UNIVERSIDAD POLITÉCNICA DE MADRID

ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA Y DISEÑO INDUSTRIAL

Máster Universitario en Ingeniería Electromecánica

TRABAJO FIN DE MÁSTER

Desarrollo de un sistema de clasificación de residuos domésticos mediante un brazo robótico

Autor: Maurizio Rocco D'Alvano Teran

Tutor: David Álvarez Sánchez

Departamento en Ingeniería Eléctrica, Electrónica Automática y Física Aplicada

Madrid, septiembre, 2025

AGRADECIMIENTOS

En primer lugar, quiero dar gracias a Dios Todopoderoso. Mi Salvador que siempre ha estado a mi lado durante los momentos buenos y malos. Él que siempre ha confiado en mí, incluso cuando yo no he confiado en Él o en mí mismo. Gracias Señor por iluminar mi camino y permitirme estar aquí hoy, que gozo de salud para aprender más de Ti y todo lo que me rodea. Y a todos los santos, a aquellos cuya intercesión han hecho posible que yo pueda crecer en la fe y en la esperanza, gracias de todo corazón.

Quiero agradecer, también, a mi familia entera que me ha apoyado con todos los recursos posibles durante mi vida, pero en especial, en este nuevo camino que estoy recorriendo en tierras tan lejanas. Particularmente, agradezco a mis padres Emma y Leonardo, que me han educado siempre con firmeza, disciplina, honestidad y sentido del aprendizaje; ellos dos fueron mis primeros modelos a seguir y, todavía hoy, son mi referencia, gracias por haberme cuidado tanto. Debo hacer mención especial, a mi hermano Gerardo, que ha sido mi compañero desde muy pequeño, él y yo siempre nos hemos apoyado como un equipo para afrontar las dificultades de la vida, muchas gracias, hermano.

No podría escribir estos agradecimientos sin mencionar a mi novia Pepa, tú que con ese corazón tan dulce me has permitido seguir adelante y nunca dejar de sentirme querido, gracias mi vida. Contigo, he podido ver la esperanza dentro de tanta oscuridad, me has dado la confianza para levantarme y me has dicho siempre las palabras necesarias que necesitaba. Especialmente, estos últimos meses, me has inspirado a esforzarme más y más, con el objetivo de ser mejor persona de Cristo. Te estaré siempre agradecido de corazón.

Por último y no menos importante, quiero dar las gracias a mis profesores que me instruyeron durante todo el máster, en particular, a mi tutor David, que siempre estuvo dispuesto y abierto a ayudarme cuando tuve obstáculos que superar. Gracias por su paciencia y su entrega a la vida académica. Finalmente, a todas aquellas personas que de una u otra forma me han apoyado en este proceso de aprendizaje, gracias.

ÍNDICE

ÍNDICE DE FIGURAS

RESUMEN

El presente Trabajo Fin de Máster desarrolla un sistema automatizado de clasificación de residuos domésticos mediante la integración de visión por computador, aprendizaje profundo y robótica colaborativa. El proyecto tiene como objetivo contribuir a la optimización del proceso de separación de desechos, mejorando la eficiencia y precisión en la gestión de residuos reciclables.

Para ello, se ha implementado un modelo de detección de objetos basado en la arquitectura YOLOv8, entrenado con bases de datos de residuos obtenidas y adaptadas desde la plataforma *Roboflow*. La cámara estéreo OAK-D Lite de Luxonis se empleó para la captura de imágenes y la estimación de coordenadas tridimensionales, mientras que el brazo robótico colaborativo UR3e de Universal Robots fue programado para ejecutar las tareas de aproximación, agarre y depósito de objetos.

El sistema fue validado en condiciones de laboratorio y domésticas controladas, alcanzando métricas de precisión y *recall* superiores al 85% y valores de mAP50 cercanos al 90%. Asimismo, se incorporó una modificación mecánica a la pinza original del robot mediante piezas diseñadas en Autodesk Inventor e impresas en 3D, lo que permitió ampliar el rango de agarre y garantizar la manipulación de la mayoría de los residuos evaluados.

Los resultados obtenidos demuestran la viabilidad del sistema como prototipo funcional de clasificación automática de residuos, constituyendo un aporte tanto académico como tecnológico en el ámbito de la robótica aplicada a la sostenibilidad medioambiental. Finalmente, se plantean líneas de mejora futura, como la optimización del modelo de detección mediante bases de datos más variadas y representativas, el uso de pinzas más avanzadas y el desarrollo de algoritmos que permitan la clasificación simultánea de múltiples objetos.

Palabras clave: Clasificación de residuos, visión por computador, YOLOv8, OAK-D Lite, UR3e, robótica colaborativa, sostenibilidad.

ABSTRACT

This master's final project presents the development of an automated domestic waste classification system through the integration of computer vision, deep learning, and collaborative robotics. The aim of the project is to contribute to the optimization of waste separation processes, enhancing efficiency and accuracy in the management of recyclable materials.

For this purpose, an object detection model based on the YOLOv8 architecture was implemented, trained with waste datasets obtained and adapted from the Roboflow platform. The Luxonis OAK-D Lite stereo camera was employed for image acquisition and 3D coordinate estimation, while the Universal Robots UR3e collaborative robotic arm was programmed to perform approach, grasping, and deposition tasks.

The system was validated under controlled laboratory and domestic conditions, achieving precision and recall metrics above 85% and mAP50 values close to 90%. In addition, a mechanical modification was introduced to the original robot gripper using 3D-printed components designed in Autodesk Inventor, which expanded its grasping range and enabled the manipulation of most of the tested waste items.

The results demonstrate the feasibility of the system as a functional prototype for automatic waste classification, providing both academic and technological contributions in the field of robotics applied to environmental sustainability. Finally, future work is proposed, including the optimization of the detection model with more varied and representative datasets, the use of advanced grippers, and the development of algorithms enabling the simultaneous classification of multiple objects.

Keywords: Waste classification, computer vision, YOLOv8, OAK-D Lite, UR3e, collaborative robotics, sustainability.

GLOSARIO

LISTA DE ABREVIATURAS

CAD	Diseño asistido por ordenador (*Computer-Aided Design*)
CNN	Red neuronal convolucional (*Convolutional Neural Network*)
IoT	Internet de las cosas (*Internet of Things*)
IoU	Intersección sobre Unión (*Intersection over Union*)
mAp	Precisión media promedio (*mean Average Precision*)
ROI	Región de interés (*Region of Interest*)
RTDE	Intercambio de datos en tiempo real (*Real-Time Data Exchange*)
YOLO	Solo mira una vez (*You Only Look Once*)

Capítulo 1. INTRODUCCIÓN

1.1. CONTEXTO Y MOTIVACIÓN

En las últimas décadas, el crecimiento de la población y, por consiguiente, del consumo inherente de productos de necesidad en el hogar han generado un aumento significativo en la cantidad de residuos domésticos [1][2]. La gestión inadecuada de estos desechos no solo impacta negativamente al medio ambiente, sino que también supone un desafío económico y logístico para los sistemas de recolección y reciclaje. La separación y clasificación de residuos que proceden del consumo en casa se realiza habitualmente por los mismos ciudadanos, lo que resulta en una actividad dependiente de la correcta disposición de las personas al momento de separar un residuo de otro, y, por lo tanto, puede conllevar a errores frecuentes y la disminución de la eficacia del proceso de reciclaje.

En este contexto, la automatización de la clasificación de residuos representa una alternativa prometedora para comodidad del ciudadano, así como mejorar la sostenibilidad y optimizar el tratamiento de desechos sólidos urbanos. Tecnologías como la visión por computador, el aprendizaje automático y la robótica colaborativa

permiten diseñar sistemas que, de forma autónoma, identifiquen y manipulen diferentes tipos de residuos, garantizando una separación eficiente.

El presente trabajo se enmarca en esta línea de pensamiento y visión tecnológica, proponiendo el desarrollo de un sistema de clasificación automática de residuos domésticos mediante la integración de algoritmos de detección de objetos, un sistema de visión por computador basado en el uso de la cámara OAK-D Lite de *Luxonis* y un brazo robótico colaborativo UR3e de *Universal Robots*. Además, el desarrollo de este tipo de sistemas permite la aplicación de conceptos obtenidos en el transcurso del máster, y también el aprendizaje de nuevos procedimientos para la resolución de problemas que suelen producirse durante la integración de tecnologías como la visión por computador, el aprendizaje automático y el movimiento real en el espacio del robot.

1.2. JUSTIFICACIÓN DEL PROYECTO

En el ámbito mundial, las organizaciones internacionales han establecidos objetivos cada vez más exigentes en materia de reciclaje y gestión de residuos. La Unión Europea, a través de sus directivas en materia de gestión de residuos [3][4][5], es la referencia para considerar este tipo de proyectos con significativa importancia. En España, dichos objetivos se traducen en la necesidad de aumentar los índices de recuperación y valorización de materiales reciclables, al tiempo que se reducen los costes de procesamiento.

Este proyecto responde a esa necesidad al proponer una solución automatizada que integra técnicas avanzadas de aprendizaje automático (redes neuronales) y visión por computador con la capacidad operativa de un brazo robótico. La clasificación automática no solo reduce la dependencia de la intervención humana en procesos repetitivos y potencialmente insalubres, sino que también incrementa potencialmente la precisión en la separación de materiales, lo que repercute en un mayor aprovechamiento de los recursos reciclables.

Además, el proyecto supone un aporte de valor académico y práctico al campo de la robótica aplicada a problemas sociales y medioambientales. La integración de detección en tiempo real con algoritmos de visión (YOLO), el uso de transformaciones homogéneas para ubicar los objetos en coordenadas del robot a partir de un *Apriltag* de referencia, y la programación de rutinas de control del UR3e mediante el protocolo de comunicación en tiempo real RTDE representan contribuciones que trascienden la aplicación inmediata y pueden servir como base para investigaciones posteriores en entornos industriales o urbanos.

1.3. OBJETIVOS

1.3.1. OBJETIVO GENERAL

El objetivo general de este trabajo es el de diseñar y desarrollar un sistema de clasificación de residuos que, mediante el uso de visión por computador y algoritmos de aprendizaje computacional, sea capaz de identificar, detectar y clasificar diferentes tipos de residuos domésticos con el uso de un brazo robótico.

1.3.2. OBJETIVOS ESPECÍFICOS

Siguiendo el planteamiento del objetivo general, se asocian los siguientes objetivos específicos:

1. Adaptar técnicas de aprendizaje computacional para el reconocimiento de tipos de residuos domésticos utilizando imágenes obtenidas de una cámara.
2. Configurar y calibrar un brazo robótico para realizar tareas específicas de selección y clasificación basadas en la información proporcionada por el sistema de visión por computador.
3. Validar la precisión y la confianza del sistema desarrollado mediante pruebas en las que se presentan los objetos de manera individual en el área de trabajo del brazo robótico.
4. Implementar el sistema de clasificación mediante visión por computador para el caso en el que se presentan varios tipos de residuos al mismo tiempo y de forma no estructurada en el espacio de trabajo del brazo robótico.

1.4. METODOLOGÍA

La metodología adoptada en este proyecto se estructura en varias fases:

1. Revisión bibliográfica y del estado de la técnica: análisis de investigaciones previas en clasificación automática de residuos, visión por computador aplicada a la robótica y técnicas de control de manipulación robótica.

2. Diseño del sistema: integración conceptual de los componentes de *hardware* (UR3e, cámara OAK-D Lite, pinza) y *software* (YOLOv8, RTDE, transformaciones de coordenadas con *Apriltag*).

3. Implementación: desarrollo de algoritmos en *Python* para la detección de objetos y estimación de posición en el espacio 3D. El uso de matrices de transformación homogéneas para calcular las coordenadas del objeto detectado entre cámara, *Apriltag* y base del robot. Creación de un algoritmo de movimiento del robot para gestionar rutinas como la posición inicial, descenso, ascenso y depósito de residuos. Configuración de la comunicación entre el ordenador y el robot mediante RTDE.

4. Validación experimental: verificación de la confiabilidad del programa de detección de objetos mediante pruebas de detección en un entorno controlado: diferentes configuraciones de residuos (objetos individuales y múltiples mezclados), evaluando precisión, ruido en la estimación de la ubicación y rendimiento global.

5. Análisis de resultados y discusión: comparación con referencias previas y valoración de limitaciones, incluyendo niveles de tolerancia en los errores de posicionamiento.

1.5. ESTRUCTURA DE LA MEMORIA

Con el fin de facilitar la comprensión del proyecto, el presente Trabajo Fin de Máster se describe brevemente de la siguiente manera:

1. En el Capítulo 1, el presente capítulo, se da la introducción al proyecto de estudio, justificación, objetivos, una breve descripción de la metodología y estructura del trabajo.

2. En el Capítulo 2, se expone el estado de la técnica, revisando principales tecnologías y enfoques relacionados con la clasificación de residuos, la visión por computador y el control de robots colaborativos.

3. En el Capítulo 3, se recogen las bases teóricas que permiten la aplicación de la tecnología desarrollada en este trabajo. Se incluyen los fundamentos teóricos y tecnológicos necesarios para el desarrollo del proyecto, incluyendo conceptos de aprendizaje computacional, visión por computador, transformaciones homogéneas de coordenadas y cinemática del brazo robótico.

4. En el Capítulo 4, se describe el desarrollo del sistema, detallando la configuración de todas sus partes: la cámara estéreo, el algoritmo de detección, el aprendizaje de la red neuronal, la metodología para el cálculo de posiciones en el espacio del objeto detectado, el uso de *Apriltags* como referencia espacial y el control del brazo robótico, incluido el protocolo de comunicación utilizado, así como también el protocolo de funcionamiento para el accionamiento de la pinza a través de la consola de programación del robot.

5. En el Capítulo 5, se presentan los resultados obtenidos en las pruebas de validación, analizando la precisión de la detección, los errores de posicionamiento y el rendimiento global del sistema.

6. Por último, en el Capítulo 6, se presentan las conclusiones del proyecto y se proponen posibles líneas de mejora que se pudieran considerar para la elaboración de sistemas futuros.

Capítulo 2. ESTADO DE LA TÉCNICA

2.1. SISTEMAS DE CLASIFICACIÓN AUTOMÁTICA DE RESIDUOS

La clasificación automática de residuos ha cobrado relevancia en los últimos años como una alternativa eficaz para afrontar los crecientes volúmenes de desechos generados a nivel doméstico e industrial. El incremento del consumo y los estilos de vida urbanos han conducido a un incremento en la generación de residuos, lo cual ha impulsado el interés en soluciones tecnológicas que optimicen la separación y el reciclaje.

Entre los métodos más investigados destacan los sistemas basados en visión por computador y aprendizaje automático, que permiten reconocer y clasificar materiales en tiempo real. Estos enfoques reducen la dependencia de la clasificación manual, caracterizada por su baja eficiencia y relevante tasa de error, y permiten un mejor aprovechamiento de materiales reciclables.

Estudios recientes han explorado diferentes arquitecturas de redes neuronales aplicadas a este campo. Por ejemplo, se ha implementado un sistema de segregación de residuos multiclase empleando YOLOv5, obteniendo resultados prometedores en la separación de materiales heterogéneos en condiciones de laboratorio [6]. De manera

complementaria, otros trabajos han aplicado YOLOv8 junto a brazos robóticos tipo Delta para realizar clasificación automatizada en entornos controlados, demostrando la viabilidad de combinar visión por computador y manipulación robótica.

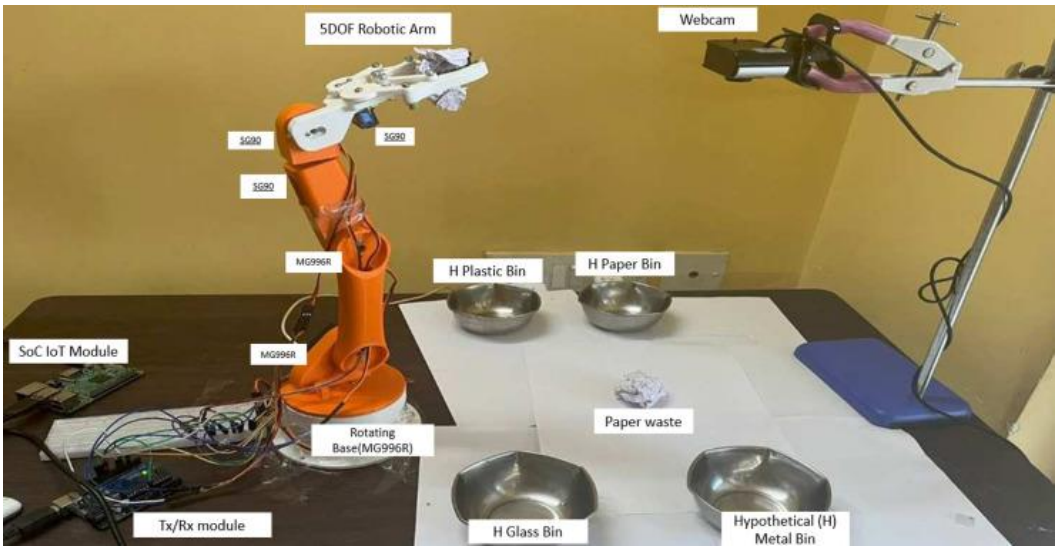

Figura 2.1. Configuración de un sistema de clasificación de residuos [6].

La aplicación de estas técnicas no se limita al ámbito doméstico. También se ha desarrollado un prototipo de clasificación de residuos médicos utilizando YOLO e Internet de las Cosas (IoT) [7], logrando una precisión elevada en entornos hospitalarios y evidenciando la adaptabilidad de estos métodos a diferentes contextos. Estas investigaciones demuestran que la integración de inteligencia artificial en sistemas de gestión de residuos no solo es viable, sino que constituye una línea de desarrollo en expansión y con gran potencial para mejorar la sostenibilidad global.

2.2. VISIÓN POR COMPUTADOR EN ROBÓTICA

La visión por computador constituye un área esencial dentro de la robótica moderna, ya que permite a los sistemas adquirir información del entorno y actuar en consecuencia. En la actualidad, su aplicación se ha visto impulsada por el desarrollo de algoritmos de aprendizaje profundo y por la disponibilidad de *hardware* especializado en adquisición de imágenes. Otras obras de referencia [8] han servido de base para el desarrollo de técnicas de segmentación, detección y reconocimiento de objetos, fundamentales para la interacción autónoma de los robots.

El uso de la visión en robótica colaborativa se vincula estrechamente con la necesidad de diseñar sistemas capaces de compartir espacios de trabajo con humanos, manteniendo criterios de seguridad, confiabilidad y eficiencia lo suficientemente

elevados. Los avances en cámaras estéreo y sensores de profundidad han facilitado la estimación tridimensional de posiciones, posibilitando tareas de manipulación más precisas.

2.2.1. MODELOS DE DETECCIÓN DE OBJETOS

La detección de objetos en tiempo real ha sido históricamente un desafío de visión artificial. Con la revolución de arquitecturas de redes neuronales convolucionales, se han alcanzado niveles de precisión y velocidad que antes no eran posibles. Entre estas arquitecturas, la familia YOLO (*You Only Look Once*) ha marcado un punto de importante desarrollo en el campo.

Desde sus primeras versiones, YOLO ha evolucionado hacia modelos más rápidos y precisos. Revisiones exhaustivas, documentan los avances desde YOLOv1 hasta YOLOv8, destacando mejoras en eficiencia computacional y robustez en detecciones [9]. Del mismo modo, Wang y Liao han mostrado cómo las versiones más recientes, como YOLOv10, combinan rapidez y precisión para aplicaciones entornos dinámicos [10].

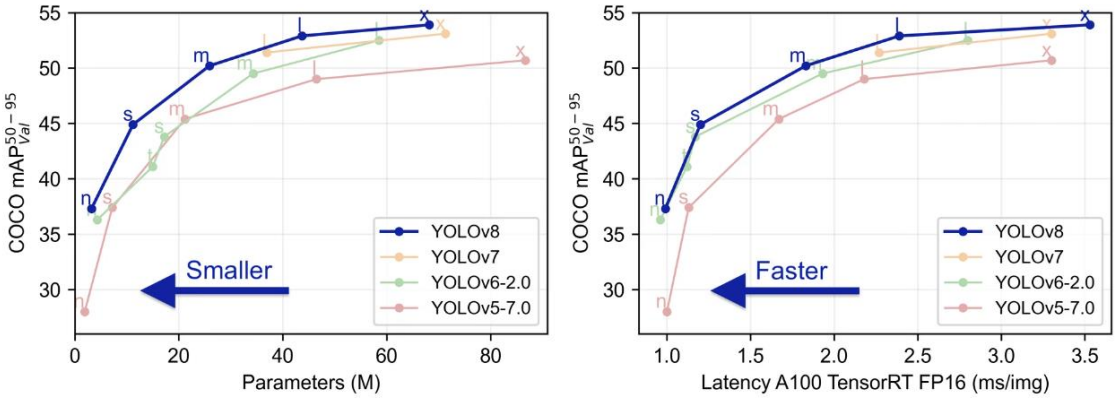

Figura 2.2. Métricas de rendimiento del modelo YOLOv8 en comparación con sus predecesores [11].

YOLOv8, desarrollado por *Ultralytics* [11], representa una de las arquitecturas más utilizadas actualmente en aplicaciones de robótica, gracias a su capacidad de ejecutar detecciones precisas en tiempo real incluso en sistemas con recursos limitados. Su integración en sistemas de detección y clasificación de residuos ha permitido avances en la separación multiclase de materiales.

2.2.2. USO DE SISTEMAS DE ETIQUETAS Y ESTIMACIÓN DE POSES

Más allá de la detección de objetos, los robots requieren conocer la posición tridimensional de los elementos para interactuar con ellos de forma segura y eficaz. Para ello, se emplean técnicas de estimación de poses que, en muchos casos, se apoyan en el uso de marcadores artificiales. Entre ellos, los *Apriltags* han demostrado ser una herramienta robusta y confiable para proporcionar referencias espaciales [12].

Estos marcadores permiten establecer un sistema de coordenadas de referencia en el espacio de trabajo, lo que resulta fundamental para transformar las coordenadas de detección de la cámara al sistema de referencia del robot.

La combinación de detección mediante modelos de redes neuronales (como YOLOv8) y la estimación de poses como *Apriltags* posibilita el desarrollo de sistemas integrados capaces de reconocer y manipular objetos en entornos no estructurados, constituyendo la base tecnológica de múltiples investigaciones recientes.

2.3. CONTROL Y PROGRAMACIÓN DE BRAZOS ROBÓTICOS

El control de robots colaborativos constituye un aspecto crítico en la robótica aplicada a la manipulación de objetos. En particular, los brazos robóticos como el UR3e de *Universal Robots* destacan por su facilidad de programación, versatilidad y capacidad de operar en entornos compartidos con seres humanos.

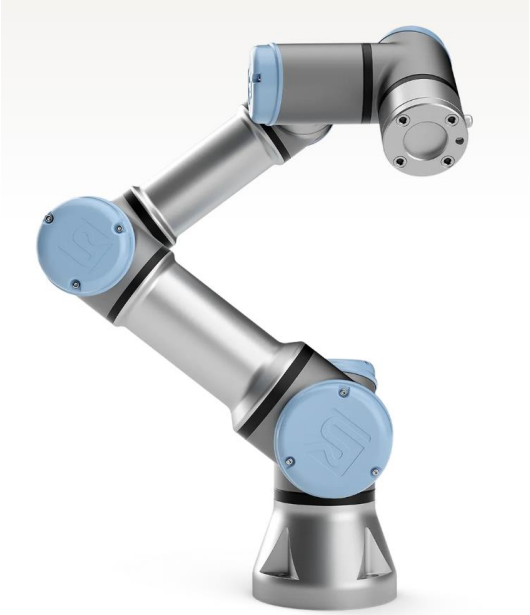

Figura 2.3. Brazo robótico colaborativo UR3e de *Universal Robots* [13].

Para establecer comunicación y control con este tipo de robots, se emplea el protocolo RTDE (*Real-Time Data Exchange*), propuesto por el fabricante [13], que permite el envío y recepción de datos en tiempo real entre el robot y un ordenador u otra aplicación externa. Esto resulta especialmente útil para implementar movimientos personalizados, controlar actuadores como pinzas (*grippers*) y monitorear el estado del robot durante su operación. Con la facilidad que conlleva ejecutar todas estas acciones con el robot desde un entorno externo, permite que la integración de la parte de detección (programa basado en el uso de una red neuronal) y del movimiento del robot sea más flexible y conveniente.

Estudios recientes han explorado el uso de técnicas avanzadas de aprendizaje automático y aprendizaje por refuerzo para optimizar la ejecución de tareas de manipulación. Gomes et al., por ejemplo, desarrollaron un estudio de caso en el que se aplicaba el aprendizaje por refuerzo a tareas de agarrar y reubicar objetos con robots colaborativos, demostrando la viabilidad de integrar enfoques adaptativos en la programación de estos robots.

La flexibilidad del UR3e, combinada con la potencia de algoritmos modernos, abre una amplia variedad de aplicaciones en clasificación automática, e incluso, en manufactura avanzada. Documentación técnica del fabricante y desarrollos recientes en el área apuntan a un crecimiento sostenido en la adopción de estas soluciones en entornos productivos.

2.4. CASOS DE ESTUDIO SIMILARES Y TRABAJOS RELACIONADOS

Los estudios de investigación científicos recientes ofrecen múltiples ejemplos de integración entre visión por computador y robótica para la clasificación de objetos. Estos casos constituyen referencias relevantes para el presente trabajo, ya que aportan tanto validaciones experimentales como aproximaciones metodológicas.

Lahoti et al. desarrollaron un sistema de clasificación de residuos domésticos empleando YOLOv5, alcanzando resultados de precisión elevados en escenarios controlados. De forma similar, otros autores han combinado YOLOv8 con brazos robóticos para realizar tareas de clasificación multiclase, utilizando diferentes configuraciones de manipulación.

Robinson et al. realizaron una revisión exhaustiva sobre visión por computador aplicada a la colaboración humano-robot, destacando la importancia de integrar algoritmos de visión artificial en entornos industriales dinámicos. Por su parte, Terras y

Cohen mostraron la relevancia de fusionar técnicas de visión con inteligencia artificial para mejorar el desempeño de robots colaborativos en tiempo real [14][15].

En aplicaciones específicas, Moktar et al propusieron un sistema de clasificación de residuos médicos que, además de emplear la arquitectura de red neuronal convolucional YOLO, integraba tecnologías IoT para el monitoreo remoto. Este último ejemplo permite ver la posibilidad de ampliar las capacidades de los sistemas de clasificación hacia escenarios más complejos y con mayores requisitos de confiabilidad.

En resumen, los trabajos consultados permiten observar que la combinación de visión por computador, algoritmos de detección de objetos y brazos robóticos constituye un campo activo de investigación, con resultados cada vez más prometedores que permitan una aplicación real bajo circunstancias complejas presentes en el campo.

Capítulo 3. FUNDAMENTOS TEÓRICOS

3.1. VISIÓN POR COMPUTADOR Y ESTIMACIÓN 3D

3.1.1. FUNDAMENTOS DE LA VISIÓN POR COMPUTADOR

La visión por computador es la disciplina que permite a una máquina interpretar información visual, emulando en cierta medida la capacidad humana de percibir, el entorno. En términos computacionales, una imagen no es más que una matriz de valores de intensidad o color, pero el reto radica en extraer de esta información atributos semánticos: reconocer qué objetos están presentes, dónde se encuentran y qué relaciones espaciales mantienen entre sí.

Históricamente, la visión por computador se poyaba en algoritmos manuales de extracción de características, como bordes, esquinas o histogramas de gradientes, que eran posteriormente utilizados en clasificadores estadísticos. Sin embargo, el advenimiento de las redes neuronales convolucionales y el aprendizaje profundo (*Deep*

learning) ha transformado el campo [16][17]. Hoy en día, la detección y el reconocimiento se llevan a cabo mediante arquitecturas entrenadas directamente con grandes cantidades de datos.

En el ámbito de la robótica, la visión es esencial para dotar al robot de percepción espacial. Esto permite que el sistema actúe en entornos no estructurados, detecte obstáculos, planifique trayectorias y ejecute tareas de manipulación con precisión.

3.1.2. Estimación de profundidad y reconstrucción 3D

Una de las limitaciones de la visión monocular es que las imágenes planas no contienen información directa sobre la distancia de los objetos a la cámara. Para resolver este obstáculo, se han desarrollado diversas técnicas de estimación de profundidad, que permiten reconstruir la estructura tridimensional del entorno a partir de información visual.

Entre los enfoques más relevantes se encuentran:

1. **Visión estéreo:** utiliza dos cámaras separadas por una distancia conocida (paralaje) para calcular disparidades entre puntos homólogos en ambas imágenes. A partir de estas disparidades se obtiene la profundidad relativa. Este principio es similar al de la percepción humana mediante los ojos.

2. **Cámaras RGB-D:** dispositivos que, además de capturar la imagen a color, incorporan sensores infrarrojos o patrones estructurados que permiten obtener directamente un mapa de profundidad. Estos sensores integran simultáneamente la textura y la geometría de la escena.

3. **Estimación monocular basada en aprendizaje:** basado en la interpretación de una sola imagen a color, son algoritmos que predicen un mapa de profundidad entrenando con datos previamente calibrados. Aunque presenta limitaciones de precisión, son útiles en escenarios donde sólo se dispone de una cámara estándar RGB.

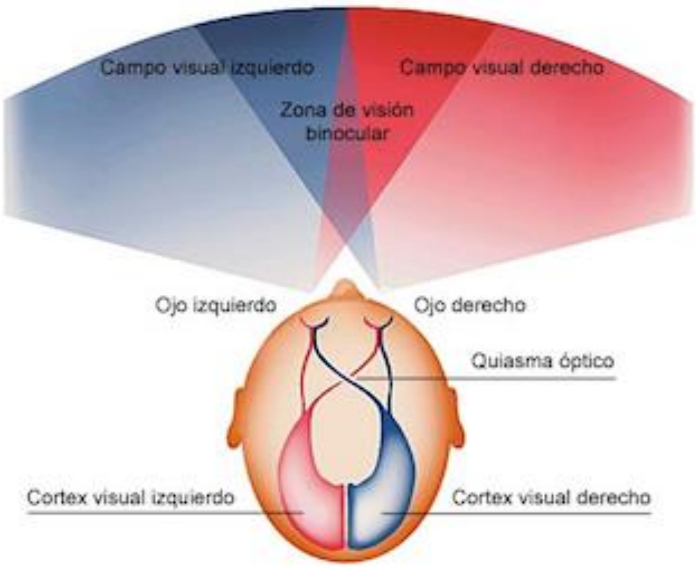

Figura 3.1. Esquema de la visión binocular, usada para detectar profundidad.

En este proyecto, la cámara OAK-D Lite combina visión estéreo con sensores de alta resolución, lo que permite obtener coordenadas 3D de los objetos detectados. Sin embargo, la estimación de profundidad no es perfectamente precisa en todos los escenarios: se ve afectada por textura repetitivas, superficies reflectantes, falta de iluminación o bordes que no se definen claramente. Para mitigar estos problemas, se suelen aplicar técnicas de filtrado, promediado de medidas y validación geométrica, garantizando mayor estabilidad en el cálculo de las posiciones.

3.2. APRENDIZAJE COMPUTACIONAL PARA CLASIFICACIÓN

3.2.1. FUNDAMENTOS DEL APRENDIZAJE COMPUTACIONAL

El aprendizaje computacional busca que un sistema sea capaz de reconocer patrones a partir de datos de entrada, generalizando después sobre ejemplos nuevos. En el contexto de este trabajo, se trata de que un modelo aprenda a identificar distintos tipos de residuos domésticos en imágenes capturadas por la cámara (o en el video obtenido de la transmisión). La tarea se formula como un problema de clasificación supervisada, en el que a cada imagen región se le asigna una etiqueta perteneciente a un conjunto discreto de clases (cartón, vidrio, plástico, etc.)

El proceso implica varias etapas: adquisición de datos, anotación manual, diseño de la arquitectura del modelo, entrenamiento, validación y evaluación. El reto principal reside en que el modelo debe ser capaz de distinguir entre objetos con gran variabilidad

y, al mismo tiempo, evitar confundir clases con gran similitud visual (por ejemplo, una botella de plástico y una botella de vidrio transparentes).

Dentro de los paradigmas de aprendizaje, el supervisado es el núcleo de esta propuesta, aunque se benefician técnicas como el aprendizaje por transferencia, que permite partir modelos previamente entrenados en grandes bases de datos para adaptarlos a un dominio más específico con menos recursos de entrenamiento.

3.2.2. Redes neuronales convolucionales y su papel en la visión por computador

Las redes neuronales convolucionales (CNNs) constituyen la base teórica y práctica de los sistemas modernos de clasificación y detección de objetos. A diferencia de las redes totalmente conectadas, que tratan la entrada como un vector plano, las CNNs explotan la estructura espacial de las imágenes mediante filtros convolucionales que recorren toda la matriz de píxeles. Estos filtros detectan características locales como bordes, texturas y patrones de color, que se combinan en capas sucesivas para construir representaciones cada vez más abstractas y complejas [18]. Se muestra gráficamente la arquitectura de una red neuronal convolucional y sus capas:

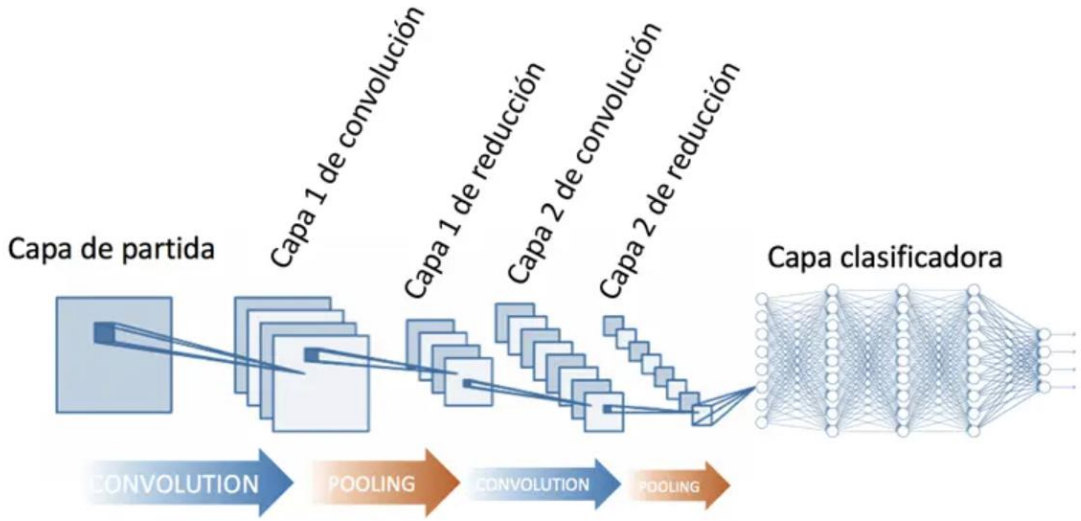

Figura 3.2. Arquitectura de una red neuronal convolucional.

La principal ventaja de este enfoque es la invariancia traslacional, es decir, la capacidad del modelo de reconocer un objeto independientemente de su ubicación en la imagen. A lo largo de los años, las arquitecturas de CNN han evolucionado incorporando técnicas como *pooling*, normalización, conexiones residuales y módulos de atención, lo que ha permitido aumentar la profundidad y complejidad de los modelos manteniendo un entrenamiento estable.

En la práctica, estas redes no sólo clasifican imágenes completas, sino que sirven de base para sistemas de detección como YOLO, en los cuales una misma arquitectura es capaz de proponer regiones de interés y clasificarlas simultáneamente.

3.2.3. De la clasificación a la detección de objetos

El paso de la clasificación de imágenes a la detección de objetos supone un salto conceptual. Mientras que en la primera se asigna una única etiqueta a toda la imagen, en la detección es necesario reconocer múltiples objetos en distintas posiciones y escalas. Para ello, además de la etiqueta de la clase, el modelo debe predecir coordenadas de cajas delimitadoras que indiquen la localización del objeto.

Este enfoque combina dos tareas: clasificación y regresión. La clasificación se refiere a predecir qué tipo de objeto está presente, mientras que la regresión predice la posición y dimensiones de la caja. Los modelos modernos implementan pérdidas compuestas que equilibran ambas tareas y emplean técnicas como la supresión de no-máximos para descartar predicciones redundantes.

En el contexto del proyecto, este aspecto es fundamental, ya que no basta con saber que hay un determinado tipo de material presente en la imagen: el robot necesita conocer dónde se encuentra cada objeto para poder planificar un movimiento de agarre. Así, la detección se convierte en la herramienta central que conecta visión con acción robótica.

3.2.4. Preparación de datos y entrenamiento

El éxito de un modelo de detección depende en gran medida de la calidad del conjunto de datos utilizado. En el caso de la clasificación de residuos, es necesario disponer de imágenes variadas que reflejen la diversidad de condiciones en las que estos objetos aparecen: diferentes niveles de iluminación, deformaciones, suciedad, variedad de fondos y grados de desgaste.

La preparación de datos incluye procesos de anotación, donde cada objeto se etiqueta cuidadosamente con su clase correspondiente y su caja delimitadora, así como técnicas de aumento de datos (*data augmentation*). Estas últimas son esenciales para mejorar la capacidad de generalización del modelo e incluyen transformaciones geométricas (rotaciones, escalados, recortes), fotométricas (ajustes de brillo, contraste, ruido) y combinatorias (mezcla de imágenes o recortes).

El entrenamiento se realiza ajustando los parámetros de la red mediante un algoritmo de optimización, normalmente basado en gradiente descendente estocástico.

Se monitorizan métricas de validación para evitar sobreajuste, y se emplean estrategias como parada temprana, regularización y normalización para garantizar un rendimiento robusto. En este tipo de sistemas, la evaluación no se limita a la precisión global, sino que incorpora métricas como precisión por clase, *recall* y mAP (*mean Average Precision*), que permiten valorar tanto la clasificación como la localización.

3.2.5. Integración en sistemas robóticos

En un sistema robótico, los resultados del modelo de detección deben integrarse en un flujo de decisión que conecta la percepción con la acción. Las detecciones 2D en la imagen se combinan con la información de profundidad de la cámara para estimar coordenadas en el espacio 3D del objeto. Posteriormente, dichas coordenadas se transforman al sistema de referencia del robot, lo que le permite ejecutar movimientos de aproximación y agarre.

Es importante considerar que la visión por computador no opera en condiciones ideales: las detecciones pueden tener ruido, incertidumbre o falsos positivos. Para contrarrestarlo, se aplican técnicas como el promediado de predicciones a lo lardo de varios fotogramas o el uso de filtros que estabilicen las posiciones estimadas. Asimismo, se definen umbrales de confianza mínimos para decidir si una detección es suficientemente fiable como para que el robot actúe.

La integración exitosa del aprendizaje computacional en un sistema robótico depende, por lo tanto, no sólo de la precisión del modelo, sino también de la capacidad de combinarlo con estrategias de control y de seguridad. De esta forma, el aprendizaje computacional se convierte en el puente entre el entorno visual y la ejecución física de tareas en el espacio.

3.3. Transformaciones de coordenadas (homogéneas)

3.3.1. Concepto de sistemas de referencia en robótica

En robótica, cada componente del sistema posee su propio marco de referencia: la base del robot, el actuador final, la cámara, los objetos en el entorno, e incluso los marcadores de calibración como los *Apriltags*. Para que un manipulador robótico pueda interactuar correctamente con el entorno, es imprescindible que todas las posiciones y orientaciones estén expresadas en un mismo sistema de referencia común.

Sin embargo, como cada elemento del sistema "observa" el mundo desde su propia perspectiva, es necesario disponer de una herramienta matemática que permita transformar coordenadas de un marco a otro. Esta es la función de las transformaciones de coordenadas, que constituyen el lenguaje común para comunicar visión, planificación y control.

3.3.2. TRANSFORMACIONES HOMOGÉNEAS EN 3D

Una transformación homogénea es una representación matricial que combina rotación y traslación en un único bloque matemático. En el espacio tridimensional, se utiliza una matriz de 4x4 que actúa sobre vectores de coordenadas extendidos (x, y, z, 1). La estructura típica tiene la siguiente forma:

$$T = \begin{bmatrix} R & t \\ 0 & 1 \end{bmatrix} \quad (1)$$

donde ***R*** es una matriz de rotación 3x3 y ***t*** un vector de traslación 3x1.

Esta formulación permite aplicar, en una sola operación matricial, tanto un cambio de orientación como un desplazamiento espacial. Además, las transformaciones pueden encadenarse mediante multiplicación de matrices, lo que hace posible describir trayectorias o concatenar distintos sistemas de referencia [19][20], por ejemplo: cámara > marcador *Apriltag* > base del robot.

Entre las ventajas del uso de coordenadas homogéneas, se encuentra la facilidad de invertir transformaciones, esto es, cambiar de un sistema a otro y viceversa teniendo una de las matrices de transformación de uno de los dos elementos que precisan relacionar.

En este proyecto, el uso de transformaciones homogéneas es fundamental, ya que proporciona el camino para que la información obtenida por la cámara de coordenadas en el espacio tridimensional se pueda traducir en valores de entrada para el movimiento del brazo robótico.

3.4. CINEMÁTICA DIRECTA E INVERSA DEL BRAZO ROBÓTICO

La cinemática, en el contexto del área de la robótica, estudia el movimiento de los manipuladores en función de las relaciones geométricas de sus componentes, sin considerar las fuerzas que lo generan. Se centra en describir cómo los desplazamientos

en las articulaciones de un robot se traducen en movimientos del actuador final (una pinza, por ejemplo) en el espacio cartesiano.

Existen dos conceptos fundamentales a considerar en este ámbito: la cinemática directa y la cinemática inversa.

1. La cinemática directa consiste en calcular la posición y orientación del actuador final a partir de los ángulos articulares conocidos. Es un problema determinista y se resuelve aplicando transformaciones geométricas a lo largo de la cadena cinemática del robot.

2. La cinemática inversa, por el contrario, busca determinar qué valores deben adoptar las articulaciones para que el efector final alcance una posición y orientación deseadas. Este problema es más complejo, ya que puede tener múltiples soluciones, ninguna solución o verse restringido por los límites físicos de las articulaciones y del espacio que rodea al robot.

El estudio de la cinemática constituye la base matemática para cualquier aplicación de manipulación por robótica, conecta las órdenes de control con los movimientos reales del robot [21][22].

3.4.1. CINEMÁTICA DIRECTA DEL UR3E

El cálculo de la cinemática directa en un robot manipulador de tipo serial, como el UR3e, se realiza a partir de los parámetros geométricos de sus eslabones y articulaciones. Para formalizar estos cálculos, se emplea comúnmente la notación de Denavit-Hartenberg (DH), un método sistemático que asigna marcos de referencia a cada articulación y describe sus relaciones mediante cuatro parámetros:

- Ángulo de la articulación (para articulaciones rotacionales).
- Desplazamiento a lo largo del eje de la articulación.
- Longitud del eslabón.
- Ángulo entre ejes consecutivos.

Con estos parámetros se construyen matrices de transformación homogénea que describen el cambio de referencia entre cada par de eslabones. El producto sucesivo de estas matrices genera la transformación global desde la base del robot hasta el actuador final designado.

En términos generales, la cinemática directa proporciona la pose (x, y, z, R) del actuador, donde (x, y, z) representa la posición cartesiana y R es una matriz de rotación que describe la orientación. Esta representación puede también expresarse en otros formatos, como ángulos de Euler o cuaterniones, dependiendo de la aplicación.

En el caso de robots industriales con seis grados de libertad, como el UR3e, la cinemática directa es siempre un problema bien definido y tiene una única solución para un conjunto dado de ángulos articulares.

3.4.2. Cinemática inversa del UR3e

La cinemática inversa plantea el problema a la inversa que el anterior: dado un punto y orientación deseados del actuador final, determinar los valores de las articulaciones que permiten alcanzarlo. A diferencia de la cinemática directa, la inversa es un problema más complejo debido a los siguientes factores:

1. **Múltiples soluciones:** un robot de seis grados de libertad puede tener varias configuraciones distintas de articulaciones que lo lleven a la misma posición final. Esto se traduce en posturas alternativas, como codos arriba o codos abajo.
2. **Ausencia de solución:** en algunos casos, la pose deseada puede encontrarse fuera del espacio de trabajo del robot, lo que hace imposible alcanzarla.
3. **Restricciones físicas:** los límites mecánicos de las articulaciones restringen el conjunto de soluciones factibles.
4. **Problemas de singularidades:** ocurren cuando diferentes configuraciones producen el mismo efecto cinemático, provocando pérdida de grados de libertad efectivos.

Para abordar estos retos, se emplean diversos métodos matemáticos de resolución, tales como los métodos geométricos que se apoyan en relaciones trigonométricas y propiedades geométricas, o los métodos algebraicos para obtener soluciones a un sistema de ecuaciones polinómicas. Sin embargo, para abordar problemas más generales que no siempre tienen una solución analítica exacta, se prefiere el uso de métodos numéricos que aproximan la solución mediante algoritmos iterativos como el descenso de gradiente o el método de Newton-Raphson.

En robots como el UR3e, se dispone tanto de soluciones analíticas predefinidas como de algoritmos numéricos, lo que permite encontrar configuraciones de forma eficiente y adaptada a cada situación.

3.4.3. RELEVANCIA PARA EL CONTROL DE MANIPULADORES ROBÓTICOS

La importancia de la cinemática directa e inversa en un proyecto de robótica radica en que constituyen la base para el control y la planificación de movimientos. Gracias a la cinemática directa, se puede predecir dónde se encuentra el actuador del robot en un instante dado, a partir de los ángulos de las articulaciones. Con la cinemática inversa, en cambio, se puede planificar cómo debe moverse el robot para alcanzar un punto concreto en espacio de trabajo.

En aplicaciones de clasificación de objetos, como la presente, la cinemática inversa es la herramienta que permite transformar las posiciones calculadas por el sistema de visión en instrucciones de movimiento para el robot. Una vez conocido el objetivo en coordenadas cartesianas, la cinemática inversa calcula los ángulos de las juntas que deben aplicarse para el actuador se ubique en la posición correcta.

Así, la cinemática no sólo es un fundamento teórico, sino el vínculo indispensable entre la percepción del entorno y la acción física del robot. Sin este marco matemático, sería imposible garantizar que el robot se mueva de manera precisa, segura y coordinada en tareas de manipulación de objetos.

Capítulo 4. DESARROLLO DEL SISTEMA

4.1. DISEÑO GENERAL DEL SISTEMA

El sistema desarrollado se ha concebido bajo un enfoque modular, en el que cada componente cumple una función diferenciada y puede ser modificado o ampliado sin afectar al resto. Esta estructura responde a la necesidad de integrar múltiples tecnologías (visión por computador, transformaciones matemáticas y control robótico) dentro de un flujo coherente y funcional. Los componentes principales del sistema son:

1. Robot UR3e, incluyendo la consola de programación (*Tablet*) y la pinza acoplada.
2. Cámara OAK-D Lite.
3. Ordenador portátil personal (AMD RYZEN 7 6800h 3.20 GHz, NVIDIA GeForce RTX 3060 6 GB, 16 GB RAM, 1 TB SSD). Sistema operativo Ubuntu 24.04.2 LTS utilizado para la ejecución del programa escrito en lenguaje de programación *Python* 3.10.11.

El entorno de trabajo en el que se desarrollaron las actividades de detección y pruebas de validación es el del laboratorio de robótica de la Escuela Técnica Superior de Ingeniería y Diseño Industrial de la Universidad Politécnica de Madrid.

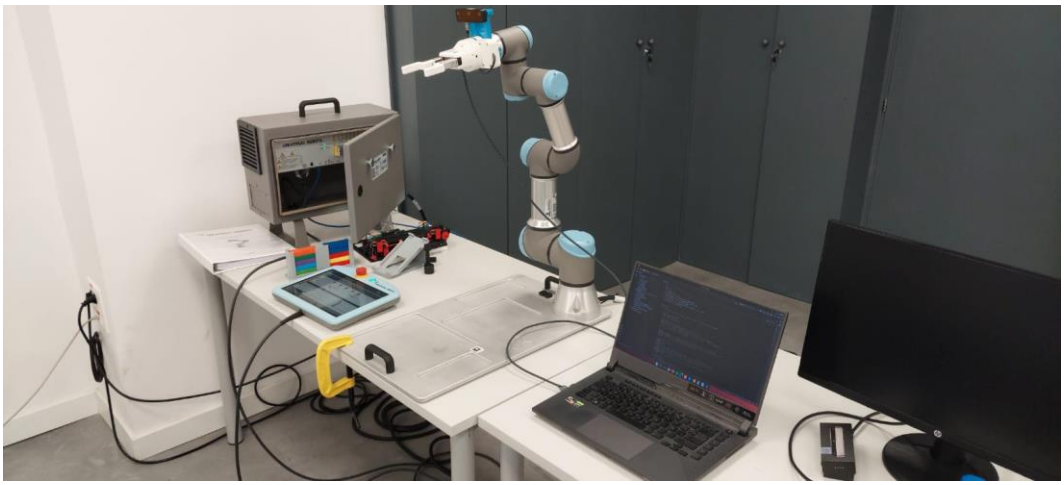

Figura 4.1. Entorno de trabajo del sistema en el laboratorio.

En líneas generales, el sistema (programado en *Python*) se puede dividir en tres bloques principales:

1. **Visión por computador:** encargado de adquirir información visual mediante la cámara OAK-D Lite y procesarla para detectar objetos de interés. Aquí intervienen dos módulos clave:
    1. Uno que implementa un detector basado en aprendizaje profundo para reconocer residuos domésticos y clasificarlos en diferentes categorías.
    2. Otro que permite localizar marcadores (*Apriltags*) en la escena observada por la cámara, proporcionando un punto de referencia espacial necesario para la calibración.
2. **Estimación de poses y transformaciones:** dedicado a convertir las coordenadas obtenidas en el sistema de referencia de la cámara hacia el sistema de referencia del robot. Se presenta un módulo que gestiona las matrices homogéneas que representan rotaciones y traslaciones, encadenando los cambios de marco desde la cámara hasta la base del robot.
3. **Control del brazo:** responsable de la comunicación y el envío de órdenes al manipulador del robot UR3e. En este módulo, se encapsulan las funciones de conexión mediante RTDE, ejecución de movimientos básicos hacia las coordenadas calculadas previamente, y dejando la

gestión de la pinza del robot en manos de la programación hecha a través de la consola de programación del robot.

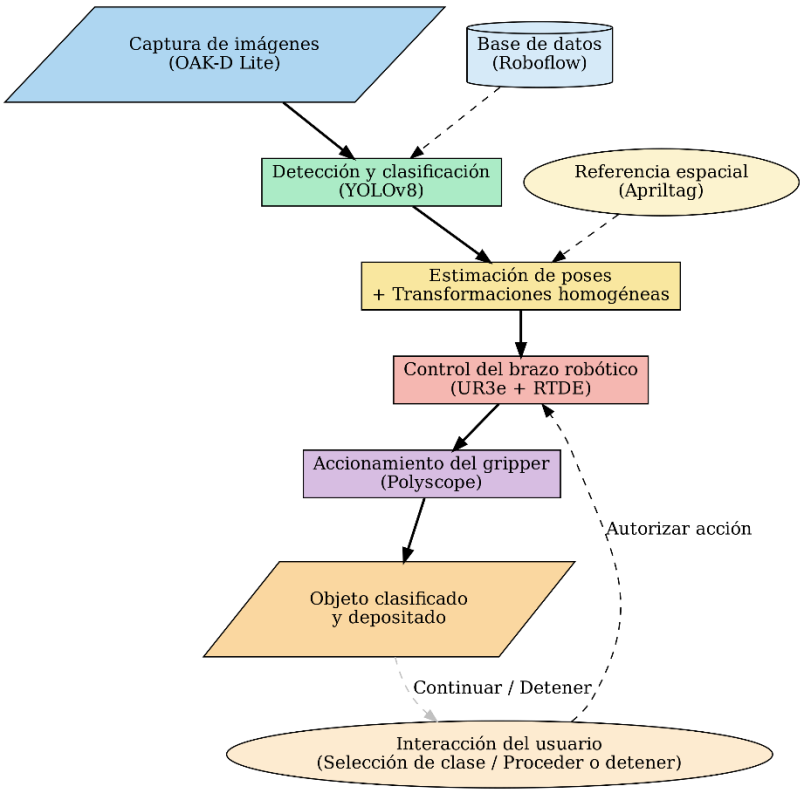

Figura 4.2. Diagrama de flujo del sistema de clasificación de residuos.

Además de estos bloques principales, se desarrollaron utilidades complementarias: un módulo de recolección de datos que facilita la captura de imágenes que se pueden utilizar en el entrenamiento del modelo, mientras que se otro módulo principal (*main*) se encarga de actuar como núcleo de integración, gestionando la interacción secuencial de todos los módulos en la ejecución del sistema completo.

Este diseño modular garantiza que cada componente pueda evolucionar de manera independiente: el detector visual puede ser sustituido por otra arquitectura sin modificar el control del robot, y las rutinas de movimiento pueden adaptarse a otro manipulador conservando la misma lógica de visón y transformación de coordenadas.

4.2. Sistema de visión

El subsistema de visión constituye el núcleo perceptivo del proyecto, al ser el encargado de identificar los residuos presentes en la escena y proporcionar la información espacial necesaria para su manipulación robótica. Para ello se integraron tanto técnicas de aprendizaje profundo para la detección y clasificación de objetos

como algoritmos de detección de marcadores visuales que permiten establecer referencias geométricas. Desde el punto de vista físico, el sistema de visión es posible gracias a la cámara OAK-D Lite de Luxonis, que se muestra a continuación indicando la posición de sus tres cámaras (una de color RGB y dos esteréo):

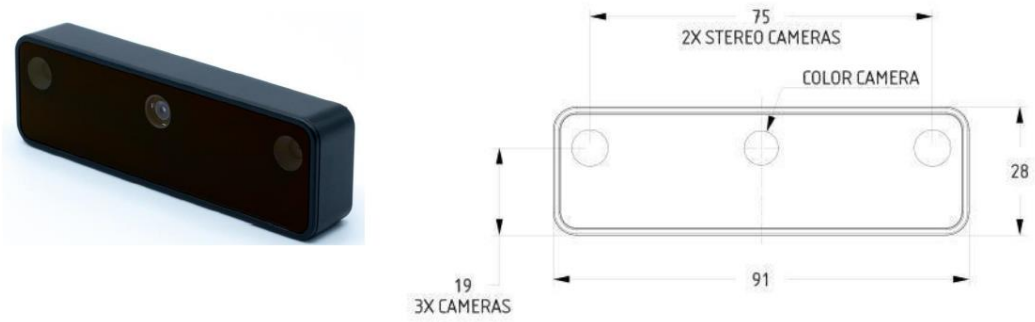

Figura 4.3. Vista de la OAK-D Lite de Luxonis y la posición de sus cámaras integradas [23].

4.2.1. ADQUISICIÓN DE IMÁGENES Y EQUIPO EMPLEADO

El sistema se apoya en la cámara OAK-D Lite de Luxonis [23], un dispositivo que combina un sensor RGB de alta resolución con cámaras estéreo para la estimación de profundidad. Este *hardware* permite obtener simultáneamente imágenes a color y mapas de disparidad, lo que facilita la reconstrucción 3D de la escena y la localización espacial de los objetos detectados.

Durante la fase de entrenamiento y pruebas preliminares, además de la cámara del sistema final, se emplearon bases de datos externas con imágenes representativas de residuos, asegurando la variabilidad en condiciones de iluminación, perspectiva y forma de objetos.

4.2.2. ENTRENAMIENTO DEL MODELO DE DETECCIÓN DE OBJETOS

Para la clasificación automática de residuos se seleccionó la arquitectura YOLOv8 [11], una de las versionas más avanzadas de la familia *You Only Look Once*, optimizada para aplicaciones en tiempo real. La elección se fundamenta en su equilibrio entre precisión y velocidad de inferencia, cualidades de gran importancia en el funcionamiento del sistema desarrollado en este proyecto.

Antes de llevar a cabo el reentrenamiento, se tomó la decisión de formar dos propuestas, diferenciadas por las clases que la constituyen para clasificar por tipo de objetos:

1. La primera, con tres clases: “Carton”, “Envases” y “Vidrio”.
2. La segunda, con cuatro clases: “Carton”, “Latas”, “Plastico” y “Vidrio”.

La idea, detrás de proponer estas dos opciones, es la de separar la clase de objetos que se clasificarían como “Envases” en dos más diferenciadas en cuanto al material que las compone.

El reentrenamiento del modelo se llevó a cabo usando la plataforma de Google Colab, aprovechando los recursos de la tarjeta gráfica GPU A100 disponible. Para este fin, se recurrió a Roboflow Universe [24], un repositorio abierto de bases de datos etiquetados. Se exploraron inicialmente diferentes bases de datos de tamaño reducido, con el fin de validar la arquitectura y observar el comportamiento de cada modelo. Posteriormente se incrementó de manera progresiva la cantidad de datos empleados hasta alcanzar una base de datos compuesta por decenas de miles de elementos etiquetados, que permitió obtener los mejores resultados en términos de exactitud y estabilidad en las predicciones.

El proceso incluyó técnicas de *data augmentation*, tales como rotaciones, cambios de escala, ajustes de brillo y contraste, con el objetivo de robustecer el modelo frente a variaciones habituales en la manipulación de residuos (objetos arrugados, sucios o que aparecen parcialmente ocultos). El entrenamiento se monitorizó mediante métricas como la precisión, el *recall* y el mAP, garantizando la validez del modelo antes de su integración en el sistema final.

4.2.3. Implementación del detector

El detector resultante fue encapsulado en un módulo que recibe como entrada imágenes obtenidas por la cámara y devuelve como salida las clases detectadas junto con las coordenadas de las cajas delimitadoras y la confianza asociada. Cabe destacar que, mientras se recibe la información obtenida de imagen por parte de la cámara, también se muestra en video a tiempo real lo que está detectando el programa.

Adicionalmente, dentro del mismo módulo de detección YOLOv8, se integró la capacidad de la cámara OKA-D Lite para estimar las coordenadas en el espacio (x, y, z) de una región de la imagen, es decir, mediante el procesamiento de la información obtenida por las dos cámaras estéreo (izquierda y derecha). Con esta información, se determina una región delimitada por un cuadrado dentro de la imagen llamada ROI (*Region of Interest*), sobre la cual, mediante el uso de la librería del fabricante *depthai* (usada para programar en lenguaje *Python*), se determina la distancia y la ubicación en el plano en el que se encuentran delimitados los píxeles de la imagen de la ROI [25]. Por lo tanto, si se localiza esta región de interés ROI sobre la localización del objeto

detectado por el modelo reentrenado de YOLOv8, se puede obtener la posición relativa del residuo detectado con respecto del marco referencial de la cámara.

En la siguiente figura, se muestra un ejemplo de cómo la ROI (en blanco) se posiciona concéntricamente dentro del recuadro de detección (en verde):

Figura 4.4. Ejemplo de detección de una lata dónde se muestra la posición de la ROI.

De esta forma, el subsistema de visión no sólo clasifica los objetos presentes de la escena, sino que también proporciona a información necesaria para estimar su ubicación tridimensional en combinación con los datos de profundidad de la cámara.

4.2.4. DETECCIÓN DE MARCADORES

Además de la identificación de residuos, se integró la detección de marcadores visuales tipo *Apriltag*, implementada en uno de los módulos de programación. Estos marcadores permiten establecer un sistema de referencia adiciona que actúa como punto de enlace entre la cámara y la base del robot. La detección del marcador ofrece una pose de referencia estable, la cual se emplea posteriormente en las transformaciones homogéneas para garantizar la coherencia espacial de todo el sistema. Se muestra en la siguiente figura la detección del marcador *Apriltag* en condiciones experimentales:

Figura 4.5. Detección del *Apriltag*.

Para este proyecto en particular, se utilizó un marcador tipo *Apriltag* de la familia "36h11" de número identificativo 4 (ID: 4) de geometría cuadrada de lado igual a 15 milímetros pegado a la plataforma de apoyo del robot, ubicado convenientemente para que pueda ser visto por la cámara al mismo tiempo que detecta los objetos presentes. También, se orientó el marcador para que su sistema de coordenadas relativo coincidiera en dirección y sentido con el sistema de coordenadas de la base del robot.

4.3. ESTIMACIÓN DE POSES Y TRANSFORMACIONES

Una vez detectados y clasificados los residuos en el sistema de visión, resulta necesario expresar sus coordenadas en un marco de referencia útil para la manipulación robótica. La cámara OAK-D Lite proporciona posiciones relativas en su propio sistema de coordenadas, pero el robot UR3e requiere conocer dichas posiciones en el marco de su base para ejecutar correctamente las trayectorias de agarre y deposición de los objetos. Esta diferencia motiva la necesidad de un módulo dedicado a la estimación de poses y transformaciones de coordenadas.

4.3.1. FUNDAMENTACIÓN GEOMÉTRICA

Cada dispositivo del sistema (cámara, *Apriltag*, base del robot) posee un marco de referencia propio. Para garantizar la coherencia entre ellos se emplean

transformaciones homogéneas, representadas por matrices 4x4 que unifican rotación y traslación en una sola estructura matemática. Estas matrices permiten describir cómo pasar de un sistema a otro mediante la concatenación de transformaciones intermedias.

En este proyecto, la secuencia geométrica principal se describe de la siguiente manera:

Figura 4.6. Esquema de la secuencia geométrica empleada.

donde cada ***T*** corresponde a una matriz homogénea obtenida por calibración o detección. Así, el marco de referencia con el cuál se mide la posición del objeto pasa de la cámara, al marcador *Apriltag*, y finalmente, al sistema de coordenadas de la base del robot.

4.3.2. USO DE *APRILTAGS* COMO REFERENCIA ESPACIAL

El sistema emplea un marcador *Apriltag* colocado en una posición fija y conocida respecto a la base del robot. La cámara detecta este marcador (mientras está en la misma posición relativa que cuando ha detectado los residuos) y, con el uso del módulo de *Python pupil-apriltags*, se calcula la matriz de rotación y el vector de traslación del marcador en tiempo real, lo que permite definir con precisión la relación espacial cámara-robot. Esta estrategia evita depender únicamente de calibraciones manuales, que suelen acarrear errores de medición e inestabilidad en entornos dinámicos.

El uso de *Apriltags* aporta la ventaja de que el sistema sea más flexible: es posible reconfigurar la escena simplemente cambiando la ubicación del marcador y actualizando la transformación correspondiente.

4.3.3. IMPLEMENTACIÓN DEL SISTEMA

La lógica de transformaciones se implementó en un módulo de cálculos matemáticos que contienes las funciones necesarias para construir las matrices

homogéneas a partir de las matrices de rotación y de los vectores de traslación convenientes:

1. Para el caso de la matriz de transformación T_tag_base, que relaciona la referencia del *Apriltag* con la de la base del robot, se toma en consideración que la ubicación del marcador y su propio sistema de coordenadas coincida en dirección y sentido con el sistema de coordenadas de la base del robot. Esto hace que la matriz de rotación sea la identidad. Para el vector de traslación, se miden las distancias reales que existen en direcciones cartesianas entre la base del robot y el marcador para construir el vector. Esta ubicación es siempre fija y conocida.

2. La matriz de transformación T_cam_tag, se calcula primero con los datos obtenidos de matriz de rotación y vector de traslación del módulo *pupil-apriltags*, que son puestos de forma ordenada en una sola matriz 4x4, y que posteriormente, se calcula su inversa para obtener la matriz de transformación que traslada el punto de referencia desde la cámara hasta el marcador. La inversa se calcula debido a que la información obtenida de la medición por cámara y el cálculo hecho por el módulo permite construir la matriz de transformación que traslada las posiciones relativas con respecto del marcador a la cámara, cuando lo que se quiere es lo contrario: que las posiciones medidas relativas a la cámara se transformen en posiciones relativas al marcador.

El flujo de trabajo consiste en tomar las coordenadas 3D de un objeto detectado por la red neuronal reentrenada en el sistema de la cámara, transformarlas al marco del *Apriltag* y, finalmente, obtener su posición relativa a la base del robot; esto ocurre, en la práctica, haciendo el cálculo matemático de multiplicar las matrices de transformación por el vector de posición relativo del objeto detectado, que se muestra en la ecuación:

$$P_{obj\_base} = T_{tag\_base} \times T_{cam\_tag} \times P_{obj\_cam} \qquad (2)$$

donde $\boldsymbol{P_{obj\_cam}}$ es el vector posición 4x1 (de coordenadas x, y, z, 1) del objeto detectado por la cámara en relación con el sistema de coordenadas de la cámara, $\boldsymbol{T_{cam\_tag}}$ y $\boldsymbol{T_{tag\_base}}$ son las matrices de transformación homogénea que permiten expresar la posición relativa del objeto detectado por la cámara en función del marcador *Apriltag* y, luego, en función del marco de referencia de la base del robot, respectivamente. Finalmente, $\boldsymbol{P_{obj\_base}}$ es el vector posición obtenido del cálculo que describe la posición en el espacio del objeto detectado según el sistema de coordenadas de la base del robot.

Visualmente, si se lee la ecuación (2) de derecha a izquierda, se observa cómo la posición relativa (a la cámara) obtenida de la detección se convierte, en primer lugar, al sistema de referencia del marcador *Apriltag*, y, en segundo lugar, al sistema de coordenadas de la base del robot, que es el objetivo de este cálculo. De esta forma, la pinza del UR3e se dirige siempre a objetivos expresados en el marco de referencia de la base del robot.

4.3.4. VALIDACIÓN DE LAS TRANSFORMACIONES

Para comprobar la coherencia de las transformaciones, se llevaron a cabo pruebas de consistencia espacial. Estas incluyeron verificar que objetos situados en posiciones conocidas fueran correctamente transformados al sistema de la base del robot. Esto se hizo por partes, primero, verificar que lo medido por el marcador pudiera ser traslado al marco referencial de la base del robot, segundo, verificar que lo medido por la cámara se pudiera comprobar con la transformación en base al sistema de coordenadas del marcador *Apriltag*, y finalmente, comprobar la consistencia de las dos transformaciones al concatenar ambas y verificar que el resultado obtenido correspondía con la posición conocida con respecto del sistema referencial de la base del robot.

4.4. CONTROL DEL BRAZO ROBÓTICO

El controlador del manipulador constituye el componente final del sistema, al ser el encargado de ejecutar físicamente la clasificación de los residuos una vez que han sido detectados y localizados. En este proyecto, el brazo colaborativo UR3e de Universal Robots se controló de manera híbrida: los movimientos generales y la planificación espacial se ejecutaron mediante *Python* utilizando el protocolo RTDE, mientras que el accionamiento de la pinza se gestionó de forma independiente a través de un programa predefinido en el entorno *Polyscope*, interfaz de usuario de la consola de programación del propio robot.

4.4.1. COMUNICACIÓN Y CONTROL DE MOVIMIENTOS

La comunicación entre el ordenador y el UR3e se estableció mediante la librería rtde_*control*, que permite enviar en tiempo real órdenes de movimiento y recibir estados del robot. Para encapsular la lógica de control se diseñó un módulo en el que se implementaron funciones básicas como:

1. Posición inicial: colocar al robot (la pinza) a un estado seguro desde el cual iniciar cualquier tarea.
2. Descenso: aproximar la pinza hacia las coordenadas del objeto detectado o hacia el sitio de deposición de residuos
3. Ascenso: luego de ejecutar la acción de agarre (o de liberación) del objeto, efectuar una subida de la pinza.
4. Desconexión: finalizar la comunicación con el robot de manera controlada.

Estas rutinas se estructuraron en la clase *RobotMotion*, lo que favorece la claridad y la reutilización del código. En cada ciclo de clasificación, el programa de *Python* se encarga de calcular la ubicación del objetivo a partir de las transformaciones descritas en la sección 4.3, y de enviar al UR3e las órdenes de movimiento correspondientes para alcanzar esa posición.

La gestión para comunicar al robot la posición a la cual debe moverse y planificar los movimientos viene dada dentro del programa *main* desde el ordenador: dónde, luego de finalizada la detección de los objetos presentes y la obtención de las ubicaciones con respecto del marco referencia de la base del robot, se le muestra al usuario las clases de objetos detectadas y se le pide que comience el proceso de clasificación seleccionando la clase que desea separar. Seguidamente, el robot empieza a recolectar cada objeto (uno a uno) de esa clase seleccionada hasta que no quede ningún residuo de ese tipo de clase; luego pide de nuevo al usuario que seleccione otra clase (si la hubiera), y si no queda ninguna clase para clasificar, el programa termina y el robot queda en su posición inicial (la misma de cuándo estaba en el proceso de detección). Cabe destacar que, si así lo desea el usuario, cada vez que se le pide instrucciones sobre cuál clase recolectar, también tiene la opción de detener totalmente el proceso.

4.4.2. Accionamiento de la pinza desde *Polyscope*

A diferencia de los movimientos, el accionamiento de la pinza no se integró en el control de *Python*, sino que se ejecutó desde un programa independiente en *Polyscope*. Esto se debe a que la pinza utilizada ya está configurada para ser controlada directamente desde la consola de programación del robot, lo que simplificó la integración al evitar dependencias adiciones en el software externo. La lógica seguida se describe a continuación:

1. El programa en *Python* envía al robot a la posición de descenso sobre el objeto.

2. Se cierra la comunicación con la función de desconexión.
3. El proceso sigue desde la consola de programación del robot y ejecuta la acción de cerrar la pinza, acompañado de un tiempo de espera que asegura la correcta sujeción y un margen de seguridad para que la comunicación se retome desde el programa externo.
4. Se establece la comunicación desde el programa en *Python*, que ordena el ascenso del robot.
5. El robot desciende mientras se desplaza a la posición de descarga designada.
6. De nuevo, se cierra la comunicación, y el programa en *Polyscope* abre la pinza para soltar el objeto.
7. Finalmente, la pinza retorna a la posición inicial, y el robot está listo para comenzar de nuevo el ciclo.

Este mecanismo, repetido en bucle, permite clasificar de manera secuencial los distintos residuos presentes en el área de trabajo.

4.4.3. AMPLIACIÓN DE LA PINZA

Aunque la pinza empleada (modelo HRC-03-118505, de la empresa Zimmer [26]) estaba integrado con el robot UR3e, se identificó una limitación importante en su rango de apertura. En su configuración original, las mordazas de la pinza permitían sujetar objetos de un ancho comprendido entre 13 y 34 mm, lo cual resultaba insuficiente para la mayoría de los objetos de prueba empleados.

Con el fin de ampliar este rango de trabajo, se diseñaron e implementaron adaptadores impresos en 3D que sustituyen a las piezas originales de contacto. El diseño CAD de estos adaptadores se hizo en el *software* Autodesk Inventor, cuidando que las nuevas geometrías se ajustaran a la de la pinza y sin comprometer la funcionalidad mecánica del agarre. Para asegurar un diseño preciso, se importó el modelo CAD existente de la pinza y sobre ese modelo se diseñó el par de adaptadores.

Las piezas de adaptación impresas fueron fijadas en lugar de los pequeños componentes plásticos de color negro ubicados al final de la pinza, ampliando el rango operativo entre los 55 y 76 milímetros. Gracias a esta modificación, el sistema es capaz de manipular los objetos (residuos) de prueba contemplados para este proyecto. En la figura a continuación se aprecia la comparación sin y con la ampliación:

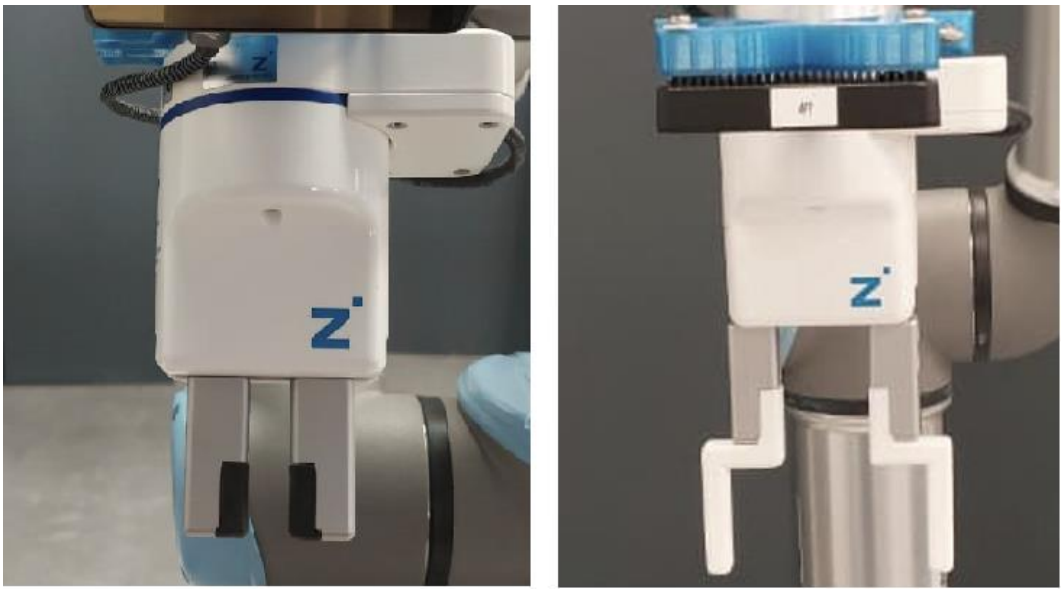

Figura 4.7. Comparación de la pinza sin ampliación (izquierda) y con ampliación (derecha).

4.5. INTEGRACIÓN DEL SISTEMA COMPLETO

El desarrollo de los distintos módulos del sistema culmina en la fase de integración, en la que se establece un flujo de trabajo unificado que conecta visión, transformaciones de coordenadas y control del manipulador robótico.

4.5.1. FLUJO DE PROCESAMIENTO

El ciclo de operación (Figura 4.2) puede describirse como una secuencia de pasos encadenados:

1. Captura de imágenes: la cámara OAK-D Lite obtiene datos visuales y de profundidad de la escena.
2. Detección de residuos: el módulo de detección procesa la imagen para identificar los objetos presentes, asignándoles clases y cajas delimitadoras.
3. Estimación tridimensional: se calculan las coordenadas 3D de cada objeto a partir de la información estéreo de la cámara.

4. Transformaciones de referencia: las coordenadas del objeto se convierten desde el marco de referencia de la cámara al marco de la base del robot, utilizando la detección del marcador *Apriltag* como referencia.

5. Planificación de movimientos: el programa (desde el ordenador) envía al UR3e las órdenes necesarias para posicionar la pinza sobre el objeto, utilizando las rutinas del módulo de control del robot.

6. Accionamiento de la pinza: el programa en *Polyscope* toma el control en los momentos clave para cerrar o abrir la pinza.

7. Deposición en la zona de descarga: el robot transporta el objeto a la posición de destino, donde se suelta siguiendo la misma coordinación *Python-Polyscope*.

8. Retorno a la posición inicial: se regresa la pinza a su ubicación inicial de detección, y el robot está listo para iniciar un nuevo ciclo de clasificación o de detección.

Este proceso se repite iterativamente desde el punto 5 al punto 8 hasta clasificar todos los objetos detectados de una clase determinada en el área de trabajo, según decida el usuario cuál clase clasificar primero o también decidir si detener el proceso.

4.5.2. SINCRONIZACIÓN Y MODULARIDAD

La integración se diseñó bajo un esquema modular en el que cada componente se ejecuta de manera independiente, pero se comunica a través de interfaces bien definidas. De esta forma, el sistema mantiene flexibilidad para futuras modificaciones:

1. El detector visual (la cámara) puede ser reemplazado por otra arquitectura más avanzada sin afectar al control del robot.

2. Las rutinas de movimiento del UR3e pueden adaptarse a otro manipulador con cambios mínimos en la lógica de transformaciones.

3. El programa en *Polyscope* puede ampliarse o sustituirse según el tipo de pinza disponible, sin que esto afecte el esquema de detección y comunicación previamente diseñado.

La sincronización entre el programa externo (escrito en lenguaje *Python*) y la consola de programación del robot se apoya en el uso de interrupciones claras: el programa externo dirige los movimientos y libera la comunicación en el momento

preciso para que la pinza actúe. Esta coordinación asegura que las dos capas de control trabajen en armonía, evitando conflictos y garantizando la repetibilidad de los ciclos.

Capítulo 5. RESULTADOS Y VALIDACIÓN

5.1. CONFIGURACIÓN EXPERIMENTAL

Las pruebas de validación se desarrollaron en dos entornos distintos:

1. Entorno doméstico, utilizando una cámara Logitech C920 HD Pro como capturador de vídeo, con iluminación ambiental no controlada y objetos comunes de uso cotidiano. En este entorno se ejecutaron las primeras pruebas para validar la confiabilidad del modelo de forma sencilla y rápida.
2. Entorno de laboratorio, utilizando la cámara OAK-D Lite de Luxonis, en un espacio más controlado, excepto por la iluminación que no tuvo cambios significativos con respecto del entorno doméstico, y con condiciones de prueba cercanas al sistema final.

En las fases iniciales de validación, los experimentos se centraron en presentar un único objeto por vez al sistema, con el fin de verificar que el modelo entrenado fuese

capaz de detectar y clasificar residuos individuales con una alta confianza. Posteriormente, se llevaron a cabo pruebas más exigentes, presentando varios objetos de forma simultánea, lo que permitió evaluar la robustez del modelo frente a situaciones de mayor complejidad y la capacidad del sistema de reconocer correctamente tanto los objetos de interés como el fondo.

Además, se observó que el proceso de detección y clasificación de cada objeto requirió un tiempo medio de entre 75 y 90 segundos, dependiendo de la interacción del usuario con el sistema, ya que este solicitaba confirmaciones para continuar o detener la operación, y del coste computacional de las detecciones mismas.

5.2. Evolución de las bases de datos y entrenamientos

5.2.1. Bases de datos empleadas

El entrenamiento de los modelos YOLOv8l (versión *large*) se llevó a cabo de manera iterativa a lo largo de un período de dos meses del presente año, empleando diferentes conjuntos de datos obtenidos a través de la plataforma *Roboflow Universe*. La base de datos fue creciendo progresivamente hasta alcanzar aproximadamente cuarenta mil elementos en sus versiones finales, es decir, más de cuarenta mil instancias de clases de objetos pues en muchas imágenes se presentaban varios objetos a la vez.

1. **La Propuesta I** se centró en tres clases: “Carton”, “Envases” y “Vidrio”.
2. **La Propuesta II** incluyó cuatro clases: “Carton”, “Latas”, “Plastico” y “Vidrio”.

Las tablas a continuación resumen la evolución de las bases de datos utilizadas en cada versión de ejecución de reentrenamiento:

Tabla 5.1. Evolución del tamaño de las BBDD de la Propuesta I.

PROPUESTA I			
Versión modelo reentrenado	***test***	***train***	***valid***
1A	657	5360	536
1B	848	7099	821

1C	1185	8729	973
1D	2793	33058	4166
1E	2793	33058	4166

Tabla 5.2. Evolución del tamaño de las BBDD de la Propuesta II.

PROPUESTA II			
Versión modelo reentrenado	***test***	***train***	***valid***
2A	848	7092	822
2B	1185	8722	974
2C	2793	33064	4166
2D	2793	33064	4166

En ambas tablas se describe el número de elementos y su distribución en los subconjuntos de carpetas que serán los datos de entrada para el reentrenamiento de YOLOv8. En la práctica estándar de aprendizaje automático, las bases de datos se dividen en tres conjuntos:

1. **Entrenamiento (*train*):** entre el 70-80% de los datos, usado para ajustar los parámetros del modelo.
2. **Validación (*valid*):** entre el 10-20%, usado para prevenir sobreajuste [27].
3. **Prueba (*test*):** entre el 10-15%, usado únicamente para evaluar el rendimiento final de un modelo entrenado y validado [16].

Esta distribución se sugiere de acuerdo con la forma en la que generalmente se distribuyen los datos alojados en grandes plataformas de bases de datos [23].

Un punto destacable es la cantidad de épocas (*epochs*) con la que se entrenaron los modelos. Una época, en este contexto de aprendizaje automático, representa una pasada completa de la base de datos durante el entrenamiento. La elección del número de épocas depende de encontrar un equilibrio entre: entrenamiento insuficiente (*underfitting*), el modelo no aprende patrones relevantes, y sobreajuste (*overfitting*), el modelo memoriza en exceso y pierde capacidad de generalización.

La mayor parte de los entrenamientos se llevaron a cabo con 50 épocas siguiendo las recomendaciones habituales en visión por computador y la documentación de *Ultralytics* [11], que sugieren este valor como suficiente para bases de datos grandes. Sin embargo, en el último caso de cada propuesta se amplió a 75 épocas, con el fin de analizar si una mayor exposición a la base de datos ofrecía mejoras significativas en la detección, evaluando en todo caso la evolución de las métricas de validación para evitar sobreajuste. Este incremento de épocas supuso tiempos de entrenamiento de alrededor de 8 horas, en comparación con su contraparte de igual base de datos que tardó 5 horas (un incremento del 60 %). Cada versión del modelo fue entrenada en Google Colab, utilizando una GPU A100.

5.2.2. Métricas de entrenamiento

Durante los entrenamientos se registraron métricas estándar de rendimiento: precisión, *recall*, *F1-score* y mAP (*mean Average Precision*). Estas métricas fueron monitorizadas a través de las gráficas generadas automáticamente por *Ultralytics* después de cada ejecución de reentrenamiento.

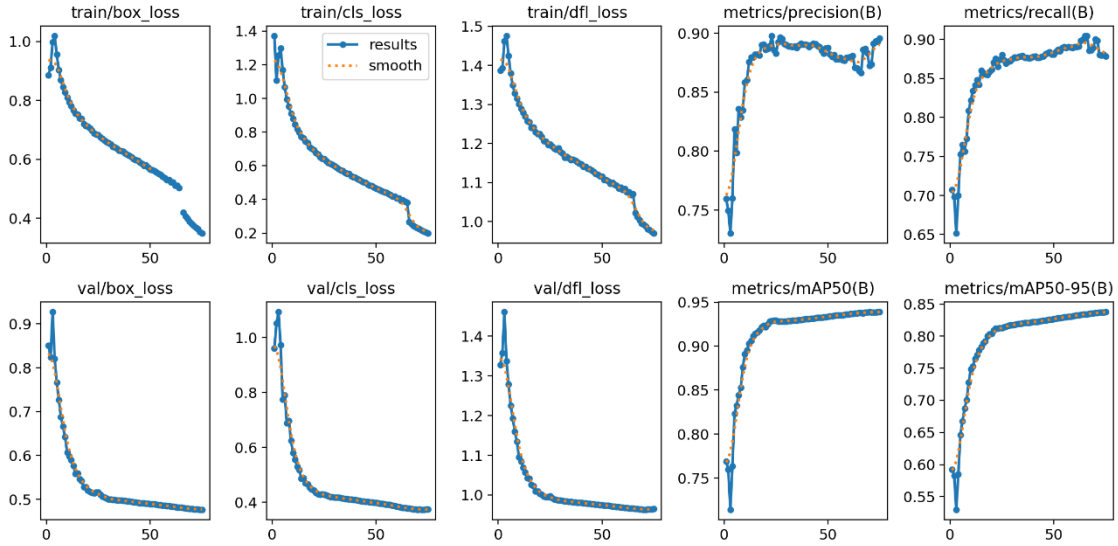

Figura 5.1. Resultados del reentrenamiento de la Propuesta I versión 1E.

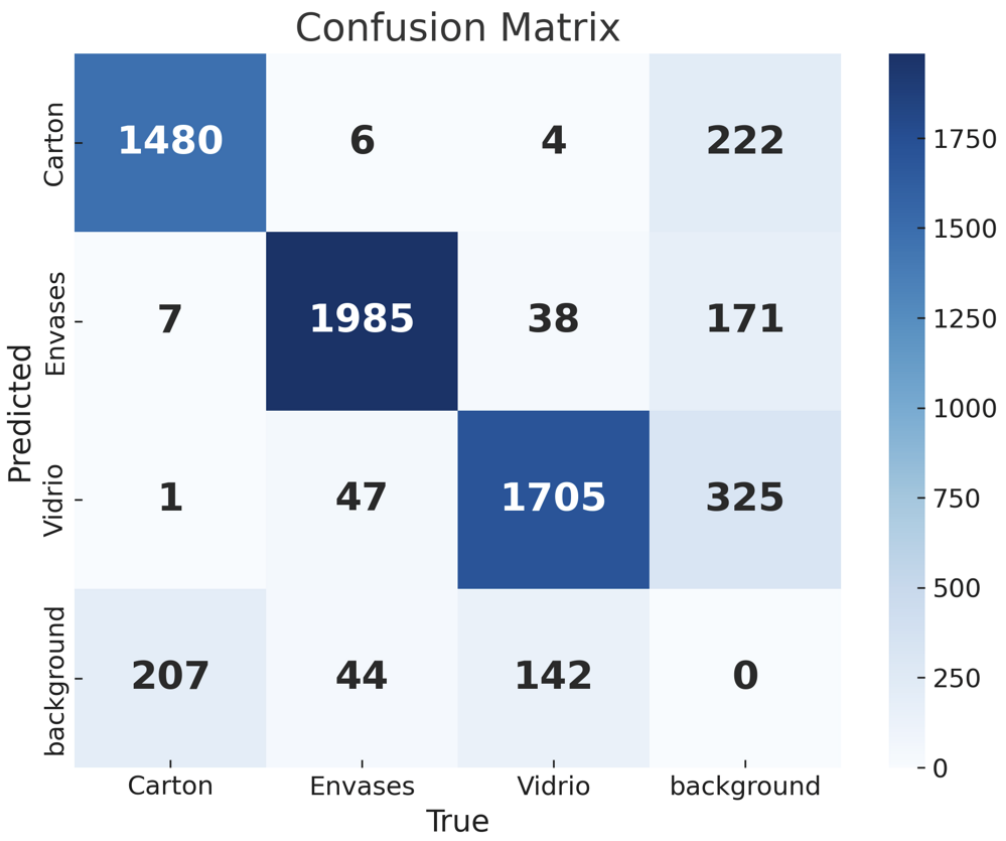

Figura 5.2. Matriz de confusión asociada al reentrenamiento de la versión 1E.

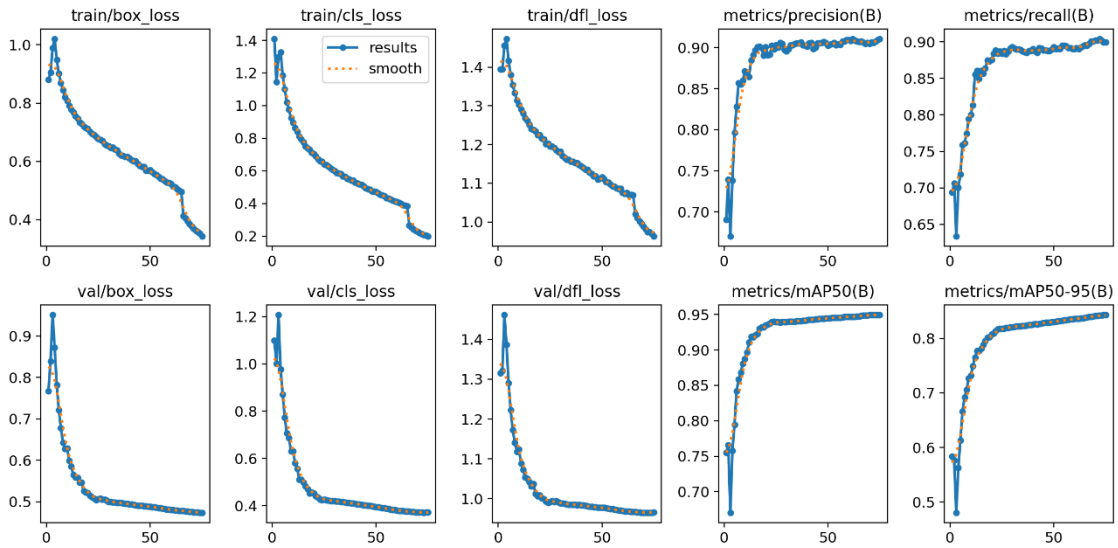

Figura 5.3. Resultados del reentrenamiento de la Propuesta II versión 2D.

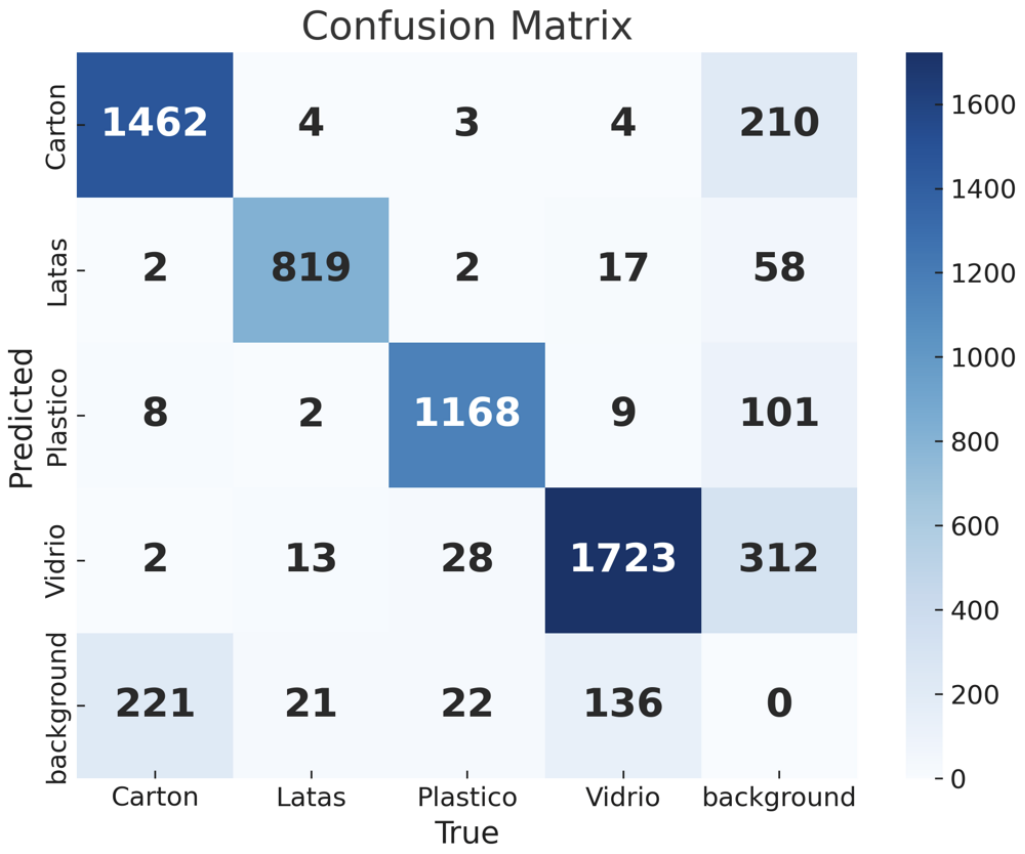

Figura 5.4. Matriz de confusión asociada al reentrenamiento de la versión 2D.

Sobre las Figuras 5.1 y 5.3 (gráficos de los resultados obtenidos del entrenamiento según el número de épocas) se describe su significado [11] siguiendo el orden de izquierda a derecha desde la primera fila superior que:

1. ***train/box_loss:*** mide el error en la predicción de las coordenadas de las cajas delimitadoras. Cuanto más bajo, mejor ajustadas están las cajas a los objetos reales.

2. ***train/cls_loss:*** es la pérdida de clasificación. Evalúa qué tan bien el modelo asigna la clase correcta a cada objeto detectado.

3. ***train/dfl_loss:*** es la pérdida de distribución de distancias (DFL, *Distribution Focal Loss*). Refina la precisión en la localización de los bordes de las cajas.

4. ***metrics/precisión(B)****:* es la precisión y mide qué porcentaje de las detecciones positivas hechas por el modelo son correctas (evita falsos positivos).

5. ***metrics/recall(B):*** es el *recall* y mide qué porcentaje de los objetos reales fueron detectados correctamente (evita falsos negativos).

6. ***val/box_loss:*** mide el error en la predicción de las coordenadas de las cajas delimitadoras, pero evaluado sobre los datos de validación (controla el sobreajuste).

7. ***val/cls_loss:*** es la pérdida de clasificación, aplicado a la validación.

8. ***val/dfl_loss:*** es la pérdida de la distribución de distancias, aplicada a los datos de validación.

9. ***metrics/Map50(B):*** es la media de la precisión promedio (*mean Average Precision*) considerando un umbral IoU de 0,5 (permisivo). Resume globalmente qué tan bien detecta el modelo de objetos.

10. ***metrics/Map50-95(B):*** es la mAP calculado con IoU desde 0,5 hasta 0,95 en pasos de 0,05 (más restrictivo). Es la métrica principal para comparar modelos.

Así pues, se obtiene de los gráficos de resultados del último modelo entrenado para cada propuesta que todas las pérdidas (*loss*) tanto en el entrenamiento como en la validación muestran una tendencia descendente y estable, lo que indica que el modelo aprende correctamente y no hay señales evidentes de sobreajuste.

La precisión y el *recall* se mantienen en valores por encima de 0,85 a lo largo del entrenamiento, lo que significa que el modelo logra un buen equilibrio: pocas falsas detecciones y pocos objetos omitidos.

Los valores de mAP50 superan el 0,90 y los de mAP50-95 rondan aldedor de 0,85-0,90, lo que es un excelente resultado en cuanto a la fiabilidad de la evolución del modelo a lo largo de las épocas. Además, el hecho de que las curvas sean suaves y converjan antes de llegar a las 75 épocas sugiere que 50 épocas ya eran suficientes para alcanzar la convergencia, y las 75 épocas sólo consolidan ligeramente los resultados.

En cuanto a las Figuras 5.2 y 5.4, las matrices de confusión de cada propuesta son herramientas que muestran cómo se desempeña un modelo de clasificación. Permite visualizar cuántos elementos fueron correctamente clasificados (aciertos) y cuántos fueron mal clasificados (errores). En el eje horizontal (*True*) se encuentran las clases reales de los objetos en la base de datos, y en el eje vertical (*Predicted*) las clases predichas por el modelo. Las celdas en la diagonal principal señala los aciertos (predicción correcta) y las celdas fuera de la diagonal principal señala los errores (confusión entre clases). Por ejemplo, en la Figura 5.4 para la clase “Carton”: 1462 objetos de cartón fueron correctamente clasificados como cartón, 210 se confundieron como “*background*” (el modelo no los detectó como objetos) y otros pocos se confundieron con las demás clases, 4 con “Latas”, 3 con “Plastico” y 4 con “Vidrio”.

La mayoría de los valores se encuentran en la diagonal principal, lo que indica un alto rendimiento del modelo en todas las clases. Y se observa que los principales errores (confusiones) provienen de objetos confundidos como parte del fondo o el contexto físico y alguna confusión puntual entre “Vidrio” y “Plastico”, lo cual se entiende porque pueden ser visualmente similares en ciertas condiciones, especialmente si están hechos de material con características de transparencia. En general, los resultados son prometedores porque los aciertos superan ampliamente a los errores en todas clases y en cada propuesta.

5.3. VALIDACIÓN EN CONDICIONES CONTROLADAS

5.3.1. VALIDACIÓN CON CÁMARA LOGITECH

En la primera fase de pruebas, se utilizaron imágenes captadas con una webcam Logitech en un entorno doméstico. Los objetos se presentaron de forma individual, lo que permitió observar el desempeño del modelo en condiciones más sencillas. Esto se observa de manera rápida haciendo un registro de los resultados obtenidos mediante matrices de confusión de los objetos probados.

PROPUESTA I

Predicción \ Valores reales	Carton	Envases	Vidrio
Carton	7	1	2
Envases	1	11	1
Vidrio	0	7	2

PROPUESTA II

Predicción \ Valores reales	Carton	Latas	Plastico	Vidrio
Carton	8	0	0	2
Latas	0	9	0	1
Plastico	0	2	9	2
Vidrio	0	1	0	8

Figura 5.5. Matrices de confusión con los modelos versiones 1A y 2A.

Los resultados mostraron un buen nivel de detección en clases como “Carton” y “Latas”, aunque se evidenciaron algunos errores en la diferenciación entre “Vidrio” y

"Envases" particularmente en el modelo de versión 1A, pues se observa que 7 elementos de "Vidrio" fueron confundidos como "Envases", probablemente debido a los residuos de características de transparencia altamente similares.

En las siguientes dos figuras se muestran claros ejemplo de confusión entre materiales de plástico y de vidrio (detectado con el modelo versión 1A y 2A):

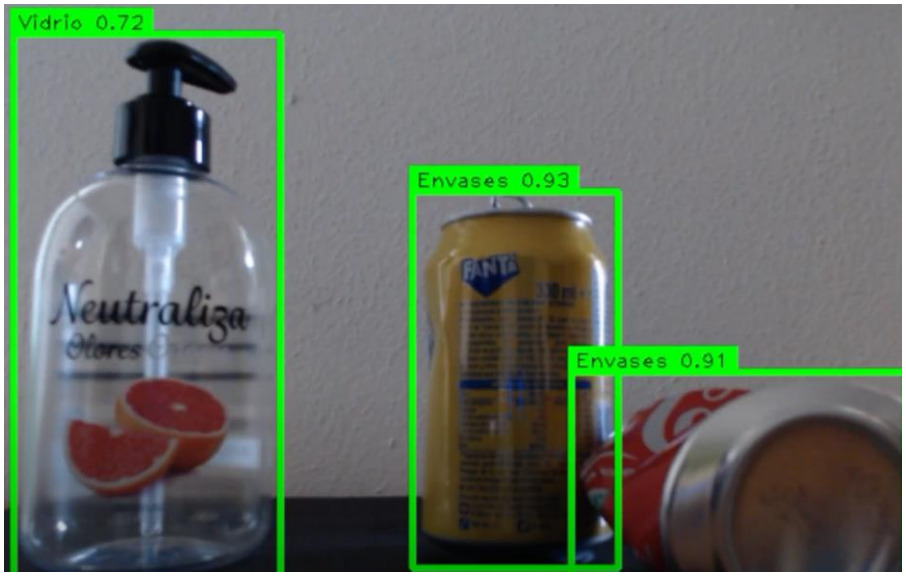

Figura 5.6. Detección con cámara Logitech y confusión de plástico con vidrio modelo versión 1A.

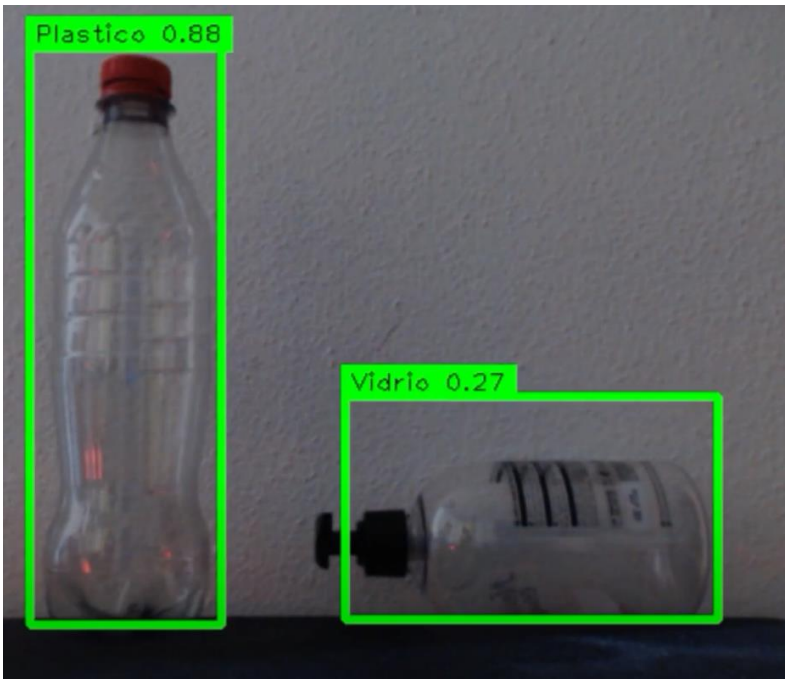

Figura 5.7. Detección con cámara Logitech y confusión de plástico con vidrio modelo versión 2A.

En contraste, bajo el mismo modelo 2A, también se detectó correctamente objetos de vidrio, pero con la observación que su orientación los hacía ser confundidos con otra clase, en el caso de la siguiente figura, con latas:

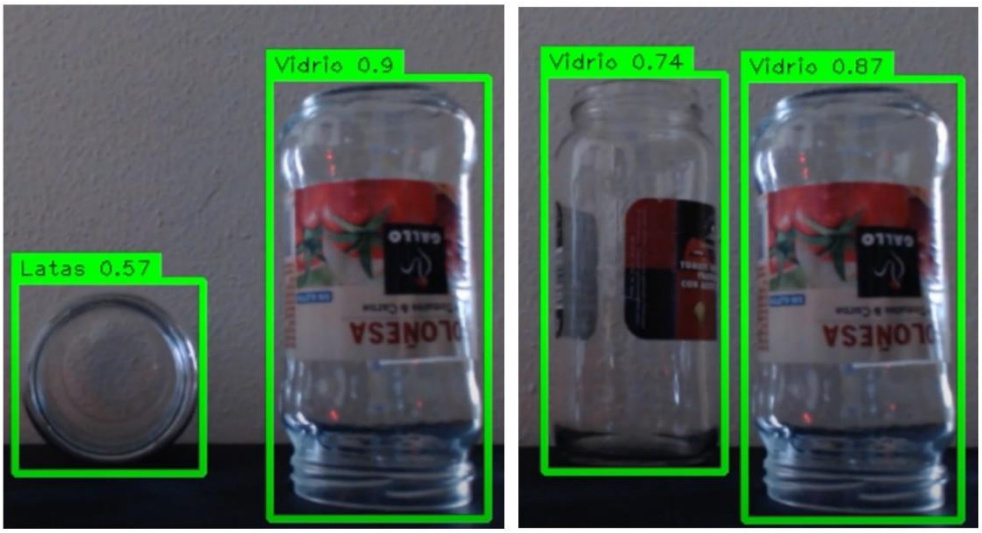

Figura 5.8. Detección con cámara Logitech y confusión de plástico con vidrio modelo versión 2A.

5.3.2. Validación con la cámara OAK-D Lite en el laboratorio

La segunda fase de validación se llevó a cabo en el laboratorio, utilizando la cámara OAK-D Lite. También en este caso se presentaron objetos de forma individual, obteniendo matrices de confusión más representativas del sistema final.

PROPUESTA I

Predicción \ Valores reales	Carton	Envases	Vidrio
Carton	13	0	1
Envases	0	16	4
Vidrio	0	4	12

PROPUESTA II

Predicción \ Valores reales	Carton	Latas	Plastico	Vidrio
Carton	15	0	0	0
Latas	1	13	0	0
Plastico	1	5	14	3
Vidrio	0	1	2	12

Figura 5.9. Matrices de confusión con los modelos versiones 1E y 2D.

Se observó una mejora en la consistencia de las detecciones respecto al entorno doméstico, gracias al incremento de la base de datos, que permitió robustecer el modelo para poder predecir detecciones en un ambiente más generalizado. Particularmente se puede observar que todavía ocurren instancias puntuales en las que un objeto hecho de material plástico se confunde con uno de vidrio y viceversa. Un ejemplo de ello, en el laboratorio, se muestra en la figura a continuación:

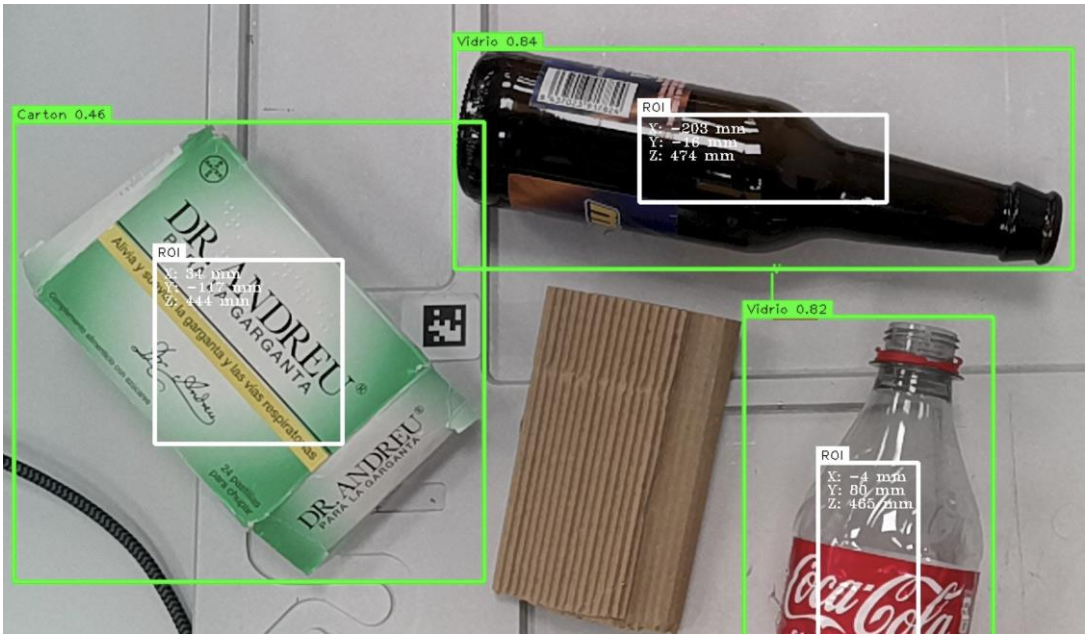

Figura 5.10. Detección de varios objetos a la vez con el modelo versión 1E.

Además, en la misma Figura 5.10, se puede observar que también existe la posibilidad de que el objeto no sea detectado por el modelo. En este caso, ha sido con un objeto hecho de cartón. En la siguiente sección se analiza este escenario en concreto.

5.4. VALIDACIÓN CON MÚLTIPLES OBJETOS Y FONDO

Finalmente, se realizaron pruebas en las que se presentaron simultáneamente tres o más objetos, incorporando además la detección del fondo (cada vez que un objeto no era detectado, se considera como una predicción de fondo). Estas pruebas resultaron ser las más exigentes, ya que pusieron a prueba la robustez del sistema frente a detecciones múltiples y posibles falsos positivos.

PROPUESTA I

Valores reales

Predicción	Carton	Envases	Vidrio	FONDO
Carton	10	0	1	3
Envases	0	12	3	2
Vidrio	0	2	13	5

PROPUESTA II

Valores reales

Predicción	Carton	Latas	Plastico	Vidrio	FONDO
Carton	14	0	0	0	3
Latas	0	15	1	0	0
Plastico	0	4	15	2	2
Vidrio	0	1	3	13	7

Figura 5.11. Matrices de confusión con los modelos versiones 1E y 2D con varios objetos presentado a la vez.

Los resultados mostraron que, aunque los modelos lograban detectar correctamente la presencia de los objetos, las estimaciones de profundidad presentaban inestabilidad debido al cambio frecuente en las regiones de interés (ROI). Esto derivó en posiciones intercambiadas entre objetos de diferentes clases. El cambio frecuente de la ROI ocurrió debido al vínculo que tiene con la caja delimitadora de la detección: si la detección no es confiable y entre cada iteración de imagen procesada la caja aparece y desaparece (debido a la poca confiablidad de la detección), entonces la ROI cambia de la misma manera, pues se posiciona en la misma ubicación que la caja de detección.

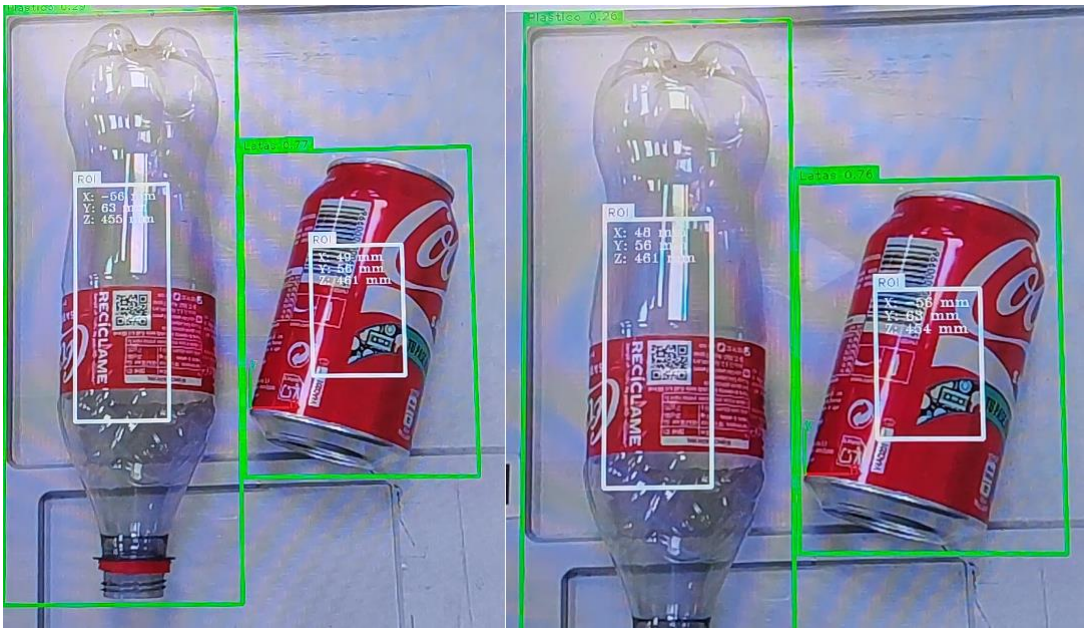

Figura 5.12. Detección de varios objetos con valores de ROI intercambiados.

En la Figura 5.12 se observa para el caso de la botella de plástico que el valor de la posición en el eje X es negativo (-56 mm) en la imagen de la izquierda, pero positivo (48 mm) en la imagen de la derecha; estas imágenes tienen el centro del sistema coordenado en el punto donde se encuentra la esquina inferior izquierda de la lata (vista en el plano). Así que se interpreta que el valor de los vectores posición de los dos objetos están cambiando durante la detección, el correcto sería el correspondiente a lo visto en la imagen de la izquierda. Esto es un fenómeno observado especialmente bajo la presencia de varios objetos y con detecciones de poca confiabilidad presentes para alguno de los objetos.

5.5. VALIDACIÓN DEL SISTEMA ROBÓTICO

En esta fase, el sistema fue probado en su ciclo completo: detección, transformación de coordenadas y manipulación física con el UR3e (se muestra en la Figura 5.13). Además de las matrices de detección, se elaboró una matriz que reflejaba lo que el robot realmente era capaz de agarrar. Esta matriz evidenció la diferencia entre lo detectado de forma correcta y lo manipulado efectivamente.

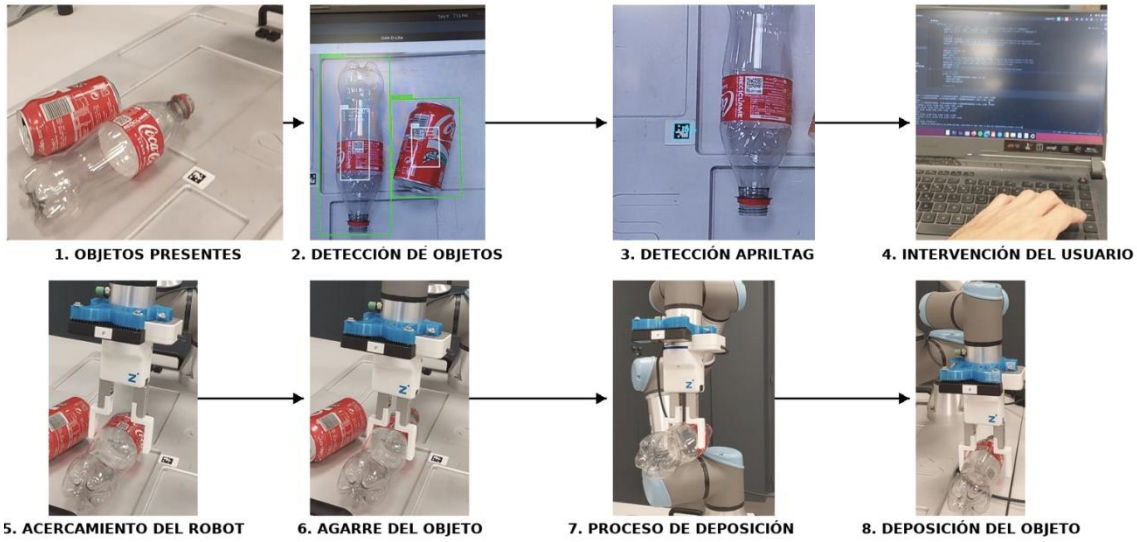

Figura 5.13. Secuencia de imágenes mostrando el proceso de clasificación.

PROPUESTA I

Valores realmente separados

Predicción para separar	Carton	Envases	Vidrio	FONDO
Carton	9	0	2	3
Envases	0	10	5	2
Vidrio	0	4	11	5

PROPUESTA II

Valores realmente separados

Predicción para separar	Carton	Latas	Plastico	Vidrio	FONDO
Carton	14	0	0	0	3
Latas	0	12	4	0	0
Plastico	0	5	12	4	2
Vidrio	0	3	4	10	7

Figura 5.14. Matrices de confusión con los modelos versiones 1E y 2D con varios objetos presentado a la vez y su validación de recogida y depósito.

Las Figuras 5.11 y 5.14 reflejan la misma instancia de estudio, es decir, el proceso completo de clasificación presentando varios objetos a la vez. Si el proceso se completa con total perfección, ambas figuras deberían ser iguales, pero no lo son. Esto es porque durante las pruebas con múltiples objetos, se detectó un fenómeno recurrente: aunque las detecciones eran correctas, las posiciones calculadas por la cámara se intercambiaban entre objetos debido a la inestabilidad de la ROI. Este problema afectó entre un 20–25% de los verdaderos positivos, lo que provocó errores en el agarre pese a detecciones correctas.

Por ejemplo, si tomamos la fila horizontal de “Envases” de la PROPUESTA I y comparamos ambas figuras (5.11 y 5.14) se observa para la primera (5.11) que el valor de los objetos denominados como “Envases” que realmente fueron detectados positivamente es de 12, sin embargo, en la segunda figura (5.14) se observa que el valor baja a 10 elementos; la diferencia en este número, en el contexto real, se traduce en que esos dos elementos de “Envases” fueron clasificados como “Vidrio”, en este caso, pues se evidencia que su valor cambió de 3 a 5 elementos.

Cabe destacar que esta forma de recolectar y agrupar los datos para observar la validación de la clasificación robótica se desarrolló como forma de comparar con las mediciones de detección (matriz de confusión), y no con el objetivo de valorar la calidad de la detección misma.

Capítulo 6. CONCLUSIONES

6.1. CONCLUSIONES

El presente Trabajo Fin de Máster ha permitido demostrar la viabilidad de integrar técnicas de visión por computador y aprendizaje automático con un sistema robótico colaborativo para abordar la clasificación automática de residuos domésticos. La arquitectura desarrollada, que combina un detector visual basado en YOLOv8, la cámara estéreo OAK-D Lite y un brazo robótico UR3e controlado mediante el protocolo RTDE, ha hecho posible completar el ciclo de detección, localización, agarre y depósito de objetos, cumpliendo con los objetivos propuestos al inicio del proyecto.

Los resultados obtenidos muestran que el sistema alcanza un rendimiento sólido en la detección y clasificación de residuos, con valores de precisión y *recall* elevados y métricas de mAP que superan el 90%. Las pruebas realizadas, tanto en entornos domésticos como en laboratorio, validan la capacidad del sistema para distinguir entre diferentes tipos de residuos como cartón, vidrio, plásticos y latas. No obstante, en escenarios con múltiples objetos se observaron errores en la asignación de coordenadas tridimensionales, atribuibles a la inestabilidad de las regiones de interés.

Este aspecto abre un campo de mejora para la estabilización de las detecciones y la gestión de escenas más complejas.

El trabajo ha incluido también la adaptación de la pinza HRC-03-118505 mediante piezas diseñadas en Autodesk Inventor y fabricadas en impresión 3D, lo que permitió superar las limitaciones de apertura originales y manipular con éxito la mayoría de los residuos de prueba. Este resultado pone de manifiesto la importancia de considerar no sólo los aspectos de visión e inteligencia artificial, sino también las adaptaciones mecánicas necesarias para adecuar el sistema a los requisitos de la aplicación. Otro aspecto relevante ha sido la incorporación de la interacción usuario-robot en el flujo de trabajo. El usuario participa seleccionando qué clase de residuos clasificar en cada iteración, lo que aporta flexibilidad y seguridad operativa. Este esquema de supervisión compartida puede considerarse un valor añadido, al equilibrar la autonomía del sistema con la toma de decisiones humanas.

En conjunto, este trabajo establece una base sólida para futuros desarrollos en la automatización de la clasificación de residuos, con aplicaciones potenciales tanto en entornos domésticos como industriales. La combinación de técnicas de visión por computador, aprendizaje automático y robótica colaborativa se muestra como una alternativa prometedora para avanzar en los objetivos de sostenibilidad y eficiencia en la gestión de desechos, abriendo la puerta a soluciones más completas y escalables en un futuro cercano.

6.2. LÍNEAS DE MEJORA Y TRABAJOS FUTUROS

A pesar de los logros alcanzados, el proyecto deja abiertas varias líneas de investigación y desarrollo. Entre ellas se encuentran:

1. La optimización del modelo de detección mediante bases de datos más variadas y representativas que incluyan condiciones de iluminación complejas, residuos deformados o parcialmente ocultos, con el fin de mejorar la robustez y reducir los falsos negativos.

2. La mejora en la estimación de posiciones 3D, como aplicar técnicas de estabilización de detecciones, como filtros de promediado o seguimiento temporal de ROIs, para minimizar los errores de asignación de coordenadas en escenas con múltiples objetos.

3. El uso de pinzas más avanzadas, como pinzas adaptativas o de agarre blando, que permitan manipular una mayor diversidad de residuos sin necesidad de modificaciones mecánicas adicionales.

4. La integración de múltiples cámaras podría ofrecer una visión más robusta y precisa del entorno, reduciendo errores en la estimación de distancias y permitiendo generar un algoritmo comparativo de los objetos detectados desde cada punto de referencia, y posiblemente mejorando la fiabilidad de las detecciones obtenidas.

5. Extender las pruebas hacia escenarios semi-controlados o domésticos reales, evaluando la aceptación y usabilidad del sistema por parte de usuarios no especializados, lo cual es fundamental para avanzar hacia aplicaciones comerciales o sociales.

BIBLIOGRAFÍA

[1] Kaza, S., Yao, L., Bhada-Tata, P., & Van Woerden, F. (2018). *What a Waste 2.0: A Global Snapshot of Solid Waste Management to 2050*. Washington, DC: World Bank.

[2] United Nations Environment Programme (UNEP). (2024). *Global Waste Management Outlook 2024*. Nairobi: UNEP.

[3] Parlamento Europeo y Consejo. (2008). *Directiva 2008/98/CE sobre los residuos y por la que se derogan determinadas Directivas*. Diario Oficial de la Unión Europea.

[4] Parlamento Europeo y Consejo. (2018). *Directiva (UE) 2018/851 por la que se modifica la Directiva 2008/98/CE relativa a los residuos*. Diario Oficial de la Unión Europea.

[5] Comisión Europea. (2020). *Plan de Acción de Economía Circular. Por una Europa más limpia y más competitiva*. Bruselas.

[6] Lahoti, J., et al. (2024). *Multi-class waste segregation using computer vision and YOLOv5*. Scientific Reports, 14, 2989.

[7] Moktar, M. H., et al. (2025). *Medical waste sorting machine development with IoT and YOLO*. Journal of Engineering and Applied Science, 72, 115–129.

[8] Szeliski, R. (2010). *Computer Vision: Algorithms and Applications*. Springer.

[9] Terven, J., & Córdova-Esparza, D. (2023). *A Comprehensive Review of YOLO Architectures in Computer Vision: From YOLOv1 to YOLOv8 and YOLO-NAS*. arXiv preprint, arXiv:2304.00501.

[10] Wang, C.-Y., & Liao, H.-Y. M. (2024). *YOLOv1 to YOLOv10: The fastest and most accurate real-time object detection systems*. arXiv preprint, arXiv:2408.09332.

[11] Ultralytics. (s.f.). *YOLOv8 Documentation*. Consultado en septiembre de 2025, de https://docs.ultralytics.com

[12] Olson, E. (2011). *AprilTag: A robust and flexible visual fiducial system*. IEEE International Conference on Robotics and Automation (ICRA), 3400–3407.

[13] Universal Robots. (s.f.). *Universal Robots*. Consultado en septiembre de 2025, de https://www.universal-robots.com

[14] Terras, N. (2025). *Integration of Deep Learning Vision Systems in Collaborative Robotics for Real-Time Applications*. Applied Sciences, 15(3), 1336.

[15] Cohen, Y. (2025). *Fusion of Computer Vision and AI in Collaborative Robotics*. Applied Sciences, 15(14), 7905.

[16] Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep Learning. MIT Press*.

[17] LeCun, Y., Bengio, Y., & Hinton, G. (2015). *Deep learning*. Nature, 521, 436–444.

[18] Krizhevsky, A., Sutskever, I., & Hinton, G. (2012*). ImageNet classification with deep convolutional neural networks*. Advances in Neural Information Processing Systems (NeurIPS), 25, 1097–1105.

[19] Zhang, Z. (2000). *A flexible new technique for camera calibration*. IEEE Transactions on Pattern Analysis and Machine Intelligence, 22(11), 1330–1334.

[20] OpenCV. (s.f.). *Basic concepts of the homography explained with code*. Consultado en septiembre de 2025, de https://docs.opencv.org/4.x/d9/dab/tutorial_homography.html

[21] Chaumette, F., & Hutchinson, S. (2006). *Visual Servo Control, Part I: Basic Approaches*. IEEE Robotics & Automation Magazine, 13(4), 82–90.

[22] Chaumette, F., & Hutchinson, S. (2007). *Visual Servo Control, Part II: Advanced Approaches*. IEEE Robotics & Automation Magazine, 14(1), 109–118.

[23] Luxonis. (s.f.). *OAK-D Lite - Luxonis*. Consultado en septiembre de 2025, de https://shop.luxonis.com/products/oak-d-lite-1

[24] Roboflow. (s.f.). *Roboflow Universe*. Consultado en mayo de 2025, de https://universe.roboflow.com/

[25] Luxonis. (s.f.). *Spatial Location Calculator. Luxonis Documentation*. Consultado en septiembre de 2025, de https://docs.luxonis.com/software/depthai/examples/spatial_location_calculator/

[26] Zimmer Group. (s.f.). *HRC-03-118505 - Zimmer Group*. Consultado en septiembre de 2025, de https://www.zimmer-group.com/es/productos/componentes/tecnologia-de-manipulacion/pinza-hrc/hrc-03/productos/hrc-03-118505

[27] Bishop, C. M. (2006). *Pattern Recognition and Machine Learning*. Springer.

ANEXOS

ANEXO A. MODELO CAD AMPLIACIÓN DE PINZA

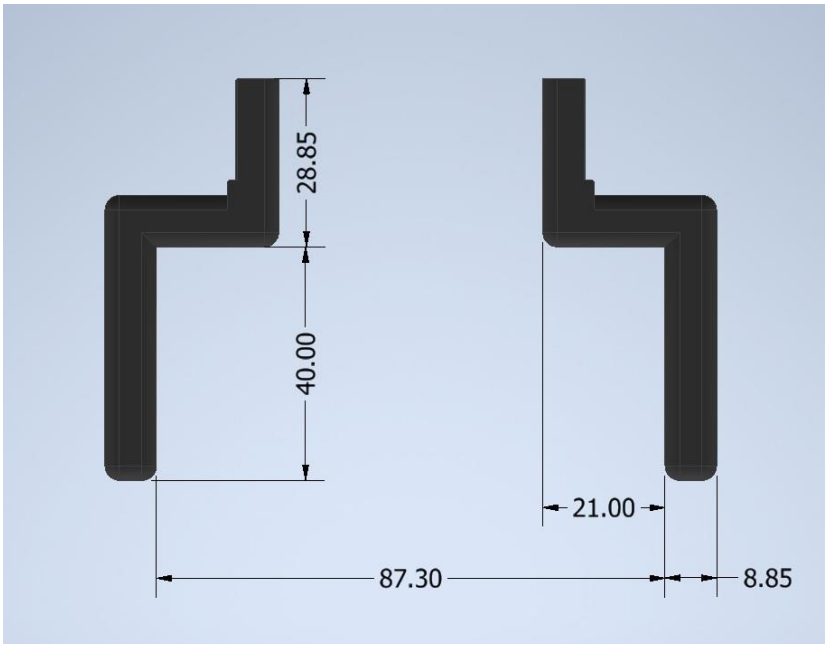

Figura A - 1. Dimensiones de las piezas de ampliación para la pinza HRC-03-118505.

ANEXO B. ENLACES

B.1. ENLACE AL REPOSITORIO DE GITHUB

https://github.com/Mau32/TFMWasteClassification

B.2. ENLACE A LA LISTA DE VIDEOS DE YOUTUBE

https://www.youtube.com/playlist?list=PL7M1fQJCSgQcr_DgHD3ZIVviH09riklLt

B.3. ENLACE A LAS BASES DE DATOS Y RESULTADOS DE REENTRENAMIENTOS ALOJADOS EN MEGA

https://mega.nz/folder/xl5zHIxQ#cZ7AY5HGoctJ8lNEyEN3RA

B.4. ENLACE A LOS ARCHIVOS CAD DE LAS PIEZAS DE AMPLIACIÓN DE LA PINZA ALOJADOS EN MEGA

https://mega.nz/folder/RhhS0ISZ#q6ZlEips8Fjq3owoxUOyAw

B.5. ENLACE A LOS *NOTEBOOKS* DE GOOGLE COLAB PARA EL ENTRENAMIENTO

https://colab.research.google.com/drive/1LJx3o_qCiY7XvwKirQhIxRT5SHodkWq7?usp=drive_link

https://colab.research.google.com/drive/13PWNDt4WnBbizawfPwXNcWRtxjsbD3Kg?usp=drive_link

ANEXO C. MATRICES DE CONFUSIÓN

A continuación, se muestran las matrices de confusión obtenidas de las pruebas realizadas de detección de objetos comunes de desechos domésticos. Se dividen en dos: Propuesta I y Propuesta II, y se detalla la versión del modelo reentrenado de YOLOv8l con la nomenclatura mostrada en Tablas 5.1 y 5.2. Estas pruebas se desarrollaron mostrando un solo objeto a la vez.

En la siguiente figura se muestran tres bolsas de objetos de desecho doméstico, que fueron utilizados para las pruebas de este trabajo. Se dividen de la siguiente manera, por tipo de material: la blanca, de material de vidrio, la amarilla con objetos de tipo envases plásticos (considerados para el depósito en recipientes amarillos) y latas, y la negra, con objetos de material de papel y cartón.

Figura C - 1. Muestra de objetos de desecho doméstico.

C.1. MATRICES DE CONFUSIÓN DE PRUEBAS PARA LA PROPUESTA I

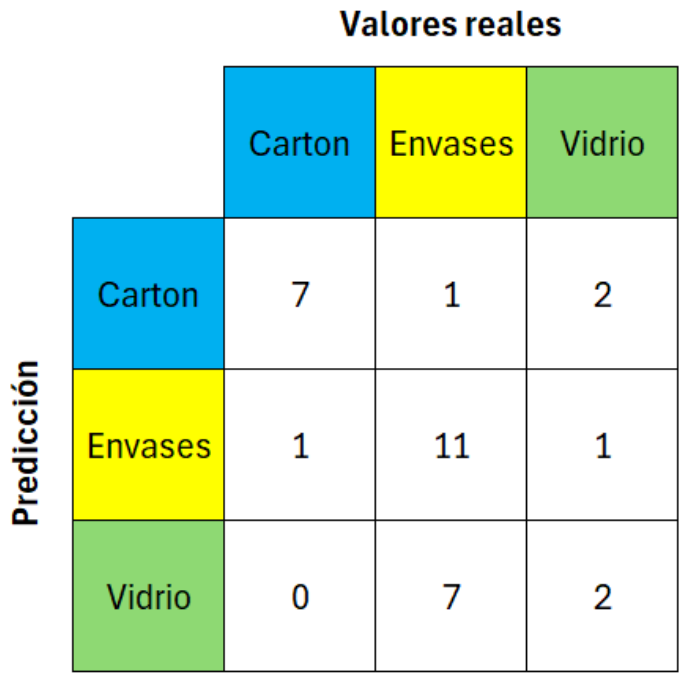

Figura C - 2. Matriz de confusión a partir de pruebas hechas con el modelo versión 1A.

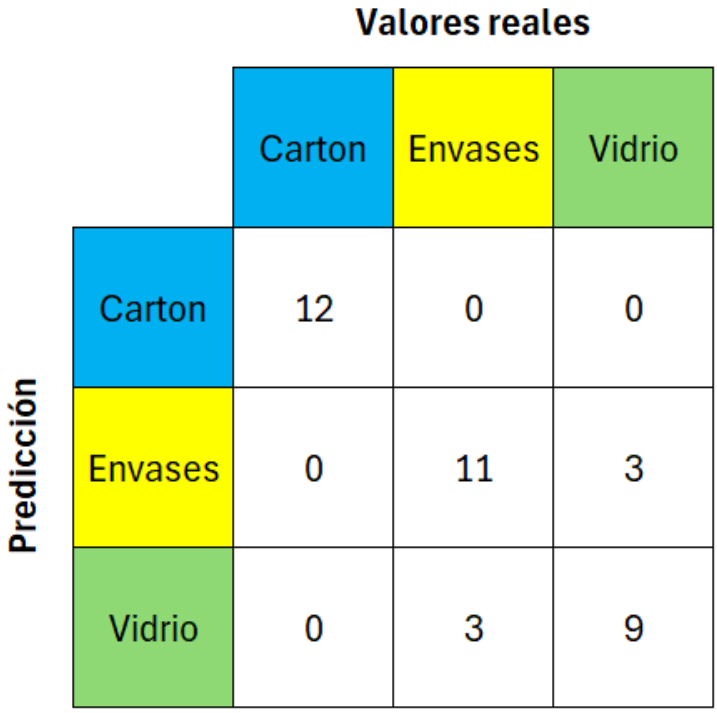

Figura C - 3. Matriz de confusión a partir de pruebas hechas con el modelo versión 1B.

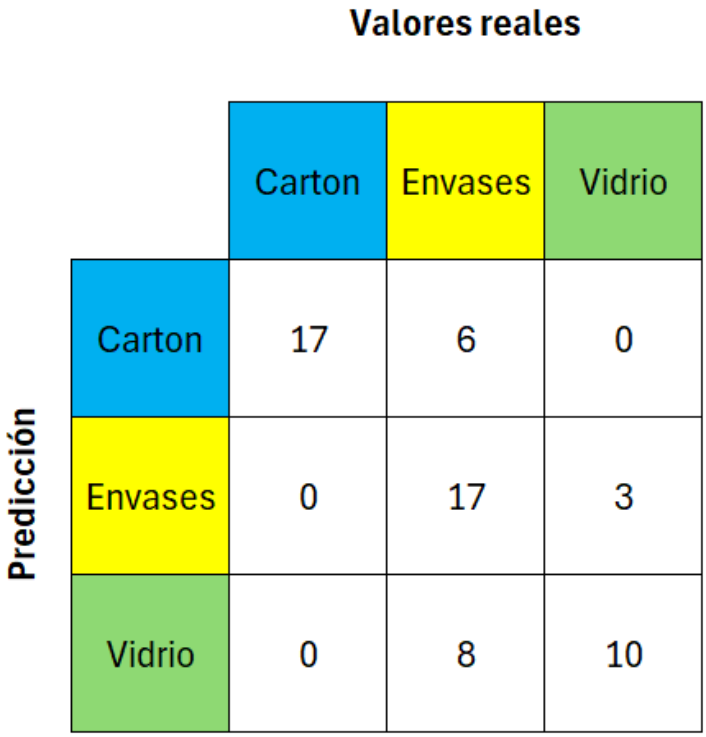

Figura C - 4. Matriz de confusión a partir de pruebas hechas con el modelo versión 1C.

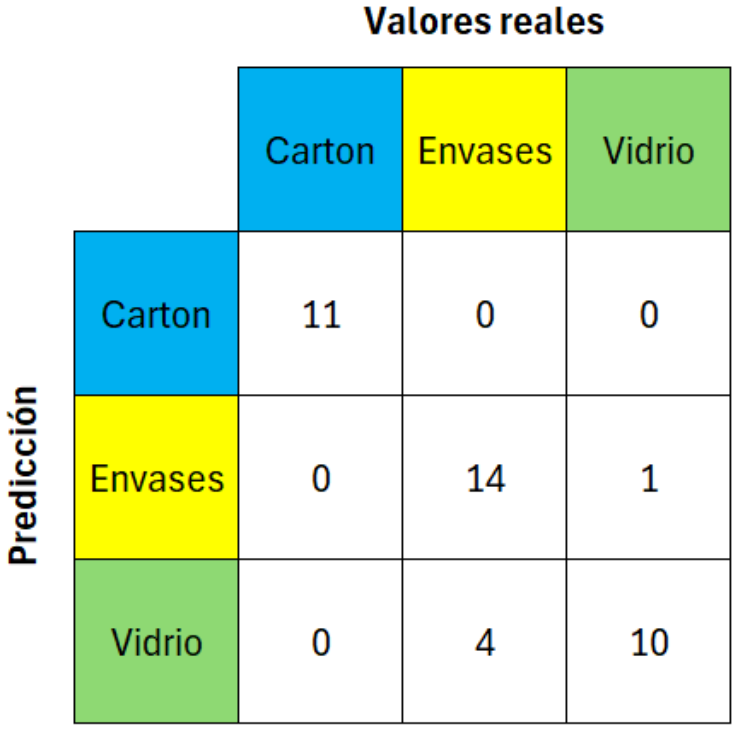

Figura C - 5. Matriz de confusión a partir de pruebas hechas con el modelo versión 1D.

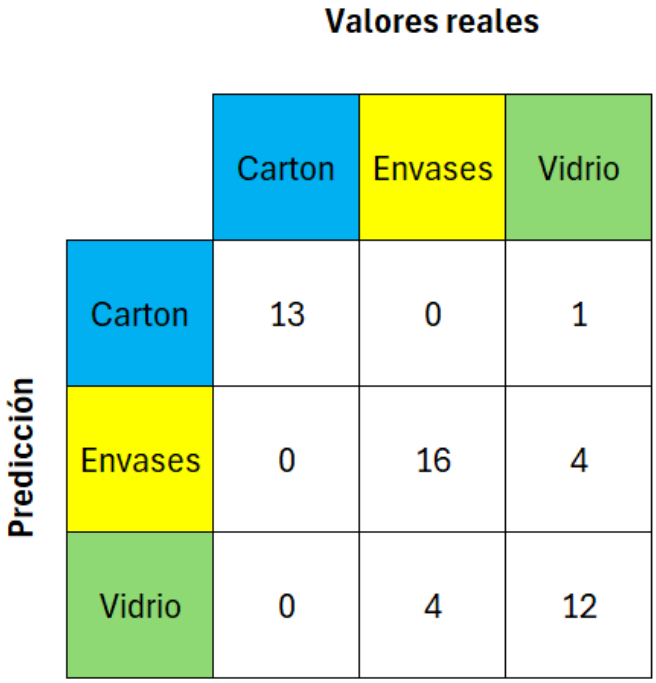

Figura C - 6. Matriz de confusión a partir de pruebas hechas con el modelo versión 1E.

C.2. MATRICES DE CONFUSIÓN DE PRUEBAS PARA LA PROPUESTA II

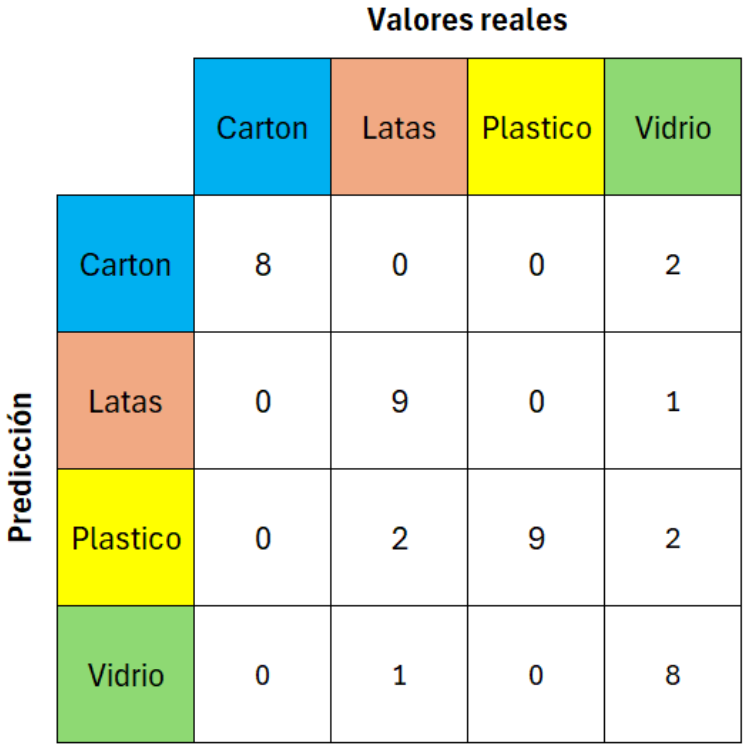

Figura C - 7. Matriz de confusión a partir de pruebas hechas con el modelo versión 2A.

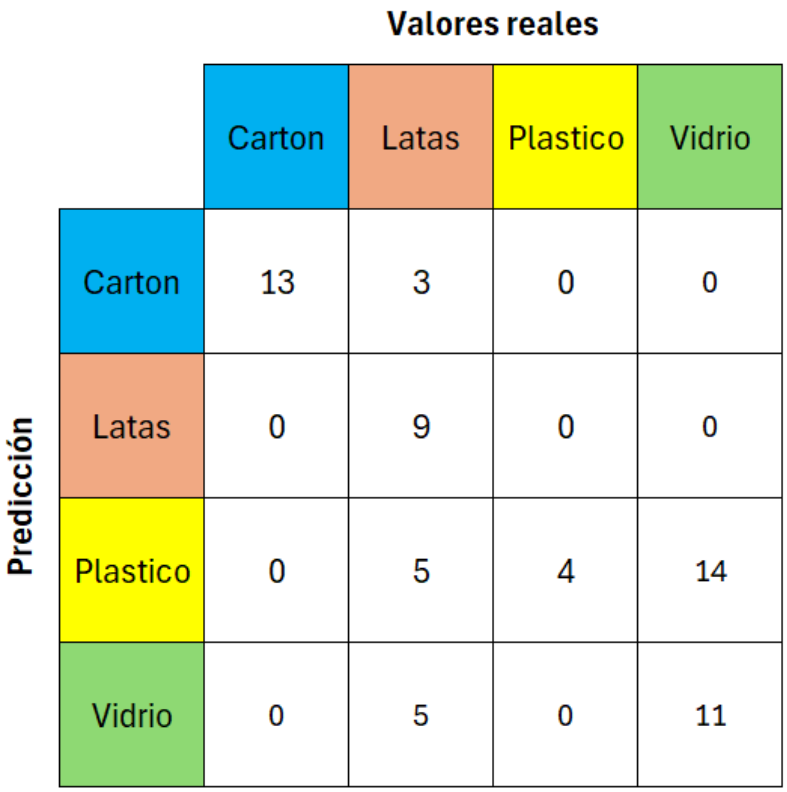

Figura C - 8. Matriz de confusión a partir de pruebas hechas con el modelo versión 2B.

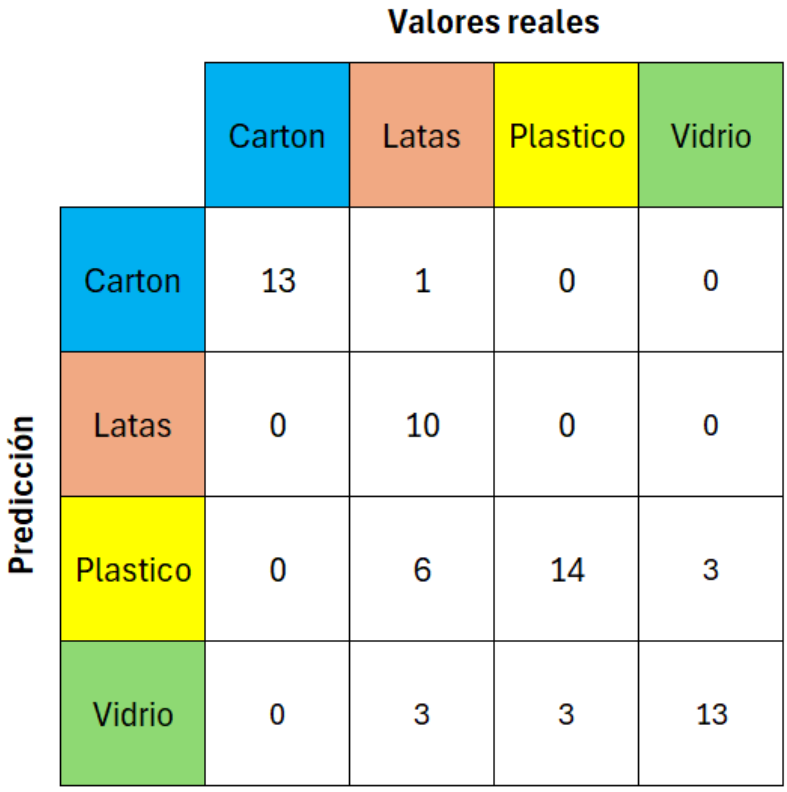

Figura C - 9. Matriz de confusión a partir de pruebas hechas con el modelo versión 2C.

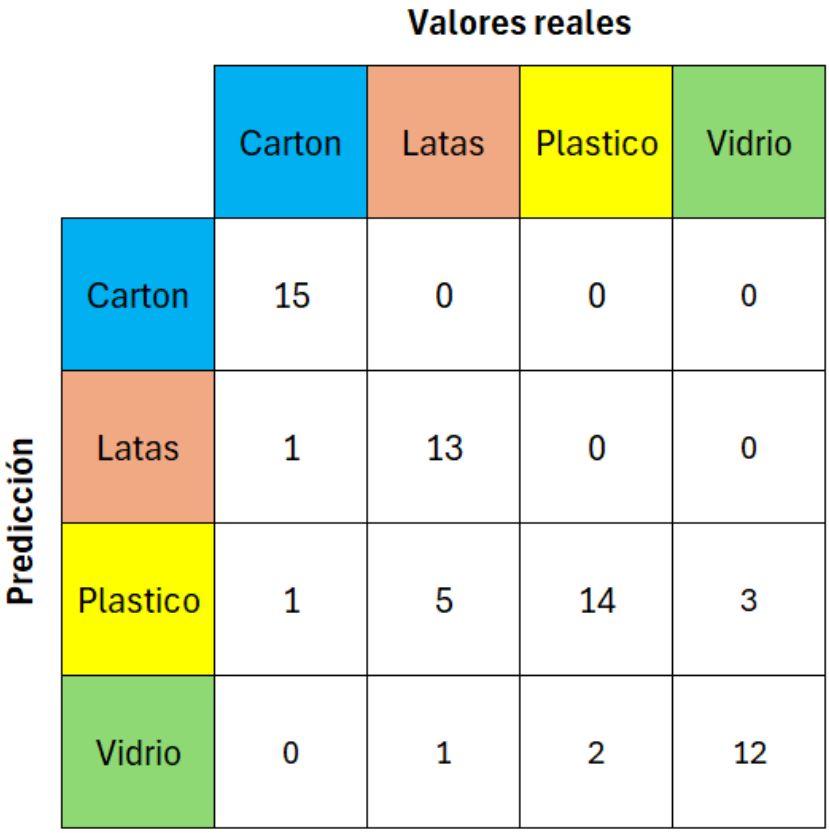

Figura C - 10. Matriz de confusión a partir de pruebas hechas con el modelo versión 2D.