ADV5G-TWINS-VIDEO

Entregable resumen del piloto de gemelos digitales en tiempo real en Industria 4.0

Versión 1.0

Autores principales	Enrique García, Joan Orti, Universitat Politècnica de València
Distribución	
Entregado	27/06/2025

Avanzando-5G-Gemelos Digitales-Industria (TSI-063000-2021-113)

Financiado por la Unión Europea NextGenerationEU
Plan de Recuperación, Transformación y Resiliencia

Detalles del proyecto

Título del proyecto	ANÁLISIS AI/ML DE VÍDEO DE GEMELOS DIGITALES EN TIEMPO REAL
Acrónimo	ADV5G-TWINS-VIDEO
Fecha de comienzo	24/05/2023
Fecha fin	27/06/2025
Nº de Expediente	MY22/ITEAM/SE/77 Lote 4

Índice de contenidos

1. Introducción 1

2. Descripción de los casos de uso 2

2.1. Caso de uso del brazo robótico 2

2.1.1. Presentación del caso de uso 2

2.1.2. Descripción de la solución 2

2.1.3. Homografía entre tablero y cámara RGB 3

2.1.4. Plano del tablero respecto a la cámara de profundidad 3

2.1.5. Transformación entre cámara de profundidad y robot 4

2.2. Caso de uso del robot de carrera 5

2.2.1. Descripción del caso de uso 5

2.2.3. Detección de personas 7

2.2.4. Detección de objetos con ArUco 8

2.2.5. Segmentación de carril 8

2.2.6. Interfaz de visualización 9

3. Resultados y métricas 9

4. Conclusiones 12

5. Trabajos futuros 13

5.1. Caso de uso del brazo robótico 13

5.2. Caso de uso del robot de carrera 14

1. Introducción

La transformación digital de la industria requiere cada vez más soluciones tecnológicas capaces de interpretar, actuar y visualizar el entorno en tiempo real. En este contexto, los **gemelos digitales** se posicionan como una herramienta clave, al ofrecer una representación dinámica y precisa de los sistemas físicos, facilitando la toma de decisiones, la supervisión remota y el control automatizado. Si a ello se le añade la capacidad de procesar datos mediante técnicas de inteligencia artificial y de transmitirlos mediante redes de **alta capacidad y baja latencia como el 5G**, el potencial de estas soluciones se multiplica significativamente.

El proyecto **ADV5G-TWINS-VIDEO** se enmarca en esta visión, y tiene como objetivo demostrar cómo la combinación de redes 5G, procesamiento en la nube/edge y análisis de vídeo mediante algoritmos de aprendizaje automático puede habilitar **gemelos digitales industriales en tiempo real**. Para ello, se han desarrollado y evaluado dos casos de uso concretos: un **brazo robótico interactivo**, basado en comandos de voz y visión artificial para manipulación de objetos, y un **robot de exploración móvil**, equipado con cámara 360° y LiDAR, capaz de detectar peatones, identificar objetos y analizar la trayectoria en un entorno físico replicado en una interfaz visual dinámica.

Este entregable recoge los principales avances técnicos del proyecto, describiendo las soluciones desarrolladas, las arquitecturas implementadas, las métricas obtenidas y las conclusiones derivadas de su validación en escenarios reales o semi realistas. A través de estos resultados, se busca evidenciar la viabilidad de desplegar sistemas complejos de análisis y visualización en tiempo real sobre redes 5G, sentando las bases para futuras aplicaciones industriales más seguras, inteligentes y conectadas.

2. Descripción de los casos de uso

2.1. Caso de uso del brazo robótico

2.1.1. Presentación del caso de uso

El proyecto desarrolla un nuevo caso de uso para un brazo robótico basado en el juego **tres en raya (tic-tac-toe)**, con el objetivo de demostrar sus capacidades de **localización, captura y traslado de objetos**. El sistema se estructura en **tres nodos distribuidos**:

- **Nodo usuario**: captura audio y video del entorno.
- **Nodo remoto**: procesa la información capturada y genera comandos.
- **Nodo local**: ejecuta los comandos y, en este caso, incluye una cámara para detectar las fichas del juego.

Esta arquitectura permite una gran flexibilidad, ya que los dispositivos del usuario pueden variar desde ordenadores personales hasta móviles o sistemas embebidos como **Raspberry Pi**, dado que el procesamiento pesado lo realiza el nodo remoto.

2.1.2. Descripción de la solución

El brazo robot:

- Recibe **comandos de voz** para realizar movimientos.
- Usa **visión artificial** para detectar la posición de las fichas.
- Manipula las piezas para jugar al tres en raya.

Inicialmente, se intentó ejecutar el juego solo con control por voz y calibración manual del tablero, pero surgieron errores acumulativos en la posición de las fichas. Esto llevó a incorporar una **cámara de profundidad RealSense™**, que proporciona información 3D precisa, mejorando significativamente la manipulación de objetos.

Este caso de uso, aunque sencillo, es extrapolable a **aplicaciones industriales** donde se requiera manipulación precisa de objetos por robots controlados por voz e inteligencia artificial.

La calibración es un paso crucial para que el brazo robot pueda ejecutar tareas con precisión. Implica establecer relaciones entre varios **sistemas de coordenadas**:

Robot: usado para posicionar el efector final.

- **Cámara RGB**: identifica la ubicación del tablero.
- **Cámara de profundidad**: localiza objetos en 3D.
- **Tablero de juego**: define la posición de cada casilla para el usuario.

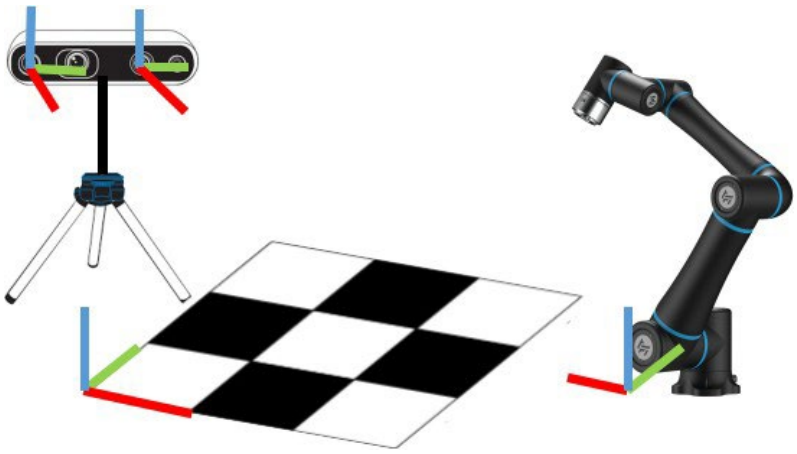

Ilustración 1: Sistemas de coordenadas implicados en la calibración del sistema

2.1.3. Homografía entre tablero y cámara RGB

Dado que el tablero está en un plano, se emplea una **homografía** para relacionar sus coordenadas con las de la imagen RGB. Usando **OpenCV**, se calculan estas correspondencias a partir de las esquinas del tablero. Para resolver ambigüedades de orientación, se fija una convención en la asignación de ejes y origen, facilitando la interpretación por parte del usuario.

2.1.4. Plano del tablero respecto a la cámara de profundidad

Las medidas de la cámara de profundidad tienen cierto **ruido (~1%)**, por lo que se aplican mejoras:

- **Promedio temporal** de 21 imágenes para reducir ruido.
- **Nube de puntos 3D** obtenida alineando imágenes RGB y de profundidad.

- Uso de **RANSAC** para eliminar outliers y clasificación robusta de puntos válidos (inliers).
- Cálculo del plano mediante **SVD**, para proyectar puntos y eliminar el ruido perpendicular al tablero.

2.1.5. Transformación entre cámara de profundidad y robot

La relación entre ambos sistemas se define como una **transformación rígida** (rotación + traslación). Se obtienen al:

1. Medir con precisión 3 puntos no colineales del tablero en el sistema de la cámara.
2. Posicionar manualmente el robot sobre esos puntos y registrar sus coordenadas.
3. Calcular la transformación por mínimos cuadrados.

Este proceso garantiza que todas las coordenadas —del robot, cámaras y tablero— estén correctamente alineadas, habilitando una manipulación precisa de objetos en el entorno del juego.

2.1.6. Recepción y procesado de los comandos de voz

Dentro de la parte que atañe a la recepción y ejecución de los comandos dictados por el usuario, el flujo de trabajo sería el siguiente:

1. Al recibir un comando de **recogida** o **colocación**, se consulta el servicio **get_chessboard**: debe haber pieza en el destino si es recogida, y hueco libre si es colocación; de lo contrario, se emite una alerta y se interrumpe la operación.
2. (Sólo recogida) **Abrir** el gripper a la distancia definida en el fichero de configuración.
3. Calibrar el punto objetivo mediante el fichero de calibración (centro de la pieza o del cuadro).
4. Mover el *end-effector* a la **Z superior** del punto objetivo al **30 %** de la velocidad y aceleración máximas (aproximación rápida y

controlada).
5. Bajar a la **Z inferior** al **10 %** de la velocidad y aceleración máximas (aproximación lenta y precisa).
6. (Recogida) **Cerrar** el gripper según configuración. (Colocación) **Abrir** el gripper para soltar la pieza.
7. Elevar de nuevo a la **Z superior** al **10 %** de la velocidad y aceleración máximas.
8. Desplazar el *end-effector* a la posición **home** al **30 %** de la velocidad y aceleración máximas.

9. Todos los movimientos se ejecutan de forma **sincrónica y punto a punto** para garantizar que las trayectorias con distintas velocidades y aceleraciones no interfieran entre sí.

2.2. Caso de uso del robot de carrera

2.2.1. Descripción del caso de uso

Este caso de uso se centra en el desarrollo de un sistema inteligente de análisis del entorno de un **robot de exploración**.

El robot utilizado es el Robot Móvil Autónomo RB-SUMMIT. Es una plataforma versátil, modular y robusta, diseñada para aplicaciones de I+D en interiores y exteriores, capaz de navegar autónomamente o ser teleoperado. Gracias a su alta movilidad, tamaño compacto y capacidad de carga de hasta 50 kg, resulta ideal para tareas en logística, transporte o agricultura, con una arquitectura de control abierta basada en ROS 2.

El robot presenta una **cámara de visión 360°** y un sensor **LiDAR**. Se pretende detectar personas y objetos en un pista del velódromo y sus posiciones relativas en 3D alrededor del robot por una parte. Por otro lado se segmentan los carriles de la pista del velódromo para analizar si la trayectoria del robot está centrada dentro del carril.

Las principales funcionalidades de la solución implementada son:

- **Rectificación de imágenes 360°** a cámaras virtuales apuntando en las direcciones delantera, derecha, trasera e izquierda

- **Detección de personas** con visión artificial
- **Detección de obstáculos** marcados con ArUco
- **Fusión** de las detecciones IA con datos tridimensionales del LiDAR
- **Segmentación del carril** del velódromo de la UPV con visión artificial
- Integración en una interfaz de **visualización** de los resultados programa en **Unity** conjuntamente con el iTEAM

El procesamiento de los datos se tiene que realizar en **tiempo real**. Para eso se han realizado múltiples optimizaciones en el sistema de análisis a lo largo del proyecto.

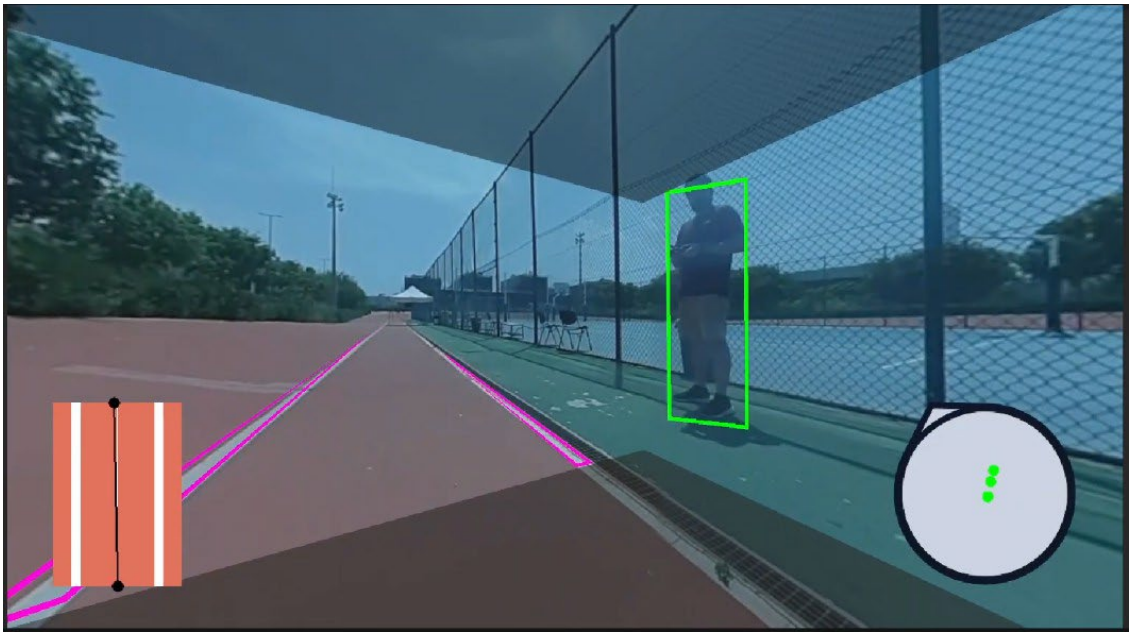

Ilustración 2: captura de pantalla de vídeo demostración de la aplicación de visualización de Unity en el entorno real del velódromo, funcionando en tiempo real

Por otro lado, se ha trabajado en una transmisión de datos eficiente a través de la **red 5G** de la UPV, integrando las fuentes de input de cámara y LiDAR para su procesamiento y posterior envío de resultados a la aplicación de visualización.

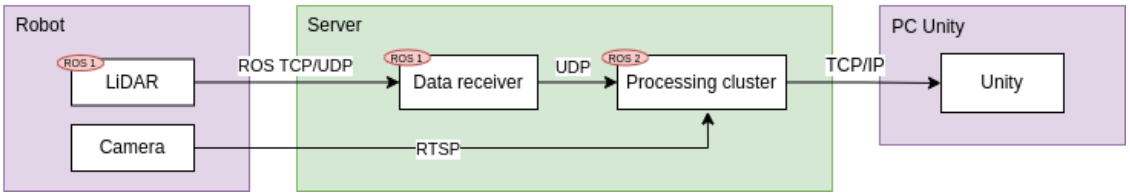

Ilustración 2: diagrama de flujo del dato en el sistema de robot de carrera

La solución ha sido además encapsulada en **Kubernetes** para su orquestación en un sistema global de gemelo digital en tiempo real.

2.2.2. Rectificación de imagen 360°

Uno de los principales retos del proyecto fue procesar correctamente la imagen panorámica generada por la cámara 360° LabPano, cuya distorsión y formato impiden la calibración del sistema cámara-LiDAR (y su explotación para ubicación de obstáculos en 3D).

Para solucionarlo, se desarrolló un método de rectificación basado en teoría de imagen panorámica, que convierte la imagen esférica en vistas planas correspondientes a cuatro direcciones (frontal, trasera, izquierda y derecha), ajustando parámetros como ángulos y campo de visión. Inicialmente, este proceso era computacionalmente costoso (170 ms por frame), pero se optimizó mediante técnicas de paralelización en CPU y, sobre todo, mediante aceleración en GPU con CUDA, reduciendo el tiempo a 40 ms, suficiente para mantener procesamiento en tiempo real.

El sistema se encapsuló en un nodo ROS que publica las imágenes rectificadas para su análisis, integrándose eficientemente con otros módulos como detección de personas y marcadores ArUco.

2.2.3. Detección de personas

La detección de personas se realiza con el modelo de IA YOLOv8, una red

neuronal convolucional optimizada para procesar imágenes en tiempo real con alta precisión. Este modelo permite identificar múltiples personas y sus ubicaciones mediante cuadros delimitadores, incluso en condiciones complejas. Se utilizan pesos preentrenados en grandes datasets, lo que garantiza una detección robusta y precisa en el caso de uso, siendo útil para aplicaciones como vigilancia, análisis de comportamiento e interacción humano-robot.

2.2.4. Detección de objetos con ArUco

Para detectar y localizar objetos estáticos específicos en la pista, se han utilizado marcadores ArUco. Estos patrones en blanco y negro permiten estimar directamente la posición y orientación de los objetos en el espacio 3D, sin necesidad de entrenar modelos. Se colocan físicamente sobre los objetos de interés, y al ser detectados por la cámara del robot, se puede calcular su ubicación precisa con respecto al robot utilizando visión por computadora.

2.2.5. Segmentación de carril

La segmentación de carril se basa en un modelo de inteligencia artificial entrenado específicamente para detectar las líneas del velódromo de la UPV, descartando enfoques clásicos por su poca fiabilidad. Para ello, se utilizó el modelo YOLOv8-seg, entrenándolo inicialmente con datasets externos como Self Driving Cars y Semantic Segmentation Makassar, enfocándose en clases relacionadas con marcas viales. Posteriormente, se aplicó un proceso de fine tuning con imágenes propias del velódromo, etiquetadas semi-automáticamente usando el modelo SAM de Meta, logrando un modelo final preciso y generalizable.

En el sistema, este modelo se emplea para procesar imágenes de cámaras frontales y traseras, generando máscaras de los carriles que se procesan para obtener valores de centrado del robot. Estos se publican en ROS y se visualizan como widgets en Unity, además de los propios polígonos tras un

suavizado de sus contornos.

2.2.6. Interfaz de visualización

Antes de la visualización en Unity, se adaptaron funcionalidades ya desarrolladas para asegurar su compatibilidad con la lógica de dicha plataforma, incluyendo un módulo de des-rectificación que convierte coordenadas desde imágenes rectificadas a coordenadas panorámicas. Este proceso se integró como un nodo ROS 2 que actúa como intermediario entre el análisis y el envío a Unity, permitiendo el envío en tiempo real de detecciones, segmentaciones e indicadores de orientación.

Finalmente, se estandarizó un protocolo de comunicación para el envío de mensajes JSON, que eran recibidos en una aplicación de Unity desarrollada por el iTEAM. Se realizó una demostración funcional en tiempo real en el velódromo de la UPV, validando la integración del sistema completo de análisis en tiempo real con una visualización inmersiva que muestra objetos detectados, segmentación de carril y widgets informativos.

3. Resultados y métricas

3.1. Caso brazo robot: rendimiento

En base a los elementos fundamentales que componen la operación del brazo robot (la cámara RealSense de tres dimensiones, un micrófono para la detección de los comandos y el propio robot) se han obtenido métricas clave que permiten evaluar el rendimiento de la aplicación. En este caso, se han separado las métricas en dos grupos temáticos: **precisión robótica** y **reconocimiento de voz**.

Dentro del primer grupo, se ha considerado relevante tener en cuenta la **desviación espacial de los objetos**, tanto en la recogida como en el depositado de piezas, así como una **tasa de éxito de manipulación**, o lo que es lo mismo, cuántas operaciones se realizan correctamente y si hay un deterioro con el paso del tiempo. Lamentablemente, debido a un fallo relacionado con el manipulador y la operación del mismo con ROS2, no ha

sido posible evaluar dichas métricas. En consecuencia, se ha planteado una **solución alternativa** que permita poder recogerlas y evaluarlas en un futuro, con una aproximación diferente.

En el caso de las métricas de audio, al estar desacopladas de la parte del robot (comunicación con el nodo usuario y el nodo remoto), sí que se han podido extraer los KPIs correspondientes. Se han considerado relevantes para la aplicación el **ancho de banda** consumido por el envío de los comandos de voz, el **tiempo de respuesta** entre el reconocimiento de los mismos y el **Word Error Rate (WER)**, correspondiente al porcentaje de palabras mal reconocidas frente al total pronunciado. Respecto a la primera métrica, gracias al formato Opus, se ha conseguido reducir el ancho de banda a unos inapreciables 20/25 Kbps, casi inapreciable para la red 5G. Por otro lado, el modelo VOSK ha obtenido en datasets populares de voz, valores por debajo del 20% (16% en el Common Voice de Mozilla, 16.72% en el MTEdx y 11.21% en el Multilingual LibriSpeech), más que aceptables para tareas de detección informales. Obviamente, al no haber podido probar los comandos directamente en el robot, no se ha podido evaluar la métrica correspondiente con la ejecución de los comandos, la cual queda pendiente de evaluar más adelante.

3.2. Caso robot de carrera: rendimiento

En cuanto a **visión**, se evaluaron tres funcionalidades principales: **detección** de **personas**, detección de marcadores **ArUco** y segmentación de **carril**. El modelo YOLOv8m demostró un excelente desempeño en la detección de peatones, con una precisión de 0.985 y un F1-score de 0.959, destacando su fiabilidad en escenarios reales como el velódromo. Para los marcadores ArUco, la precisión fue del 100% en todas las pruebas, aunque el recall disminuyó en condiciones reales respecto al entorno controlado. En segmentación de carril, se obtuvieron métricas elevadas, con precisión de 0.963 y mAP@50 de 0.958, aunque con un posible ligero sobreajuste a las condiciones de las características de cámara con las que se tomaron datos en primera instancia..

Además, se evaluó la **precisión** en la devolución de **coordenadas 3D** combinando cámara y LiDAR, logrando un RMSE promedio de 0.47 metros. Se identificaron errores mayores en distancias cortas, los cuales aportaban error considerable al RMSE, y eran atribuibles a paralaje y limitaciones de visibilidad en vistas rectificadas.

Los **tiempos de procesamiento** variaron según el módulo, con la detección de personas alcanzando los 48.6 ms, el análisis de centrado en carril 47 ms, la detección de marcadores ArUco 10 ms, la rectificación de imagen 360° unos 44 ms, la fusión RGB-LiDAR 23 ms, la rectificación de polígonos de carril 6 ms y la rectificación de bounding boxes 3 ms. Al combinar los distintos tiempos, el flujo completo de procesamiento por frame se sitúa en torno a los 116 ms en el caso de detección 3D (peatones con LiDAR), 92 ms para el análisis de carril y 55 ms para la detección de ArUco. Estos valores permiten alcanzar tasas de procesamiento de aproximadamente 8 FPS para detección 3D, 11 FPS para análisis de carril y hasta 20 FPS en el caso de detección de ArUco. En conjunto, estas cifras son compatibles con el funcionamiento en tiempo real del sistema, teniendo en cuenta las tasas de entrada de los sensores (1 FPS para LiDAR y 30 FPS para la cámara).

En lo relativo a **métricas de red**, se analizaron cuatro tramos clave de comunicación. La transmisión del LiDAR desde el robot al sistema central se limitó a 1 Hz para adaptarse al ancho de banda de 2.5 MB/s de la red 5G empleada, con una latencia promedio de 45 ms. La comunicación entre el procesador y la visualización en Unity se realizó por socket TCP/IP, con una latencia baja de 35 ms y un uso de ancho de banda de apenas 19 KB/s, gracias a la reducción de datos como la simplificación de polígonos. Las tasas de envío por tipo de mensaje fueron consistentes con las tasas de procesamiento.

En conjunto, estas métricas demuestran que el sistema del robot de carrera es capaz de operar en tiempo real con precisión y eficiencia, incluso en un entorno con limitaciones reales de red y computación. Las soluciones implementadas han permitido equilibrar el rendimiento de análisis con la calidad de los resultados y la estabilidad de las comunicaciones.

4. Conclusiones

El desarrollo de sistemas inteligentes de análisis del entorno para robots industriales ha demostrado cómo la combinación de inteligencia artificial con redes de comunicación avanzadas, como la 5G, permite crear soluciones capaces de interpretar el entorno en tiempo real. Estos sistemas utilizan sensores como cámaras y LiDAR, junto con algoritmos de visión artificial, para detectar objetos, identificar personas, analizar la posición del robot respecto a su entorno y fusionar información visual y espacial para obtener representaciones 3D precisas.

La comunicación mediante red 5G juega un papel clave al posibilitar la transmisión rápida y estable de datos desde el robot al servidor de procesamiento, permitiendo latencias bajas y un ancho de banda suficiente para manejar múltiples flujos de información simultáneos. Esta infraestructura de comunicación es esencial para garantizar el funcionamiento fluido y en tiempo real del sistema, sobre todo en aplicaciones distribuidas o fuera del entorno local.

En el caso del brazo robótico, esta arquitectura permite la teleoperación del mismo, simulando una situación habitual en entornos industriales. Con la retroalimentación proporcionada por la cámara de visión 3D acerca de la situación del tablero, además de la recepción de comandos desde el nodo remoto, el robot es capaz de realizar movimientos clave posicionando las piezas de forma precisa. Lamentablemente, tal y como se ha mencionado con anterioridad, no se ha podido probar en un entorno real por problemas de hardware del robot.

Para el robot de carrera, se ha alimentado un gemelo digital en tiempo real, que representa de manera virtual y dinámica el entorno físico del robot. Esta representación se visualiza en Unity, lo que facilita tareas de supervisión, análisis remoto y toma de decisiones en entornos industriales. Las mediciones de métricas se han realizado en un entorno que simula condiciones cercanas al entorno industrial final, validando así la viabilidad y

el potencial de estas soluciones para su integración en procesos reales.

Por ello, a pesar de no haber podido profundizar más en el apartado correspondiente del manipulador robótico, se puede considerar que la operación remota en ambos casos de uso es posible y además demostrable con el caso de uso del robot de carrera, pudiendo dar pie a muchas aplicaciones compatibles con esta dinámica de trabajo.

5. Trabajos futuros

5.1. Caso de uso del brazo robótico

Este caso de uso deja abierta la puerta a más que posibles optimizaciones, debido sobre todo a la imposibilidad de probarlo con el hardware real. No obstante, es de sobra conocido el papel fundamental de la robótica en el campo de la teleoperación y la automatización de procesos industriales. Desde un bin-picking caótico, pasando por la operación del robot en situaciones de difícil acceso para los humanos, el abanico de posibilidades es extenso.

Concretamente, la posibilidad de implementar un motor de juego o una lógica más avanzada en el nodo remoto, es interesante. Más aún cuando entra en juego la comunicación a través de 5G, facilitando una respuesta rápida y directa. Obviamente, el grado de autonomía de la aplicación se convierte en un parámetro fundamental a definir para futuras aplicaciones, dependiendo de la casuística de la operación a realizar.

Otra línea de investigación es el encapsulamiento del software en dispositivos embebidos, atractivos para la industria debido a su fácil mantenimiento y su bajo coste. Se conoce que hoy en día, dispositivos como la NVIDIA Jetson, con un apartado gráfico relativamente potente para el tamaño de la placa, pueden jugar un papel fundamental a al hora de orquestar operaciones robotizadas con un peso importante en la parte de visión por computador y aprendizaje profundo. Esto, junto a la fácil implementación y mantenimiento de los productos de UR, puede llegar a convertirse en un estándar industrial.

5.2. Caso de uso del robot de carrera

Una primera línea de continuación natural sería el despliegue del sistema en entornos reales más complejos, como almacenes logísticos, plantas industriales o espacios agrícolas, donde puedan evaluarse su fiabilidad y adaptabilidad frente a condiciones más variables. Para ello, sería necesario un conjunto de datos de entrenamiento completo y variado, especialmente en lo relativo a segmentación de carril y detección de personas u obstáculos. Un dataset más diverso permitiría mejorar la generalización del sistema y su rendimiento ante situaciones no vistas anteriormente.

En paralelo, podría explorarse la incorporación de funciones más avanzadas como la conducción autónoma básica en entornos controlados, así como módulos de análisis predictivo de obstáculos. Esto permitiría no solo detectar la situación actual del entorno, sino anticipar posibles interferencias en la trayectoria del robot o cambios relevantes, aumentando así la autonomía y utilidad del sistema para tareas de supervisión, transporte o patrullado.

Por último, desde el punto de vista de las comunicaciones, una mejora importante sería la optimización del canal de transmisión del LiDAR. Actualmente, la necesidad de reducir el ancho de banda llevó a limitar su frecuencia a 1 Hz, lo cual puede ser suficiente para ciertos casos, pero no para maniobras más ágiles o entornos dinámicos. Esto contribuiría a consolidar una solución robusta, escalable y preparada para un despliegue operativo más ambicioso.