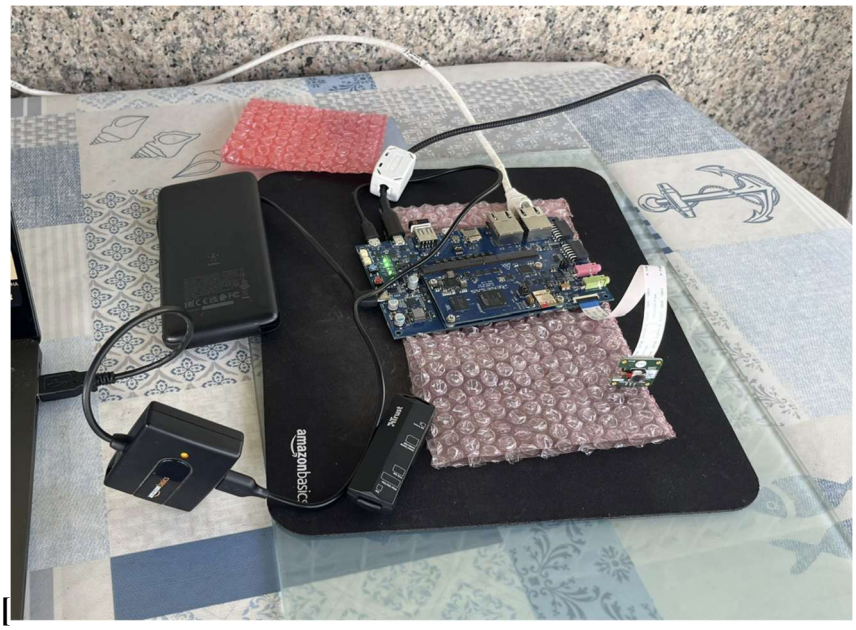


Ilustración 5 Placa Renesas con la cámara y los cables Ethernet y de alimentación

5.4 Configuración del Entorno de Desarrollo

El desarrollo del sistema ha requerido la utilización de diferentes herramientas hardware y software destinadas tanto a la programación de la Plataforma como a la ejecución y validación de las distintas funcionalidades implementadas. En este apartado se describen los principales elementos empleados durante el proceso de desarrollo.

5.4.1 Sistema Operativo de Desarrollo

El entorno principal de desarrollo utilizado durante la realización del Proyecto ha sido un ordenador ejecutando el sistema operativo Ubuntu Linux. La elección de este entorno se debe a la compatibilidad ofrecida por el ecosistema software de *Renesas Electronics*, así como a la disponibilidad de herramientas de desarrollo y depuración orientadas a sistemas embebidos.

Además del desarrollo del software, el entorno Ubuntu se ha utilizado para la gestión de la tarjeta microSD de la plataforma, la ejecución del servidor local y la realización de pruebas de comunicación entre dispositivos.

La Figura 6 muestra el Entorno Linux con la microSD empleada

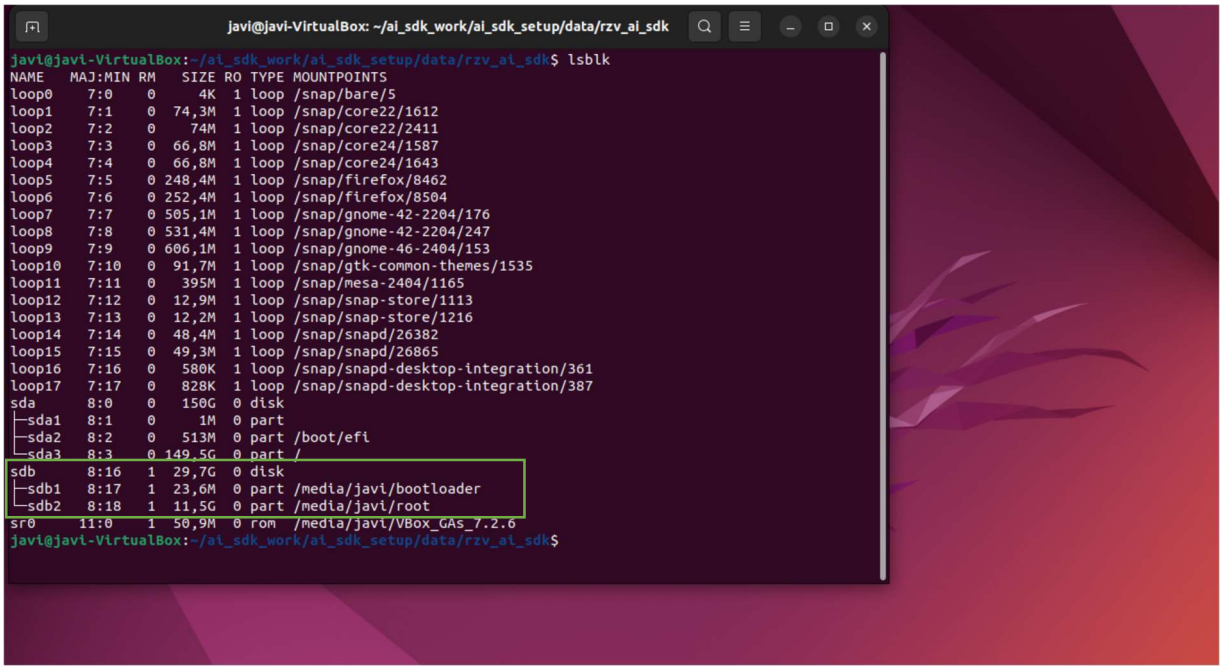


Ilustración 6 Entorno Linux con la microSD empleada

5.4.2 Configuración de la Plataforma

La preparación de la Plataforma se realizó utilizando un entorno Ubuntu como equipo de desarrollo. En dicho entorno se instaló el SDK proporcionado por *Renesas Electronics*, que incluye las herramientas, librerías y ejemplos necesarios para compilar y preparar aplicaciones destinadas a la placa.

Una vez instalado el SDK, se preparó la tarjeta microSD utilizada por la Plataforma. En ella se incorporaron los elementos necesarios para el arranque del sistema, incluyendo el *bootloader*, la imagen del sistema operativo Linux embebido y los ficheros de aplicación requeridos para la ejecución de las demos y programas desarrollados.

Durante el proceso de desarrollo, el ordenador Ubuntu se utilizó para modificar, compilar y organizar los programas. Posteriormente, los ejecutables y archivos necesarios se instalaron la tarjeta microSD para poder ejecutarlos desde la Plataforma.

De este modo, el flujo de trabajo seguido fue el siguiente:

La Figura 7 muestra el Entorno de desarrollo Ubuntu:

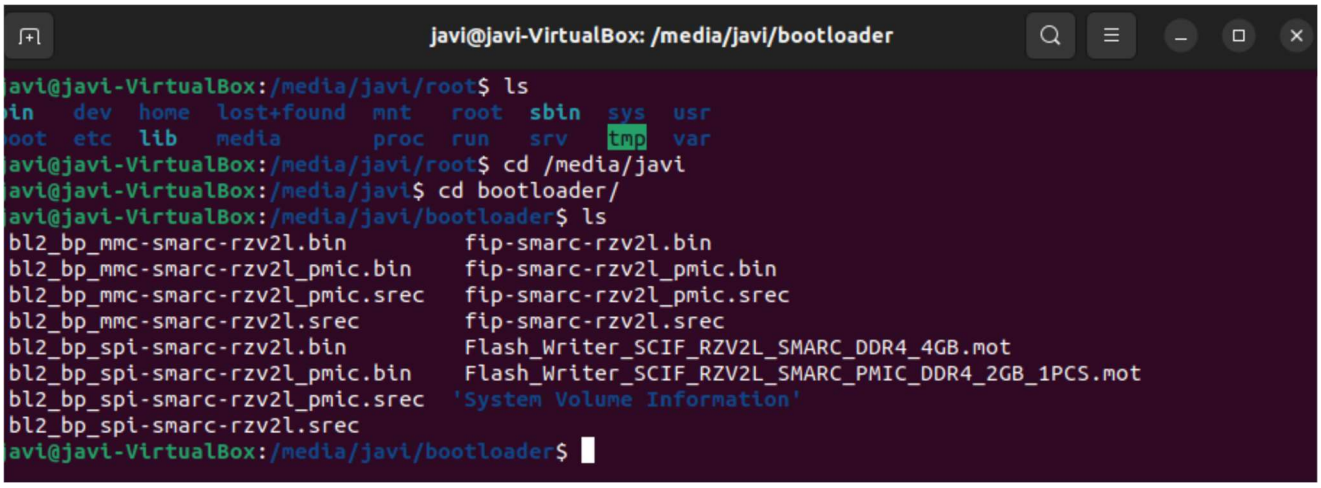


Ilustración 7 Entorno de desarrollo Ubuntu utilizado durante la preparación de la tarjeta microSD y gestión del SDK de Renesas Electronics

5.4.3 Acceso Remoto y Depuración

Con objeto de facilitar el desarrollo, configuración y depuración del sistema, el acceso a la Plataforma se realizó mediante una conexión serie utilizando la herramienta PuTTY.

Esta aplicación permitió establecer una comunicación directa con la consola Linux de la Plataforma desde el ordenador de desarrollo, proporcionando acceso remoto al sistema operativo y a las aplicaciones ejecutadas sobre la placa.

A través de la consola fue posible realizar diferentes tareas durante el desarrollo del Proyecto, entre las que destacan la ejecución de aplicaciones, la modificación de archivos de configuración, la monitorización del estado del sistema y la verificación del correcto funcionamiento de las comunicaciones de red.

Asimismo, el acceso remoto permitió supervisar en tiempo real la ejecución de la aplicación de detección, facilitando la identificación y resolución de incidencias surgidas durante las fases de integración y validación del sistema.

Durante el proceso de desarrollo se utilizaron diferentes comandos de administración del sistema Linux para verificar el estado de la Plataforma, gestionar archivos y comprobar la configuración de red. Estas herramientas resultaron fundamentales para la depuración de la aplicación y la validación del correcto intercambio de información entre los distintos componentes del sistema.

La Figura 8 muestra Plataforma Renesas RZ/V2L conectada a un monitor HDMI, teclado y ratón USB durante las tareas de desarrollo y validación del sistema

Ilustración 8 Plataforma Renesas RZ/V2L conectada a un monitor HDMI, teclado y ratón USB durante las tareas de desarrollo y validación del sistema

5.4.4 Aplicación de Referencia y Desarrollo de la Solución

El desarrollo del sistema no se realizó desde cero. Como punto de partida se empleó una aplicación de referencia proporcionada por *Renesas Electronics* dentro de su ecosistema software para la plataforma RZ/V2L.

Esta aplicación de referencia permite ejecutar modelos de detección de objetos acelerados mediante el acelerador hardware DRP-AI, proporcionando una base funcional para el desarrollo de aplicaciones de visión artificial en tiempo real sobre la Plataforma.

La utilización de una aplicación base permitió reducir significativamente el tiempo necesario para la puesta en marcha del sistema, posibilitando centrar el desarrollo en la adaptación y personalización de la solución para satisfacer los requisitos específicos del Proyecto.

A partir de esta aplicación inicial se realizaron diferentes modificaciones orientadas a implementar la funcionalidad deseada. Entre las principales adaptaciones realizadas destacan:

- Selección de las detecciones correspondientes a bicicletas.
- Implementación de la lógica de cálculo de ocupación del aparcamiento.
- Incorporación de mecanismos de comunicación mediante HTTP.
- Desarrollo de la interfaz de intercambio de información con el servidor local.
- Adaptación del sistema para la visualización del estado de ocupación de las plazas.

De este modo, la aplicación original proporcionada por *Renesas Electronics* evolucionó hasta convertirse en una solución específica para la monitorización automática de aparcamientos de bicicletas.

Ilustración 9 Aplicación de referencia de detección de objetos proporcionada por Renesas Electronics y utilizada como punto de partida para el desarrollo del sistema.

5.5 Implementación del Sistema de Detección

Como se ha comentado anteriormente, el sistema de detección desarrollado se basa en una aplicación de referencia proporcionada por *Renesas Electronics*[23] para la plataforma RZ/V2L. Esta aplicación permite ejecutar modelos de detección de objetos utilizando el acelerador hardware DRP-AI, facilitando la implementación de aplicaciones de visión artificial en tiempo real sobre sistemas embebidos.

Durante la ejecución del sistema, las imágenes capturadas por la cámara son procesadas por el modelo de detección integrado en la aplicación. Como resultado, el sistema obtiene una serie de detecciones asociadas a diferentes objetos presentes en la escena, junto con la información correspondiente a su posición y nivel de confianza.

Con objeto de adaptar la aplicación a los requisitos específicos del proyecto, fue necesario implementar mecanismos adicionales que permitieran identificar las bicicletas presentes en la imagen y determinar el estado de ocupación del aparcamiento.

5.5.1 Detección de Bicicletas

El objetivo principal del sistema desarrollado consiste en identificar automáticamente la presencia de bicicletas en el aparcamiento monitorizado. Para ello, se aprovechó la capacidad de detección de objetos proporcionada por la aplicación de referencia de *Renesas Electronics*.

Durante la ejecución de la aplicación, el modelo de inteligencia artificial procesa cada imagen capturada por la cámara y genera un conjunto de detecciones correspondientes a los diferentes objetos identificados en la escena. Cada detección incluye información relativa a la clase del objeto, la posición de la caja delimitadora (*bounding box*) y el nivel de confianza asociado.

Con el fin de adaptar la aplicación a los requisitos específicos del Proyecto, se implementó un mecanismo de filtrado encargado de seleccionar únicamente aquellas detecciones correspondientes a bicicletas. De este modo, el sistema ignora el resto de objetos identificados por el modelo y centra el análisis exclusivamente en los elementos relevantes para la determinación del estado de ocupación del aparcamiento.

Adicionalmente, se estableció un umbral mínimo de confianza para reducir la aparición de falsas detecciones. Únicamente aquellas detecciones cuyo nivel de confianza supera dicho umbral son consideradas válidas y utilizadas en las etapas posteriores del procesamiento.

Este procedimiento permite mejorar la robustez del sistema frente a posibles errores de clasificación y garantizar que el cálculo de ocupación se realiza utilizando únicamente información considerada fiable.

La Figura 10 muestra Fragmento de código

```
print_det.clear();
bicycle_count = 0;
/* Draw bounding box on RGB image. */
for (i = 0; i < det.size(); i++)
{
    if (det[i].prob == 0) continue;
    /* COCO class 1 = bicycle */
    if (det[i].c != 1) continue;

    bicycle_count++;

    print_det.push_back(det[i]);

    /* Clear string stream for bounding box labels */
    stream.str("");

    /* Draw the bounding box on the image */
    stream << std::fixed << std::setprecision(2) << det[i].prob;
    result_str = "bicycle " + stream.str();
```

Ilustración 10. Fragmento de código encargado de filtrar las detecciones del modelo para seleccionar únicamente los objetos clasificados como bicicletas

La figura muestra el fragmento de código encargado de seleccionar y procesar únicamente aquellas detecciones correspondientes a bicicletas. Este bloque constituye uno de los elementos fundamentales de la aplicación desarrollada, ya que permite adaptar la aplicación de referencia proporcionada por *Renesas Electronics* a los requisitos específicos del Proyecto. Mientras que la aplicación original es capaz de detectar diversos tipos de objetos, la solución propuesta requiere centrarse exclusivamente en la identificación de bicicletas presentes en el aparcamiento monitorizado.

5.5.1.1 Pasos del algoritmo

- ✓ El primer paso del algoritmo consiste en vaciar el vector **print_det** mediante la instrucción:

  ```
  print_det.clear();
  ```

  Este vector se utiliza para almacenar las detecciones que posteriormente serán representadas sobre la imagen procesada. Dado que la aplicación funciona de manera continua y procesa sucesivamente múltiples imágenes capturadas por la cámara, resulta necesario eliminar las detecciones almacenadas durante el ciclo anterior antes de comenzar el procesamiento de una nueva imagen. De este modo, se evita que detecciones antiguas permanezcan visibles sobre la imagen actual, garantizando que únicamente se representen los objetos realmente presentes en cada instante.

- ✓ A continuación, el contador global de bicicletas detectadas se reinicia mediante la instrucción:

  ```
  bicycle_count = 0;
  ```

  Esta variable desempeña un papel esencial dentro del sistema, puesto que almacena el número total de bicicletas identificadas en la escena durante cada iteración del algoritmo. Al igual que ocurre con el vector de detecciones, este contador debe reiniciarse al comienzo de cada nuevo procesamiento para evitar la acumulación de resultados procedentes de imágenes anteriores.

- ✓ Posteriormente, el algoritmo recorre secuencialmente todas las detecciones generadas por el modelo de inteligencia artificial:

  ```
  for (i = 0; i < det.size(); i++)
  ```

  El vector **det** contiene el conjunto completo de objetos detectados por el modelo YOLO ejecutado sobre el acelerador DRP-AI. Cada uno de estos elementos almacena información relativa a un objeto concreto, incluyendo su clase, coordenadas espaciales y probabilidad asociada a la detección.

  La utilización de un bucle permite analizar individualmente cada uno de los objetos detectados. Este enfoque proporciona una gran flexibilidad, ya que permite aplicar diferentes criterios de selección dependiendo de las necesidades de la aplicación.

- ✓ Dentro del bucle, la primera comprobación realizada consiste en descartar aquellas detecciones cuya probabilidad sea nula:

  ```
  if (det[i].prob == 0) continue;
  ```

  La variable **prob** representa el nivel de confianza asociado a cada detección. En determinadas circunstancias, el modelo puede generar estructuras de datos sin información válida o detecciones cuya probabilidad haya sido descartada durante etapas previas del postprocesado. Mediante esta comprobación, el sistema ignora automáticamente dichos elementos y continúa evaluando la siguiente detección disponible.

  El uso de la instrucción continue permite abandonar inmediatamente la iteración actual y avanzar al siguiente elemento del vector sin ejecutar el resto del código contenido en el bucle. Esta estrategia reduce la carga computacional y simplifica el proceso de selección.

- ✓ Una vez descartadas las detecciones inválidas, el algoritmo realiza la comprobación más importante del sistema:

  ```
  if (det[i].c != 1) continue;
  ```

La variable **c** almacena el identificador numérico correspondiente a la clase del objeto detectado. El modelo YOLO utilizado ha sido entrenado utilizando el conjunto de datos COCO (*Common Objects in Context*), ampliamente utilizado en aplicaciones de visión artificial. Este conjunto de datos define un total de ochenta clases distintas, asignando un identificador específico a cada categoría de objeto.

Dentro de dicha nomenclatura, el identificador numérico 1 corresponde a la clase *bicycle*. Por tanto, esta condición permite descartar automáticamente cualquier detección que no represente una bicicleta.

En consecuencia, objetos tales como personas, automóviles, motocicletas, perros o cualquier otra categoría contemplada por el modelo son ignorados completamente por el sistema.

Este filtrado constituye una de las principales modificaciones introducidas sobre la aplicación original de *Renesas Electronics*, ya que permite especializar el comportamiento del modelo para una aplicación concreta de monitorización de aparcamientos de bicicletas.

✓ Cuando una detección supera todas las comprobaciones anteriores, el sistema considera que se ha identificado correctamente una bicicleta. En ese momento, el contador global se incrementa mediante la instrucción:

```
bicycle_count++;
```

De esta manera, el sistema mantiene actualizado el número total de bicicletas presentes en la escena. Este valor será utilizado posteriormente para calcular el estado de ocupación del aparcamiento y determinar el número de plazas libres disponibles.

✓ Además del incremento del contador, la detección válida se almacena dentro del vector **print_det**:

```
print_det.push_back(det[i]);
```

Este vector contiene exclusivamente aquellas detecciones correspondientes a bicicletas y será utilizado posteriormente para representar gráficamente los resultados sobre la imagen capturada.

Gracias a esta estrategia, el sistema no delimita sobre la imagen todos los objetos detectados originalmente por el modelo, sino únicamente aquellos que resultan relevantes para la aplicación desarrollada.

- ✓ Una vez almacenada la detección, el sistema prepara la información que será mostrada visualmente al usuario. Para ello, en primer lugar, se limpia el contenido previo del flujo de caracteres:

  ```
  stream.str("");
  ```

  El objeto *stream* se emplea para construir dinámicamente las cadenas de texto que posteriormente serán representadas sobre la imagen.

- ✓ Posteriormente, el sistema genera la etiqueta asociada a la detección:

  ```
  stream << std::fixed << std::setprecision(2) << det[i].prob;

  result_str = "bicycle " + stream.str();
  ```

  Este fragmento concatena la palabra "bicycle" con el nivel de confianza proporcionado por el modelo, expresado con dos cifras decimales.

  Por ejemplo, una detección cuya probabilidad sea del 94,57 % aparecerá representada sobre la imagen mediante una etiqueta similar a: bicycle 0.95

La representación del nivel de confianza proporciona al usuario una indicación adicional sobre la fiabilidad de la detección realizada, permitiendo evaluar visualmente el comportamiento del sistema durante las pruebas experimentales.

En conjunto, este bloque de código transforma las salidas genéricas proporcionadas por el modelo de inteligencia artificial en información específica para la aplicación desarrollada. Gracias a este procedimiento, el sistema es capaz de ignorar objetos irrelevantes, contabilizar automáticamente las bicicletas presentes y generar la información necesaria para la posterior determinación del estado de ocupación del aparcamiento.

5.5.2 Cálculo del Estado de Ocupación

Una vez identificadas las bicicletas presentes en la escena mediante el proceso de detección descrito en el apartado anterior, resulta necesario transformar dicha información en un indicador útil para el usuario final. En el contexto del presente proyecto, este indicador corresponde al número de plazas libres y ocupadas existentes en el aparcamiento en cada instante de tiempo.

Para ello, se desarrolló un mecanismo específico encargado de calcular el estado de ocupación a partir del número de bicicletas detectadas por el sistema de visión artificial. Este procedimiento constituye uno de los elementos fundamentales de la aplicación, ya que permite convertir la información proporcionada por el modelo de inteligencia artificial en un dato directamente interpretable por el usuario.

El aparcamiento utilizado durante las pruebas experimentales dispone de un total de seis plazas destinadas al estacionamiento de bicicletas. Debido a ello, la lógica implementada considera inicialmente este número como el total de plazas disponibles dentro del sistema.

Durante cada ciclo de ejecución, el modelo de detección procesa la imagen capturada por la cámara y genera un conjunto de objetos detectados. Tras aplicar el proceso de filtrado descrito anteriormente, únicamente se conservan las detecciones correspondientes a bicicletas, obteniéndose finalmente el número total de bicicletas presentes en la escena.

A partir de este valor, el número de plazas libres se calcula mediante la siguiente expresión:

$$N_{libres} = N_{plazas} - N_{bicicletas}$$

N_{libres} = N_{totales} - N_{bicicletas}

donde:

- N_{libres} representa el número de plazas libres.
- N_{plazas} corresponde al número total de plazas monitorizadas.
- $N_{bicicletas}$ indica el número de bicicletas detectadas por el sistema.

En el caso concreto del prototipo desarrollado, el valor de N_{plazas} se ha fijado en seis, coincidiendo con el número de plazas presentes en el aparcamiento seleccionado para la validación experimental, el bici-hangar de Don Cicleto localizado en la Estación de tren Guixar de Vigo.

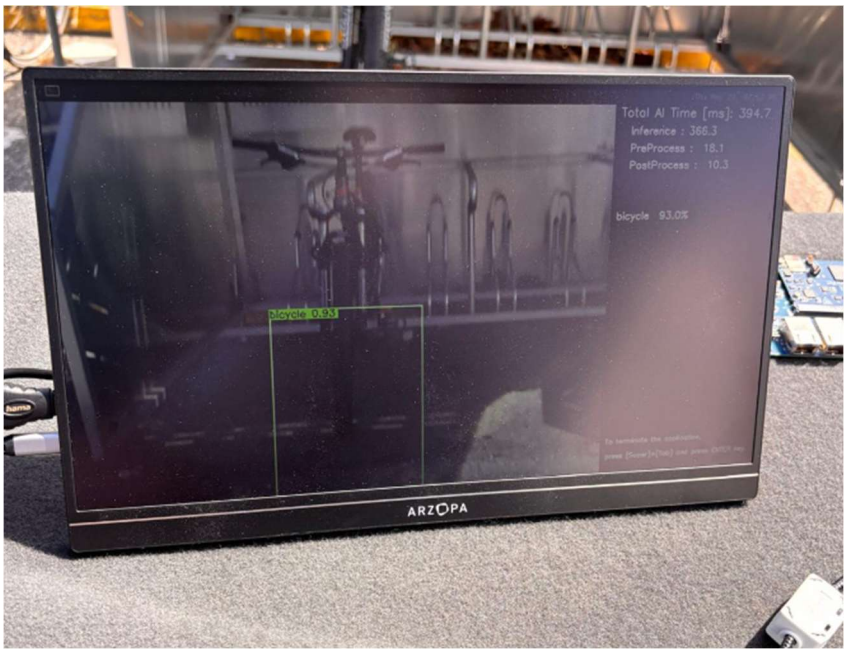


Ilustración 11. Prueba en Bici Hangar Don Cicleto de Estación Guixar (Vigo)

La implementación software de este procedimiento se realiza mediante el fragmento de código mostrado en la figura siguiente:

```
stream.str("");
int free_spaces = 6 - bicycle_count;
if (free_spaces < 0)
{
    free_spaces = 0;
}
char cmd[512];
```

Ilustración 12. Fragmento de código encargado del cálculo del número de plazas libres

Como puede observarse, el algoritmo obtiene inicialmente el número de plazas libres restando el número de bicicletas detectadas al número total de plazas disponibles. Sin embargo, se ha incorporado adicionalmente una comprobación de seguridad destinada a evitar la aparición de valores negativos.

Esta situación podría producirse en determinadas circunstancias, por ejemplo, debido a falsas detecciones generadas por el modelo de inteligencia artificial, a la presencia de bicicletas fuera del área monitorizada o a la detección simultánea de objetos no pertenecientes al aparcamiento. Para garantizar la coherencia de la información mostrada al usuario, cualquier resultado inferior a cero es automáticamente corregido y sustituido por el valor cero.

Una vez calculado el número de plazas libres, la aplicación genera la información que posteriormente será mostrada al usuario y enviada al servidor local. Para ello, el sistema actualiza dinámicamente el texto representado sobre la imagen procesada, permitiendo visualizar en tiempo real el estado de ocupación del aparcamiento.

Las siguientes figuras muestran un ejemplo del resultado obtenido tras aplicar la lógica de cálculo implementada:

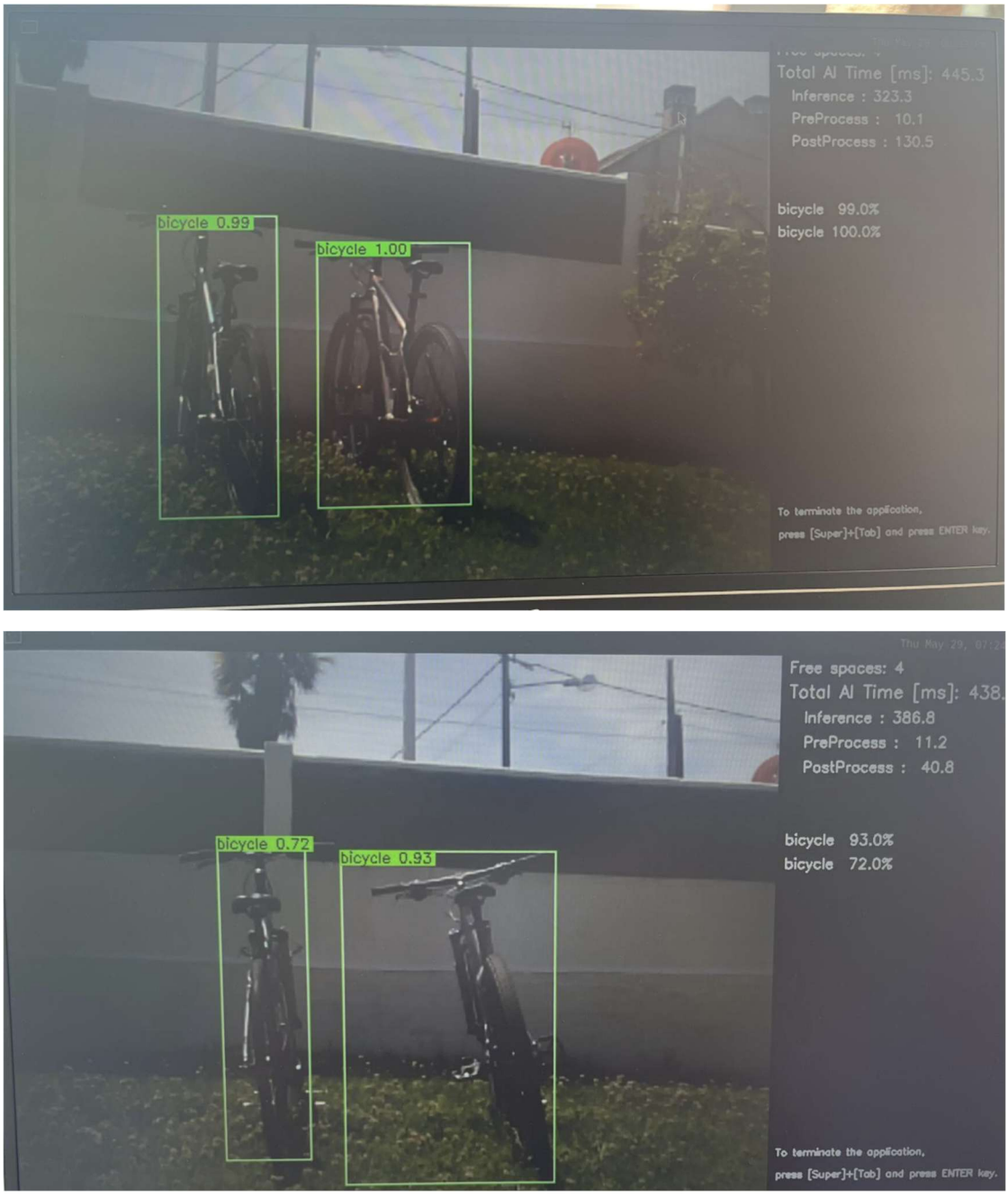


Ilustración 13 y 14: Detección de 2 bicicletas y Visualización en pantalla del número de plazas libres calculado por el sistema

5.6 Implementación del Sistema de Comunicaciones

Una vez implementados el sistema de detección de bicicletas y el mecanismo de cálculo del estado de ocupación, fue necesario desarrollar un subsistema de comunicaciones que permitiera transmitir los resultados obtenidos hacia un entorno de visualización externo. Esta parte del desarrollo resulta fundamental, ya que el objetivo del proyecto no se limita

a detectar bicicletas sobre una imagen, sino que pretende proporcionar una información útil y consultable por el usuario final.

En la solución desarrollada, la Plataforma realiza el procesamiento local de las imágenes y calcula el número de plazas ocupadas y libres. Sin embargo, la visualización de dicha información no se realiza directamente sobre la placa, sino a través de una interfaz web alojada en un servidor local. Por este motivo, fue necesario establecer un mecanismo de intercambio de datos entre ambos elementos.

La arquitectura finalmente implementada se basa en una comunicación mediante HTTP dentro de una red local. La Plataforma actúa como cliente, generando y enviando peticiones al servidor, mientras que el equipo Ubuntu actúa como servidor local, recibiendo la información y actualizando el estado mostrado en la web.

Esta decisión permite separar claramente las funciones del sistema. Por un lado, la plataforma embebida se encarga de la adquisición de imágenes, inferencia del modelo y cálculo de ocupación. Por otro lado, el servidor local se ocupa de recibir los datos, almacenarlos temporalmente y mostrarlos al usuario. Esta separación facilita el desarrollo, la depuración y la futura ampliación de la solución.

5.6.1 Arquitectura de Comunicaciones

La arquitectura de comunicaciones implementada sigue un modelo cliente-servidor. En este modelo, la Plataforma desempeña el papel de cliente, ya que es el dispositivo que inicia la comunicación cada vez que dispone de nuevos datos de ocupación. El servidor local, ejecutado sobre un ordenador Ubuntu, permanece a la espera de recibir peticiones procedentes de la Plataforma.

Ambos dispositivos se encuentran conectados a la misma red local. Durante las pruebas se utilizó un router como elemento de interconexión, permitiendo que la placa y el ordenador Ubuntu dispusieran de conectividad dentro de una misma red IP. Esta configuración permitió validar el funcionamiento de la solución sin necesidad de utilizar una infraestructura cloud ni exponer el servidor a Internet.

El flujo de comunicación se inicia cuando la aplicación de detección ejecutada en la Plataforma obtiene el número de bicicletas detectadas. A continuación, se calcula el número de plazas libres y se construye un mensaje con los datos principales del estado del aparcamiento. Este mensaje se envía al servidor mediante una petición HTTP POST. El servidor recibe la petición, interpreta los datos recibidos y actualiza la información que posteriormente será mostrada en la página web.

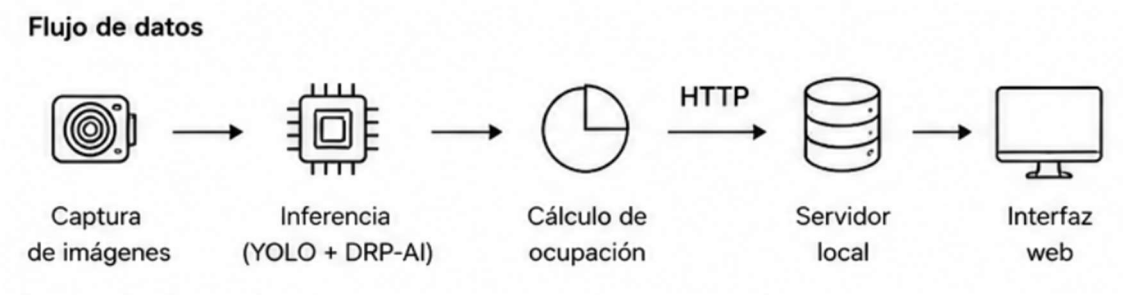


Ilustración 2 Diagrama del Sistema Propuesto (imagen incluida en punto 4.3)

Esta arquitectura presenta una ventaja importante frente a una solución completamente integrada en la Plataforma: permite que ésta se centre en las tareas de visión artificial, que son las más exigentes desde el punto de vista computacional, mientras que el ordenador Ubuntu se encarga de la parte de servidor y visualización. De este modo, se reduce la complejidad software ejecutada sobre la Plataforma.

Al utilizar una red local el sistema puede ser probado y validado de forma sencilla en un entorno controlado, siendo el más adecuado para validar su funcionalidad.

Como mejora a futuro el sistema podría emplearse sustituyendo el servidor local por un servidor remoto o una plataforma *cloud*.

5.6.2 Configuración de la Red Local

Para que la comunicación entre la Plataforma y el servidor local fuera posible, ambos dispositivos debían encontrarse dentro de la misma red IP. Durante el desarrollo se utilizaron distintas configuraciones de red, comenzando con pruebas directas entre la Plataforma y el equipo Ubuntu y finalizando con una arquitectura basada en un router.

El servidor Ubuntu debía disponer de una dirección IP accesible desde la Plataforma. Esta dirección se utilizó posteriormente dentro del código de la aplicación para construir la URL de destino de las peticiones HTTP. En las pruebas realizadas, el servidor local quedó asociado a la dirección 192.168.100.35, utilizada por la aplicación embebida para enviar los datos de ocupación.

Antes de integrar la comunicación dentro del código principal, fue necesario comprobar la conectividad básica entre los dispositivos. Para ello se utilizaron herramientas estándar de Linux destinadas a verificar interfaces de red, direcciones IP y puertos abiertos. Estas comprobaciones permitieron asegurar que el servidor estaba escuchando correctamente y que la placa podía alcanzar la dirección configurada.

La Figura 15 muestra Prueba exitosa con el servidor

```
    root@smarc-rzv2l:~# curl -X POST http://192.168.1.35:5000/update \
>   -H "Content-Type: application/json" \
>   -d '{"ocupadas":2,"libres":4}'
{"status":"ok"}
```

Ilustración 15 Prueba exitosa con el servidor desde la consola de la placa mediante Putty, donde se puede comprobar que la IP es 192.168.1.35

También se comprobó que el servidor web estaba escuchando en el puerto configurado. En el prototipo se utilizó el puerto 5000, habitual en aplicaciones web ligeras desarrolladas con Python y Flask. Esta comprobación permitió confirmar que el servicio estaba correctamente levantado antes de realizar pruebas desde la Plataforma.

Esta fase de configuración fue necesaria para evitar errores de conexión durante la integración. En sistemas distribuidos, una parte importante de los problemas no se encuentra en el código de aplicación, sino en la configuración de red, direcciones IP, rutas, firewalls o puertos. Por ello, validar estos elementos antes de modificar la aplicación principal permitió aislar errores y acelerar el proceso de desarrollo.

5.6.3 Selección del Protocolo HTTP

Para la transmisión de información se seleccionó el protocolo HTTP, utilizando concretamente peticiones de tipo POST. Esta elección se realizó por varios motivos técnicos y prácticos.

En primer lugar, HTTP es un protocolo ampliamente utilizado y soportado por prácticamente cualquier sistema operativo moderno. Esto facilita su integración tanto en el lado de la plataforma embebida como en el lado del servidor local. La Plataforma ejecuta Linux, por lo que dispone de herramientas y bibliotecas compatibles con este tipo de comunicación.

En segundo lugar, HTTP permite utilizar una arquitectura sencilla basada en cliente y servidor. En el Proyecto no se requiere mantener una conexión permanente con el servidor, sino únicamente enviar información cuando dispone de nuevos datos. Esto encaja bien con el funcionamiento del sistema, ya que la información de ocupación se actualiza de forma periódica tras cada ciclo de detección.

En tercer lugar, el uso de peticiones POST resulta adecuado porque la Plataforma no está solicitando información al servidor, sino enviando nuevos datos generados localmente. Por este motivo, se decidió utilizar una ruta específica del servidor encargada de recibir actualizaciones del estado del aparcamiento de bicicletas.

La información enviada se codifica en formato JSON. Este formato fue seleccionado por su sencillez, legibilidad y compatibilidad con aplicaciones web. Además, al organizar la información en pares clave-valor, permite representar de forma clara los datos principales del sistema: número total de plazas, número de plazas ocupadas y número de plazas libres.

Un ejemplo de mensaje enviado por la plataforma es el siguiente:

```
{
  "total": 6,
  "ocupadas": 4,
  "libres": 2
}
```

Este formato resulta fácil de interpretar tanto por el servidor como por cualquier posible sistema futuro que se quiera integrar, como una base de datos, una aplicación móvil o una plataforma de monitorización remota.

5.6.4 Implementación del Envío desde la Plataforma

Una vez calculado el número de bicicletas detectadas y el número de plazas libres disponibles, resulta necesario transmitir dicha información al servidor local para que pueda ser posteriormente visualizada a través de la interfaz web desarrollada. Para ello, se implementó un mecanismo de comunicación basado en peticiones HTTP, integrado directamente en la aplicación ejecutada sobre la Plataforma.

La Figura 16 muestra el fragmento de código encargado de construir y enviar la información generada por el sistema de detección al servidor local.

```
char cmd[512];

snprintf(cmd, sizeof(cmd),
"curl -s -X POST http://192.168.1.35:5000/update "
"-H \"Content-Type: application/json\" "
"-d '{\"total\":6,\"ocupadas\":%d,\"libres\":%d}' >/dev/null 2>&1",
bicycle_count,
free_spaces);

system(cmd);

stream.str("");
stream << "Free spaces: " << free_spaces;
str = stream.str();
```

Ilustración 4 Parte del código responsable de construir y enviar la petición al servidor

El proceso se inicia con la declaración de un vector de caracteres denominado **cmd**, cuya función consiste en almacenar el comando que posteriormente será ejecutado por el sistema operativo Linux de la Plataforma.

`char cmd[512];`

Este vector dispone de una longitud de 512 caracteres, tamaño suficiente para contener la totalidad del comando que se desea ejecutar, incluyendo la dirección del servidor, las cabeceras HTTP y los datos que serán transmitidos.

A continuación, la función `snprintf()` se utiliza para construir dinámicamente el comando que será enviado al servidor. Esta función permite generar cadenas de texto de forma segura, evitando posibles desbordamientos de memoria y permitiendo introducir variables cuyo valor cambia durante la ejecución de la aplicación.

El comando generado utiliza la herramienta *Curl*, ampliamente empleada en sistemas Linux para realizar transferencias de datos mediante distintos protocolos de comunicación, entre ellos HTTP y HTTPS. La elección de esta herramienta permitió simplificar considerablemente la implementación del sistema de comunicaciones, evitando la necesidad de desarrollar manualmente un cliente HTTP específico para la Plataforma.

Dentro del comando puede observarse la siguiente instrucción:

curl -s -X POST http://192.168.1.35:5000/update

La opción -X POST indica que la petición HTTP generada será de tipo POST. Este tipo de peticiones se utiliza habitualmente cuando un cliente necesita enviar información a un servidor para que ésta sea procesada o almacenada.

En el Proyecto, la Plataforma actúa como cliente, mientras que el ordenador Ubuntu que ejecuta el servidor local actúa como servidor. Cada vez que se completa un ciclo de detección, la plataforma establece una conexión con el servidor y le transmite la información correspondiente al estado actual del aparcamiento de bicicletas.

La dirección http://192.168.1.35:5000/update identifica el recurso encargado de recibir las actualizaciones procedentes de la plataforma embebida. En dicha dirección pueden distinguirse varios elementos:

- **192.168.1.35** corresponde a la dirección IP del ordenador Ubuntu utilizado como servidor local.
- **5000** representa el puerto TCP en el que se encuentra escuchando la aplicación web desarrollada mediante Flask.
- **/update** constituye la ruta específica del servidor destinada a procesar las actualizaciones enviadas por la plataforma.

Posteriormente, el comando incorpora la siguiente cabecera HTTP:

-H "Content-Type: application/json"

Esta cabecera informa al servidor de que los datos contenidos en el cuerpo de la petición se encuentran codificados utilizando el formato JSON (*JavaScript Object Notation*). El empleo de JSON resulta especialmente adecuado en aplicaciones *IoT* debido a su sencillez, legibilidad y elevada compatibilidad con aplicaciones web modernas.

Los datos transmitidos al servidor se generan mediante la siguiente instrucción:

```
-d '{"total":6,"ocupadas":%d,"libres":%d}'
```

Este fragmento define la estructura de la información enviada. En primer lugar, se especifica el número total de plazas monitorizadas, fijado en seis para el prototipo desarrollado. A continuación, se incluyen dos parámetros variables:

- **ocupadas**, que representa el número de bicicletas detectadas por el sistema.
- **libres**, que indica el número de plazas disponibles calculadas automáticamente.

Los símbolos *%d* actúan como marcadores de posición que son sustituidos automáticamente durante la ejecución del programa por los valores almacenados en las variables *bicycle_count* y *free_spaces*, respectivamente. Gracias a este mecanismo, el contenido de la petición refleja en todo momento el estado real del aparcamiento.

Por ejemplo, si el sistema detecta dos bicicletas durante un determinado instante, el mensaje JSON generado será equivalente al siguiente:

```
{
  "total": 6,
  "ocupadas": 2,
  "libres": 4
}
```

Asimismo, puede observarse la presencia de la instrucción:

```
>/dev/null 2>&1
```

Esta redirección tiene como finalidad descartar tanto la salida estándar como los posibles mensajes de error generados durante la ejecución del comando *Curl*. De esta forma, se evita la aparición continua de mensajes en la consola de la aplicación, manteniendo una visualización más limpia y facilitando el seguimiento del proceso de detección en tiempo real.

Finalmente, una vez construido el comando completo, éste se ejecuta mediante la función:

`system(cmd);`

La función `system()` permite invocar directamente el intérprete de comandos del sistema operativo Linux y ejecutar el contenido almacenado en la variable cmd. Como consecuencia, la Plataforma establece automáticamente la conexión con el servidor local y transmite la información correspondiente al estado de ocupación del aparcamiento.

Este procedimiento se repite periódicamente durante toda la ejecución de la aplicación, permitiendo actualizar de forma automática y prácticamente en tiempo real la información mostrada en la interfaz web. Gracias a esta implementación, el sistema desarrollado no sólo es capaz de detectar bicicletas, sino también de compartir dicha información con otros dispositivos conectados a la red, constituyendo así una solución *IoT* completa orientada a la monitorización inteligente de aparcamientos de bicicletas.

5.6.5 Validación del Subsistema de Comunicaciones

Una vez implementado el mecanismo encargado de transmitir la información generada por la Plataforma al servidor local, resultó imprescindible llevar a cabo una fase exhaustiva de validación experimental. Esta etapa tuvo como objetivo comprobar no sólo el correcto funcionamiento individual de cada uno de los componentes implicados, sino también verificar la correcta integración entre la Plataforma, la infraestructura de red local y el servidor encargado de proporcionar la interfaz de visualización.

La validación del subsistema de comunicaciones constituye una etapa especialmente importante dentro del desarrollo del Proyecto, ya que el sistema diseñado no se limita únicamente a realizar tareas de detección mediante inteligencia artificial, sino que persigue proporcionar información útil y accesible para el usuario final. Por este motivo, garantizar la correcta transmisión de los datos entre los distintos elementos de la arquitectura resulta fundamental para asegurar el funcionamiento global del sistema.

Con el fin de reducir la complejidad del proceso de depuración y facilitar la identificación de posibles errores, la validación se realizó siguiendo una metodología incremental. En lugar de verificar directamente el sistema completo, se optó por validar inicialmente cada uno de los subsistemas de forma independiente, integrándolos progresivamente hasta obtener la solución final.

De esta manera, el procedimiento seguido durante la fase de validación se estructuró en las siguientes etapas:

1. Verificación de la conectividad física y lógica entre dispositivos.
2. Comprobación del correcto funcionamiento del servidor local.
3. Validación manual del intercambio de información mediante peticiones HTTP.
4. Integración del mecanismo de comunicaciones dentro de la aplicación principal.

5. Validación del funcionamiento del sistema completo en condiciones reales.

5.6.5.1 Verificación de la Conectividad de Red

El primer paso consistió en comprobar que tanto la Plataforma como el ordenador Ubuntu encargado de ejecutar el servidor local se encontraban correctamente conectados a la misma red local.

Para ello, ambos dispositivos fueron conectados a un router, que actuó como elemento central de la infraestructura de comunicaciones. Esta configuración permitió disponer de un entorno de pruebas sencillo, fácilmente reproducible y suficientemente representativo para validar el funcionamiento del prototipo.

Antes de iniciar cualquier prueba de intercambio de información, fue necesario verificar la configuración de red del servidor Ubuntu. Concretamente, se comprobó la dirección IP asignada al equipo, puesto que esta dirección debía ser posteriormente incorporada al código de la aplicación ejecutada sobre la Plataforma.

La obtención de la dirección IP se realizó mediante herramientas estándar proporcionadas por Linux, tales como *hostname* o *ip*.

La Figura 17 muestra la dirección IP del servidor Ubuntu:

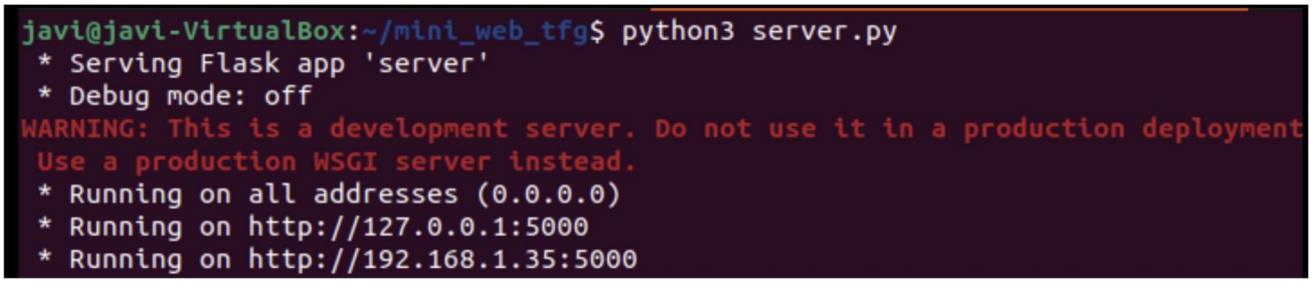


Ilustración 17 dirección IP del servidor Ubuntu (192.168.1.35)

A partir de la información obtenida, se identificó la dirección IP del servidor, que posteriormente fue utilizada dentro del código de la aplicación para construir las peticiones HTTP dirigidas al servidor local.

Además de verificar la dirección IP, también se comprobó que el servidor permanecía correctamente a la escucha en el puerto seleccionado para la web. En el prototipo desarrollado se empleó el puerto TCP 5000, habitual en aplicaciones implementadas mediante el *framework* Flask.

Para comprobar el estado del servidor se utilizaron diferentes herramientas de monitorización de puertos disponibles en Linux.

La correcta visualización del puerto en estado de escucha confirmó que la aplicación web se encontraba preparada para aceptar conexiones procedentes de la Plataforma.

5.6.5.2 Validación Manual del Servidor

Una vez verificado el correcto funcionamiento de la infraestructura de red, se procedió a validar el comportamiento del servidor local de manera aislada.

La validación aislada del servidor resultó especialmente útil durante las primeras etapas del desarrollo, puesto que permitió comprobar el funcionamiento de la aplicación web sin necesidad de ejecutar simultáneamente el sistema completo sobre la Plataforma.

Con este propósito se utilizaron peticiones HTTP generadas manualmente mediante la herramienta *Curl*. Gracias a esta aproximación fue posible verificar que el servidor era capaz de:

- Recibir correctamente peticiones HTTP de tipo POST.
- Interpretar adecuadamente los datos recibidos en formato JSON.
- Procesar la información recibida.
- Devolver respuestas adecuadas al cliente.

Un ejemplo del comando utilizado durante las pruebas se muestra a continuación:

```
curl -X POST http://192.168.1.35:5000/update \
-H "Content-Type: application/json" \
-d '{"total":6,"ocupadas":2,"libres":4}'
```

La Figura 18 muestra envío manual de información al servidor

```
root@smarc-rzv2l:~# curl -X POST http://192.168.1.35:5000/update \
>   -H "Content-Type: application/json" \
>   -d '{"ocupadas":2,"libres":4}'
{"status":"ok"}
```

Ilustración 58 Envío manual de información al servidor utilizando Curl. El status: ok confirma el correcto funcionamiento

La recepción de este mensaje confirmó que el servidor era capaz de interpretar adecuadamente la información enviada y procesarla sin errores.

Este tipo de pruebas permitió además validar la estructura JSON seleccionada para el intercambio de información antes de integrarla definitivamente dentro de la aplicación principal.

5.6.5.3 Integración con la Plataforma

Tras validar el servidor de forma independiente, se procedió a integrar el mecanismo de comunicación dentro de la aplicación ejecutada sobre la Plataforma.

Durante esta fase, el sistema completo operó de forma automática. El flujo de funcionamiento puede resumirse mediante las siguientes etapas:

1. Captura de una imagen mediante la cámara.
2. Ejecución de la inferencia utilizando el acelerador DRP-AI.
3. Obtención de las detecciones generadas por el modelo.
4. Filtrado de las detecciones correspondientes a bicicletas.
5. Cálculo del número de plazas libres.
6. Construcción del mensaje JSON.
7. Envío automático de la información al servidor local.

La integración de todas estas etapas permitió disponer de un sistema completamente autónomo capaz de detectar bicicletas y transmitir automáticamente la disponibilidad del aparcamiento sin intervención humana.

5.6.5.4 Monitorización de las Peticiones Recibidas

Con objeto de verificar que la Plataforma transmitía correctamente la información al servidor, se monitorizó la actividad de la web durante la ejecución del sistema.

Cada vez que la Plataforma enviaba una actualización, el servidor registraba automáticamente la recepción de la correspondiente petición HTTP.

La observación de estos registros permitió confirmar la existencia de una comunicación estable entre ambos dispositivos.

La aparición continuada de códigos HTTP 200 OK confirmó que las peticiones eran procesadas satisfactoriamente por el servidor.

Asimismo, la ausencia de errores durante periodos prolongados de funcionamiento permitió verificar la estabilidad del mecanismo de comunicación implementado.

5.6.5.5 Validación Funcional del Sistema Completo

Finalmente, se realizaron diferentes pruebas funcionales utilizando distintas configuraciones de ocupación del aparcamiento de bicicletas.

Durante estas pruebas se modificó el número de bicicletas presentes en la escena y se comprobó que la información mostrada por la interfaz web coincidía con el número de bicicletas detectadas por el sistema.

Para cada escenario considerado, se verificó:

- El número de bicicletas detectadas.
- El número de plazas libres calculadas.
- La información enviada mediante HTTP.
- La información mostrada en la interfaz web.

Los resultados obtenidos demostraron una total coherencia entre las diferentes etapas del sistema.

La Figura 19 muestra Demo de la web funcionando en tiempo real, detectando dos bicicletas

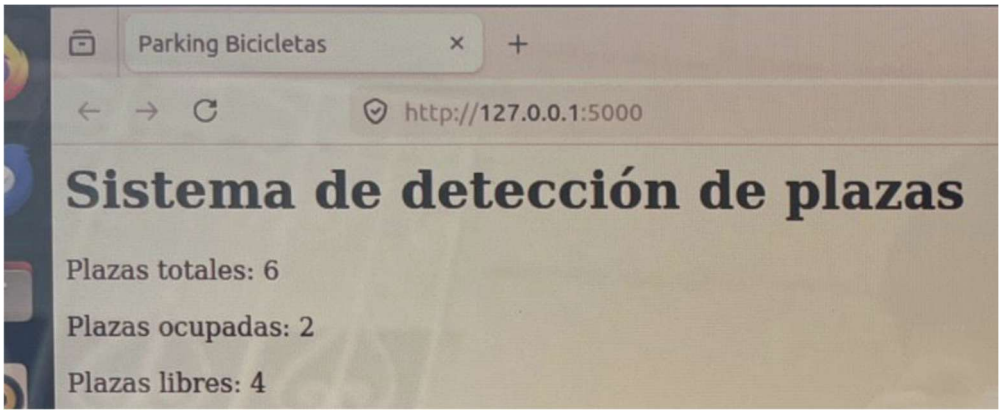


Ilustración 19 Demo de la web funcionando en tiempo real, detectando dos bicicletas

La Figura 20 muestra de la base de datos de la web, guardando los datos obtenidos

Ilustración 20 Muestra de la base de datos de la web, guardando los datos obtenidos

La correcta correspondencia entre la información mostrada por la aplicación de detección y la interfaz web confirmó la validez de la arquitectura propuesta y demostró la viabilidad del sistema desarrollado para aplicaciones de monitorización inteligente de aparcamientos de bicicletas.

En consecuencia, puede afirmarse que el subsistema de comunicaciones desarrollado cumple satisfactoriamente los requisitos establecidos inicialmente, permitiendo la transmisión automática, fiable y prácticamente en tiempo real de la información generada por el sistema de visión artificial.

5.7 Desarrollo de la Interfaz Web de Visualización

Uno de los principales objetivos perseguidos durante el desarrollo del presente proyecto consistía en proporcionar al usuario final una herramienta sencilla que permitiera consultar el estado de ocupación del aparcamiento monitorizado. Aunque la Plataforma es capaz de ejecutar el modelo de inteligencia artificial y mostrar localmente determinados resultados mediante un monitor conectado directamente a la placa, esta solución presenta importantes limitaciones desde el punto de vista de la usabilidad y la accesibilidad.

En primer lugar, la necesidad de disponer físicamente de un monitor conectado a la plataforma dificulta enormemente la utilización práctica del sistema en un escenario real. En un entorno de despliegue operativo, el usuario no debería tener que acceder físicamente al dispositivo embebido para conocer la disponibilidad del aparcamiento. Por el contrario, la información debería encontrarse disponible de forma remota y accesible desde cualquier dispositivo autorizado conectado a la red.

Además, la visualización directa sobre la Plataforma limita considerablemente la capacidad de ampliación futura del sistema. Por ejemplo, la integración con aplicaciones móviles, plataformas *IoT* o sistemas de supervisión remota resultaría significativamente más compleja si la información permaneciese exclusivamente almacenada en la Plataforma.

Por estos motivos, se desarrolló una interfaz web capaz de mostrar de forma clara y accesible la información generada por el sistema de detección. Esta solución permite desacoplar completamente el proceso de inferencia ejecutado sobre la Plataforma de la capa de presentación destinada al usuario final.

La utilización de tecnologías web proporciona además numerosas ventajas adicionales. Entre ellas destacan la independencia respecto al sistema operativo utilizado por el usuario, la posibilidad de acceder a la información desde diferentes dispositivos y la facilidad de mantenimiento y ampliación del sistema. Gracias a este enfoque, el usuario puede consultar el estado del aparcamiento utilizando cualquier navegador web convencional sin necesidad de instalar software adicional.

5.7.1 Selección de la Tecnología de Desarrollo

Para implementar la interfaz de visualización se optó por desarrollar una aplicación web ligera ejecutada sobre un ordenador con sistema operativo Ubuntu. Esta decisión se fundamentó principalmente en criterios de simplicidad, rapidez de desarrollo y compatibilidad con el resto de elementos del sistema.

El servidor web se desarrolló utilizando el lenguaje de programación Python junto con el *framework* Flask. Flask es un *framework* minimalista orientado al desarrollo de aplicaciones web ligeras y servicios basados en HTTP. Su reducido tamaño y sencillez de utilización lo convierten en una herramienta especialmente adecuada para proyectos de investigación y prototipos funcionales como el desarrollado en este TFG.

La elección de Python como lenguaje de implementación también se justificó por varios motivos. En primer lugar, Python es uno de los lenguajes más utilizados actualmente en aplicaciones relacionadas con la inteligencia artificial, la visión artificial y el Internet de las Cosas, lo que facilita enormemente la integración entre los distintos componentes software del sistema.

Por otra parte, Python dispone de una extensa colección de bibliotecas destinadas al desarrollo web, simplificando considerablemente la implementación de funcionalidades relacionadas con la recepción, tratamiento y visualización de datos.

Finalmente, la experiencia previa adquirida durante el desarrollo de otras partes del proyecto utilizando este lenguaje contribuyó a reducir el tiempo necesario para implementar y validar la aplicación web.

5.7.2 Funcionamiento General de la Aplicación Web

La aplicación web desarrollada funciona siguiendo un esquema cliente-servidor clásico. Dentro de esta arquitectura, el servidor permanece continuamente a la espera de recibir nuevas peticiones procedentes de la Plataforma.

Cada vez que la Plataforma finaliza un ciclo de detección, genera una petición HTTP que contiene información relativa al estado actual del aparcamiento. Esta información es recibida y procesada por el servidor, que actualiza internamente los datos almacenados.

Posteriormente, cuando un usuario accede a la página web mediante un navegador, el servidor genera dinámicamente la interfaz incorporando la información más reciente disponible.

Este mecanismo permite desacoplar completamente las tareas de procesamiento intensivo realizadas por la Plataforma de las funciones relacionadas con la interacción con el usuario.

Desde un punto de vista funcional, la secuencia de funcionamiento del sistema puede resumirse mediante las siguientes etapas:

1. Captura de imágenes por parte de la cámara conectada a la Plataforma.
2. Ejecución del modelo de inteligencia artificial sobre la Plataforma.
3. Obtención del número de bicicletas detectadas.
4. Cálculo del número de plazas ocupadas y libres.
5. Generación y envío de una petición HTTP al servidor local.
6. Actualización de la información almacenada por el servidor.
7. Visualización de la información actualizada mediante la interfaz web.

Gracias a esta arquitectura, el sistema es capaz de proporcionar información prácticamente en tiempo real sin necesidad de transmitir imágenes completas a través de la red.

5.7.3 Diseño de la Interfaz de Usuario

Durante el desarrollo de la interfaz web se prestó especial atención a aspectos relacionados con la simplicidad, claridad y facilidad de utilización.

Dado que el principal objetivo del sistema consiste en informar al usuario acerca de la disponibilidad de plazas en el aparcamiento, se decidió diseñar una interfaz minimalista centrada exclusivamente en mostrar la información esencial.

Esta decisión responde a varios motivos. En primer lugar, una interfaz excesivamente compleja podría dificultar la interpretación rápida de la información mostrada. Además, la incorporación de elementos gráficos innecesarios incrementaría la complejidad del desarrollo sin aportar beneficios significativos para los objetivos planteados en el Proyecto.

Como consecuencia, la interfaz desarrollada presenta únicamente la información estrictamente necesaria para el usuario final.

Concretamente, la aplicación muestra:

- El número total de plazas monitorizadas.
- El número de plazas ocupadas detectadas por el sistema.
- El número de plazas libres disponibles.
- El estado actualizado del aparcamiento.

La información se presenta de forma estructurada y fácilmente interpretable, permitiendo al usuario conocer instantáneamente la situación del aparcamiento sin necesidad de realizar ninguna acción adicional.

Otro aspecto importante considerado durante el diseño fue la posibilidad de acceder a la interfaz desde dispositivos con diferentes resoluciones de pantalla. Aunque el prototipo desarrollado fue validado principalmente utilizando un ordenador portátil, la naturaleza web de la solución permite acceder igualmente desde teléfonos móviles, tabletas u otros dispositivos equipados con un navegador compatible.

5.7.4 Actualización Automática de la Información

Uno de los requisitos fundamentales establecidos durante el diseño del sistema consistía en garantizar que la información mostrada al usuario reflejase en todo momento el estado real del aparcamiento.

Para satisfacer este requisito, la aplicación web fue diseñada para actualizar automáticamente los datos almacenados cada vez que el servidor recibe una nueva petición procedente de la Plataforma.

Este enfoque presenta varias ventajas frente a otras posibles alternativas. En primer lugar, evita la necesidad de almacenar secuencias completas de imágenes o vídeo, reduciendo significativamente el tráfico generado dentro de la red local.

Además, al transmitir únicamente información procesada, se disminuye notablemente la carga computacional asociada al sistema y se mejora la privacidad de los usuarios, puesto que las imágenes capturadas por la cámara no abandonan la Plataforma.

Otro beneficio importante consiste en la reducción de la latencia. Dado que la Plataforma procesa localmente las imágenes y transmite únicamente los resultados obtenidos, el tiempo transcurrido entre la detección de una bicicleta y la actualización de la información mostrada al usuario resulta muy reducido.

En consecuencia, el sistema proporciona una experiencia de utilización fluida y una representación actualizada del estado del aparcamiento de bicicletas.

5.7.5 Validación Experimental de la Interfaz

Una vez implementada la aplicación web, se realizaron diversas pruebas experimentales destinadas a verificar su correcto funcionamiento.

Durante estas pruebas se evaluaron diferentes escenarios de ocupación modificando el número de bicicletas presentes en el aparcamiento monitorizado.

Para cada uno de los escenarios considerados se comprobó que:

- La Plataforma detectaba correctamente las bicicletas presentes.
- El número de plazas libres calculado era coherente con la situación observada.

- El servidor recibía correctamente la información enviada.
- La interfaz web mostraba los datos actualizados.
- La información visualizada coincidía con el estado real del aparcamiento.

Los resultados obtenidos permitieron confirmar el correcto funcionamiento de la interfaz desarrollada y verificar la adecuada integración entre todos los componentes software del sistema.

Asimismo, se comprobó que la aplicación permanecía estable durante periodos prolongados de funcionamiento continuo, sin observarse pérdidas de información ni interrupciones del servicio.

En conjunto, la incorporación de esta interfaz transforma el prototipo desarrollado en una solución *IoT* completa, permitiendo no sólo detectar automáticamente la presencia de bicicletas mediante técnicas de inteligencia artificial, sino también proporcionar información útil y accesible a los usuarios de forma remota.

5.8 Integración y Funcionamiento Global del Sistema

Una vez desarrollados e implementados individualmente todos los subsistemas descritos en los apartados anteriores, se procedió a la fase de integración global de la solución. Esta etapa constituye una de las fases más críticas dentro del desarrollo de sistemas complejos, ya que permite verificar no solamente el correcto funcionamiento de cada componente de forma aislada, sino también su capacidad para operar conjuntamente dentro de una arquitectura común.

La integración del sistema supuso combinar diferentes tecnologías pertenecientes a ámbitos muy diversos, incluyendo sistemas embebidos, visión artificial, inteligencia artificial, comunicaciones de red y desarrollo web. La correcta interacción entre todos estos elementos resulta esencial para garantizar el funcionamiento satisfactorio de la solución propuesta.

A diferencia de aplicaciones convencionales ejecutadas íntegramente sobre un único dispositivo, el sistema desarrollado se encuentra distribuido entre varios elementos hardware y software que deben intercambiar información de forma continua. Por este motivo, la integración no consistió únicamente en unir diferentes fragmentos de código, sino en diseñar una arquitectura completa capaz de coordinar el funcionamiento de todos los componentes involucrados.

Desde un punto de vista general, el sistema final está compuesto por cuatro bloques funcionales principales:

- Subsistema de adquisición de imágenes.
- Subsistema de procesamiento e inferencia.

- Subsistema de comunicaciones.
- Subsistema de visualización.

Cada uno de estos bloques desempeña funciones específicas dentro del flujo global de funcionamiento y presenta requisitos particulares en términos de rendimiento, sincronización y procesamiento.

5.8.1 Subsistema de adquisición de imágenes

El primer elemento de la arquitectura corresponde al sistema de captura de imágenes. Este subsistema está formado por la cámara conectada a la Plataforma y constituye la fuente primaria de información del sistema.

Durante el funcionamiento normal, la cámara captura de forma continua imágenes del aparcamiento monitorizado. Estas imágenes son transferidas inmediatamente a la aplicación ejecutada sobre la plataforma embebida para su posterior procesamiento.

La calidad y estabilidad de las imágenes adquiridas condicionan directamente el rendimiento global del sistema. Por este motivo, durante el desarrollo experimental se procuró mantener unas condiciones de iluminación relativamente estables.

Cada imagen capturada representa una instantánea del estado del aparcamiento en un instante temporal determinado. A partir de esta información visual, el resto de componentes del sistema llevan a cabo las tareas necesarias para determinar la disponibilidad de plazas.

5.8.2 Subsistema de procesamiento e inferencia

Una vez adquirida la imagen, ésta es transferida al subsistema encargado del procesamiento y análisis visual.

Este subsistema se ejecuta íntegramente sobre la Plataforma y constituye el núcleo principal de la solución desarrollada.

El procesamiento comienza con la ejecución del modelo de inteligencia artificial acelerado mediante el hardware DRP-AI integrado en la plataforma. Gracias al uso de este acelerador especializado, resulta posible realizar inferencias en tiempo real manteniendo un consumo energético reducido.

El modelo genera como salida un conjunto de detecciones correspondientes a los distintos objetos presentes en la escena. Cada detección incluye información relativa a:

- Clase del objeto detectado.
- Coordenadas espaciales de la región detectada.
- Nivel de confianza asociado a la predicción.

Sin embargo, el objetivo del Proyecto no consiste en detectar cualquier tipo de objeto, sino únicamente bicicletas. Por este motivo, se implementó un proceso adicional de filtrado encargado de descartar todas aquellas detecciones que no correspondiesen a la clase de interés.

Posteriormente, el sistema contabiliza automáticamente el número total de bicicletas presentes en la escena y calcula el número de plazas disponibles.

Este procesamiento se realiza de manera completamente automática y sin intervención del usuario, permitiendo obtener información actualizada sobre el estado del aparcamiento en cada ciclo de ejecución.

5.8.3 Subsistema de comunicaciones

Una vez obtenido el estado de ocupación del aparcamiento, la información debe ser transmitida al resto del sistema para su posterior visualización.

Para ello, se desarrolló un subsistema de comunicaciones basado en protocolos estándar de red, concretamente utilizando HTTP sobre una red local.

La Plataforma actúa como cliente dentro de la arquitectura propuesta, siendo responsable de generar y enviar las actualizaciones correspondientes al estado del aparcamiento.

Por su parte, el ordenador Ubuntu que ejecuta el servidor local permanece continuamente a la espera de nuevas peticiones.

Esta arquitectura presenta múltiples ventajas. En primer lugar, permite desacoplar completamente las tareas de procesamiento intensivo de las tareas de visualización. Además, facilita enormemente la escalabilidad del sistema, ya que futuras ampliaciones podrían incorporar nuevos clientes, aplicaciones móviles o servicios cloud sin necesidad de modificar la lógica principal de detección.

Otro aspecto importante es la utilización de mensajes en formato JSON para intercambiar información entre dispositivos. Este formato presenta una gran interoperabilidad y constituye actualmente uno de los estándares más utilizados en aplicaciones *IoT* y sistemas distribuidos.

Gracias a ello, la solución desarrollada podría integrarse fácilmente con otras plataformas software en futuras versiones del sistema.

5.8.4 Subsistema de visualización

El último elemento de la arquitectura corresponde al subsistema de visualización.

Este subsistema tiene como finalidad proporcionar una representación clara e intuitiva de la información generada por la Plataforma.

La aplicación web desarrollada recibe los datos procedentes del servidor y los presenta al usuario final mediante una interfaz sencilla y fácilmente interpretable.

La principal ventaja de esta aproximación reside en que el usuario no necesita interactuar directamente con la Plataforma ni disponer de conocimientos técnicos específicos. Basta con acceder a la interfaz web utilizando un navegador convencional para consultar el estado actualizado del aparcamiento.

Asimismo, al tratarse de una solución basada en tecnologías web, la información puede consultarse desde múltiples dispositivos, incluyendo ordenadores personales, tabletas o teléfonos inteligentes.

5.8.5 Flujo Global de Funcionamiento

Considerando conjuntamente todos los subsistemas descritos anteriormente, el funcionamiento global de la solución desarrollada puede resumirse mediante la siguiente secuencia de operaciones:

1. La cámara captura una imagen del aparcamiento.
2. La imagen es transferida a la Plataforma.
3. El modelo de inteligencia artificial ejecuta el proceso de inferencia.
4. Se obtienen las detecciones correspondientes a los objetos presentes en la escena.
5. El sistema filtra exclusivamente las detecciones correspondientes a bicicletas.
6. Se contabiliza el número total de bicicletas detectadas.
7. Se calcula el número de plazas libres y ocupadas.
8. Se genera un mensaje JSON con la información obtenida.
9. La plataforma envía el mensaje al servidor local mediante una petición HTTP.
10. El servidor actualiza los datos almacenados.
11. La interfaz web muestra al usuario la información actualizada.

La Figura 21 representa gráficamente el flujo global de funcionamiento del sistema integrado.

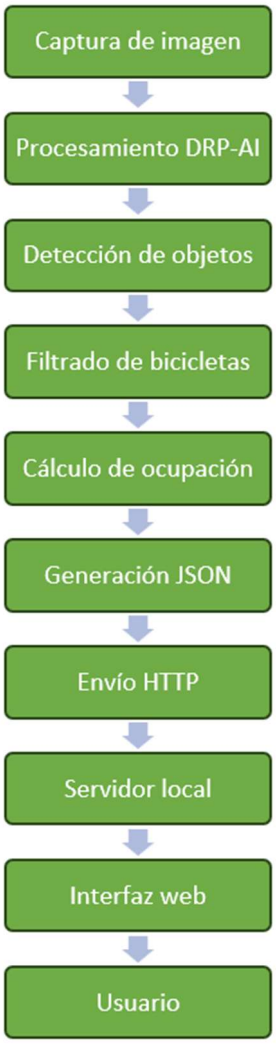


Ilustración 21. Flujo global de funcionamiento del sistema integrado

5.8.6 Consideraciones sobre la Arquitectura Implementada

Uno de los principales aspectos positivos de la arquitectura desarrollada es su carácter altamente modular.

Cada uno de los subsistemas implementados puede evolucionar de forma independiente sin afectar significativamente al resto de componentes.

Por ejemplo, sería posible sustituir el modelo de inteligencia artificial actual por futuros modelos más avanzados manteniendo inalterado el sistema de comunicaciones y la interfaz web. Del mismo modo, el servidor local podría reemplazarse por una

infraestructura cloud sin necesidad de modificar el algoritmo de detección ejecutado sobre la Plataforma.

La modularidad facilita enormemente el mantenimiento, la reutilización y la escalabilidad futura del sistema.

Por otra parte, la utilización de estándares ampliamente aceptados, como HTTP y JSON, garantiza la interoperabilidad de la solución con tecnologías y plataformas desarrolladas por terceros.

Desde el punto de vista experimental, la integración realizada permitió comprobar que todos los componentes desarrollados interactúan correctamente entre sí, proporcionando una solución funcional capaz de monitorizar automáticamente un aparcamiento de bicicletas y comunicar dicha información a los usuarios prácticamente en tiempo real.

La obtención de este sistema integrado constituye el principal resultado tecnológico del Proyecto y demuestra la viabilidad del uso combinado de técnicas de *Edge AI*, sistemas embebidos y tecnologías *IoT* para el desarrollo de soluciones inteligentes orientadas a la gestión de infraestructuras urbanas.

5.8.7 Sincronización entre Subsistemas

Uno de los aspectos más importantes considerados durante la integración del sistema fue la sincronización entre los distintos subsistemas que componen la solución desarrollada.

A diferencia de aplicaciones ejecutadas sobre un único dispositivo, en las que todos los procesos comparten el mismo entorno de ejecución, la solución propuesta distribuye las tareas entre varios elementos hardware independientes. Como consecuencia, resulta necesario garantizar que la información generada por cada uno de los subsistemas sea transferida y procesada de forma coordinada.

En el sistema desarrollado, la sincronización se realiza de forma implícita mediante la propia secuencia de ejecución de la aplicación principal. Cada ciclo de funcionamiento comienza con la captura de una nueva imagen y continúa secuencialmente con la inferencia, el filtrado de detecciones, el cálculo del estado de ocupación y el envío de la información al servidor.

Este enfoque secuencial simplifica notablemente el diseño del sistema, ya que evita la necesidad de implementar mecanismos adicionales de sincronización entre procesos o técnicas de comunicación más complejas.

Además, dado que el volumen de información intercambiado entre la Plataforma y el servidor local es relativamente reducido, la latencia introducida por el sistema de comunicaciones resulta despreciable frente al tiempo total requerido para completar el ciclo de procesamiento.

Durante las pruebas experimentales realizadas no se observaron problemas de sincronización ni inconsistencias entre la información generada por la plataforma y la

mostrada por la interfaz web, lo que demuestra la adecuación del mecanismo implementado para el alcance del proyecto.

5.8.8 Escalabilidad del Sistema

Otro aspecto relevante de la arquitectura desarrollada es su capacidad de escalabilidad.

Aunque el prototipo implementado se ha diseñado específicamente para monitorizar un aparcamiento compuesto por seis plazas, la arquitectura software utilizada permite extender fácilmente la solución a escenarios de mayor tamaño.

Por ejemplo, sería posible aumentar el número de plazas monitorizadas modificando únicamente determinados parámetros software relacionados con el cálculo de ocupación, manteniendo inalterado el resto del sistema.

Asimismo, la arquitectura cliente-servidor implementada permitiría integrar simultáneamente varias Plataformas monitorizando diferentes aparcamientos y transmitiendo toda la información a un único servidor centralizado.

Esta posibilidad abre la puerta al desarrollo de sistemas distribuidos capaces de supervisar múltiples infraestructuras urbanas de manera simultánea.

Del mismo modo, el servidor local podría evolucionar hacia una solución basada en servicios cloud, permitiendo el acceso remoto a la información desde cualquier ubicación geográfica con acceso a Internet.

La utilización de protocolos de comunicación estándar y formatos de intercambio ampliamente aceptados facilita considerablemente este proceso de escalado.

5.8.9 Fiabilidad y Robustez de la Solución Integrada

La integración final del sistema permitió también evaluar la robustez de la solución propuesta frente a diferentes situaciones de funcionamiento.

Durante las pruebas realizadas, el sistema fue capaz de operar de manera continua durante periodos prolongados de tiempo sin experimentar interrupciones significativas en el flujo de procesamiento ni pérdidas apreciables de información.

La utilización de tecnologías maduras y ampliamente utilizadas, como Linux, HTTP, Python o Flask, contribuyó de forma significativa a mejorar la estabilidad general de la solución.

Por otra parte, la propia modularidad de la arquitectura facilita enormemente las tareas de mantenimiento y depuración. En caso de producirse algún fallo en uno de los subsistemas, éste puede analizarse y corregirse de manera aislada sin necesidad de modificar el resto de componentes del sistema.

Este aspecto resulta especialmente importante en aplicaciones *IoT* y sistemas embebidos, donde la capacidad de identificar rápidamente posibles incidencias constituye un requisito fundamental para garantizar la disponibilidad del servicio.

5.8.10 Consideraciones Finales

En conjunto, la integración satisfactoria de todos los componentes hardware y software desarrollados ha permitido obtener una solución completamente funcional capaz de monitorizar automáticamente un aparcamiento de bicicletas utilizando técnicas de visión artificial e inteligencia artificial ejecutadas en el sistema.

La solución propuesta no sólo es capaz de detectar la presencia de bicicletas y calcular automáticamente el estado de ocupación del aparcamiento, sino también de comunicar dicha información a un servidor externo y ponerla a disposición de los usuarios mediante una interfaz web accesible desde diferentes dispositivos.

La obtención de este prototipo integrado constituye el principal logro tecnológico alcanzado durante el desarrollo del Proyecto y demuestra el potencial de las tecnologías *Edge AI* para el desarrollo de soluciones inteligentes orientadas a la gestión eficiente de infraestructuras urbanas.

CAPÍTULO 6. RESULTADOS EXPERIMENTALES Y VALIDACIÓN

6.1 Introducción

Una vez completado el desarrollo e integración de todos los componentes hardware y software que constituyen la solución propuesta, resulta necesario evaluar experimentalmente el comportamiento del sistema con el fin de determinar su grado de cumplimiento respecto a los objetivos inicialmente planteados.

La validación experimental constituye una etapa fundamental dentro del desarrollo de cualquier sistema basado en técnicas de inteligencia artificial y visión artificial, ya que permite verificar no sólo el correcto funcionamiento individual de cada uno de los componentes implementados, sino también analizar el comportamiento global de la solución cuando ésta opera en condiciones reales de funcionamiento.

En el Proyecto la evaluación experimental tiene como objetivo principal determinar la capacidad del sistema para monitorizar automáticamente un aparcamiento de bicicletas utilizando técnicas de visión artificial ejecutadas sobre una plataforma *Edge AI*.

Concretamente, a lo largo de este capítulo se analizarán aspectos relacionados con la precisión del sistema de detección, la fiabilidad del subsistema de comunicaciones, el comportamiento de la interfaz web desarrollada y la capacidad de la solución para operar de forma integrada y autónoma.

Para llevar a cabo dicha evaluación, se diseñó un conjunto de pruebas experimentales orientadas a validar cada uno de los subsistemas que componen la arquitectura desarrollada.

En primer lugar, se analizará el entorno experimental utilizado durante las pruebas, describiendo detalladamente tanto la infraestructura hardware empleada como las condiciones bajo las cuales se realizaron los experimentos.

Posteriormente, se estudiará el comportamiento del sistema de detección evaluando su capacidad para identificar correctamente bicicletas presentes en el aparcamiento monitorizado.

A continuación, se validará el funcionamiento del subsistema de comunicaciones implementado entre la Plataforma y el servidor local, verificando la correcta transmisión de información mediante peticiones HTTP.

Asimismo, se analizará el comportamiento de la interfaz web desarrollada, comprobando su capacidad para representar adecuadamente el estado del aparcamiento y actualizar automáticamente la información mostrada al usuario.

Finalmente, se realizará una evaluación global del sistema, discutiendo las principales fortalezas y limitaciones observadas durante el desarrollo experimental y analizando posibles líneas de mejora para futuros trabajos.

Los resultados presentados en este capítulo permitirán determinar la viabilidad técnica de la solución propuesta y valorar el potencial de las plataformas *Edge AI* para el desarrollo de sistemas inteligentes orientados a la gestión eficiente de infraestructuras urbanas.

6.2 Entorno experimental

6.2.1 Escenario de Pruebas

La evaluación experimental del sistema desarrollado se llevó a cabo utilizando un escenario diseñado específicamente para reproducir, en la medida de lo posible, las condiciones de funcionamiento previstas para una futura implementación real del sistema.

El objetivo principal de esta fase experimental consistió en validar la capacidad de la solución propuesta para detectar automáticamente bicicletas estacionadas y determinar el estado de ocupación de un aparcamiento a partir de la información visual capturada por una cámara.

El escenario seleccionado para la realización de las pruebas experimentales consistió en un aparcamiento de bicicletas compuesto por un total de seis plazas de estacionamiento. La elección de este entorno respondió tanto a criterios prácticos como técnicos, ya que permitía disponer de un espacio suficientemente representativo para evaluar el funcionamiento del sistema, manteniendo al mismo tiempo un elevado grado de control sobre las condiciones experimentales.

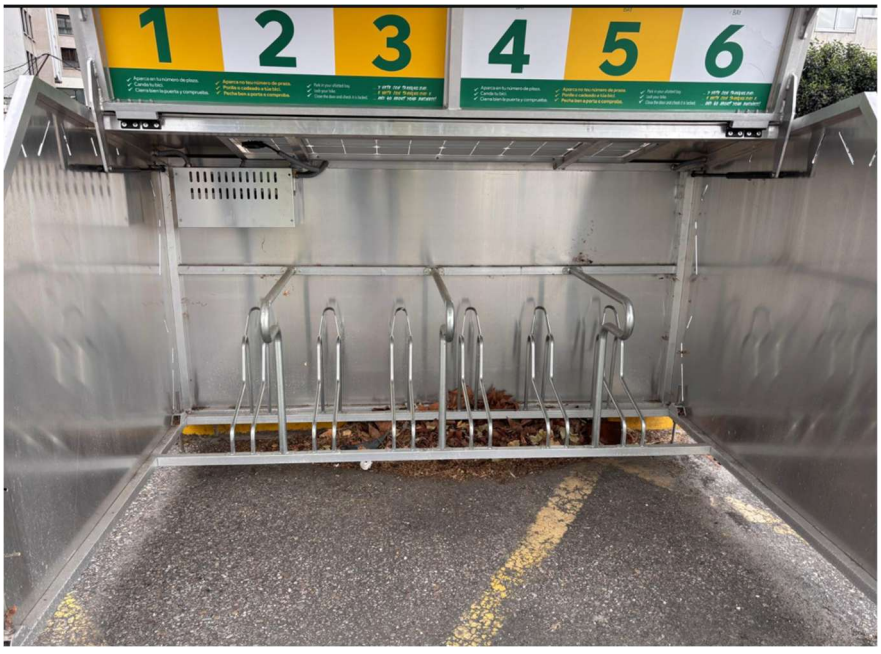
Desde el punto de vista metodológico, la utilización de un aparcamiento de dimensiones reducidas presenta diversas ventajas. En primer lugar, facilita la realización de pruebas repetitivas y controladas, permitiendo modificar fácilmente la disposición de las bicicletas presentes en la escena y analizar el comportamiento del sistema ante diferentes niveles de ocupación.

Además, la existencia de un número conocido y limitado de plazas simplifica considerablemente el proceso de validación, puesto que permite comparar directamente el número real de bicicletas presentes en el aparcamiento con el número de detecciones generadas por el sistema de inteligencia artificial. Esta comparación resulta esencial para determinar la precisión de la solución desarrollada y evaluar el grado de cumplimiento de los objetivos inicialmente establecidos.

Durante la fase experimental se llevaron a cabo múltiples pruebas considerando diferentes configuraciones de ocupación del aparcamiento. Concretamente, se realizaron ensayos correspondientes a situaciones en las que el aparcamiento se encontraba completamente vacío, así como escenarios con diferentes niveles de ocupación parcial.

La evaluación de distintos escenarios permitió analizar el comportamiento del sistema bajo diversas condiciones de funcionamiento y comprobar su capacidad para adaptarse a variaciones en el número de bicicletas presentes en la escena.

Las Figuras 22 y 23 muestran el escenario experimental empleado durante el proceso de validación.



Ilustraciones 22 y 23 Escenario experimental utilizado para la validación del sistema, compuesto por un aparcamiento de seis plazas monitorizado mediante una cámara conectada a la plataforma Renesas RZ/V2L.

6.2.2 Utilización de un Entorno Real

Una de las principales decisiones adoptadas durante el diseño experimental consistió en utilizar un entorno físico real para la validación del sistema en lugar de recurrir exclusivamente a imágenes pregrabadas o escenarios simulados.

La utilización de un entorno real presenta importantes ventajas desde el punto de vista de la validez experimental. En primer lugar, permite evaluar el comportamiento del sistema frente a situaciones que aparecen de forma natural durante el funcionamiento normal y que resultan difíciles de reproducir mediante simulaciones.

Entre estos factores pueden mencionarse pequeñas variaciones en la iluminación, ligeros cambios en la posición de las bicicletas, diferencias en la orientación de los objetos presentes en la escena o pequeñas oclusiones parciales producidas por elementos del entorno.

Asimismo, la utilización de un escenario real proporciona una evaluación más rigurosa de la solución propuesta, ya que obliga al sistema a operar bajo condiciones próximas a las que podrían encontrarse en una implementación práctica.

No obstante, la utilización de entornos reales también introduce ciertos inconvenientes. Por ejemplo, las condiciones de adquisición no pueden mantenerse completamente constantes, pudiendo aparecer variaciones asociadas a factores externos no controlados experimentalmente.

A pesar de ello, se consideró que las ventajas derivadas de utilizar un escenario real superaban ampliamente las limitaciones asociadas a esta aproximación, proporcionando resultados más representativos y cercanos a una futura aplicación práctica del sistema.

6.2.2.1 Representatividad del Escenario Seleccionado

Aunque el escenario experimental utilizado durante las pruebas presenta unas dimensiones reducidas, se considera suficientemente representativo para validar la viabilidad técnica de la solución propuesta.

Las funcionalidades principales desarrolladas en el Proyecto, tales como la detección automática de bicicletas, el cálculo del número de plazas libres o la transmisión de información a través de la red, no dependen directamente del número exacto de plazas monitorizadas.

Por este motivo, la utilización de un aparcamiento de seis plazas resulta adecuada para demostrar el correcto funcionamiento del sistema y verificar la interacción entre los diferentes subsistemas implementados.

Además, la arquitectura software desarrollada presenta un carácter modular, lo que facilita futuras ampliaciones orientadas a la monitorización de aparcamientos de mayor tamaño. En consecuencia, el escenario utilizado puede considerarse un prototipo representativo a pequeña escala de un sistema de monitorización más complejo.

Desde un punto de vista académico, la utilización de un escenario de dimensiones reducidas también presenta la ventaja de simplificar la interpretación de resultados y facilitar la identificación de posibles errores o limitaciones del sistema.

6.2.2.2 Configuraciones Experimentales Analizadas

Con el objetivo de obtener una evaluación lo más completa posible del comportamiento del sistema, se diseñaron diferentes configuraciones experimentales variando el número de bicicletas presentes en la escena.

Las pruebas realizadas incluyeron situaciones en las que no existían bicicletas en el aparcamiento, así como escenarios con distintos niveles de ocupación parcial.

Este procedimiento permitió analizar la respuesta del sistema ante diferentes estados del aparcamiento y verificar la estabilidad del algoritmo implementado para el cálculo de plazas disponibles.

Además, la realización de múltiples configuraciones experimentales permitió estudiar el comportamiento del sistema ante cambios progresivos en el nivel de ocupación y comprobar que la información mostrada en la interfaz web coincidía con el estado real del aparcamiento.

Durante todas las pruebas se registró tanto el número real de bicicletas presentes como el número de bicicletas detectadas por el sistema, permitiendo realizar posteriormente un análisis detallado de la precisión obtenida.

6.2.2.3 Posicionamiento de la Cámara

Otro aspecto fundamental dentro del diseño experimental fue la ubicación de la cámara encargada de capturar las imágenes del aparcamiento.

La cámara fue instalada en una posición fija, manteniéndose inalterada durante toda la fase experimental. Esta decisión se adoptó con el objetivo de garantizar la reproducibilidad de las pruebas y asegurar que todas las configuraciones experimentales fueran evaluadas bajo las mismas condiciones de adquisición.

La posición seleccionada permitió obtener una visión completa de la totalidad de las plazas monitorizadas, maximizando la cantidad de información disponible para el algoritmo de detección.

La utilización de una posición fija presenta además la ventaja de simplificar considerablemente el procesamiento posterior de las imágenes, ya que evita la necesidad de recalibrar el sistema ante cambios en la orientación de la cámara.

6.2.2.4 Reproducibilidad de los Experimentos

Con objeto de garantizar la validez científica de los resultados obtenidos, todas las pruebas experimentales se realizaron manteniendo constantes las principales condiciones de funcionamiento del sistema.

Concretamente, durante toda la fase experimental se conservaron invariables los siguientes aspectos:

- La posición y orientación de la cámara.
- La plataforma hardware utilizada.
- El modelo de inteligencia artificial empleado.
- La configuración software del sistema.
- Los parámetros asociados al proceso de inferencia.
- La arquitectura de comunicaciones implementada.

Gracias a estas medidas fue posible asegurar que las diferencias observadas entre experimentos se debieran exclusivamente a variaciones en la ocupación del aparcamiento y no a modificaciones introducidas por factores externos.

La adopción de una metodología experimental reproducible constituye un requisito fundamental dentro de cualquier trabajo de investigación, ya que permite garantizar la consistencia de los resultados obtenidos y facilita la comparación con futuros desarrollos realizados sobre la misma plataforma.

En conjunto, el escenario experimental diseñado permitió validar de forma satisfactoria el funcionamiento del sistema desarrollado y proporcionó un entorno adecuado para la realización de las pruebas presentadas en los apartados siguientes.

6.2.3 Configuración Hardware

La infraestructura hardware empleada durante el desarrollo y validación experimental del sistema se encuentra compuesta por diversos dispositivos, cada uno de ellos encargado de desempeñar funciones específicas dentro de la arquitectura global de la solución propuesta.

La selección de los diferentes componentes hardware se realizó teniendo en consideración factores tales como la capacidad de procesamiento requerida, la disponibilidad de interfaces de comunicación compatibles, la facilidad de integración y la adecuación de los dispositivos a los objetivos planteados en el Proyecto.

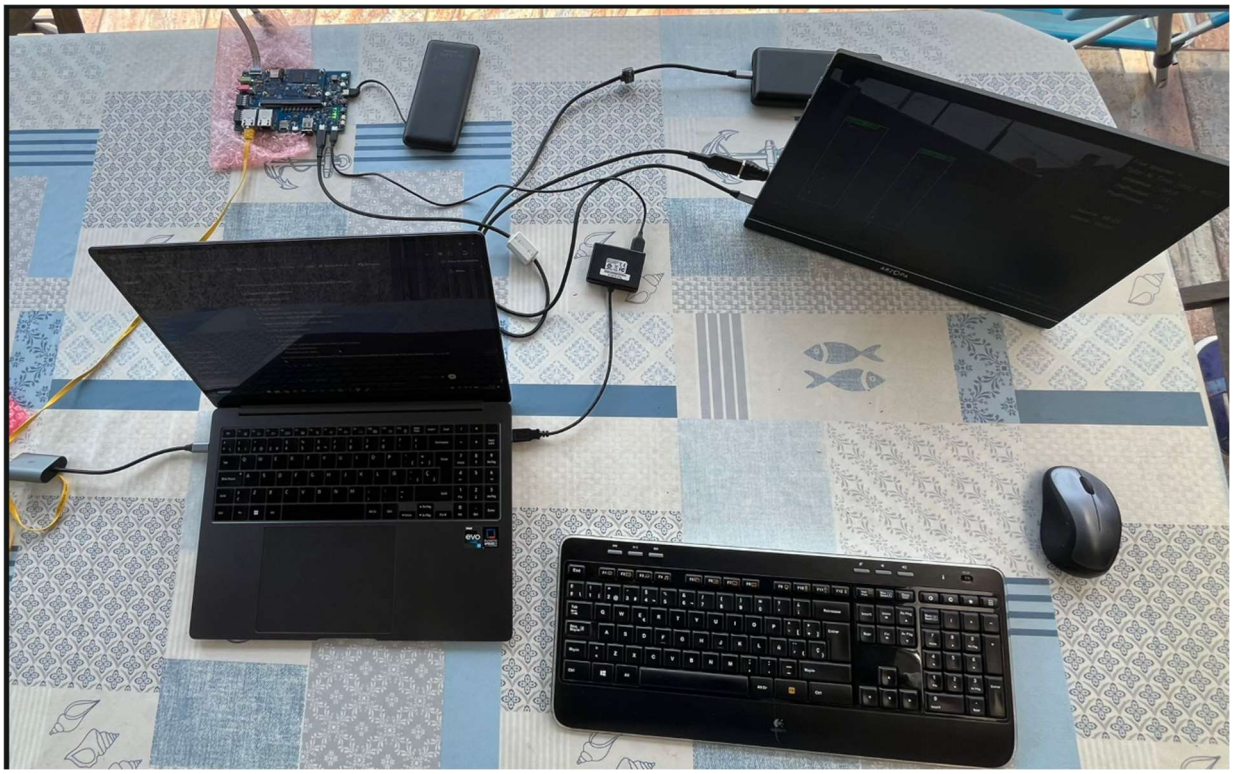


Figura 24. Principales dispositivos hardware utilizados durante el desarrollo y validación experimental del sistema propuesto

6.2.3.1 Plataforma Renesas RZ/V2L

El elemento principal de la arquitectura desarrollada corresponde a la plataforma Renesas RZ/V2L, utilizada como dispositivo de procesamiento embebido y núcleo central del sistema.

Esta plataforma fue seleccionada debido a su orientación específica hacia aplicaciones de inteligencia artificial ejecutadas en el borde de la red (*Edge AI*), así como por la incorporación de recursos hardware especializados para la aceleración de tareas de inferencia.

La plataforma integra un procesador basado en arquitectura ARM de 64 bits y dispone de un acelerador hardware DRP-AI (*Dynamically Reconfigurable Processor for Artificial Intelligence*), específicamente diseñado para la ejecución eficiente de modelos de inteligencia artificial en sistemas embebidos.

La incorporación de este acelerador constituye uno de los aspectos más relevantes de la plataforma, puesto que permite ejecutar algoritmos de visión artificial manteniendo un consumo energético significativamente inferior al requerido por soluciones basadas exclusivamente en procesadores de propósito general.

Desde el punto de vista funcional, la Plataforma fue responsable de realizar las siguientes tareas:

- Captura de imágenes procedentes de la cámara.
- Ejecución del sistema operativo Linux.
- Procesamiento de las imágenes adquiridas.
- Ejecución del modelo de inteligencia artificial.
- Filtrado de detecciones correspondientes a bicicletas.
- Cálculo del estado de ocupación del aparcamiento.
- Generación de los mensajes de comunicación.
- Transmisión de información al servidor local.

Como consecuencia, puede afirmarse que la Plataforma constituye el elemento central de toda la arquitectura desarrollada.

La elección de una plataforma *Edge AI* frente a otras alternativas más convencionales se justifica principalmente por la posibilidad de realizar el procesamiento localmente, evitando la necesidad de transmitir continuamente imágenes hacia servidores externos.

Este enfoque presenta múltiples ventajas, entre las que destacan:

- Reducción del tráfico generado en la red.
- Menor dependencia de la conectividad externa.
- Disminución de la latencia.
- Mejora de la privacidad de los usuarios.
- Incremento de la autonomía del sistema.

Además, el empleo de una plataforma específicamente orientada a aplicaciones de inteligencia artificial facilita enormemente la implementación de soluciones *IoT* inteligentes destinadas a entornos urbanos.

6.2.3.2 Cámara de Captura de Imágenes

La adquisición de información visual se realizó mediante una cámara conectada directamente a la Plataforma.

La cámara constituye el elemento encargado de proporcionar la información de entrada necesaria para el funcionamiento del sistema, siendo responsable de capturar continuamente imágenes del aparcamiento monitorizado.

La calidad de las imágenes adquiridas influye directamente sobre el rendimiento global del sistema de detección. Por este motivo, durante el desarrollo experimental se prestó especial atención a aspectos relacionados con la ubicación de la cámara, el encuadre obtenido y las condiciones de iluminación existentes.

Durante todas las pruebas realizadas, la cámara permaneció instalada en una posición fija con objeto de garantizar la reproducibilidad de los experimentos.

La utilización de una cámara fija simplifica considerablemente el problema de detección, ya que evita la necesidad de realizar procedimientos adicionales de calibración o compensación asociados a movimientos del dispositivo de captura.

Asimismo, mantener constante la posición de la cámara permite asegurar que todas las pruebas experimentales se realizan bajo condiciones equivalentes, facilitando la comparación objetiva de los resultados obtenidos.

Desde el punto de vista del sistema de inteligencia artificial, la cámara puede considerarse el sensor principal de la arquitectura propuesta, puesto que toda la información utilizada durante el proceso de inferencia procede de las imágenes adquiridas por este dispositivo.

6.2.3.3 Ordenador Utilizado como Servidor Local

Otro de los elementos fundamentales dentro de la arquitectura desarrollada corresponde al ordenador utilizado como servidor local.

Este equipo ejecuta el sistema operativo Ubuntu Linux y alberga la aplicación web desarrollada mediante el *framework* Flask.

La utilización de un servidor local permite separar físicamente las tareas de procesamiento visual realizadas por la Plataforma de las tareas relacionadas con la visualización y gestión de la información.

Esta separación funcional presenta importantes ventajas desde el punto de vista de la modularidad y escalabilidad del sistema.

Entre las funciones desempeñadas por el servidor local pueden destacarse las siguientes:

- Recepción de peticiones HTTP procedentes de la Plataforma.
- Procesamiento de mensajes JSON.
- Actualización de la información relativa al estado del aparcamiento.
- Generación dinámica de la interfaz web.
- Presentación de la información al usuario final.

La elección de Ubuntu Linux como sistema operativo se fundamentó principalmente en su estabilidad, amplia disponibilidad de herramientas de desarrollo y excelente compatibilidad con las tecnologías software empleadas durante el proyecto.

Adicionalmente, la utilización de un ordenador convencional como servidor simplificó considerablemente las tareas de depuración y validación realizadas durante el desarrollo experimental.

6.2.3.4 Infraestructura de Red

La comunicación entre la Plataforma embebida y el servidor local se realizó mediante una red local proporcionada por un router.

Este dispositivo fue el encargado de proporcionar conectividad IP a todos los elementos de la arquitectura, permitiendo el intercambio de información mediante el protocolo HTTP.

La utilización de una red local presenta diversas ventajas dentro del contexto del Proyecto.

En primer lugar, permite disponer de un entorno de comunicaciones completamente controlado, facilitando las tareas de validación y depuración.

Asimismo, elimina la dependencia de servicios externos o conexiones a Internet, incrementando la robustez y autonomía de la solución desarrollada.

Por otra parte, el empleo de un router demuestra que el sistema puede implementarse utilizando infraestructuras de comunicaciones sencillas y fácilmente disponibles.

Aunque en futuras versiones sería posible sustituir esta arquitectura por soluciones basadas en infraestructuras cloud, la configuración utilizada durante el presente trabajo resultó adecuada para validar la viabilidad técnica de la propuesta.

6.2.3.5 Dispositivos Auxiliares de Desarrollo

Además de los elementos descritos anteriormente, durante el desarrollo experimental se emplearon diversos dispositivos auxiliares destinados a facilitar las tareas de configuración, depuración y validación del sistema.

Entre ellos destacan un monitor HDMI, un teclado USB y un ratón USB conectados directamente a la Plataforma.

Estos periféricos permitieron interactuar localmente con el sistema operativo ejecutado sobre la Plataforma, facilitando la ejecución de aplicaciones, la monitorización del sistema y la realización de pruebas experimentales.

La utilización de estos dispositivos simplificó significativamente el proceso de desarrollo y permitió acelerar las tareas de integración y depuración de la solución propuesta.

En conjunto, la infraestructura hardware utilizada proporcionó todos los recursos necesarios para implementar, validar y evaluar experimentalmente el sistema desarrollado, permitiendo demostrar la viabilidad del empleo de tecnologías *Edge AI* para la monitorización inteligente de aparcamientos de bicicletas.

6.2.4 Configuración Software

Además de la infraestructura hardware descrita en el apartado anterior, el desarrollo del sistema requirió la utilización de un conjunto de herramientas software encargadas de proporcionar el entorno necesario para la implementación, ejecución y validación de la solución propuesta.

La correcta selección de las tecnologías software empleadas constituye un aspecto fundamental dentro del diseño de cualquier sistema embebido basado en inteligencia artificial. La elección de cada herramienta se realizó considerando criterios relacionados con la compatibilidad con la plataforma hardware seleccionada, la facilidad de integración, la disponibilidad de documentación técnica y la adecuación a los objetivos específicos del proyecto.

Desde un punto de vista general, la arquitectura software desarrollada puede dividirse en cuatro grandes bloques funcionales:

- Sistema operativo y entorno de ejecución sobre la Plataforma.
- Aplicación de detección de bicicletas ejecutada sobre la Plataforma.
- Subsistema servidor encargado de recibir y procesar la información.
- Aplicación web utilizada para la visualización de resultados.

Cada uno de estos bloques desempeña funciones específicas dentro del funcionamiento global del sistema y su integración resulta esencial para garantizar el correcto comportamiento de la solución desarrollada.

6.2.4.1 Sistema Operativo de la Plataforma

La Plataforma utilizada durante el desarrollo ejecuta una solución Linux proporcionada por el fabricante.

La elección de Linux como sistema operativo presenta numerosas ventajas dentro del ámbito de los sistemas embebidos y las aplicaciones de inteligencia artificial. Entre ellas destacan su estabilidad, flexibilidad, amplia disponibilidad de herramientas de desarrollo y compatibilidad con un gran número de bibliotecas software.

Asimismo, Linux constituye actualmente uno de los sistemas operativos más ampliamente utilizados en aplicaciones *Edge AI* e Internet de las Cosas, proporcionando un entorno robusto y bien documentado para el desarrollo de soluciones inteligentes.

El sistema operativo ejecutado sobre la plataforma fue responsable de gestionar todos los recursos hardware disponibles, incluyendo la memoria, los dispositivos de entrada y salida, las interfaces de comunicación y el acelerador hardware DRP-AI.

Además, Linux proporcionó el entorno necesario para la ejecución de la aplicación principal desarrollada en lenguaje C++, permitiendo el acceso a funcionalidades

relacionadas con el procesamiento de imágenes, la gestión de archivos y las comunicaciones de red.

Durante el desarrollo experimental, la interacción con el sistema operativo se realizó principalmente mediante la conexión de un monitor HDMI, un teclado USB y un ratón USB directamente a la Plataforma. Esta configuración permitió ejecutar aplicaciones, supervisar el funcionamiento del sistema y realizar tareas de depuración de manera adecuada.

6.2.4.2 SDK y Herramientas de Desarrollo de *Renesas*

Para el desarrollo de la aplicación de detección se empleó el entorno software proporcionado por *Renesas Electronics* para la plataforma RZ/V2L.

El fabricante proporciona un conjunto de herramientas y bibliotecas destinadas a simplificar el desarrollo de aplicaciones basadas en inteligencia artificial sobre plataformas *Edge AI*. Estas herramientas incluyen ejemplos de referencia, documentación técnica y componentes software específicos para la utilización del acelerador DRP-AI.

La utilización del ecosistema software oficial proporcionado por *Renesas Electronics* facilitó la integración de modelos de inteligencia artificial dentro de la Plataforma.

Durante las primeras etapas del Proyecto, el SDK fue instalado sobre un ordenador ejecutando Ubuntu Linux. Posteriormente, las aplicaciones desarrolladas, junto con los diferentes componentes software necesarios para su ejecución, fueron transferidos a la tarjeta microSD utilizada por la Plataforma.

Esta metodología permitió separar claramente las tareas de desarrollo realizadas sobre el ordenador anfitrión de la ejecución final sobre la Plataforma.

Asimismo, la utilización del SDK oficial garantiza la compatibilidad entre el software desarrollado y el hardware específico de la Plataforma, reduciendo la probabilidad de problemas asociados a la integración.

6.2.4.3 Aplicación de Detección Basada en C++

La lógica principal del sistema fue implementada utilizando el lenguaje de programación C++.

La elección de este lenguaje se fundamenta principalmente en sus elevadas prestaciones y en su amplia utilización dentro del ámbito de los sistemas embebidos y la visión artificial.

La aplicación desarrollada se basa en una aplicación de referencia proporcionada por *Renesas Electronics*, sobre la cual se realizaron diversas modificaciones orientadas a satisfacer los requisitos específicos del proyecto.

Entre las principales funcionalidades implementadas dentro de la aplicación pueden destacarse las siguientes:

- Captura y procesamiento de imágenes.
- Ejecución del modelo de inteligencia artificial.
- Filtrado de las detecciones correspondientes a bicicletas.
- Cálculo del estado de ocupación del aparcamiento.
- Generación de mensajes JSON.
- Envío automático de información al servidor local mediante HTTP.

La utilización de C++ permitió implementar estas funcionalidades manteniendo un elevado rendimiento y aprovechando eficientemente los recursos hardware disponibles en la plataforma.

Asimismo, el lenguaje proporciona acceso directo a bibliotecas de bajo nivel y facilita la integración con las herramientas suministradas por el fabricante.

6.2.4.4 Aplicación Servidor Desarrollada en Python

La recepción y procesamiento de la información generada por la Plataforma se implementó utilizando el lenguaje de programación Python.

Python fue seleccionado debido a su sencillez, elevada productividad y excelente soporte para el desarrollo de aplicaciones web y sistemas distribuidos.

Además, Python constituye actualmente uno de los lenguajes más utilizados dentro del ámbito de la inteligencia artificial y el Internet de las Cosas, lo que facilita enormemente la integración entre diferentes componentes software.

La aplicación servidor permanece continuamente a la espera de nuevas peticiones procedentes de la Plataforma.

Cada vez que se recibe una nueva actualización, la aplicación procesa el mensaje recibido, extrae la información correspondiente al estado del aparcamiento y actualiza los datos almacenados internamente.

Posteriormente, esta información es utilizada para generar dinámicamente la interfaz web mostrada al usuario.

La utilización de Python permitió desarrollar esta funcionalidad de forma rápida y flexible, facilitando además las tareas de depuración realizadas durante la fase experimental.

6.2.4.4.1 *Framework* Flask

Para la implementación del servidor web se empleó el *framework* Flask.

Flask es un *framework* minimalista desarrollado en Python orientado a la creación de aplicaciones web ligeras y servicios basados en HTTP.

La principal ventaja de Flask reside en su simplicidad. A diferencia de otros frameworks más complejos, Flask proporciona únicamente los componentes esenciales necesarios para desarrollar aplicaciones web, permitiendo al desarrollador mantener un elevado grado de control sobre la arquitectura de la aplicación.

En el contexto del presente proyecto, Flask fue utilizado para:

- Definir las rutas HTTP del servidor.
- Recibir las peticiones enviadas por la Plataforma.
- Procesar los mensajes JSON recibidos.
- Generar dinámicamente la interfaz web.
- Proporcionar acceso a la información desde cualquier navegador conectado a la red local.

La utilización de Flask simplificó considerablemente el desarrollo del servidor y permitió obtener una solución funcional con un reducido número de líneas de código.

6.2.4.4.2 Tecnologías Web Empleadas

La interfaz de usuario desarrollada se implementó utilizando tecnologías web convencionales.

Estas tecnologías permiten generar interfaces accesibles desde cualquier navegador moderno, independientemente del sistema operativo empleado por el usuario.

La principal ventaja de esta aproximación reside en que elimina la necesidad de instalar aplicaciones específicas sobre los dispositivos cliente.

De esta manera, cualquier usuario conectado a la misma red local puede acceder a la información proporcionada por el sistema utilizando únicamente un navegador web convencional.

Además, la utilización de tecnologías web facilita futuras ampliaciones orientadas a la incorporación de nuevas funcionalidades, tales como almacenamiento histórico, representación gráfica de estadísticas o integración con dispositivos móviles.

6.2.4.4.3 Herramientas de Desarrollo y Validación

Durante el desarrollo experimental se emplearon diversas herramientas auxiliares destinadas a facilitar las tareas de implementación, depuración y validación del sistema.

Entre las herramientas utilizadas pueden destacarse:

- Ubuntu Linux como entorno principal de desarrollo.

- Visual Studio Code como editor de código.
- PuTTY para la realización de comprobaciones iniciales sobre la Plataforma.
- Herramientas estándar de Linux para la monitorización de la red.
- Comandos de terminal para la validación de las comunicaciones HTTP.

La utilización conjunta de estas herramientas permitió acelerar el proceso de desarrollo y simplificó significativamente las tareas de integración de todos los componentes software.

En conjunto, la infraestructura software empleada proporcionó un entorno robusto, flexible y perfectamente adaptado a las necesidades del proyecto, permitiendo desarrollar e integrar satisfactoriamente todos los componentes de la solución propuesta.

6.2.4.5 Justificación de las Tecnologías Software Seleccionadas

La selección de las tecnologías software utilizadas durante el desarrollo no se realizó de manera arbitraria, sino que respondió a la necesidad de satisfacer simultáneamente diversos requisitos funcionales y no funcionales del sistema.

Entre los principales criterios considerados durante el proceso de selección pueden destacarse los siguientes:

- Compatibilidad con la plataforma hardware empleada.
- Facilidad de integración entre diferentes componentes software.
- Disponibilidad de documentación técnica y soporte por parte del fabricante.
- Sencillez de desarrollo y mantenimiento.
- Posibilidad de reutilización en futuras versiones del sistema.
- Escalabilidad y flexibilidad de la arquitectura resultante.

Uno de los principales objetivos del proyecto consistía en desarrollar una solución funcional dentro del marco temporal disponible para la realización del Proyecto. Por este motivo, se priorizó la utilización de herramientas ampliamente documentadas y con un elevado grado de madurez tecnológica.

La utilización de tecnologías maduras presenta numerosas ventajas. En primer lugar, reduce significativamente el tiempo necesario para la implementación de nuevas funcionalidades. Además, facilita las tareas de depuración y disminuye el riesgo asociado a la aparición de problemas de compatibilidad entre componentes.

Otro aspecto especialmente relevante fue la disponibilidad de ejemplos de referencia y documentación oficial proporcionada por el fabricante de la plataforma. Este factor

resultó determinante durante las primeras fases del desarrollo, puesto que permitió reducir la complejidad asociada a la puesta en marcha del sistema.

Asimismo, la utilización de herramientas ampliamente adoptadas por la comunidad científica y tecnológica facilita la reproducibilidad del trabajo realizado y simplifica futuras tareas de mantenimiento o ampliación de la solución desarrollada.

6.2.4.6 Interacción entre los Diferentes Componentes Software

Desde un punto de vista arquitectónico, el sistema desarrollado puede interpretarse como un conjunto de componentes software independientes que cooperan entre sí para proporcionar la funcionalidad global requerida.

Cada uno de estos componentes desempeña funciones específicas y mantiene relaciones de intercambio de información con el resto de elementos del sistema.

La aplicación principal ejecutada sobre la Plataforma constituye el núcleo central de la solución. Este componente es responsable de adquirir las imágenes procedentes de la cámara, ejecutar el modelo de inteligencia artificial y generar la información relativa al estado del aparcamiento.

Una vez calculado el número de plazas libres y ocupadas, la aplicación genera un mensaje estructurado en formato JSON y lo transmite al servidor local utilizando el protocolo HTTP.

El servidor implementado mediante Flask recibe la información enviada por la plataforma embebida y actualiza internamente las variables correspondientes al estado del aparcamiento.

Finalmente, la interfaz web consulta dichos datos y los presenta al usuario final mediante una representación gráfica fácilmente interpretable.

La separación de funcionalidades entre componentes software independientes proporciona importantes ventajas desde el punto de vista del diseño del sistema.

En primer lugar, favorece la modularidad de la arquitectura, permitiendo modificar o sustituir determinados componentes sin afectar significativamente al resto del sistema.

Por ejemplo, sería posible reemplazar el servidor Flask por otro *framework* web más avanzado sin necesidad de modificar el código encargado de realizar la detección de bicicletas.

De igual forma, el modelo de inteligencia artificial empleado podría sustituirse por futuras versiones más sofisticadas manteniendo inalterada la lógica de comunicaciones y la interfaz web.

Este elevado grado de desacoplamiento constituye uno de los aspectos más positivos de la arquitectura software desarrollada.

6.2.4.7 Gestión del Flujo de Información

Otro aspecto relevante de la configuración software implementada es la gestión del flujo de información entre los diferentes componentes.

Durante el funcionamiento normal del sistema, el flujo de datos sigue una secuencia claramente definida.

Inicialmente, la cámara captura una imagen del aparcamiento. Dicha imagen es procesada localmente sobre la Plataforma mediante el modelo de inteligencia artificial acelerado por DRP-AI.

Posteriormente, el sistema extrae exclusivamente la información relevante para la aplicación, concretamente el número de bicicletas detectadas y el número de plazas disponibles.

En lugar de transmitir imágenes completas a través de la red, únicamente se envían los resultados finales obtenidos tras el procesamiento.

Esta decisión presenta importantes ventajas.

En primer lugar, reduce significativamente el tráfico generado dentro de la red local. La transmisión continua de imágenes o secuencias de vídeo requeriría un ancho de banda considerablemente superior al necesario para transmitir pequeños mensajes JSON.

Además, la reducción del volumen de información intercambiada contribuye a disminuir la latencia global del sistema y mejora la escalabilidad de la solución.

Otro aspecto especialmente relevante es la privacidad. Dado que las imágenes capturadas por la cámara permanecen siempre dentro de la plataforma embebida y no son enviadas a dispositivos externos, se minimiza la exposición de información visual potencialmente sensible.

Esta aproximación resulta coherente con la filosofía *Edge AI* descrita en capítulos anteriores, donde el procesamiento local constituye uno de los principios fundamentales del diseño.

6.2.4.8 Mantenimiento y Evolución del Software

La arquitectura software desarrollada fue diseñada teniendo en consideración posibles ampliaciones futuras del sistema.

Durante la implementación se procuró mantener una estructura modular y fácilmente extensible, evitando dependencias innecesarias entre componentes.

Esta filosofía de diseño facilita enormemente las tareas de mantenimiento, puesto que cualquier modificación realizada sobre un componente concreto presenta un impacto reducido sobre el resto del sistema.

Por ejemplo, futuras líneas de desarrollo podrían incorporar funcionalidades adicionales tales como:

- Almacenamiento histórico de datos de ocupación.
- Generación automática de estadísticas de utilización.
- Integración con bases de datos relacionales.
- Desarrollo de aplicaciones móviles específicas.
- Incorporación de mecanismos de autenticación y control de acceso.
- Integración con plataformas cloud.
- Monitorización simultánea de múltiples aparcamientos.

Gracias a la estructura modular implementada, la incorporación de estas funcionalidades podría realizarse sin necesidad de rediseñar completamente la solución desarrollada.

Este aspecto resulta especialmente importante en aplicaciones IoT, donde la capacidad de adaptación y evolución del sistema constituye un requisito fundamental.

6.2.4.9 Robustez de la Configuración Software

Durante las pruebas experimentales realizadas, la configuración software implementada demostró un comportamiento estable y satisfactorio.

Todos los componentes fueron capaces de operar de forma continuada durante periodos prolongados sin producir fallos críticos que comprometieran el funcionamiento global del sistema.

Asimismo, la utilización de tecnologías ampliamente contrastadas, como Linux, Python, Flask o HTTP, contribuyó significativamente a incrementar la robustez y fiabilidad de la solución.

La experiencia obtenida durante el desarrollo pone de manifiesto la importancia de seleccionar adecuadamente las herramientas software utilizadas, especialmente en proyectos multidisciplinares donde deben coexistir componentes pertenecientes a ámbitos tecnológicos muy diferentes.

En conjunto, la configuración software implementada proporcionó una base sólida y suficientemente flexible para el desarrollo del sistema propuesto, permitiendo alcanzar satisfactoriamente los objetivos establecidos inicialmente para el proyecto.

6.2.5 Posicionamiento y Configuración de la Cámara

Uno de los aspectos más importantes dentro del diseño experimental del sistema desarrollado corresponde a la ubicación y configuración de la cámara encargada de capturar las imágenes del aparcamiento. La calidad de la información visual

proporcionada por este dispositivo influye de forma directa sobre el rendimiento global del sistema de detección, condicionando tanto la precisión de las detecciones obtenidas como la robustez de la solución frente a diferentes situaciones de funcionamiento.

En sistemas basados en visión artificial, la posición de la cámara constituye un factor crítico, ya que determina el campo de visión disponible, la perspectiva desde la que se observan los objetos presentes en la escena y el grado de visibilidad de las diferentes zonas monitorizadas.

Por este motivo, antes de iniciar la fase experimental, fue necesario analizar cuidadosamente la ubicación más adecuada para el dispositivo de captura con objeto de maximizar la cantidad de información disponible para el sistema de inteligencia artificial.

6.2.5.1 Criterios para la Selección de la Posición de la Cámara

La selección de la posición final de la cámara se realizó teniendo en consideración diversos criterios técnicos y funcionales.

En primer lugar, se buscó garantizar que todas las plazas del aparcamiento permanecieran visibles simultáneamente dentro del campo de visión del dispositivo. Este requisito resulta imprescindible para permitir la monitorización global del aparcamiento utilizando una única cámara.

Adicionalmente, se procuró minimizar la aparición de oclusiones entre bicicletas. Las oclusiones constituyen uno de los principales problemas en aplicaciones de visión artificial, puesto que pueden ocultar parcial o totalmente determinados objetos presentes en la escena, dificultando considerablemente su detección.

Otro criterio importante considerado durante el posicionamiento de la cámara fue la necesidad de obtener una perspectiva suficientemente amplia para capturar la totalidad del aparcamiento, evitando al mismo tiempo perspectivas excesivamente inclinadas que pudieran introducir deformaciones significativas en la representación visual de las bicicletas.

Asimismo, se intentó reducir la presencia de elementos externos irrelevantes dentro del campo de visión, con objeto de disminuir la probabilidad de generar falsas detecciones y optimizar el rendimiento general del sistema.

Como resultado de este proceso, la cámara fue instalada en una posición fija que permitía visualizar simultáneamente la totalidad de las plazas monitorizadas.

6.2.5.2 Configuración Fija del Sistema de Captura

Durante toda la fase experimental, la cámara permaneció instalada en una posición fija y su orientación no fue modificada.

La utilización de una configuración estática presenta importantes ventajas desde el punto de vista experimental y operacional.

En primer lugar, permite garantizar la reproducibilidad de las pruebas realizadas. Mantener constante la posición de la cámara asegura que todas las configuraciones experimentales son evaluadas bajo condiciones equivalentes, facilitando la comparación objetiva entre diferentes escenarios.

Además, una cámara fija simplifica significativamente el procesamiento posterior de las imágenes. En sistemas con cámaras móviles resulta necesario incorporar algoritmos adicionales destinados a compensar movimientos, recalibrar la escena o corregir variaciones geométricas.

La utilización de una cámara estática elimina completamente esta necesidad, permitiendo concentrar todos los recursos computacionales disponibles en las tareas de detección y análisis.

Desde el punto de vista de una posible implementación real, la utilización de cámaras fijas constituye además una solución habitual en sistemas de videovigilancia urbana y monitorización inteligente de infraestructuras.

6.2.5.3 Campo de Visión y Cobertura del Aparcamiento

Otro aspecto relevante dentro de la configuración del sistema de captura corresponde al campo de visión proporcionado por la cámara.

El campo de visión determina la región del entorno que puede ser observada simultáneamente por el sistema y condiciona directamente el número máximo de plazas que pueden monitorizarse utilizando un único dispositivo.

Durante el diseño experimental se verificó que el campo de visión disponible permitía cubrir la totalidad de las seis plazas presentes en el aparcamiento.

Esta cobertura completa resulta especialmente importante, puesto que la ausencia de determinadas zonas dentro de la imagen podría provocar errores en el cálculo del estado de ocupación.

Además, la monitorización simultánea de todas las plazas permite simplificar considerablemente la arquitectura global del sistema, evitando la necesidad de incorporar múltiples cámaras y mecanismos adicionales de fusión de información.

No obstante, debe señalarse que el tamaño máximo del aparcamiento que puede monitorizarse utilizando una única cámara depende directamente de las características ópticas del dispositivo y de la geometría específica del escenario.

En futuros desarrollos orientados a aparcamientos de mayores dimensiones podría resultar necesario incorporar múltiples cámaras distribuidas convenientemente.

6.2.5.4 Influencia de la Perspectiva sobre la Detección

La perspectiva desde la que se capturan las imágenes constituye otro factor que puede influir significativamente sobre el comportamiento de los algoritmos de detección.

Las redes neuronales utilizadas en aplicaciones de visión artificial suelen mostrar un comportamiento más robusto cuando los objetos observados presentan características visuales similares a las empleadas durante el proceso de entrenamiento.

Por este motivo, perspectivas excesivamente inclinadas o posiciones poco habituales podrían afectar negativamente a la precisión del sistema.

Durante las pruebas realizadas se observó que la posición seleccionada proporcionaba una representación suficientemente clara de las bicicletas presentes en la escena, permitiendo obtener resultados satisfactorios en la mayoría de las situaciones analizadas.

Asimismo, el ángulo de observación empleado permitió minimizar el solapamiento entre objetos y facilitó la identificación individual de las bicicletas presentes en el aparcamiento.

Este aspecto resulta especialmente importante en escenarios con varias bicicletas simultáneamente presentes, donde la separación visual entre objetos puede condicionar significativamente la calidad de las detecciones obtenidas.

6.2.5.6 Consideraciones sobre la Calibración del Sistema

Dado que la cámara permaneció fija durante toda la fase experimental y que el sistema se diseñó específicamente para monitorizar un único escenario, no fue necesario implementar procedimientos avanzados de calibración geométrica.

Sin embargo, en aplicaciones de mayor complejidad podría resultar conveniente incorporar técnicas adicionales destinadas a corregir posibles distorsiones ópticas, compensar cambios de perspectiva o definir regiones específicas de interés dentro de la escena.

Asimismo, futuras versiones del sistema podrían incorporar mecanismos automáticos de calibración capaces de adaptar dinámicamente el funcionamiento del algoritmo a diferentes configuraciones espaciales.

Este tipo de funcionalidades resultaría especialmente útil en escenarios donde la posición de la cámara pudiera variar con el tiempo o cuando fuese necesario desplegar la solución sobre múltiples emplazamientos diferentes.

6.2.6 Metodología Experimental

Con el objetivo de evaluar rigurosamente el comportamiento del sistema desarrollado, se diseñó una metodología experimental específica orientada a validar tanto el funcionamiento individual de los diferentes subsistemas implementados como el comportamiento global de la solución cuando todos sus componentes operan conjuntamente.

La definición de una metodología experimental adecuada constituye un elemento esencial dentro de cualquier proceso de investigación o desarrollo tecnológico. Una metodología

correctamente definida permite garantizar la validez de los resultados obtenidos, facilitar la reproducibilidad de los experimentos y establecer criterios objetivos para determinar el grado de cumplimiento de los objetivos inicialmente planteados.

Dado el carácter multidisciplinar del sistema desarrollado, que integra elementos pertenecientes a ámbitos tan diversos como la visión artificial, la inteligencia artificial, las comunicaciones de red y el desarrollo web, fue necesario diseñar un procedimiento experimental capaz de evaluar simultáneamente el correcto funcionamiento de todos estos componentes.

Por este motivo, la estrategia de validación adoptada se estructuró en diferentes fases sucesivas, permitiendo verificar progresivamente cada uno de los módulos implementados antes de proceder a la evaluación del sistema completo.

Esta aproximación presenta importantes ventajas desde el punto de vista experimental. En primer lugar, facilita la identificación de posibles errores, ya que permite aislar cada componente y analizarlo de forma independiente. Asimismo, simplifica considerablemente las tareas de depuración y reduce la complejidad asociada al proceso de integración.

6.2.6.1 Estrategia de Validación Incremental

La validación del sistema se realizó siguiendo una metodología incremental.

En lugar de evaluar directamente la solución completa desde las primeras etapas del desarrollo, se optó por verificar inicialmente el correcto funcionamiento de cada uno de los subsistemas por separado.

Esta estrategia permitió garantizar que cada componente cumplía adecuadamente la función para la que había sido diseñado antes de proceder a su integración con el resto del sistema.

El proceso de validación siguió la secuencia mostrada a continuación:

1. Verificación del correcto arranque y funcionamiento de la Plataforma.
2. Comprobación de la ejecución del sistema operativo Linux sobre la Plataforma.
3. Validación de la aplicación de referencia proporcionada por *Renesas Electronics*.
4. Verificación del correcto funcionamiento del modelo de inteligencia artificial.
5. Validación del mecanismo de detección y filtrado de bicicletas.
6. Comprobación del algoritmo encargado de calcular el estado de ocupación.
7. Verificación del sistema de comunicaciones basado en HTTP.
8. Validación del servidor local desarrollado mediante Flask.

9. Comprobación del funcionamiento de la interfaz web.
10. Integración y evaluación del sistema completo.

La utilización de este procedimiento permitió detectar y corregir problemas durante las primeras fases del desarrollo, evitando que dichos errores se propagasen a etapas posteriores del proyecto.

Además, esta estrategia facilitó enormemente el proceso de integración final, puesto que todos los componentes habían sido previamente validados de forma independiente.

6.2.6.2 Definición de los Escenarios Experimentales

Para evaluar adecuadamente el comportamiento del sistema se definieron diferentes escenarios experimentales caracterizados por distintos niveles de ocupación del aparcamiento.

La utilización de múltiples escenarios resulta especialmente importante en sistemas basados en visión artificial, ya que permite analizar el comportamiento de la solución frente a diferentes configuraciones espaciales y estudiar la estabilidad de los resultados obtenidos.

Las configuraciones consideradas durante las pruebas incluyeron:

- Escenarios sin bicicletas presentes en la escena.
- Escenarios con una única bicicleta.
- Escenarios con diferentes niveles de ocupación parcial.
- Escenarios con distintas distribuciones espaciales de las bicicletas.

La modificación de la disposición de las bicicletas dentro del aparcamiento permitió estudiar el comportamiento del sistema frente a situaciones variadas y verificar su capacidad para adaptarse a diferentes configuraciones.

Asimismo, se realizaron pruebas colocando las bicicletas en distintas posiciones y orientaciones dentro de las plazas disponibles. De este modo, fue posible evaluar la robustez del sistema frente a pequeñas variaciones en la escena observada.

Debe señalarse que el objetivo principal de estas pruebas no consistía únicamente en comprobar la capacidad del sistema para detectar bicicletas, sino también verificar que el número de plazas libres calculado coincidía con el estado real del aparcamiento.

6.2.6.3 Variables Analizadas

Durante la realización de las pruebas experimentales se registraron diferentes variables con el objetivo de evaluar cuantitativamente el comportamiento del sistema.

Las principales variables consideradas fueron las siguientes:

- Número real de bicicletas presentes en la escena.
- Número de bicicletas detectadas por el sistema.
- Número de plazas libres calculadas.
- Número de plazas ocupadas calculadas.
- Correcta recepción de la información por parte del servidor local.
- Correcta actualización de la interfaz web.
- Existencia de errores de comunicación.
- Estabilidad del sistema durante la ejecución prolongada.

El análisis conjunto de estas variables permitió obtener una visión global del comportamiento de la solución desarrollada.

Además, la comparación entre el número real de bicicletas presentes y el número detectado por el sistema hizo posible evaluar la precisión alcanzada por el algoritmo de detección implementado.

Por otra parte, la verificación de la información mostrada en la interfaz web permitió validar simultáneamente el funcionamiento del subsistema de comunicaciones y del servidor local.

6.2.6.4 Procedimiento de Ejecución de las Pruebas

Para garantizar la uniformidad de los experimentos realizados, todas las pruebas siguieron un procedimiento de ejecución común.

Inicialmente, se configuraba el escenario experimental situando el número deseado de bicicletas dentro del aparcamiento.

Una vez preparada la escena, se iniciaba la aplicación ejecutada sobre la Plataforma.

Durante la ejecución del sistema, la cámara capturaba continuamente imágenes del aparcamiento y el modelo de inteligencia artificial procesaba la información visual disponible.

Posteriormente, la aplicación filtraba las detecciones obtenidas, contabilizaba el número de bicicletas presentes y calculaba el estado de ocupación del aparcamiento.

A continuación, la información generada era transmitida automáticamente al servidor local mediante una petición HTTP.

Finalmente, se verificaba que la información mostrada en la interfaz web coincidiese con el estado real observado en el aparcamiento.

Este procedimiento fue repetido sucesivamente para todas las configuraciones experimentales consideradas.

La repetición sistemática de las pruebas permitió aumentar la fiabilidad de los resultados obtenidos y reducir la influencia de posibles factores aleatorios.

6.2.6.5 Criterios de Validación

Con objeto de evaluar objetivamente el comportamiento del sistema, se definieron una serie de criterios de validación.

Una prueba experimental se consideró satisfactoria cuando se cumplían simultáneamente las siguientes condiciones:

- El número de bicicletas detectadas coincidía con el número real de bicicletas presentes.
- El número de plazas libres calculado era correcto.
- El servidor local recibía correctamente la información enviada por la plataforma.
- La interfaz web mostraba adecuadamente el estado actualizado del aparcamiento.
- No se producían errores durante la ejecución de la aplicación.
- El sistema mantenía un funcionamiento estable durante toda la duración de la prueba.

La utilización de criterios homogéneos para todas las pruebas realizadas permitió comparar objetivamente los diferentes escenarios experimentales analizados.

6.2.6.6 Reproducibilidad Experimental

Con el fin de garantizar la reproducibilidad de los resultados obtenidos, todas las pruebas se realizaron manteniendo constantes las principales condiciones de funcionamiento del sistema.

Durante toda la fase experimental permanecieron invariables los siguientes elementos:

- Posición y orientación de la cámara.
- Plataforma hardware utilizada.
- Modelo de inteligencia artificial empleado.
- Configuración software del sistema.
- Parámetros asociados al proceso de inferencia.
- Arquitectura de comunicaciones implementada.

La conservación de estas condiciones permitió asegurar que las diferencias observadas entre experimentos se debieran exclusivamente a modificaciones en la configuración del escenario y no a variaciones introducidas por factores externos.

La reproducibilidad constituye un requisito fundamental dentro de cualquier actividad experimental, ya que garantiza la consistencia de los resultados obtenidos y facilita futuras comparaciones con desarrollos posteriores.

6.3 Validación del Sistema de Detección

6.3.1 Introducción a la Validación del Sistema de Detección

La validación del sistema de detección constituye uno de los aspectos más relevantes dentro de la evaluación experimental realizada, puesto que la capacidad para identificar correctamente la presencia de bicicletas en el aparcamiento determina directamente el comportamiento global de la solución desarrollada.

El objetivo principal del sistema implementado consiste en determinar automáticamente el estado de ocupación de un conjunto de plazas de aparcamiento a partir del análisis de imágenes capturadas mediante una cámara fija. Como consecuencia, resulta imprescindible evaluar rigurosamente la capacidad del algoritmo de inteligencia artificial para detectar correctamente las bicicletas presentes en la escena y discriminar adecuadamente entre situaciones de ocupación y ausencia de vehículos.

La evaluación del subsistema de detección se llevó a cabo mediante la ejecución de diferentes pruebas experimentales sobre el escenario descrito en apartados anteriores. Durante estas pruebas se analizaron distintos niveles de ocupación del aparcamiento, permitiendo estudiar el comportamiento del sistema bajo diversas condiciones de funcionamiento.

La metodología de validación se diseñó con el objetivo de verificar no solamente la precisión de las detecciones obtenidas, sino también la estabilidad del sistema, su capacidad de generalización y la influencia de diversos factores externos sobre el rendimiento alcanzado.

Para cada uno de los escenarios evaluados se comparó el número real de bicicletas presentes en la escena con el número de detecciones generadas por el sistema.

Asimismo, se verificó que la información obtenida tras el proceso de inferencia fuese correctamente utilizada por el algoritmo encargado de calcular el estado de ocupación del aparcamiento.

La combinación de estos análisis permitió obtener una visión completa del comportamiento del sistema y determinar el grado de cumplimiento de los objetivos inicialmente establecidos.

6.3.2 Evaluación en Escenarios sin Bicicletas

Uno de los primeros escenarios considerados durante la fase experimental correspondió a situaciones en las que el aparcamiento se encontraba completamente vacío.

La evaluación de este tipo de escenarios resulta especialmente importante, ya que permite determinar la capacidad del sistema para evitar falsas detecciones.

En aplicaciones reales de monitorización, la aparición de falsas detecciones puede provocar errores en el cálculo del estado de ocupación y reducir significativamente la utilidad práctica del sistema.

Por este motivo, se realizaron múltiples pruebas capturando imágenes del aparcamiento en ausencia total de bicicletas.

Durante estas pruebas, el sistema procesó las imágenes adquiridas utilizando el modelo de inteligencia artificial implementado y generó las correspondientes predicciones.

Los resultados obtenidos mostraron que el algoritmo fue capaz de identificar correctamente la ausencia de bicicletas en la escena, no produciéndose detecciones erróneas en la mayoría de las situaciones evaluadas.

Este comportamiento resulta especialmente relevante, puesto que demuestra la capacidad del modelo para discriminar adecuadamente entre objetos pertenecientes a la clase de interés y otros elementos presentes en el entorno.

La correcta identificación de este tipo de situaciones constituye un requisito fundamental para garantizar la fiabilidad del sistema durante el funcionamiento real.

6.3.3 Evaluación en Escenarios con Una Bicicleta

Una vez verificado el correcto comportamiento del sistema en ausencia de bicicletas, se procedió a evaluar su funcionamiento en escenarios en los que existía una única bicicleta estacionada dentro del aparcamiento.

Este tipo de escenarios constituye una situación especialmente relevante desde el punto de vista experimental, ya que permite analizar la capacidad básica del sistema para detectar correctamente la presencia de bicicletas.

Durante las pruebas realizadas, la bicicleta fue situada en diferentes posiciones dentro del aparcamiento, permitiendo estudiar el comportamiento del sistema frente a pequeñas variaciones en la localización espacial del objeto.

Para cada una de las configuraciones consideradas, el sistema ejecutó el proceso completo de detección, identificación y cálculo del estado de ocupación.

Los resultados obtenidos mostraron que el modelo de inteligencia artificial fue capaz de detectar correctamente la presencia de la bicicleta en la mayoría de los casos analizados.

Asimismo, el algoritmo implementado para el cálculo de plazas disponibles generó correctamente la información correspondiente al estado del aparcamiento.

Además de la detección propiamente dicha, se verificó que la información generada por la plataforma era transmitida correctamente al servidor local y posteriormente mostrada en la interfaz web desarrollada.

Ilustración 25. Ejemplo representativo correspondiente a una situación en la que existe una única bicicleta presente en la escena

6.3.4 Evaluación en Escenarios con Varias Bicicletas

Con objeto de incrementar la complejidad de las pruebas experimentales, también se realizaron ensayos considerando la presencia simultánea de dos bicicletas dentro del campo de visión de la cámara.

La utilización de este tipo de escenarios permitió analizar el comportamiento del sistema ante situaciones en las que existen múltiples objetos pertenecientes a la misma clase dentro de la escena.

Desde el punto de vista de la visión artificial, la detección simultánea de varios objetos puede resultar más compleja debido a la posible aparición de solapamientos parciales, diferencias de escala o variaciones en la orientación relativa de los objetos.

Durante las pruebas realizadas, las bicicletas fueron colocadas en diferentes posiciones dentro del aparcamiento, permitiendo evaluar la robustez del sistema frente a distintas configuraciones espaciales.

Los resultados obtenidos mostraron que el sistema fue capaz de detectar correctamente ambas bicicletas en las diferentes configuraciones evaluadas, asignando adecuadamente las correspondientes cajas delimitadoras sobre la imagen capturada.

Asimismo, el algoritmo encargado de calcular el número de plazas libres generó resultados coherentes con el estado real observado en el aparcamiento.

La siguiente figura muestra un ejemplo correspondiente a uno de los escenarios evaluados durante esta fase experimental.

Ilustración 26. Resultado obtenido por el sistema en un escenario con varias bicicletas presentes simultáneamente en la escena.

Los resultados obtenidos ponen de manifiesto la capacidad del sistema para gestionar correctamente situaciones caracterizadas por la presencia de múltiples bicicletas dentro del área monitorizada.

6.3.5 Análisis Cualitativo de las Detecciones Obtenidas

Además del análisis cuantitativo basado en el número de bicicletas detectadas, se realizó una evaluación cualitativa de las predicciones generadas por el modelo de inteligencia artificial.

Este análisis tuvo como objetivo estudiar aspectos relacionados con la localización espacial de las detecciones, la precisión de las cajas delimitadoras y la estabilidad general del sistema.

Durante las pruebas realizadas se observó que las regiones detectadas se ajustaban adecuadamente a la posición real de las bicicletas presentes en la escena.

Asimismo, las detecciones obtenidas presentaron niveles de confianza suficientemente elevados, permitiendo discriminar correctamente entre objetos pertenecientes a la clase de interés y otros elementos presentes en el entorno.

Otro aspecto especialmente relevante observado durante las pruebas fue la estabilidad del sistema frente a pequeñas variaciones en la posición y orientación de las bicicletas.

Aunque las bicicletas no fueron colocadas exactamente en la misma posición durante todos los experimentos, el sistema fue capaz de mantener un comportamiento consistente, generando resultados satisfactorios en la mayoría de las situaciones analizadas.

Ilustración 27. Bicicletas en posición delantera y lateral.

Este comportamiento sugiere que el modelo utilizado presenta una adecuada capacidad de generalización dentro del escenario experimental considerado.

6.4 Validación del Subsistema de Comunicaciones

El subsistema de comunicaciones constituye uno de los elementos fundamentales de la arquitectura desarrollada, ya que es el encargado de transferir la información generada por la plataforma *Edge AI* hacia el servidor local responsable de la visualización de resultados.

La correcta operación de este subsistema resulta imprescindible para garantizar el funcionamiento global del sistema, puesto que cualquier fallo en la transmisión de información impediría actualizar adecuadamente el estado del aparcamiento mostrado al usuario.

Por este motivo, se llevó a cabo una fase específica de validación orientada a verificar la fiabilidad, estabilidad y correcto funcionamiento del mecanismo de comunicaciones implementado.

Las pruebas realizadas tuvieron como objetivo comprobar tanto la capacidad de la Plataforma para enviar información a través de la red como la capacidad del servidor local para recibir, procesar y almacenar correctamente los datos transmitidos.

6.4.1 Verificación Inicial de la Conectividad de Red

Antes de integrar el mecanismo de comunicaciones dentro de la aplicación principal, fue necesario verificar la correcta conectividad entre los distintos dispositivos que componen la arquitectura del sistema.

Para ello, se configuró una red local utilizando un router convencional, conectando a dicha red tanto la Plataforma como el ordenador Ubuntu encargado de ejecutar el servidor local.

Una vez establecida la infraestructura de red, se procedió a comprobar que ambos dispositivos pertenecían al mismo segmento de red y disponían de conectividad IP.

Durante esta fase se emplearon diversas herramientas estándar proporcionadas por el sistema operativo Linux para verificar el estado de las interfaces de red y comprobar la asignación correcta de direcciones IP.

La utilización de estas herramientas permitió confirmar la correcta configuración de la infraestructura de comunicaciones antes de proceder a la implementación definitiva del sistema.

```
root@smarc-rzv2l:~# ping -c 4 192.168.1.35
PING 192.168.1.35 (192.168.1.35): 56 data bytes
64 bytes from 192.168.1.35: seq=0 ttl=64 time=4.622 ms
64 bytes from 192.168.1.35: seq=1 ttl=64 time=2.060 ms
64 bytes from 192.168.1.35: seq=2 ttl=64 time=1.799 ms
64 bytes from 192.168.1.35: seq=3 ttl=64 time=2.747 ms

--- 192.168.1.35 ping statistics ---
4 packets transmitted, 4 packets received, 0% packet loss
round-trip min/avg/max = 1.799/2.807/4.622 ms
root@smarc-rzv2l:~#
```

Ilustración 27. Comprobación de la conectividad de red entre la Plataforma y el servidor local

6.4.2 Validación del Servidor Local

Una vez verificada la conectividad básica entre dispositivos, se procedió a validar el correcto funcionamiento del servidor local desarrollado mediante Flask.

Durante esta fase, el servidor fue ejecutado sobre el ordenador Ubuntu permaneciendo continuamente a la espera de nuevas peticiones HTTP procedentes de la plataforma embebida.

Inicialmente, se realizaron pruebas manuales utilizando herramientas de línea de comandos con objeto de comprobar el correcto funcionamiento del servicio antes de integrarlo dentro de la aplicación principal.

Estas pruebas permitieron verificar que el servidor era capaz de recibir correctamente peticiones HTTP y procesar adecuadamente la información recibida.

Asimismo, se comprobó que la aplicación permanecía estable durante periodos prolongados de ejecución, no observándose interrupciones inesperadas del servicio.

La utilización de un servidor basado en Flask simplificó considerablemente el proceso de validación, proporcionando mecanismos sencillos para monitorizar las peticiones recibidas y depurar posibles errores durante el desarrollo.

6.4.3 Pruebas de Envío de Información Mediante HTTP

Una vez validado el funcionamiento individual del servidor, se integró el mecanismo de comunicaciones dentro de la aplicación ejecutada sobre la Plataforma.

El sistema fue diseñado para generar automáticamente peticiones HTTP cada vez que se actualiza el estado del aparcamiento.

Concretamente, la plataforma construye un mensaje en formato JSON que contiene la información relativa al número total de plazas, el número de plazas ocupadas y el número de plazas libres.

Posteriormente, este mensaje es enviado al servidor local utilizando el protocolo HTTP.

Durante las pruebas realizadas se comprobó que las peticiones generadas por la plataforma eran recibidas correctamente por el servidor en la totalidad de los escenarios evaluados.

Además, se verificó que los datos transmitidos coincidían exactamente con la información calculada previamente por el sistema de detección.

```
root@smarc-rzv2l:~# curl http://192.168.1.35:5000

    <html>
    <head>
        <title>Parking Bicicletas</title>
    </head>
    <body>
        <h1>Sistema de detección de plazas</h1>

        <p>Plazas totales: 6</p>
        <p>Plazas ocupadas: 0</p>
        <p>Plazas libres: 6</p>
        <p>Última actualización: Sin datos</p>

    </body>
    </html>
    root@smarc-rzv2l:~#
```

Ilustración 28. Ejemplo de recepción correcta de información enviada desde la Plataforma hacia el servidor local.

6.4.4 Evaluación de la Estabilidad del Sistema de Comunicaciones

Además de verificar la correcta transmisión de información, también se evaluó la estabilidad del subsistema de comunicaciones durante periodos prolongados de funcionamiento.

Para ello, el sistema fue mantenido en ejecución continua mientras se realizaban diferentes modificaciones sobre el estado de ocupación del aparcamiento.

Durante estas pruebas se observó que la plataforma era capaz de enviar sucesivas actualizaciones sin producir errores de comunicación ni pérdidas apreciables de información.

Asimismo, el servidor local permaneció operativo durante toda la duración de las pruebas, procesando correctamente todas las peticiones recibidas.

Este comportamiento pone de manifiesto la robustez de la arquitectura de comunicaciones implementada y demuestra la adecuación del protocolo seleccionado para el tipo de aplicación considerada.

Otro aspecto especialmente relevante observado durante la fase experimental fue la baja latencia introducida por el sistema de comunicaciones.

Dado que la totalidad de la infraestructura opera dentro de una red local, el tiempo necesario para transferir la información desde la plataforma hasta el servidor resultó prácticamente despreciable desde el punto de vista del usuario.

Como consecuencia, la información mostrada en la interfaz web se actualiza prácticamente en tiempo real.

6.4.5 Resultados

Los resultados obtenidos durante la fase de validación permiten concluir que el subsistema de comunicaciones desarrollado satisface adecuadamente los requisitos establecidos para la aplicación considerada.

La utilización de protocolos estándar basados en HTTP ha permitido implementar una solución sencilla, robusta y fácilmente extensible.

Además, la arquitectura cliente-servidor desarrollada proporciona un elevado grado de modularidad, facilitando futuras ampliaciones orientadas a la integración con servicios externos o infraestructuras cloud.

Otro aspecto positivo observado durante las pruebas es la estabilidad del sistema, no detectándose pérdidas de información ni errores significativos durante la transmisión de datos.

Por otra parte, el reducido volumen de información intercambiado contribuye a minimizar el tráfico generado dentro de la red y simplifica considerablemente la gestión de las comunicaciones.

En conjunto, las pruebas realizadas demuestran la viabilidad del mecanismo de comunicaciones implementado y validan su utilización como elemento de intercambio de información dentro de la arquitectura global del sistema desarrollado.

6.5 Validación de la Interfaz Web

Una vez validado el correcto funcionamiento del subsistema de comunicaciones, se procedió a evaluar la interfaz web desarrollada para la visualización del estado de ocupación del aparcamiento.

La interfaz web constituye el elemento encargado de presentar al usuario final la información generada por el sistema de detección. Como consecuencia, su correcto

funcionamiento resulta esencial para garantizar la utilidad práctica de la solución desarrollada.

El principal objetivo de esta fase experimental consistió en verificar que la información mostrada por la aplicación web coincidiese en todo momento con el estado real del aparcamiento y con los resultados generados por la Plataforma.

Adicionalmente, se evaluó la capacidad de la interfaz para actualizar automáticamente la información recibida, así como la estabilidad general del sistema durante periodos prolongados de funcionamiento.

6.5.1 Verificación de la Recepción de Datos

La primera etapa de la validación consistió en comprobar que la interfaz web era capaz de mostrar correctamente la información recibida desde el servidor local.

Para ello, se realizaron diferentes pruebas modificando progresivamente el número de bicicletas presentes en el aparcamiento y observando la información mostrada en la página web.

Durante estas pruebas se verificó que las variables correspondientes al número total de plazas, el número de plazas ocupadas y el número de plazas libres eran actualizadas correctamente tras cada nueva detección realizada por la Plataforma.

Los resultados obtenidos permitieron comprobar que la información visualizada por la interfaz coincidía en todo momento con los datos calculados previamente por el sistema de detección.

Este comportamiento confirma el correcto funcionamiento de la cadena completa de procesamiento de información, desde la captura de imágenes hasta la presentación final de resultados al usuario.

6.5.2 Actualización del Estado del Aparcamiento

Uno de los requisitos fundamentales establecidos durante el diseño del sistema consistía en garantizar que la información mostrada al usuario reflejase adecuadamente el estado real del aparcamiento.

Para validar este aspecto, se realizaron diferentes modificaciones en la ocupación del aparcamiento mientras el sistema permanecía en funcionamiento.

Concretamente, se añadieron y retiraron bicicletas del escenario experimental con objeto de provocar cambios en el estado de ocupación.

Durante estas pruebas se observó que la interfaz web actualizaba correctamente la información mostrada tras la recepción de nuevas peticiones procedentes de la Plataforma.

Este comportamiento permitió comprobar que la arquitectura implementada es capaz de mantener sincronizados todos los elementos del sistema.

Además, la actualización automática de la información elimina la necesidad de intervención manual por parte del usuario, mejorando significativamente la experiencia de utilización del sistema.

6.5.3 Evaluación de la Usabilidad de la Interfaz

Además de verificar el correcto funcionamiento técnico de la aplicación, también se realizó una evaluación cualitativa de la interfaz desarrollada desde el punto de vista de la usabilidad.

Durante el diseño de la interfaz se priorizó la simplicidad y la facilidad de interpretación de la información mostrada.

Esta decisión responde a la necesidad de proporcionar al usuario una representación clara e intuitiva del estado del aparcamiento, evitando la incorporación de elementos gráficos innecesarios que pudieran dificultar la interpretación de los resultados.

La interfaz presenta únicamente la información esencial para el usuario, concretamente:

- Número total de plazas monitorizadas.
- Número de plazas ocupadas.
- Número de plazas libres.

La simplicidad de este diseño facilita la interpretación inmediata de la información y reduce significativamente la curva de aprendizaje asociada a la utilización del sistema.

Asimismo, al tratarse de una aplicación web accesible mediante un navegador convencional, la solución puede utilizarse desde diferentes dispositivos sin necesidad de instalar software adicional.

Este aspecto incrementa notablemente la flexibilidad de la arquitectura desarrollada y favorece su posible despliegue futuro en entornos reales.

6.5.4 Estabilidad de la Aplicación Web

Otro de los aspectos analizados durante la fase experimental fue la estabilidad de la aplicación web durante periodos prolongados de funcionamiento.

Para ello, el sistema permaneció operativo mientras se realizaban sucesivas actualizaciones del estado del aparcamiento.

Durante las pruebas realizadas no se observaron interrupciones inesperadas del servicio ni errores significativos durante la actualización de la información.

Asimismo, el servidor local fue capaz de procesar correctamente todas las peticiones recibidas desde la Plataforma sin producir pérdidas de información.

La estabilidad observada durante las pruebas confirma la adecuación de las tecnologías software seleccionadas para la implementación de la interfaz.

En particular, la utilización del *framework* Flask demostró ser suficiente para satisfacer las necesidades funcionales de la aplicación desarrollada.

6.5.5 Resultados

Los resultados obtenidos durante la fase de validación permiten concluir que la interfaz web desarrollada cumple satisfactoriamente los requisitos establecidos durante el diseño del sistema.

La aplicación fue capaz de mostrar correctamente la información procedente de la plataforma embebida y mantener actualizados los datos mostrados al usuario.

Asimismo, la arquitectura implementada demostró un comportamiento estable y robusto durante todas las pruebas experimentales realizadas.

Otro aspecto especialmente positivo observado durante la validación fue la sencillez de utilización de la interfaz, permitiendo acceder a la información del sistema mediante cualquier navegador web conectado a la red local.

En conjunto, las pruebas realizadas demuestran la viabilidad de la solución propuesta y validan la utilización de tecnologías web ligeras para la visualización remota de información generada por sistemas *Edge AI.*

6.6 Evaluación Global del Sistema

Una vez finalizadas las diferentes fases de validación experimental descritas en los apartados anteriores, resulta posible realizar una evaluación global del sistema desarrollado, analizando conjuntamente el comportamiento de todos los subsistemas que componen la solución propuesta.

La evaluación integral del sistema constituye un aspecto especialmente relevante, ya que permite determinar hasta qué punto la arquitectura implementada satisface los objetivos planteados inicialmente y valorar la viabilidad de la solución desde un punto de vista técnico.

Los resultados obtenidos durante las pruebas experimentales permiten afirmar que el sistema desarrollado es capaz de monitorizar automáticamente el estado de ocupación de un aparcamiento de bicicletas utilizando técnicas de visión artificial e inteligencia artificial ejecutadas sobre una plataforma *Edge AI.*

La integración satisfactoria de todos los componentes hardware y software ha permitido obtener una solución completamente funcional capaz de capturar imágenes, detectar

bicicletas, calcular el número de plazas disponibles, transmitir la información a través de una red local y presentar los resultados al usuario mediante una interfaz web.

Desde un punto de vista funcional, el sistema ha demostrado un comportamiento estable durante la totalidad de las pruebas realizadas, no observándose fallos críticos que comprometieran el funcionamiento global de la solución.

6.6.1 Cumplimiento de los Objetivos Iniciales

Uno de los principales objetivos establecidos al comienzo del proyecto consistía en estudiar la viabilidad de utilizar plataformas *Edge AI* para desarrollar sistemas inteligentes orientados a la monitorización de infraestructuras urbanas.

Los resultados obtenidos permiten concluir que este objetivo ha sido alcanzado satisfactoriamente.

La Plataforma ha demostrado disponer de capacidad suficiente para ejecutar algoritmos de inteligencia artificial en tiempo real, permitiendo realizar el procesamiento completo de las imágenes directamente sobre el dispositivo embebido.

Este aspecto resulta especialmente relevante, ya que confirma la idoneidad de este tipo de plataformas para aplicaciones de monitorización inteligente donde la baja latencia y el procesamiento local constituyen requisitos fundamentales.

Otro objetivo importante consistía en desarrollar un sistema capaz de comunicar automáticamente la información generada por la plataforma de procesamiento a dispositivos externos.

La implementación del subsistema de comunicaciones basado en HTTP y JSON permitió alcanzar satisfactoriamente este objetivo, proporcionando un mecanismo sencillo y robusto para el intercambio de información entre dispositivos.

Finalmente, el desarrollo de una interfaz web accesible desde la red local permitió completar la arquitectura propuesta y ofrecer al usuario una herramienta sencilla para consultar el estado actualizado del aparcamiento.

En conjunto, puede afirmarse que los objetivos técnicos inicialmente planteados han sido alcanzados satisfactoriamente.

6.6.2 Principales Fortalezas de la Solución Desarrollada

Durante la fase experimental se identificaron diversos aspectos positivos asociados a la arquitectura desarrollada.

Una de las principales fortalezas del sistema reside en la utilización de técnicas de *Edge AI* para realizar el procesamiento directamente sobre la plataforma embebida.

Este enfoque presenta numerosas ventajas frente a arquitecturas tradicionales basadas en procesamiento remoto.

En primer lugar, permite reducir significativamente la latencia del sistema, ya que las imágenes son procesadas localmente sin necesidad de ser enviadas a servidores externos.

Además, el procesamiento local disminuye considerablemente el tráfico generado dentro de la red, puesto que únicamente se transmiten los resultados obtenidos y no las imágenes completas capturadas por la cámara.

Otro aspecto especialmente relevante es la mejora de la privacidad.

Dado que las imágenes permanecen en todo momento dentro de la plataforma embebida, se minimiza la exposición de información visual potencialmente sensible.

Desde el punto de vista arquitectónico, otra fortaleza importante corresponde al carácter modular de la solución desarrollada.

La arquitectura implementada separa claramente las funciones de captura y procesamiento, comunicaciones y visualización, permitiendo modificar o sustituir cada uno de estos componentes de forma independiente.

Esta modularidad facilita considerablemente futuras ampliaciones y simplifica las tareas de mantenimiento y depuración.

Asimismo, la utilización de protocolos de comunicación estándar, tales como HTTP y JSON, favorece la interoperabilidad del sistema con otras aplicaciones y plataformas software.

Por otra parte, el uso de tecnologías ampliamente extendidas, como Linux, Python o Flask, contribuye a mejorar la robustez global de la solución y facilita su mantenimiento a largo plazo.

6.6.3 Escalabilidad y Potencial de Aplicación

La arquitectura desarrollada presenta un importante potencial de escalabilidad.

La modularidad del sistema permite incorporar nuevas funcionalidades sin necesidad de rediseñar completamente la solución existente.

Por ejemplo, sería posible integrar bases de datos destinadas al almacenamiento histórico de la información generada, permitiendo realizar estudios estadísticos sobre la utilización del aparcamiento.

Asimismo, la sustitución del servidor local por una infraestructura basada en servicios cloud permitiría proporcionar acceso remoto a la información desde cualquier ubicación con conexión a Internet.

Otra posible línea de evolución consistiría en ampliar el sistema para monitorizar simultáneamente múltiples aparcamientos distribuidos geográficamente.

Desde el punto de vista de las Smart Cities, este tipo de soluciones presenta un elevado interés, ya que permite optimizar la utilización de infraestructuras urbanas y proporcionar información útil tanto a usuarios como a gestores municipales.

Además, la arquitectura desarrollada podría adaptarse fácilmente a otras aplicaciones de monitorización basadas en visión artificial, tales como el control de plazas de aparcamiento convencionales, la supervisión de espacios públicos o la gestión inteligente de infraestructuras urbanas.

6.6.4 Consideraciones Finales

La evaluación global realizada permite concluir que la solución desarrollada constituye un prototipo funcional plenamente operativo capaz de satisfacer los objetivos técnicos establecidos durante las fases iniciales del proyecto.

Los resultados experimentales obtenidos demuestran la viabilidad del uso combinado de técnicas de visión artificial, inteligencia artificial embebida y tecnologías *IoT* para el desarrollo de sistemas inteligentes de monitorización.

Asimismo, el trabajo realizado pone de manifiesto el enorme potencial de las plataformas *Edge AI* para la implementación de aplicaciones inteligentes distribuidas caracterizadas por la necesidad de realizar procesamiento local en tiempo real.

En conjunto, la solución desarrollada establece una base sólida para futuros trabajos orientados a la mejora y ampliación de las funcionalidades implementadas, constituyendo un primer paso hacia el desarrollo de sistemas inteligentes de gestión de infraestructuras urbanas basados en tecnologías *Edge AI*.

CAPÍTULO 7. CONCLUSIONES, TRABAJOS FUTUROS Y MEJORAS

7.1 Conclusiones

El Proyecto ha permitido desarrollar e implementar un sistema funcional para la monitorización del estado de ocupación de un aparcamiento de bicicletas mediante técnicas de visión artificial e inteligencia artificial ejecutadas sobre una plataforma embebida. El sistema desarrollado integra la captura de imágenes, el procesamiento local mediante algoritmos de detección de objetos, la estimación del número de plazas ocupadas y la transmisión de dicha información a una interfaz web.

Uno de los principales resultados obtenidos es la validación de la viabilidad de emplear plataformas de computación en el borde (*edge computing*) para ejecutar modelos de inteligencia artificial sin necesidad de recurrir a servidores externos para realizar la inferencia. Esta aproximación presenta importantes ventajas, entre las que destacan la reducción del tráfico de datos en la red, la disminución de la latencia del sistema y una mayor protección de la privacidad, ya que las imágenes capturadas son procesadas localmente sin necesidad de ser enviadas a servicios remotos.

Asimismo, se ha comprobado que una arquitectura basada en comunicaciones HTTP resulta suficiente para transmitir de forma fiable la información generada por el sistema de detección. Durante las pruebas experimentales realizadas, la plataforma embebida ha sido capaz de enviar correctamente el número de plazas ocupadas y libres a un servidor web desarrollado mediante el framework Flask, actualizándose posteriormente la información mostrada en la interfaz de usuario. De esta forma, se ha validado el funcionamiento completo de la cadena de procesamiento, desde la adquisición de la imagen hasta la visualización final de los resultados.

Desde el punto de vista funcional, el sistema desarrollado permite ofrecer al usuario información actualizada sobre el estado del aparcamiento, mostrando el número total de plazas, las plazas ocupadas, las plazas libres y la hora de la última actualización. Aunque la interfaz implementada presenta un carácter demostrativo y no comercial, cumple satisfactoriamente con los objetivos inicialmente planteados para el Proyecto.

Por otro lado, el empleo de técnicas de detección de objetos basadas en modelos de inteligencia artificial constituye una alternativa flexible frente a soluciones tradicionales basadas en sensores físicos individuales instalados en cada plaza. Mediante una única cámara es posible supervisar simultáneamente múltiples plazas de estacionamiento, reduciendo la complejidad de instalación y facilitando la escalabilidad del sistema.

Finalmente, el desarrollo de este trabajo ha puesto de manifiesto la importancia de la integración entre hardware, software y comunicaciones en el diseño de sistemas *IoT* inteligentes. El proyecto no se limita únicamente a la aplicación de algoritmos de inteligencia artificial, sino que aborda el despliegue completo de una solución funcional

que incluye procesamiento embebido, comunicaciones en red y visualización remota de la información generada.

7.2 Trabajos futuros

A pesar de que los objetivos establecidos para el Proyecto han sido alcanzados satisfactoriamente, existen diversas líneas de trabajo que permitirían ampliar las capacidades del sistema y optimizarlo a un entorno de explotación real.

En primer lugar, una evolución natural del sistema consistiría en sustituir el servidor local utilizado durante las pruebas por una infraestructura accesible públicamente a través de Internet. Para ello podrían emplearse servidores cloud, plataformas *IoT* especializadas o servidores virtuales privados (VPS). Esta modificación permitiría consultar el estado del aparcamiento desde cualquier ubicación y en cualquier momento, eliminando la restricción actual de acceso únicamente desde la red local.

No obstante, la publicación del servicio en Internet requeriría incorporar mecanismos adicionales de ciberseguridad. Entre las medidas que deberían implementarse destacan el cifrado de las comunicaciones mediante HTTPS, la autenticación de usuarios, la gestión de permisos de acceso, la validación de las peticiones recibidas y la monitorización de posibles incidentes de seguridad. La incorporación de estas funcionalidades resultaría imprescindible para garantizar un funcionamiento seguro en un entorno real.

Otra posible línea de desarrollo consiste en realizar una identificación individual del estado de cada plaza de aparcamiento. En la implementación desarrollada, el sistema estima el número global de bicicletas detectadas para calcular el número de plazas libres y ocupadas. Sin embargo, una versión más avanzada podría asociar cada detección a una plaza concreta mediante la definición de regiones de interés o mediante técnicas de segmentación, permitiendo indicar exactamente qué plazas se encuentran disponibles.

Asimismo, podría incorporarse una base de datos persistente destinada al almacenamiento histórico de la información de ocupación. Esto permitiría generar estadísticas de uso, identificar patrones temporales, determinar franjas horarias de máxima demanda y proporcionar información de utilidad para la planificación y gestión de infraestructuras ciclistas.

7.3 Posibles Mejoras

Aunque los resultados obtenidos durante el desarrollo experimental permiten validar la viabilidad técnica de la solución propuesta, el sistema implementado presenta todavía un margen de mejora y evolución. La arquitectura desarrollada constituye un prototipo funcional orientado, principalmente, a demostrar la aplicabilidad de técnicas de visión artificial e inteligencia artificial embebida para la monitorización automática de aparcamientos de bicicletas. No obstante, existen diversas líneas de mejora que podrían abordarse en futuras investigaciones, con objeto de incrementar las prestaciones, robustez y funcionalidad del sistema.

Una de las principales líneas de mejora se encuentra relacionada con la ampliación de las ubicaciones. Durante el desarrollo del proyecto, las pruebas se llevaron a cabo sobre un único escenario físico. Aunque los resultados obtenidos han sido satisfactorios, sería recomendable extender la validación experimental a diferentes emplazamientos y configuraciones espaciales.

Otra línea de mejora consiste en aumentar el número de cámaras utilizadas por el sistema. La solución desarrollada emplea una única cámara fija para monitorizar la totalidad del aparcamiento, lo que simplifica considerablemente la arquitectura global, pero limita el tamaño máximo del área monitorizable. En escenarios reales de mayores dimensiones podría resultar necesario desplegar múltiples dispositivos de captura distribuidos espacialmente.

La incorporación de varias cámaras permitiría extender el área cubierta por el sistema y reducir los problemas asociados a oclusiones parciales entre bicicletas. No obstante, esta ampliación requeriría el desarrollo de mecanismos adicionales de sincronización y fusión de información capaces de integrar adecuadamente las detecciones procedentes de diferentes puntos de observación.

Asimismo, una posible evolución del sistema consistiría en sustituir la arquitectura actual basada en un servidor local por una infraestructura cloud. Aunque la utilización de un servidor local ha permitido simplificar el desarrollo y validar satisfactoriamente la solución propuesta, la migración hacia plataformas cloud proporcionaría una mayor flexibilidad y accesibilidad.

Una arquitectura basada en servicios cloud permitiría consultar la información del sistema desde cualquier ubicación geográfica con acceso a Internet, eliminando la restricción actual que limita el acceso a dispositivos conectados a la misma red local. Además, este enfoque facilitaría la integración con aplicaciones móviles, servicios web externos o plataformas de gestión urbana inteligente.

Desde el punto de vista de la interfaz de usuario, la aplicación web desarrollada podría evolucionar significativamente incorporando nuevas funcionalidades orientadas a mejorar la experiencia de utilización.

Entre las posibles mejoras pueden destacarse:

- Incorporación de mecanismos de autenticación y control de acceso.
- Desarrollo de perfiles de usuario diferenciados.
- Almacenamiento histórico de información sobre ocupación.
- Generación automática de estadísticas de utilización.
- Representación gráfica de tendencias temporales.
- Implementación de sistemas de notificaciones o alertas.

- Desarrollo de aplicaciones móviles específicas.

La disponibilidad de información histórica abriría además la posibilidad de aplicar técnicas avanzadas de análisis de datos y aprendizaje automático con objeto de predecir patrones futuros de utilización del aparcamiento.

Por ejemplo, el sistema podría incorporar modelos predictivos capaces de estimar la probabilidad de disponibilidad de plazas en determinados intervalos temporales, proporcionando así información adicional de gran utilidad para los usuarios.

Otra línea mejora particularmente interesante consiste en mejorar el modelo de inteligencia artificial utilizado durante el proyecto.

Aunque el modelo basado en YOLO ha proporcionado resultados satisfactorios, la rápida evolución experimentada por las técnicas de aprendizaje profundo hace posible considerar la utilización de arquitecturas más avanzadas en futuras versiones del sistema.

La evaluación comparativa de diferentes modelos de detección permitiría analizar el compromiso existente entre precisión, tiempo de inferencia y consumo de recursos computacionales sobre plataformas Edge AI.

Asimismo, podría estudiarse la posibilidad de realizar un entrenamiento específico del modelo utilizando imágenes obtenidas directamente en el escenario de despliegue definitivo. Este proceso de ajuste fino (*fine tuning*) podría contribuir a incrementar la precisión del sistema y mejorar su capacidad de adaptación a entornos concretos.

Otra mejora potencial consistiría en incorporar mecanismos automáticos de calibración y adaptación del sistema a diferentes configuraciones espaciales.

Actualmente, la solución desarrollada asume una posición fija de la cámara y un escenario previamente conocido. Sin embargo, la incorporación de algoritmos capaces de identificar automáticamente las plazas disponibles o recalibrar dinámicamente el sistema permitiría simplificar considerablemente el proceso de despliegue en nuevos emplazamientos.

Desde el punto de vista de las comunicaciones, también podrían incorporarse mecanismos adicionales orientados a incrementar la seguridad y fiabilidad del sistema.

La implementación de protocolos cifrados, mecanismos de autenticación mutua entre dispositivos y técnicas de protección frente a accesos no autorizados resultaría especialmente recomendable en escenarios reales de explotación.

Finalmente, una de las líneas de evolución más prometedoras consiste en la integración del sistema desarrollado dentro del ecosistema de las ciudades inteligentes (*Smart Cities*).

La información generada por la plataforma podría combinarse con datos procedentes de otros sensores urbanos para desarrollar soluciones integradas orientadas a optimizar la

movilidad sostenible, fomentar el uso de la bicicleta y mejorar la gestión global de infraestructuras urbanas.

En conjunto, las posibles mejoras descritas ponen de manifiesto el elevado potencial de evolución de la arquitectura desarrollada y evidencian que la solución implementada constituye una base sólida sobre la que desarrollar futuros sistemas inteligentes de monitorización basados en tecnologías Edge AI e Internet de las Cosas.

CAPÍTULO 8. ESTUDIO ECONÓMICO

8.1 Introducción

El presente capítulo tiene como objetivo realizar una estimación económica del coste asociado al desarrollo del sistema de detección de ocupación de plazas de aparcamiento para bicicletas presentado en este Proyecto.

Dado que el Proyecto se ha desarrollado en un entorno académico y tiene como finalidad la validación de una prueba de concepto, el análisis económico realizado no pretende representar el coste de un producto comercial definitivo, sino proporcionar una estimación razonable de los recursos necesarios para reproducir el sistema desarrollado.

Para ello se consideran tanto los costes asociados al hardware empleado como los costes derivados del desarrollo de ingeniería, incluyendo las horas dedicadas al diseño, implementación, pruebas y documentación del proyecto.

8.2 Costes de hardware

En la Tabla 8.1 se recogen los principales elementos hardware utilizados durante el desarrollo del proyecto (*valores aproximados para usuario particular basados en* **[24]**)

Elemento	Cantidad	Coste unitario (€)	Coste total (€)
Placa Renesas RZ/V2L SMARC EVK	1	300	300
Cámara USB	1	40	40
Tarjeta microSD	1	15	15
Router doméstico para pruebas	1	50	50
Cables y accesorios	1	20	20
Total hardware			**425 €**

Imagen 29. Costes aproximados Hardware

Debe señalarse que algunos de los elementos empleados han sido proporcionados por la universidad con fines de desarrollo y evaluación. Sin embargo, para la realización del estudio económico se considera su coste de adquisición en el mercado.

8.3 Costes de software

El desarrollo del proyecto se ha realizado partiendo de herramientas software de código abierto o disponibles mediante licencias académicas, por lo que no ha sido necesario asumir costes adicionales de licenciamiento.

Entre las principales herramientas utilizadas se encuentran:

- Sistema operativo Linux Ubuntu.
- Lenguaje de programación Python.

- Librería OpenCV.
- Framework Flask.
- Entorno de virtualización VirtualBox.
- Herramientas proporcionadas por Renesas dentro del AI Software Development Kit.

Por tanto, el coste asociado al software puede considerarse nulo desde el punto de vista económico.

8.4 Costes de ingeniería

La partida económica más significativa del proyecto corresponde al tiempo dedicado al desarrollo e integración del sistema.

Se estima que el desarrollo completo del Proyecto ha requerido aproximadamente 300 horas de trabajo, incluyendo actividades de investigación bibliográfica, diseño del sistema, implementación software, integración hardware, realización de pruebas experimentales, análisis de resultados y redacción de la memoria.

Considerando un coste horario de ingeniería de 35 €/h, **[25]**valor habitualmente utilizado para estimaciones preliminares de proyectos tecnológicos, se obtiene:

Concepto	Horas	Coste unitario (€/h)	Coste total (€)
Desarrollo e implementación	300	35	10.500 €
Total ingeniería			**10.500 €**

Imagen 30. Costes aproximados Servicios profesionales

8.5 Coste total del proyecto

La Tabla 8.5 resume el coste total estimado del proyecto.

Concepto	Coste (€)
Hardware	425 €
Software	- €
Ingeniería	10.500 €
Coste total estimado	**10.925 €**

Imagen 31. Coste estimado total

Por tanto, el coste total estimado para el desarrollo del sistema asciende a aproximadamente **10.925 €**.

Es importante destacar que la mayor parte del coste corresponde al trabajo de ingeniería y desarrollo, mientras que el coste del hardware representa únicamente una pequeña

fracción del coste global. Este comportamiento es habitual en proyectos de investigación y desarrollo, donde las tareas de diseño, integración y validación constituyen la actividad predominante.

8.6 Análisis de viabilidad económica

Desde un punto de vista económico, el sistema desarrollado presenta un elevado potencial de aplicación debido a la utilización de un número reducido de elementos hardware y a la posibilidad de supervisar múltiples plazas de aparcamiento mediante una única cámara.

Además, el empleo de tecnologías de código abierto y plataformas de computación en el borde permite reducir significativamente los costes de despliegue y operación respecto a otras soluciones basadas en infraestructuras centralizadas o sensores individuales por plaza.

En consecuencia, el sistema puede considerarse económicamente viable como solución de monitorización para aparcamientos de bicicletas de pequeño y mediano tamaño, especialmente en entornos urbanos, universitarios o empresariales.

CAPÍTULO 9. REFERENCIAS

[1] IBM, What is Computer Vision?

https://www.ibm.com/topics/computer-vision

[2] OpenCV Documentation

https://docs.opencv.org/

[3] IBM, What is Artificial Intelligence (AI)?

https://www.ibm.com/topics/artificial-intelligence

[4] IBM, What is Machine Learning?

https://www.ibm.com/topics/machine-learning

[5] IBM, What is Deep Learning?

https://www.ibm.com/topics/deep-learning

[6] IBM, What are Convolutional Neural Networks?

https://www.ibm.com/topics/convolutional-neural-networks

[7] TensorFlow, Convolutional Neural Networks Tutorial

https://www.tensorflow.org/tutorials/images/cnn

[8] IBM, What is Object Detection?

https://www.ibm.com/topics/object-detection

[9] NVIDIA, Object Detection: Key Concepts and Techniques

https://developer.nvidia.com/

[10] J. Redmon, S. Divvala, R. Girshick y A. Farhadi,

"You Only Look Once: Unified, Real-Time Object Detection"

https://pjreddie.com/media/files/papers/YOLOv1.pdf

[11] Ultralytics YOLO Documentation

https://docs.ultralytics.com/

[12] OpenCV Documentation

https://docs.opencv.org/

[13] OpenCV Foundation

https://opencv.org/

[14] IBM, What is Edge AI?

https://www.ibm.com/think/topics/edge-ai

[15] NVIDIA, What is Edge AI?

https://www.nvidia.com/en-us/glossary/edge-ai/

[16] Renesas Electronics, RZ/V2L Microprocessor

https://www.renesas.com/

[17] Renesas Electronics, DRP-AI Documentation

https://renesas-rz.github.io/

[18] Mozilla Developer Network (MDN). *An overview of HTTP*.
https://developer.mozilla.org/en-US/docs/Web/HTTP/Overview

[19] Mozilla Developer Network (MDN). *HTTP request methods*.
https://developer.mozilla.org/en-US/docs/Web/HTTP/Methods

[20] European Commission, Smart Cities and Communities

https://smart-cities-marketplace.ec.europa.eu/

[21] IBM, What is Computer Vision?

https://www.ibm.com/topics/computer-vision

[22] Bosch, Parking Lot Sensor Technologies

https://www.bosch.com/

[23] R01_object_detection

https://github.com/renesas-rz/rzv_ai_sdk/tree/main/R01_object_detection

[24] Página web Digikey España componentes y piezas electrónicas

https://www.digikey.es/es/products/filter/placas-y-kits-de-evaluaci%C3%B3n-y-demostraci%C3%B3n/787?s=N4IgTCBcDaIA4BsCGBjJACATgUwHbYGckCQBdAXyA

[25] Colegio Oficial de Graduados en Ingeniería e Ingenieros Técnicos Industriales de Bizkaia

https://www.ingenierosbizkaia.eus/colegiadas-os/servicios-profesionales/ejercicio-profesional/

ANEXO: ALINEACIÓN DEL PROYECTO CON LOS OBJETIVOS DE DESARROLLO SOSTENIBLE (ODS)

El Proyecto contribuye al cumplimiento de diversos Objetivos de Desarrollo Sostenible (ODS) establecidos en la Agenda 2030 de las Naciones Unidas.

El sistema desarrollado tiene como finalidad monitorizar la ocupación de aparcamientos de bicicletas mediante técnicas de visión artificial ejecutadas en una plataforma embebida, proporcionando información actualizada sobre la disponibilidad de plazas. Esta funcionalidad puede favorecer el uso de medios de transporte sostenibles y contribuir al desarrollo de infraestructuras urbanas inteligentes.

A continuación, se analiza la alineación del proyecto con los ODS más relevantes.

ODS principal: ODS 11 – Ciudades y comunidades sostenibles

El Objetivo de Desarrollo Sostenible número 11 persigue lograr que las ciudades y los asentamientos humanos sean inclusivos, seguros, resilientes y sostenibles. Entre sus metas se encuentra la promoción de sistemas de transporte sostenibles y la reducción del impacto ambiental asociado a la movilidad urbana.

El sistema desarrollado contribuye directamente a este objetivo mediante la mejora de la gestión de aparcamientos para bicicletas. La disponibilidad de información en tiempo real sobre la ocupación de las plazas facilita el uso de la bicicleta como medio de transporte habitual, reduciendo las dificultades asociadas al estacionamiento y favoreciendo una movilidad urbana más eficiente.

Asimismo, la utilización de infraestructuras inteligentes permite optimizar el uso de los espacios disponibles, evitando situaciones de saturación o infrautilización de los aparcamientos y mejorando la experiencia de los usuarios.

Aunque el proyecto se ha desarrollado como una prueba de concepto sobre un aparcamiento de seis plazas, la arquitectura propuesta es escalable y podría extenderse fácilmente a instalaciones de mayor tamaño distribuidas en diferentes localizaciones urbanas.

ODS secundario: ODS 3 – Salud y bienestar

La promoción del uso de la bicicleta como medio de transporte tiene un impacto positivo sobre la salud de la población, fomentando la realización de actividad física de manera cotidiana.

La existencia de sistemas que faciliten el estacionamiento seguro y la consulta de plazas disponibles puede incrementar la utilización de la bicicleta en desplazamientos diarios, contribuyendo indirectamente a la mejora de la salud y el bienestar de los ciudadanos.

Además, el aumento del uso de la bicicleta frente al vehículo privado favorece la reducción de la contaminación atmosférica y acústica en entornos urbanos, con el consiguiente beneficio para la salud pública.

ODS secundario: ODS 9 – Industria, innovación e infraestructura

El proyecto incorpora tecnologías avanzadas de inteligencia artificial, visión por computador y computación en el borde (*edge computing*) ejecutadas sobre una plataforma embebida, promoviendo el desarrollo de infraestructuras digitales innovadoras.

La integración de algoritmos de detección de objetos con sistemas de comunicaciones y servicios web constituye un ejemplo de aplicación práctica de tecnologías emergentes dentro del ámbito de las ciudades inteligentes (*smart cities*).

Asimismo, el empleo de herramientas de código abierto y plataformas reconfigurables favorece la reproducibilidad del sistema y facilita futuras actividades de investigación y desarrollo.

ODS secundario: ODS 12 – Producción y consumo responsables

El sistema propuesto contribuye a realizar un uso más eficiente de la infraestructura existente al optimizar la ocupación de los aparcamientos para bicicletas.

Frente a soluciones tradicionales basadas en sensores individuales para cada plaza, la utilización de una única cámara para supervisar múltiples posiciones reduce la cantidad de hardware necesario y simplifica las tareas de instalación y mantenimiento.

Por otro lado, el empleo de una plataforma embebida de bajo consumo y software actualizable permite prolongar la vida útil del sistema y reducir la generación de residuos electrónicos.

ODS secundario: ODS 13 – Acción por el clima

El fomento del transporte ciclista constituye una medida reconocida para reducir las emisiones de gases de efecto invernadero asociadas al transporte urbano.

Aunque el impacto directo del presente proyecto es limitado debido a su carácter académico y demostrativo, una implantación a gran escala de sistemas de monitorización inteligente de aparcamientos para bicicletas podría favorecer la adopción de medios de transporte no contaminantes y contribuir a la disminución de la huella de carbono de las ciudades.

El sistema implementado demuestra la viabilidad técnica de desplegar soluciones escalables capaces de monitorizar múltiples aparcamientos distribuidos en una ciudad.

Además, la arquitectura propuesta permite supervisar simultáneamente varias plazas utilizando una única cámara y una única plataforma de procesamiento, reduciendo significativamente los recursos hardware necesarios frente a soluciones basadas en sensores individuales.

Por tanto, el proyecto constituye una contribución tecnológica orientada al desarrollo de infraestructuras urbanas inteligentes, sostenibles y alineadas con los principios de la Agenda 2030 de las Naciones Unidas.

ANEXO II: CRONOGRAMA DEL PROYECTO

La dedicación total estimada al proyecto fue de aproximadamente 300 horas.

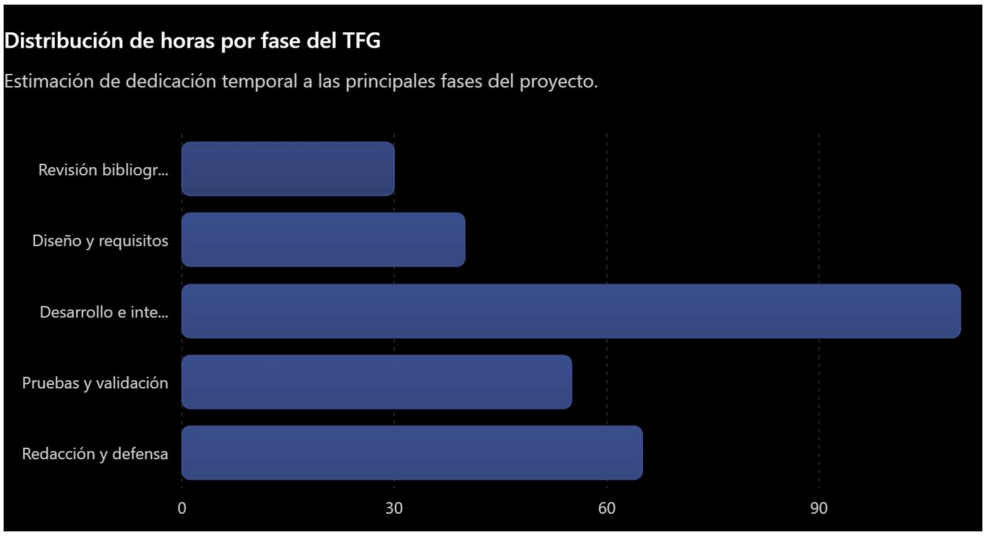
La Figura 32 muestra el diagrama de Gantt correspondiente a la planificación temporal del proyecto.

Actividad	Horas	sep-25	oct-25	nov-25	dic-25	ene-26	feb-26	mar-26	abr-26	may-26	jun-26
Revisión bibliográfica y estado del arte	30										
Análisis y definición de requisitos	20										
Selección y adquisición de hardware	25										
Configuración del SoM y entorno Linux	40										
Implementación del software de IA para la detección de bicicletas	45										
Validación del modelo IA	30										
Desarrollo del servidor web (API + Base de datos)	35										
Ajustes finales y documentación	20										
Redacción de la memoria	45										
Revisión final y preparación defensa	10										

Imagen 32. Cronograma

Las actividades en las siguientes fases principales:

1. Investigación y planificación (septiembre 2025 - diciembre 2025)
2. Desarrollo e implementación (enero 2026 - abril 2026)
3. Validación experimental (abril 2026 - mayo 2026)
4. Documentación y cierre del proyecto (abril 2026 - junio 2026)




Horas por fase
120
100
80
60
40
20
0
Investigación
Planificación
Preparación t...
Desarrollo
Comunicacio...
Validación
Documentaci...
Horas