

# **Detección de defectos en objetos en movimiento mediante Redes Neuronales Convolucionales con optimizaciones específicas para hardware NVIDIA**

---

Alumno: Haro Armero, Abel

Tutor: Flich Cardo, José

Co-tutor: López Rodríguez, Pedro Juan

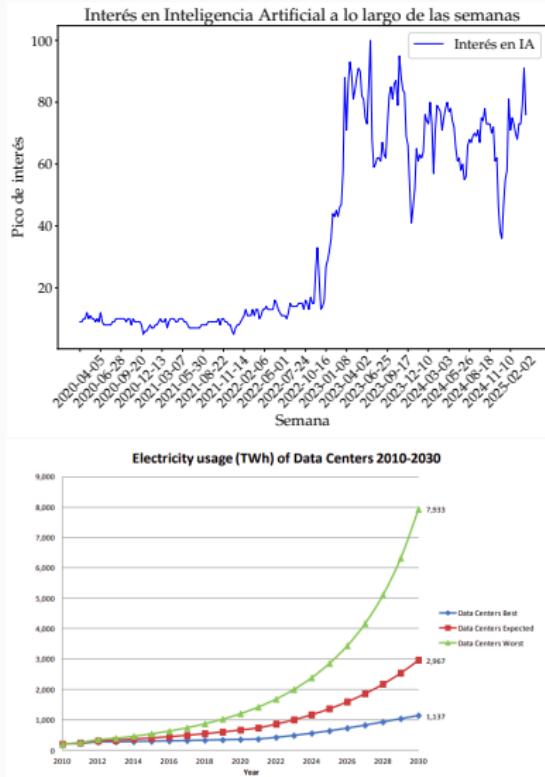
11 de julio de 2025

# Índice

1. Introducción
2. Motivación
3. Objetivos
4. Conceptos previos
5. Propuesta de solución
6. Desarrollo de la solución
7. Demo del sistema
8. Resultados
9. Conclusiones

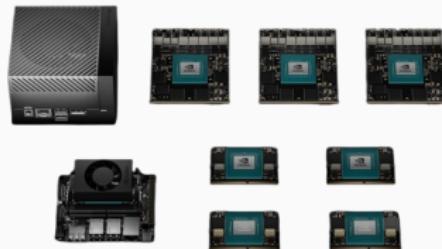
# Introducción

- Crecimiento exponencial de la Inteligencia Artificial.
- Avances en visión por computador gracias a las CNNs.
- Desafíos energéticos impulsan soluciones optimizadas.
- Dispositivos NVIDIA Jetson para IA eficiente en entornos embebidos.
- Desarrollo de sistema de detección de defectos en objetos en movimiento con CNNs, optimizado para hardware NVIDIA.



# Motivación

- Emular la interpretación humana de imágenes en máquinas.
- IA y visión por computador revolucionan la tecnología.
- NVIDIA Jetson impulsa la IA en edge computing.
- Automatización de la detección y clasificación de objetos en movimiento en la industria.

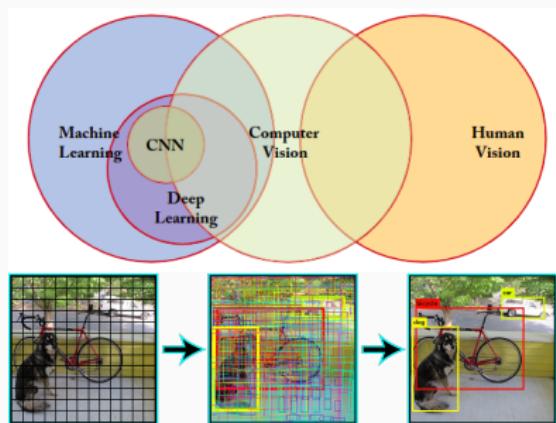


# Objetivos

- Estudiar el estado del arte en CNNs, aceleradores y optimización.
- Crear un conjunto de datos para entrenamiento y evaluación.
- Entrenar y validar modelos CNN para detección de defectos en tiempo real.
- Implementar un sistema de visión artificial integrado con hardware NVIDIA.
- Analizar y optimizar cuellos de botella para mejorar rendimiento y eficiencia.
- Evaluar el sistema con métricas de precisión, latencia y consumo.
- Realizar un análisis comparativo para encontrar la configuración óptima.

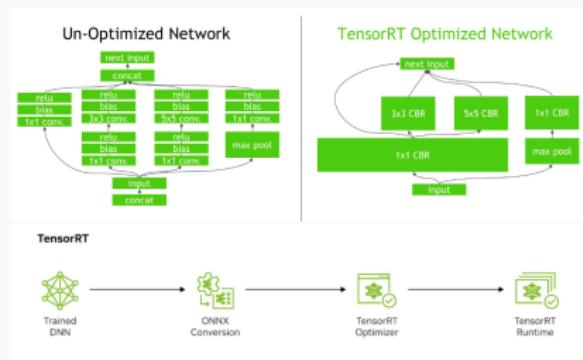
# Conceptos Previos - Redes Neuronales Convolucionales (CNNs)

- Las CNNs son un tipo de red neuronal profunda diseñadas para procesar datos con una estructura de cuadrícula, como imágenes.
- El objetivo es localizar dentro de una imagen objetos y clasificarlos, detectando defectos o características específicas.
- Utilizan capas convolucionales para extraer características jerárquicas.
- Existen dos tipos principales de CNNs, detectores de dos etapas como R-CNN y detectores de una etapa como YOLO.



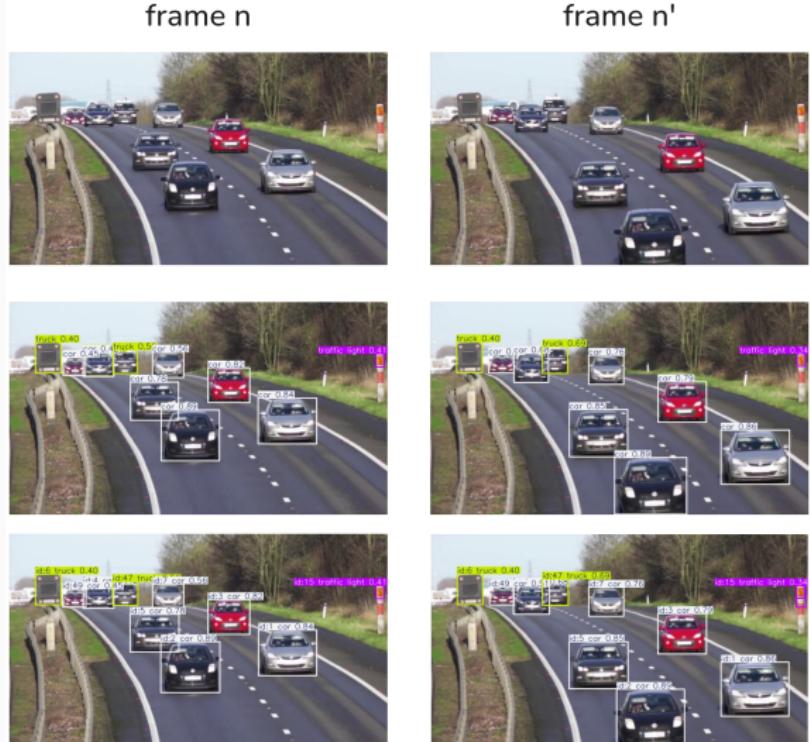
# Conceptos Previos - Hardware NVIDIA Jetson

- Arquitecturas hardware especializadas para el deep learning.
- NVIDIA Jetson: SoC (System on a Chip) que integra CPU y GPU para eficiencia energética y procesamiento paralelo.
- Jetpack SDK y TensorRT: El SDK de NVIDIA proporciona herramientas para desarrollar aplicaciones de IA, incluyendo TensorRT para optimizar de manera eficiente las redes neuronales en el hardware de NVIDIA.

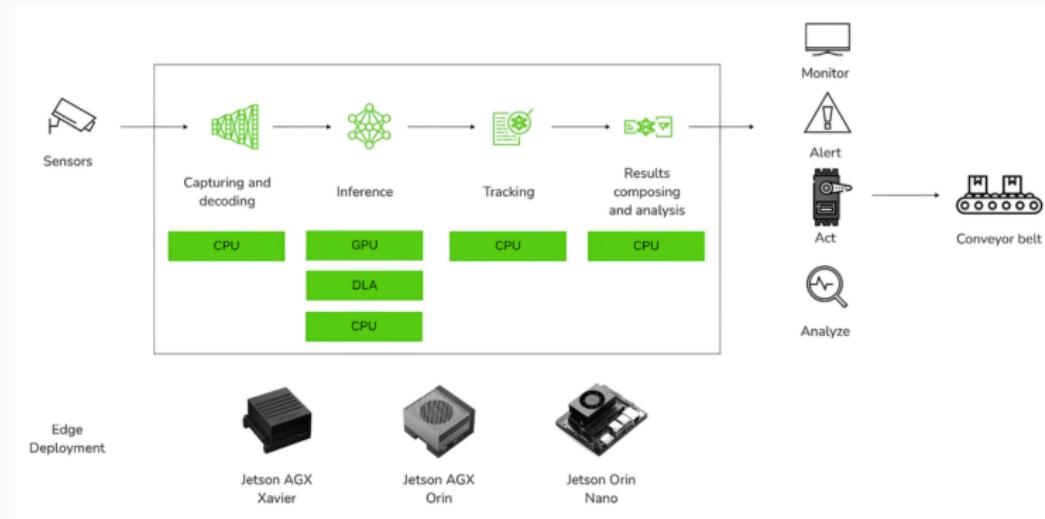


# Conceptos Previos - Seguimiento de objetos (MOT)

- El seguimiento de objetos es una técnica que permite identificar y seguir la trayectoria de un objeto a lo largo del tiempo en una secuencia de imágenes.
- Rastrea objetos combinando detección y análisis de movimiento.
- BYTETrack es el algoritmo de seguimiento que realiza el seguimiento de objetos a partir de las detecciones de un detector de objetos.

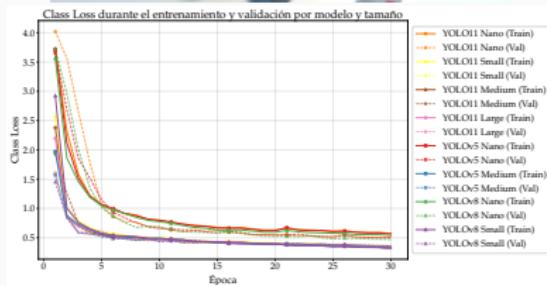


# Propuesta de solución



# Desarrollo de la solución - Entrenamiento y validación de modelos

- Creación de un conjunto de datos con imágenes de canicas de distintos colores y defectos.
- Entrenamiento de modelos CNN para detección de defectos.
- Validación y ajuste de hiperparámetros para mejorar precisión.
- Exportación de los modelos a TensorRT para optimizar su rendimiento en hardware NVIDIA.



# Desarrollo de la solución - Segmentación de las etapas (1/5)

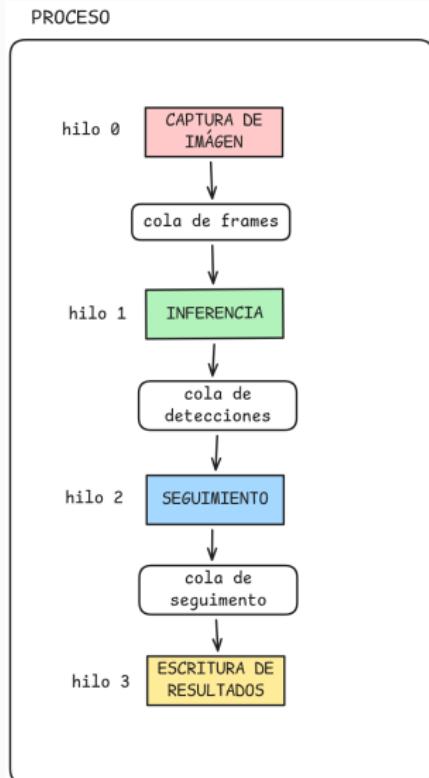
- El sistema se divide en cuatro etapas, captura de vídeo, detección de objetos, seguimiento de objetos y visualización.
- El objetivo de segmentar las etapas es permitir que cada una de ellas operen de forma independiente para mejorar la velocidad y eficiencia del sistema.
- Se han propuesto cuatro formas de segmentación para mejorar el procesamiento secuencial, (1) segmentación por **hilos**, (2) segmentación por **procesos**, (3) segmentación por **procesos con memoria compartida** y (4) segmentación por **heterogénea**



# Desarrollo de la solución - Segmentación de las etapas (2/5)

## Segmentación por hilos:

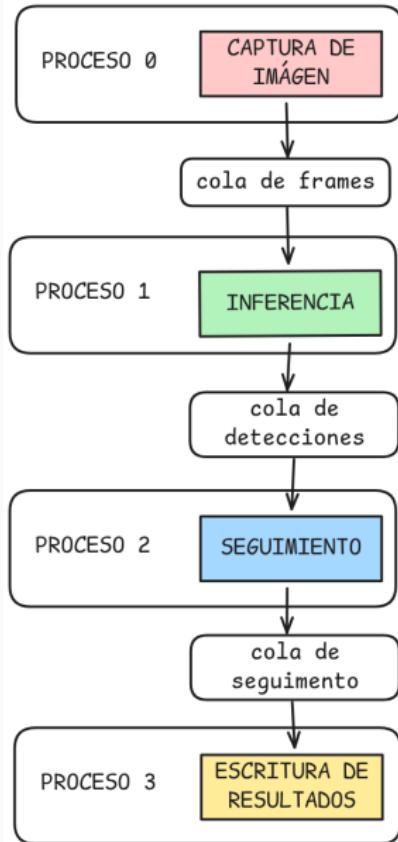
- Cada etapa se ejecuta en un hilo separado.
- La información se comparte entre hilos mediante colas que se consumen de forma asíncrona.
- Permite un procesamiento paralelo pero no concurrente debido al GIL (Global Interpreter Lock) de Python.



## Desarrollo de la solución - Segmentación de las etapas (3/5)

### Segmentación por procesos:

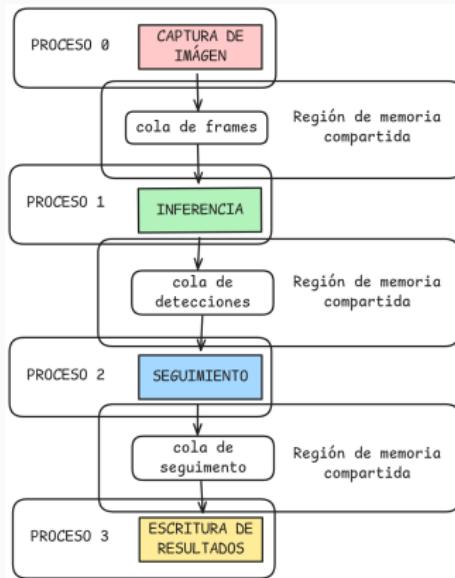
- Cada etapa se ejecuta en un proceso separado.
- La comunicación entre procesos se realiza mediante colas implementadas sobre pipes.
- Permite un procesamiento concurrente, pero con mayor latencia en la comunicación.



# Desarrollo de la solución - Segmentación de las etapas (4/5)

## Segmentación por procesos con memoria compartida:

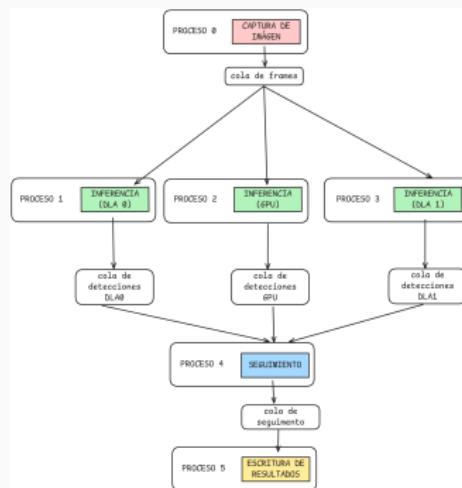
- Cada etapa se ejecuta en un proceso separado, pero comparten memoria.
- Las colas de comunicación se implementan sobre memoria entre procesos evitando la sobrecarga de pipes.
- Permite un procesamiento concurrente con menor latencia en la comunicación.



# Desarrollo de la solución - Segmentación de las etapas (5/5)

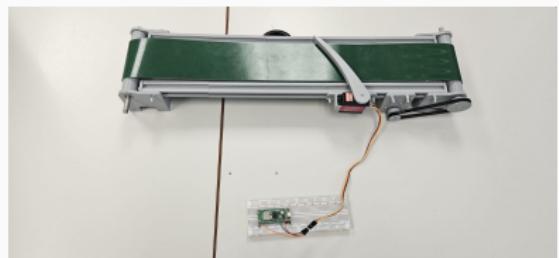
## Segmentación por heterogénea:

- Se puede ejecutar tanto con hilos como con procesos (con y sin memoria compartida).
- Ejecuta la etapa de detección en la GPU o DLAs del dispositivo Jetson.
- Permite explotar todos los recursos para mejorar el rendimiento.
- Los modelos de no se ejecutan completamente en la DLA.



## Desarrollo de la solución - Prueba de concepto

Como prueba simple de concepto, se ha construido una pequeña cinta transportadora con un motor. El sistema captura el vídeo de la cinta transportadora y detecta canicas de distintos colores y defectos. El objetivo es demostrar que el sistema es capaz de detectar objetos en movimiento y clasificarlos en tiempo real.



# VIDEO DEMO

Sistema de detección de defectos

Haz clic para reproducir

# Resultados

- Para evaluar el rendimiento del sistema, se han realizado pruebas con diferentes configuraciones de segmentación y optimización. Los resultados se han medido en términos de precisión, latencia y consumo energético.
- Existen todas estas configuraciones para evaluar el rendimiento del sistema:
  - Cantidad de objetos media en el vídeo.
  - Tipo de segmentación: hilos, procesos, procesos con memoria compartida y heterogénea.
  - Tipo de modelo y talla: YOLOv5nu, mu, YOLOv8n, s y YOLO11n, s, m, l.
  - Precisión del modelo: FP32, FP16 y INT8.
  - Modo de energía del dispositivo Jetson (MAXN, 30W8core, 30W4core, 15W4core, etc).
  - Dispositivo Jetson: Jetson AGX Xavier, Jetson AGX Orin y Jetson Orin Nano.

## Resultados - Cantidad de objetos

- Se ha evaluado el rendimiento del sistema con diferentes cantidades de objetos en el vídeo.
- 4 vídeos, (1) media de 17 objetos, (2) media de 43 objetos, (3) media de 84 objetos y (4) carga variable de 0 a 180 objetos.
- Para el resto de la configuración se ha utilizado los parámetros más óptimos encontrados en la sección de análisis comparativo.

## Resultados - Tipo de segmentación

- Se ha evaluado el rendimiento del sistema con diferentes tipos de segmentación.
- Se han utilizado los vídeos con una media de 17 objetos y 43 objetos.
- Para el resto de la configuración se ha utilizado los parámetros más óptimos encontrados en la sección de análisis comparativo.

# Conclusiones

- Se ha desarrollado un sistema de detección de defectos en objetos en movimiento.
- Se ha optimizado el rendimiento para hardware NVIDIA.
- Los resultados muestran una mejora en precisión y eficiencia.