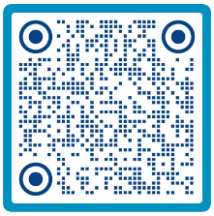
LA UNIVERSIDAD EN INTERNET

Universidad Internacional de La Rioja
Escuela Superior de Ingeniería y Tecnología

Grado en Ingeniería Informática

Detección de personas y/u objetos en vías ferroviarias a tiempo real

Trabajo fin de estudio presentado por:	Domingo Martínez Núñez
Línea de investigación:	Investigación científica
Codirector:	Fernando López Hernández
Codirector:	Javier Rainer Granados
Fecha:	marzo 2023
Repositorio:	https://github.com/Domy5/Rail_Surveillance/



UNIR-ESIT
TFE: DETECCIÓN DE PERSONAS Y/U OBJETOS EN VÍAS FERROVIARIAS A TIEMPO REAL.
2023 – Domingo Martínez Núñez.

AGRADECIMIENTOS

Es con un corazón lleno de gratitud que les escribo estas líneas de agradecimiento, a mi familia y amigos, por todo ese tiempo robado que han sufrido junto a mi para que pudiera cumplir mis sueños.

Como no podía ser de otra manera, quiero rendir homenaje al profesor Mikel Perales, un hombre excepcional que, a pesar de su prematura partida, ha dejado una huella imborrable en mi vida y en la de muchos otros estudiantes. La pasión y dedicación que él ponía en sus enseñanzas nunca serán olvidadas.

También quiero agradecer al claustro de profesores de la universidad, que han elegido esta hermosa profesión y han sido fuente de conocimiento y sabiduría para todos nosotros.

Pero mi más profundo agradecimiento va a mi director de TFG, Fernando, quien ha sido un guía incansable durante todo este proceso. Gracias por ayudarme a plasmar mis ideas y estar siempre disponible para responder a mis dudas y ofrecerme su apoyo.

Gracias a todos por hacer que esta experiencia universitaria sea inolvidable.

A usted, por tomar tiempo para leer este documento, sin su atención e interés, todo esto carecería de significado.

"La única forma de hacer un gran trabajo,

es amar lo que haces"

–Steve Jobs–

DEDICATORIA

A mis hijas, Noa y Abigail, mi amada Cristina, mis queridos padres y hermano mayor, este momento es especial para mí, ya que puedo presentar mi trabajo final de grado y poner fin a esta etapa universitaria. Pero lo que hace que este momento sea aún más especial es tenerlos a todos aquí, a mi lado, para compartirlo.

Quiero agradecerles por su apoyo incondicional y por estar a mi lado en todo momento, durante esta aventura. Sus consejos, su amor y su paciencia han sido clave para alcanzar esta meta. No puedo expresar con palabras lo agradecido que estoy por tenerlos como mi familia.

A mis hijas, quiero decirles que las quiero muchísimo y que les prometo seguir trabajando duro para hacer de este mundo un lugar mejor. A Cristina, mi pareja, gracias por ser mi compañera de vida y por ayudarme a mantenerme enfocado en mis objetivos.

A mis padres y hermano, les debo todo lo que soy hoy en día. Gracias por ser mi constante apoyo y fuente de inspiración en todo momento. Espero haberles hecho sentir orgullosos de mí y de mi logro.

Con todo mi amor y gratitud,

Domingo Martinez Núñez.

“Estudio para enseñar no teoría, sino hábitos “

Resumen

Este proyecto investiga, desarrolla y evalúa un sistema para la detección de personas y/u objetos en vías ferroviarias a tiempo real. Después de la revisión de posibles alternativas, ideas y tecnologías existentes se han identificado técnicas de detección en imágenes por sustracción de fondo como MOG2 (algoritmo de segmentación primer plano/fondo basado en mezclas gaussianas), GMG (combinación de estimación estadística de la imagen de fondo y segmentación bayesiana por píxel), KNN (vecinos más próximos), CNT (basado en el recuento) y técnicas de clasificación de objetos mediante varias CNNs como las últimas versiones YOLO. Estas técnicas permiten clasificar trenes, personas y objetos de gran tamaño, implementando una novedosa estrategia para mitigar la oclusión ocasionada por la perspectiva de las cámaras usadas y la conjunción de las dos técnicas que generan las alarmas y pre-alarmas. Para evaluarlo hemos implementado un sistema de control y notificación automatizado con la fusión de técnicas actuales de VxC con unas métricas de 15 FPS de media por video y de latencia por procesado de fotograma de 0,54 segundos, cumpliendo con las expectativas en cuanto a su rendimiento y con respecto al análisis de efectividad y viabilidad del sistema en régimen de normalidad/anormalidad. El sistema obtiene ciertas mejoras en el rendimiento sobre los sistemas actuales, lo cual permite anticiparse a los accidentes y proporcionando más información al operador. Todo esto, da como resultado que el sistema propuesto funciona, es viable, eficaz y económico.

Palabras clave: Visión artificial, visión por computador, visión por ordenador, inteligencia artificial, aprendizaje automático, seguridad, trenes, transporte de viajeros, seguridad ferroviaria.

Abstract

This project investigates, develops and evaluates a system for real-time detection of people and/or objects on railway tracks. After the review of possible alternatives, ideas and existing technologies, detection techniques have been identified in images by background subtraction such as MOG2 (foreground/background segmentation algorithm based on Gaussian mixtures), GMG (combination of statistical estimation of the background image and Bayesian segmentation per pixel), KNN (nearest neighbors), CNT (based on counting) and object classification techniques using various CNNs such as the latest YOLO versions. These techniques allow classifying trains, people and large objects, implementing a novel strategy to mitigate the occlusion caused by the perspective of the cameras used and the conjunction of the two techniques that generate the alarms and pre-alarms. To evaluate it we have implemented an automated control and notification system with the fusion of current VxC techniques with metrics of 15 FPS average per video and latency per frame processing of 0.54 seconds, meeting expectations in terms of performance and with respect to the analysis of effectiveness and feasibility of the system in normal/abnormal regime. The system achieves certain performance improvements over current systems, allowing it to anticipate accidents and providing more information to the operator. All this results in the proposed system being functional, feasible, effective and economical.

Keywords: Artificial vision, computer vision, artificial intelligence, machine learning, passenger transport, railway safety.

Índice

Índice de figuras

Índice de tablas

1. INTRODUCCIÓN

El esfuerzo de este trabajo es la investigación e implementación de una herramienta software que permita la detección de personas y/u objetos en áreas delimitadas como peligrosas en vías ferroviarias, a través de las cámaras de CCTV[1] ya existentes en las instalaciones, generando una alarma automática que pueda detener la circulación de trenes o agilizar la detección de dichos incidentes, notificando a los agentes encargados de la seguridad. Para ello se implementan y evalúan las últimas técnicas de visión por computador, como la detección de objetos y la sustracción de fondo en imágenes.

Nuestra investigación ofrece una contribución significativa al campo de la seguridad ferroviaria, ya que combina dos técnicas innovadoras para lograr una detección precisa y eficiente de objetos, generando alarmas y pre-alarmas en tiempo real con una sola cámara. En comparación con otros trabajos que utilizan múltiples cámaras, dispositivos o sensores adicionales.

1.1. Motivación

Este trabajo busca contribuir a minimizar el impacto de este tipo de accidentes, por caídas, de personas en las vías ferroviarias, así como la caída de objetos a las vías que puedan generar una situación de peligro y la violación de reglas ferroviarias en horas de servicio (grafiteros, vandalismo, cruzamiento de vías, agresiones, etc.). Todo ello perturba la buena circulación de las líneas ferroviarias.

En las instalaciones ferroviarias ocasionalmente se sufren caídas de pasajeros a la plataforma de vía o bajadas sin autorización para recuperar pequeños objetos, resultando en ocasiones heridos o muertos por atropellos. Muchos de los accidentes registrados de este tipo suelen ser por suicidios[2]. De la misma forma los objetos de mediano o gran tamaño pueden ocasionar un choque con el frontal del tren sin que el conductor pueda hacer nada por

[1] CCTV: (Closed Circuit Television, Circuito cerrado de televisión) encargado de las imágenes de seguridad en instalaciones privadas.
[2] https://www.larazon.es/sociedad/20221015/y4wbf3aj3fhbxlxo5umpzxvqaa.html

evitarlo, ya que los frenos de los convoyes están tarados a cierta fuerza de presión de freno para el confort de los pasajeros del interior, evitando así frenadas bruscas. En ocasiones, ni los propios viajeros en el andén son conscientes de que alguien está en las vías hasta que es demasiado tarde. El procesado automático de las imágenes de la cámara nos permite vigilar de forma más continuada y sistemática que la que podría visualizar un humano (evitando fatigas visuales [1] y pueda atender otras funciones de su puesto), clasificando objetos y personas sin interrupciones. Este sistema debería permitir determinar diferentes situaciones de peligro y alertar más rápidamente a los responsables para que tome las decisiones pertinentes.

1.2. Problema y solución propuesta

Las instalaciones ferroviarias cuentan con una cantidad considerable de cámaras de seguridad, pero la mayoría de ellas no tienen ningún tipo de ayuda ni automatismo. Con asistencia de técnicas de Inteligencia Artificial (IA) y visión artificial o visión por computador (VxC) podremos detectar de forma autónoma, la presencia no autorizada en las vías con circulación de trenes, para poder alertar al personal de seguridad, personal de estaciones o los controladores de tráfico, que podrán detener los trenes. De esta manera se evitarían lesiones, accidentes mortales y materiales, de la misma manera se minimizaría el perjuicio ocasionado de pérdida de tiempo a los viajeros hasta que se restablece el servicio normal.

La experimentación se realiza en grabaciones reales de vías ferroviarias de Metro de Madrid S.A. Hoy en día Metro Madrid. S.A. Costa de 302 estaciones y aproximadamente 4800 cámaras[3] de CCTV. Unas 600 cámaras de CCTV aproximadamente pertenecen a la visión de andenes. El personal de seguridad costa de un grupo de trabajadores, que recepcionan y tramitan todas las incidencias de seguridad de la explotación, los cuales avisan a los controladores de circulación de dicha línea. Si se percatan que hubiera algún percance en las vías, dicho controlador hace una llamada de emergencia al tren para que se detenga inmediatamente. De igual forma el personal de estaciones o los propios viajeros con ayuda

[3] Fuente Metromadrid: http://www.metromadrid.es/es/nota-de-prensa/2019-09-10/la-comunidad-invierte-mas-de-44-millones-de-euros-en-mejorar-el-servicio-de-metro-de-madrid

de los interfonos amarillos, cuando perciben alguna situación de peligro, pueden realizar una comunicación al control de estaciones para trasmitir dicha incidencia.

Las comunicaciones de emergencia a veces tardan en llegar o son confusas para indicar la detención al maquinista del tren (el cual, en circunstancias normales, tiene que conducir el tren en modo automático "ATO[4]"). Un tiempo que es crucial en situaciones de peligro, con un intervalo medio entre trenes de 5' por vía, siendo de entrada a las estaciones de menos de 2'30'' aproximadamente.

La solución que estudia este trabajo consiste en un sistema de control y notificación automatizado con técnicas de VxC para que, de todas las circunstancias peligrosas generadas en vía, se notifique de inmediato a los responsables de seguridad, conductor o controlador de tráfico de la línea y poder así evitarlas.

1.3. Objetivos del TFE

El objetivo general de este trabajo es evaluar la viabilidad y rendimiento de la videovigilancia automática de vías ferroviarias.

Este objetivo se descompone en los siguientes objetivos específicos:

1. Desarrollar un software o sistema de videovigilancia automática de las vías ferroviarias que sea viable, eficaz y económico.
2. Revisar de posibles alternativas, ideas y tecnologías existentes de los proyectos ya desarrollados con anterioridad a través del estado de la cuestión.
3. Identificar técnicas de detección en imágenes de objetos en un área acotada, escoger los métodos de detección, para alcanzar los objetivos planteados, analizando su efectividad.
4. Identificar técnicas de clasificación de objetos como trenes, personas, objetos de gran tamaño.

[4] ATO: (Automatic Train Operation, Operación Automática del Tren) modo de conducción del tren de forma automática sin la intervención del conductor.

5. Elegir las estrategias para mitigar la oclusión ocasionada por las personas, con respecto a un área de interés (ROI[5]), por las perspectivas de las cámaras y el ángulo de las imágenes.
6. Implementar la solución propuesta con los métodos de detección elegidos.
7. Experimentación y evaluación de resultados de los métodos elegidos.
8. Validación de la solución propuesta.
9. Sacar conclusiones, líneas futuras y viabilidad de su puesta en funcionamiento.

1.4. Estructura del documento

El resto de este documento se organiza como sigue. El Capítulo 2 revisa el marco teórico y estado actual de las tecnologías base utilizadas para el proyecto las propuestas y soluciones similares existentes en la literatura. En Capítulo 3 contextualización, describe la situación actual del trasporte de viajeros, la soluciones y herramientas existentes. El Capítulo 4 expone la metodología de investigación y desarrollo, la solución propuesta, las métricas utilizadas y los recursos disponibles para la investigación. El Capítulo 5 realiza pruebas de evaluación y rendimiento para validar el software desarrollado. El Capítulo 6 concluye el trabajo revisando los conocimientos y resultados adquiridos, aportando líneas futuras y aporta una valoración personal del proyecto.

[5] ROI: (Region of Interest, Área o región de una imagen), delimitada por un contorno, sobre la que se evalúan determinados parámetros.

2. MARCO TEÓRICO

Este capítulo revisa soluciones similares en la literatura. También se describen los problemas y las tecnologías que han sido necesarias para el desarrollo de esta investigación, así como una breve explicación de conceptos básicos de las redes neuronales convoluciones (CNNs) necesarias para implementar la solución propuesta.

2.1. Estado de la cuestión

La VxC permite transformar las imágenes digitales en comprensibles para un ordenador a través de diferentes algoritmos y técnicas, como puedan ser el procesado de imagen, Machine Learning (ML) y Deep Learning (DL), pudiendo clasificar objetos, determinar zonas de exclusión e identificando situaciones de peligro.

Es por ello que muchos estudios usan diferentes técnicas para detectar personas u objetos en plataformas de vía ferroviarias, como por ejemplo el sistema de vigilancia de andenes usando tecnología de procesamiento de imágenes propuesto por Changmu Lee del Equipo de investigación avanzada de EMU, Instituto de Investigación Ferroviaria de Corea, Uiwang-City, Gyeonggi-Do, Corea [2] plantea mediante el uso de múltiples cámaras en perpendicular a lo largo de la vía, supervisar casi toda la longitud de la línea de vía en el andén.

Changmu Lee, divide el proceso de supervisión de la línea en dos pasos: la detección del estado del tren y la detección de objetos y personas en la vía.

La detección del estado del tren se realiza mediante cámaras repartidas perpendiculares al andén, cada cámara tiene estados diferentes para poder así determinar la aposición del tren a lo largo del andén. El sistema utiliza diferencias de fotogramas, umbralización, etiquetado y fusión. Esta información es combinada con los resultados de detección de los sensores láser, para evitar falsos positivos por ruido en la imagen. Este sistema usa más de cinco fotogramas consecutivos para transitar de un estado a otro.

En la Tabla 1 se muestra la descripción de los estados de la detección del tren, pudiendo tener cuatro tipos de estado para cada cámara:

Tabla 1. Estado de la detección del tren.

Estado del Tren	*Descripción*
OFF	No hay tren en el área de monitoreo
IN	El tren se acerca.
ON	El tren está detenido u ocupa toda el área
OUT	El tren está saliendo del área de monitoreo.

La detección del objeto/persona permite determinar dos posibles sucesos, tal y como se aprecia en la Tabla 2 de los estados de las transiciones del estado del tren. Un objeto caído en la zona y un cambio global repentino en las condiciones de iluminación.

Para determinar el objeto caído, el sistema sólo tiene en cuenta los movimientos en zona de vigilancia en estado OFF (zona en la que no hay tren en el área de monitoreo) y, que procedan de la zona fuera de peligro, haciendo uso de diferentes algoritmos de redes neuronales profundas basados en la técnica de *backtracking*[6], comparando los movimientos de fotogramas anteriores.

[6] Método de retroceso o vuelta atrás, es una técnica general de resolución de problemas, aplicable tanto a problemas de optimización, juegos y otros tipos.

Tabla 2**.** Transiciones de seguimiento del estado de un tren.

Estado del Tren	*Descripción*
OFF -IN	Se han producido más de cinco fotogramas **con** movimiento consecutivos.
IN-ON	Se han producido más de cinco fotogramas **sin** movimiento consecutivos.
ON-OUT	Se han producido más de cinco fotogramas **con** movimiento consecutivos.
OUT-OFF	Se han producido más de cinco fotogramas **sin** movimiento consecutivos.

En el artículo de T. Xiao et al. titulado “Método de Detección de Obstáculos de Tránsito Ferroviario Urbano Basado en Tecnología Multisensorial” [3] se propone un método de detección de tecnología multisensorial sin contacto. Para ello se instala un dispositivo *On-Board*[7] de detección de obstáculos basado en una cámara y un dispositivo LiDAR[8], permitiendo detectar los obstáculos en el área de la órbita en tiempo real, y determinar si el tren debe frenar automáticamente o el maquinista debe frenar manualmente. Finalmente, este comando de control se transmite al centro de control del tren y al conductor del tren, a través del transpondedor, la estación base inalámbrica y otros equipos en la vía en diferentes niveles para realizar la alerta temprana y el frenado del tren.

Recientemente (2022) A. D. Petrović et al. [4] han propuesto una combinación de redes neuronales profundas (YOLOv5 con CNN) para la detección de señales ferroviarias y las propias vías con VxC con método de detección de bordes Canny y trasformada de Hough.

[7] Dispositivos embarcados en el tren.
[8] LiDAR (Light Detection And Ranging) es una tecnología que permite medir distancias desde un emisor láser a un objeto o superficie.

Haixia Pan & Tian [5] proponen una red de aprendizaje multitarea que detecta, clasifica líneas de seguimiento y hace segmentación de líneas. Esta aproximación supone mejoras sobre otras redes multitarea como son la de prestar más atención a la precisión que a la memoria. La segmentación ayuda a los resultados de clasificación. El algoritmo de detección de línea de seguimiento en la red multitarea puede efectivamente lidiar con el problema de la línea de vía bloqueada, evita la desventaja de la red de segmentación en la detección de líneas de seguimiento con una proporción de píxeles más delgada y pequeña. Sus diseños sin anclaje evitan el problema de que el tamaño del marco de anclaje a priori no sea adecuado para objetivos pequeños.

Recientemente en Docklands Light Railway (DLR) del metropolitano de Londres se está experimentando con el desarrollo del proyecto CCTV AI Trial [6]. El sistema de alertas se ha desplegado para prevenir accidentes en las líneas de la red y garantizar que no sean falsas alarmas. Aún este proyecto está en fase de pruebas.

2.2. Tecnologías base

El sistema propuesto se sustenta en dos tipos de tecnologías diferentes combinadas entre sí, para generar pre-alarmas y alarmas, la sustracción de fondo que detecta la caída de cualquier tipo de objeto a la zona de vías y la clasificación de imágenes para detección de objetos, que determina si hay personas en dicha zona y la presencia de tren, usando las siguientes tecnologías.

2.2.1. OpenCV

El proyecto se apoya a su vez en dos librerías muy extendidas en el campo de Machine Learning. La primera librería es **OpenCV** que provee de una infraestructura para aplicaciones de visión artificial. Es multiplataforma y de código abierto, creada por Intel con machine learning y aprendizaje automático. Proporciona métodos para la detección de movimiento, seguimiento de objetos o como la sustracción de fondo, la cual se usará en este proyecto. La librería está escrita en C++, aunque tiene interfaces para varios lenguajes como Python, Java, Etc. Para ejecuciones en GPUs (siendo muy recomendable para su rendimiento) se debe compilar siguiendo unas instrucciones que se describen en el Anexo A

2.2.2. PyTorch

La otra librería es **PyTorch** para Python basada en la librería de Torch de aprendizaje automático también de código abierto, está enfocada a la realización de cálculos numéricos mediante programación de tensores[9]. Tiene la capacidad para ejecutarse en GPUs, desarrollada originalmente por Facebook AI Research (FAER).

2.2.3. Sustracción de fondo

Las técnicas de sustracción de fondo consisten en la resta de una imagen o modelo de fondo, contra otra imagen. Los pixeles diferentes se procesan y representan en una imagen binaria como se muestra en la Figura 1.

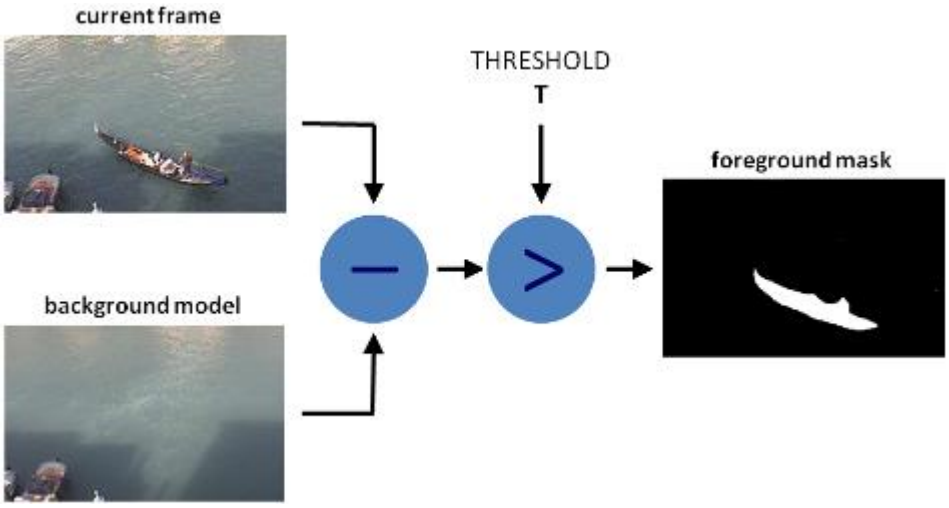


Figura 1. Ejemplo de sustracción de fondo

fuente: https://docs.opencv.org/

La técnica de sustracción de fondo también se usa en videos, para la detección de objetos en movimiento, consiste en descartar el *backgraound* o fondo del *foreground* o primer plano, para captar una escena de video la cámara tiene que estar estática.

Los algoritmos de sustracción de fondo comúnmente usados son:

- **MOG** Es un algoritmo de segmentación de fondo o de primer plano basado en una mezcla gaussiana [7], modela cada píxel del *foreground* con una mezcla de distribución

[9] Tensor: matriz multidimensional que contiene elementos de un solo tipo de datos.

gaussianas de los pixeles vecinos. Los pesos de la mezcla representan las proporciones de tiempo que esos colores permanecen en la escena. Los colores de fondo probables son los que permanecen más tiempo y más estáticos Figura 2.

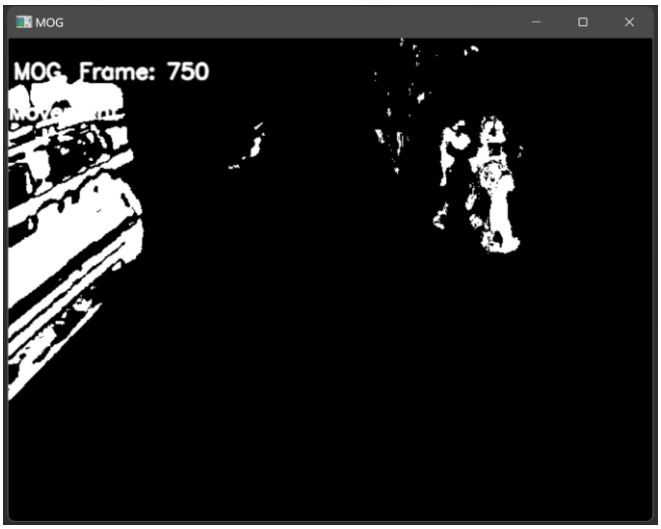


Figura 2. Ejemplo del algoritmo MOG en fotograma 750 del video 1 de pruebas.

- **MOG2** Es un algoritmo de segmentación de fondo o de primer plano basado en una mezcla gaussiana y como mejora sobre MOG, proporciona una mejor adaptabilidad a diferentes escenas debido a cambios de iluminación [8] Figura 3.

Figura 3. Ejemplo del algoritmo MOG2 en fotograma 750 del video 1 de pruebas.

- **GMG** Este algoritmo combina la estimación estadística de la imagen de fondo y la segmentación bayesiana por píxel [9], para el modelado del *backgraound. E*ste algoritmo usa los 120 primeros fotogramas (por defecto) y emplea un algoritmo de segmentación

de primer plano probabilístico que identifica posibles objetos de primer plano mediante la inferencia bayesiana. Las estimaciones son adaptativas, las observaciones más nuevas tienen un mayor peso que las observaciones antiguas para adaptarse a la iluminación variable. Se realizan varias operaciones de filtrado morfológico, como el cierre y la apertura, para eliminar el ruido no deseado. Figura 4.

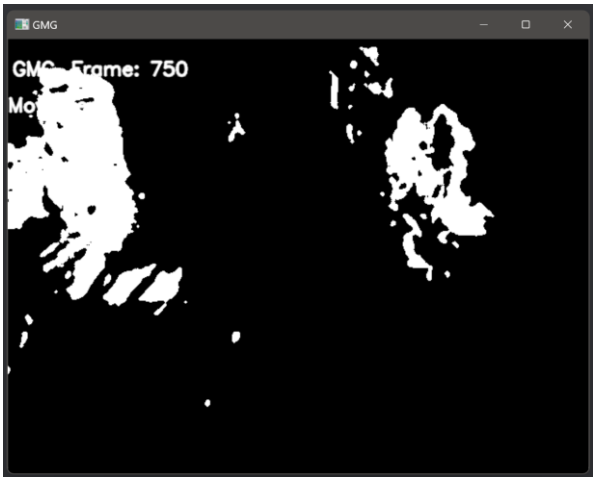


Figura 4. Ejemplo del algoritmo GMG en fotograma 750 del video 1 de pruebas.

- **KNN** Es un algoritmo de segmentación de fondo o de primer plano basado en vecinos más cercanos [10], siendo muy eficiente si el número de píxeles de primer plano es bajo. Figura 5.

Figura 5. Ejemplo del algoritmo KNN en fotograma 750 del video 1 de pruebas.

- **CNT** Resta de fondo basada en conteo, es mucho más rápido que cualquier otra solución de sustracción de fondo en OpenCV sin NVidia CUDA, según su Sagi Zeevi [11] Figura 6.

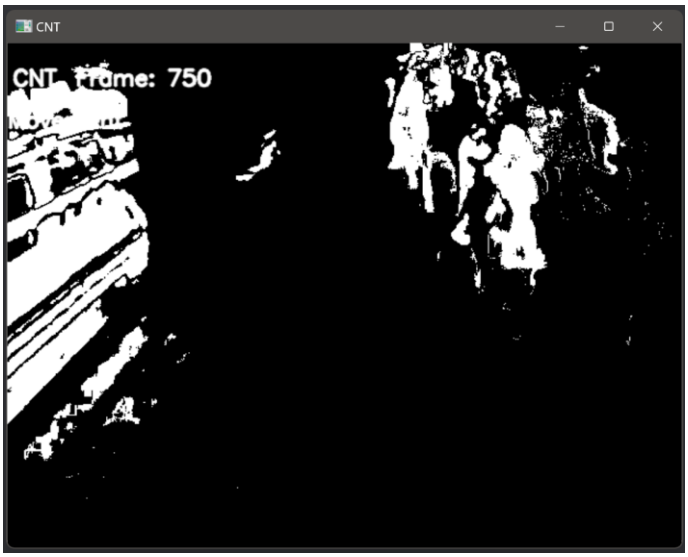


Figura 6. Ejemplo del algoritmo CNT en fotograma 750 del video 1 de pruebas.

2.2.4. Redes neuronales convolucionales (CNN)

Para entender este tipo de redes lo primero que hay que explicar son los conceptos de Kernel o núcleo y el proceso de convolución.

Los ***kernel* o núcleos** son pequeñas matrices de datos como los de la Figura 7, que dependiendo de la distribución de sus valores pueden extraer las características o transformaciones deseadas de la matriz origen tales como, ejes verticales, ejes horizontales, oblicuas, vértices, etc. Estos kernel son matrices de tamaño impar 3x3, 5x5, etc. para facilitar tener un píxel central que será el píxel llamado píxel principal.

-1	0	1
-2	0	2
-1	0	1

Sobel Vertical

-1	-2	-1
0	0	0
1	2	1

Sobel Horizontal

-1	-1	-1
-1	-8	-1
-1	-1	-1

Deteción de ejes

-2	-1	0
-1	1	1
0	0	2

Repujado

0	0	0
1	-4	1
0	0	0

Detectar Bordes

0	0	0
-1	1	0
0	0	0

Realzar Bordes

0	-1	0
-1	5	-1
0	-1	0

Enfocar

0	0	0
0	1	0
0	0	0

Desenfocar

Figura 7. Tipos de kernel 3x3 y sus resultados.

La **convolución** es el proceso que sufre una matriz de datos obtenida a partir de una imagen (tras un pre-procesado cada píxel se normalizará en un dato de la nueva matriz). Dicha matriz destacará sus características dependiendo del kernel o núcleo aplicado. En la Figura 8 se comprueba el proceso tras calcular la suma de los productos de cada valor del píxel del input con el kernel elegido (en el ejemplo kernel del enfoque). Para conseguir una matriz *Output* de igual tamaño que *Input* se usa un método llamado ***Padding*,** que consiste en completar el perímetro de la matriz *Input,* con tantas columnas y filas con valor "0" como haga falta, para que el píxel principal del kernel coincida con el primer píxel de la matriz *Output*.

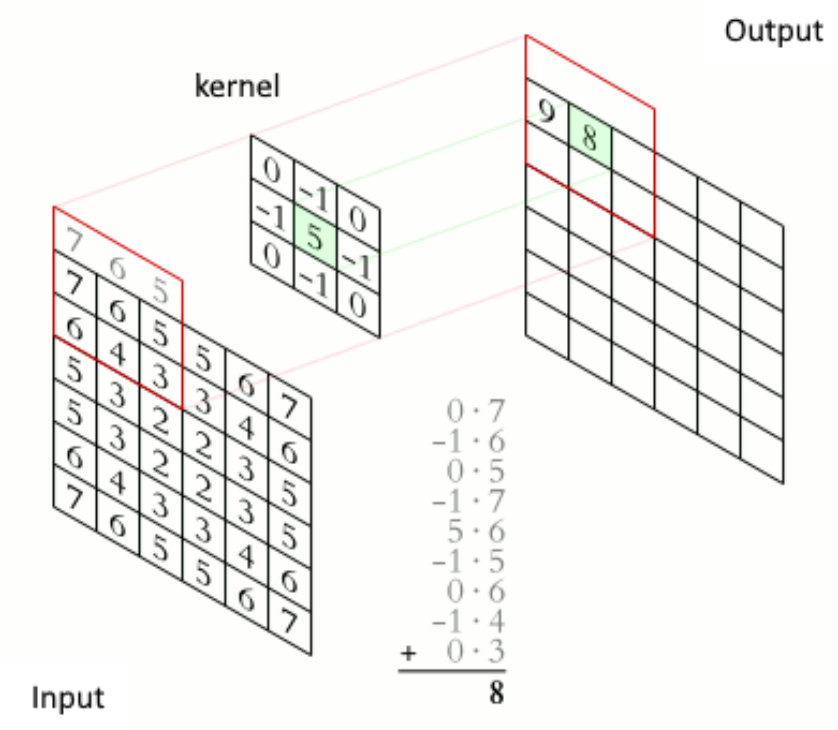


Figura 8. Ejemplo simple de convolución.

Fuente: Wikipedia

De forma matemática podemos definir la operación de convolución a partir de esta ecuación:

$$g[x,y] = \sum_{s=-a}^{a} \sum_{t=-b}^{b} w[s,t] f[x-s, y-t]$$

Siendo ***f*** la matriz origen y ***w*** la matriz kernel, recorridas desde **-a** hasta **a** en el caso de la matriz origen y desde **-b** hasta **b** en la matriz kernel.

La Figura 9 muestra ejemplos del resultado de convoluciones de imágenes aplicando diferentes kernels.

Original	**Eje Básico**	**Desenfoque Gaussiano**	**Sobel Horizontal Vertical**
			

Figura 9. Ejemplo de aplicación de kernels.

En una red convolucional se usan varios kernels. A un conjunto de matrices de salida se le llaman comúnmente filtros o ***feature mapping***, cada filtro usado será una neurona en la primera capa oculta de neuronas, a la que se le aplicará una función de activación.

La función de activación más comúnmente utilizada en las capas ocultas de este tipo de redes es la **función de no linealidad o ReLU** (Rectifier Linear Unit) siendo su ecuación la de la Figura 10:

$$\boldsymbol{f(x) = \max(0, x)}$$

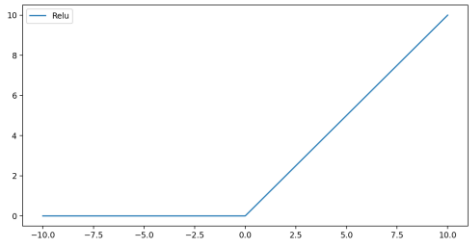


Figura 10. Gráfica de ReLU

Después de aplicar la función de activación se aplica una reducción de las neuronas a través de lo que se conoce como **subsampling**. Esto se hace porque si se usaran todas las nuevas salidas, para las entradas de una nueva capa se haría una red inabarcable. El subsampling más común es el ***Max Pooling*** **o capa de agrupación**, que consiste en reducir la salida de cada filtro, eligiendo el valor "más alto" de los pixeles, este representa mejor la característica de dicha región de la imagen, lo que suele corresponder al píxel más intenso en la salida del filtro. Lo que a su vez mejora la precisión de la red neuronal en tareas de detección de objetos, generando una matriz de tamaño más reducido, como se muestra en le Figura 11.

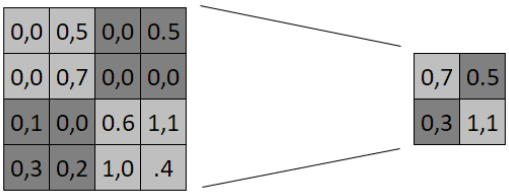


Figura 11. Subsampling de tipo Max Pooling.

En las siguientes capas ocultas se aplicará los kernel para obtener la convolución. Se obtendrá un nuevo feature mapping, al que se aplica el Max-Pooling para obtener una nueva salida que será la entrada de la siguiente capa oculta.

Por último, se aplica una función llamada ***Softmax*** que será la encargada de la clasificación en probabilidad de neuronas de salida o, dicho de otro modo, normalizar la salida de una red a una distribución de probabilidad sobre las clases de salida señaladas. De forma matemática podemos definir la función a partir de esta ecuación:

$$S(y_i) = \frac{e^{y_i}}{\sum_j e^{y_i}}$$

Siendo y_i el valor a normalizar de los valores reales en el rango [0, 1].

Una vez elegida la cantidad de capas ocultas se puede aplicar la técnica de ***dropout***. El dropout es la técnica por la cual se desactivan neuronas aleatorias en cada iteración de la red, esta técnica previene el llamado overfitting o sobre aprendizaje, evitando que la red caiga en memorización de los datos de entrenamiento. Si estos últimos son escasos o no se ha entrenado la red lo suficiente. Si el modelo de red cae en overfitting no generalizará bien con nuevos datos. La Figura 12 muestra a la derecha, una red neuronal estándar con dos capas ocultas, a la izquierda, una red neuronal después de aplicar dropout.

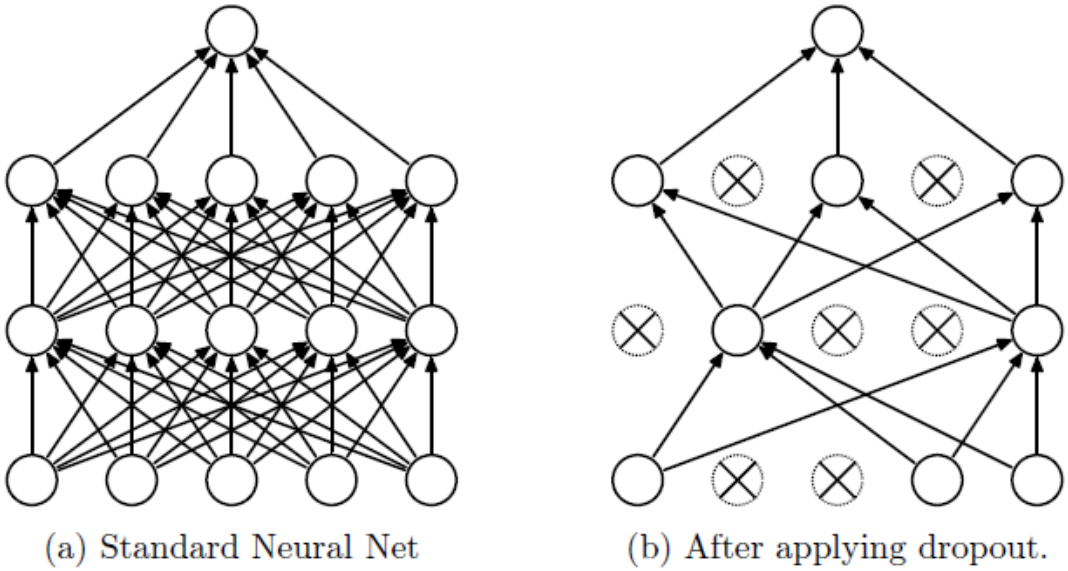


Figura 12. Ejemplo de aplicación de método dropout.

Fuente: "Dropout: A Simple Way to Prevent Neural Networks from Overfitting" [12]

En resumen, como se puede comprobar en la Figura 13 la arquitectura completa de una red neuronal convolucional está compuesta de múltiples capas después de aplicar los ***kernel*** o filtros elegidos. Dichas capas se pueden combinar siempre en el orden **Conv+ReLU**, que generará tantos volúmenes como filtros aplicados en un ***feature maps***, al pasar por la capa de ***pooling*** que reduce el tamaño de los volúmenes no afectando a su profundidad. Esta secuencia se puede combinar tantas veces como el arquitecto de la red vea conveniente, la capa de ***fatten*** trasformar las matrices en vectores aplanando los datos de todos los volúmenes. La capa ***fully connected*** conecta por completo a todas las neuronas de salida o tipos de clases a predecir hasta la capa de ***Softmax.***

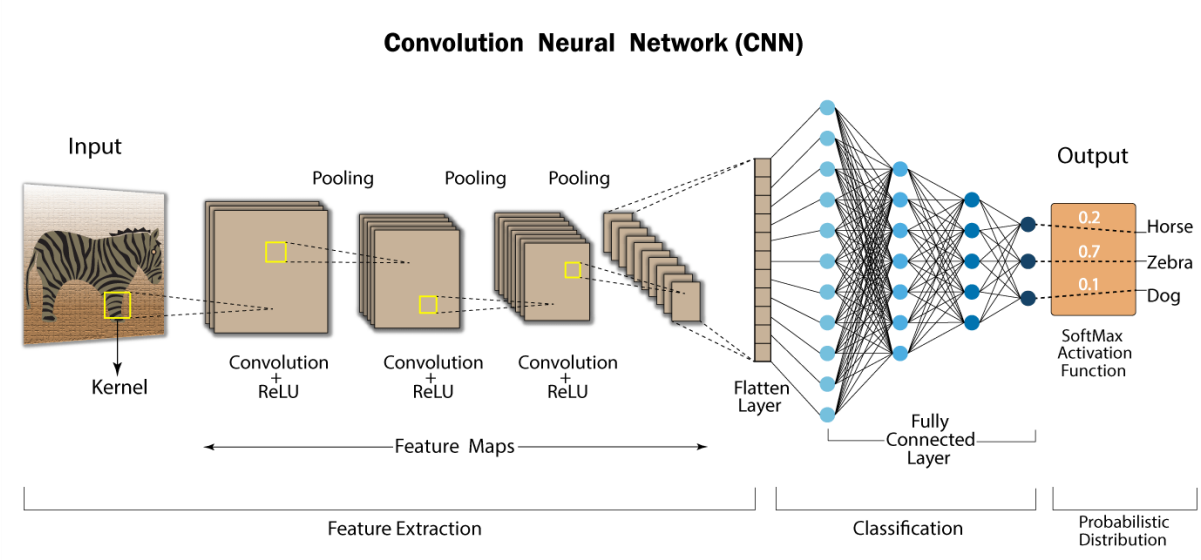


Figura 13. Arquitectura de las redes neuronales convolucionales.

Fuentes: Medium.com

2.2.5. YOLO

YOLO (You Look Only Once), el nombre del sistema surge porque para clasificar/predecir los objetos (**Class**) y determinar la región de estos (**BBox**[10]), solo se procesa la imagen una vez. Como se puede ver en el esquema de la Figura 14, la imagen se procesa primero en las capas de convolución que generan los Feature Maps (mapas de características), después se pasa a las Fully Connected (capas que conectan por completo a todas las neuronas de salida o tipos de clases a predecir), que dan como resultado las Class y los BBox solamente en una iteración. Las redes anteriores procesaban la imagen una vez para clasificar/predecir y otra vez para determinar la región.

[10] BBox: del inglés "bounding box" es un término utilizado en el ámbito de la visión artificial y el procesamiento de imágenes para describir un rectángulo que rodea un objeto de interés en una imagen o video. En general se usa como una forma de delimitar un objeto dentro de una imagen o video, permitiendo una detección y seguimiento de objetos en movimiento, clasificación de objetos, entre otras aplicaciones.

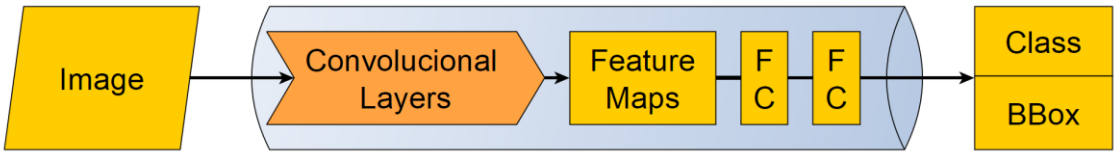


Figura 14. Esquema YOLOv1.

Las tres primeras versiones del sistema YOLO, YOLOv2, YOLOv3 fueron desarrolladas por Joseph Redmon [13] [14] con **Darknet,** una única red neuronal convolucional (**CNN**) de código abierto escrita en C y CUDA[11]. La explicación del funcionamiento básico de este tipo de redes se realizó en la sección Redes neuronales convolucionales (CNN).

La versión YOLOv4 la desarrollo Alexey Bochkovskiy [15], reescrita en PyTorch, pero aun con Darknet como CNN. Como mejoras sobre las anteriores versiones se puede destacar el tiempo de inferencia del modelo, también se optimiza la eficiencia del cálculo, siendo así mayor la complejidad de la estructura con dos etapas de detección. La Figura 15 muestra estas dos capas con la típica estructura de módulos de redes neuronales. La primera etapa o preliminar, para detectar regiones de importancia. En el módulo **Backbone** una red **CSPDarknet53**, que aumenta la capacidad de aprendizaje y que gracias a la unión del módulo de agrupación de pirámide espacial permite mejorar el campo receptivo y distinguir características importantes. En el módulo **Neck** módulo de agrupación de pirámides espaciales y agregación de ruta **PANet**, principalmente. PANet se implementa como sustitución de las redes piramidales de características empleadas para la detección en YOLOv3. Por último, el módulo **Head** como YOLOv3 [16], que engloba a la vez la primera y segunda etapa, la segunda etapa aporta la capa densa llamada ***Sparse Prediction***, la cual se encarga de clasificar si se ha detectado un objeto en la región de importancia.

[11] CUDA: plataforma de computación paralela y un modelo de programación desarrollado por NVIDIA para la informática general en unidades de procesamiento gráfico (GPU).

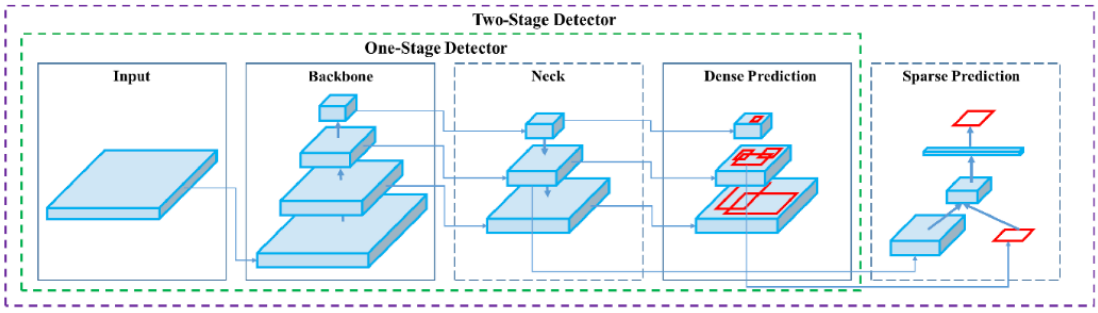


Figura 15. Esquema de las 2 etapas en YOLOv4.

Fuente: «YOLOv4: Optimal Speed and Accuracy of Object Detection» [15]

La versión YOLOv5 Ultralytics[12] de Glenn Jocher, usada para este trabajo, está escrita en PyTorch con API´s para el lenguaje de programación Python. En esta versión se ha mejorado el canal de capacitación de imágenes con nuevas estrategias de aumento y ajustes en la arquitectura con una sola capa, siendo mucho más fácil de utilizar, por ser la más probada, fácil y rápida de implementar. Actualmente es uno de los mejores sistemas en detección y clasificación de imágenes, avalado por la comunidad científica y citado en multitud de trabajos de investigación. Actualmente en la plataforma ***Github***[13] cuenta con cerca de 32.000 estrellas y 11.500 fork (bifurcación del código) que avalan su popularidad. Esta versión de YOLO puede discriminar entre 80 tipos de objetos, detecta menos falsos positivos, siendo uno de los sistemas de detección más rápidos y fiables.

La arquitectura de YOLOv5 no ha sido publicada[14] pero a través de la herramienta TensorBoard podremos extraer su arquitectura para el estudio, como se muestra en la Figura 16, donde se aprecia los diferentes módulos de las redes neuronales. El módulo Backbone consta de una serie de capas de filtros convolucionales intercaladas con capas de agrupación que van disminuyendo su tamaño a medida que se avanza en la red, lo que permite la extracción de características importantes a diferentes escalas. Es en la capa UpSampling2D, donde empieza el módulo Neck, la cual realiza un sobremuestreo de entrada

[12] YOLOv5: https://ultralytics.com/about
[13] Github: famoso repositorios de código online para gestión de proyectos y control de versiones (VCS) perteneciente a la empresa Microsoft.
[14] https://github.com/ultralytics/yolov5/issues/1333

se encarga de aumentar el tamaño de las características extraídas por la fase backbone. Esto se hace mediante una interpolación, es decir, se crean nuevos pixeles intercalados entre los pixeles originales para aumentar el tamaño de la imagen. La capa de UpSampling2D se utiliza en YOLOv5 para mejorar la precisión de la detección de objetos. Se obtiene una mejor resolución de las regiones de la imagen que contienen los objetos a detectar. Esto permite una mayor precisión en la detección de estos objetos. Por último, el módulo Head costa de capas que se encargan de clasificar si se ha detectado un objeto en la región de importancia.

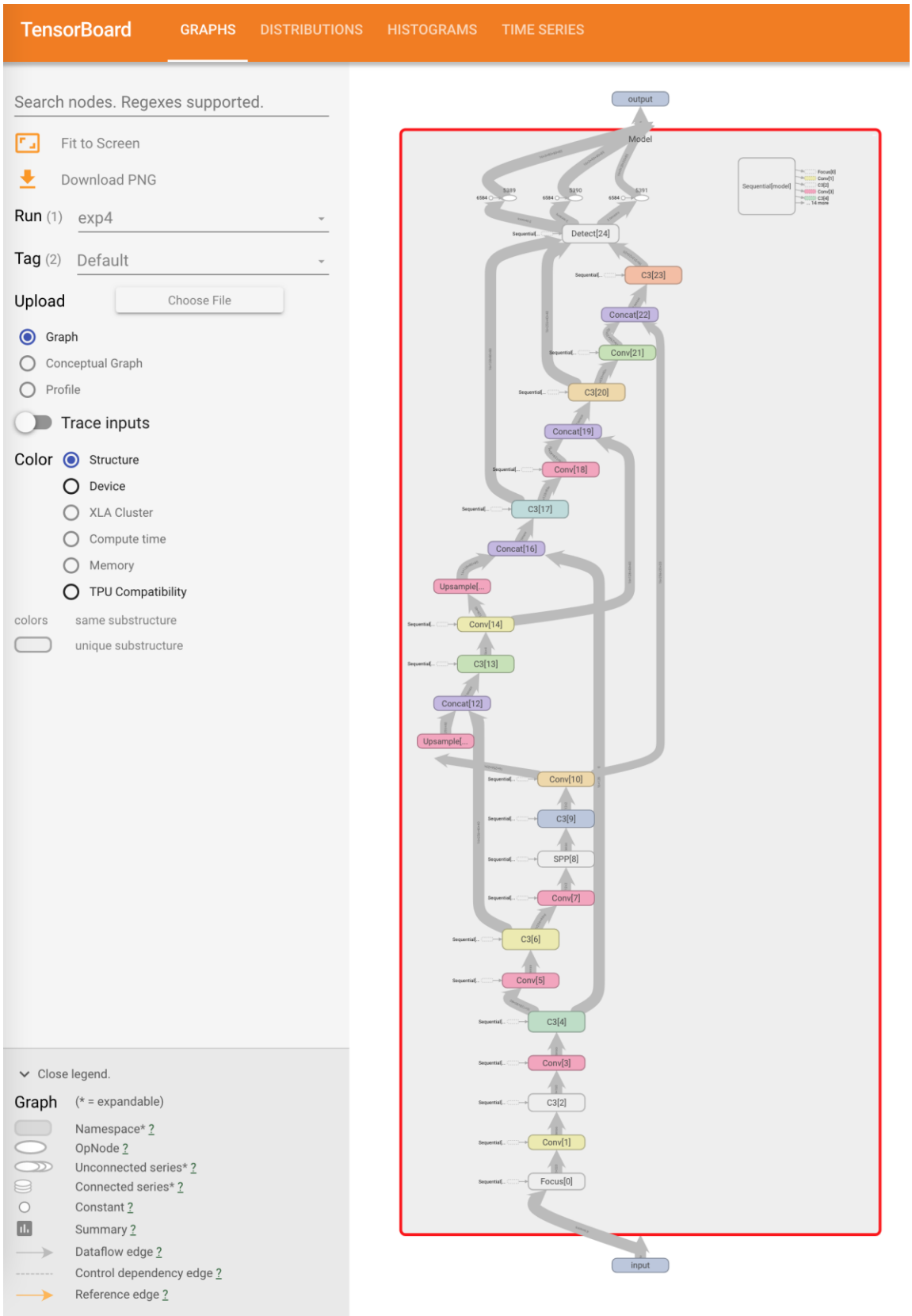


Figura 16. Estructura de YOLOv5 extraída con la herramienta TensorBoard.

Fuente: https://github.com/

En la Tabla 3 se compara las diferentes versiones de YOLO, el lenguaje de implementación y las diferentes redes CNN utilizadas.

Tabla 3. Comparación de versiones YOLO.

	Implementación	*Redes empleadas*
YOLO	C, CUDA	Googlenet
YOLO9000	C, CUDA	Darknet-19
YOLOv2	C++	Darknet
Fast-YOLO	C++	Darknet
YOLOv3	C++	Darknet
YOLOv4	PyTorch	AlexeyAB/darknet
YOLOv5	PyTorch	YOLO v5l, YOLO v5x, YOLO v5m y YOLO v5s

3. CONTEXTUALIZACIÓN

A continuación, describiremos la situación de los factores que influyen para el buen desarrollo de la idea principal del proyecto. Para realizar esta contextualización conviene resaltar la magnitud de la empresa de transportes para más adelante asemejar otras herramientas y tecnologías existentes para solucionar el problema propuesto.

3.1. El trasporte de viajeros

El transporte público ferroviario de viajeros es cada vez más importante en las grandes ciudades, siendo uno de los preferidos por los ciudadanos para su movilidad, por la reducción del tiempo de desplazamiento, el cuidado del medio ambiente, siendo el medio más seguro, accesible y económico. También gracias a su fiabilidad y sostenibilidad con respecto a los sistemas de transporte privados. Con respecto a otros medios de transportes públicos, el transporte ferroviario y más concretamente la red de metro, se consolida año tras año como el más usado por los madrileños, como se pude comprobar en la Tabla 4 comparativa.

Tabla 4. Demanda de transporte público, billetes por modos de transporte.

Fuente: http://www.madrid.org/

	2016	*2017*	*2018*	*2019*	*2020*
Red de metro	587,6	629,4	660,2	680,2	349,8
Red de autobús (EMT)	430,1	427,9	420,2	439,8	241,6
Red de concesiones carretera	219,8	227,6	236,1	248,7	143,8
Autobuses interurbanos	177,2	182,2	188,9	199,3	116,2
Autobuses urbanos	42,6	45,4	47,2	49,4	27,6
Red de metro ligero	15,8	16,9	18,2	18,8	9,5
Red de ferrocarril de cercanías	184,6	192,5	203,4	203	109,4

Según fuentes de MetroMadrid. S.A.[15] la afluencia de viajeros actualmente va estabilizándose después de los años atípicos 2020 y 2021 tras la crisis sanitaria provocada por la pandemia de COVID-19. La tendencia en 2022 va en aumento y el crecimiento es positivo alcanzando el 63,7% recuperando la demanda de junio 2019, siendo así los datos de 2019 los más normalizados con 2,2 millones al día y 677,5 millones al año de viajeros, como se muestra en la Figura 17.

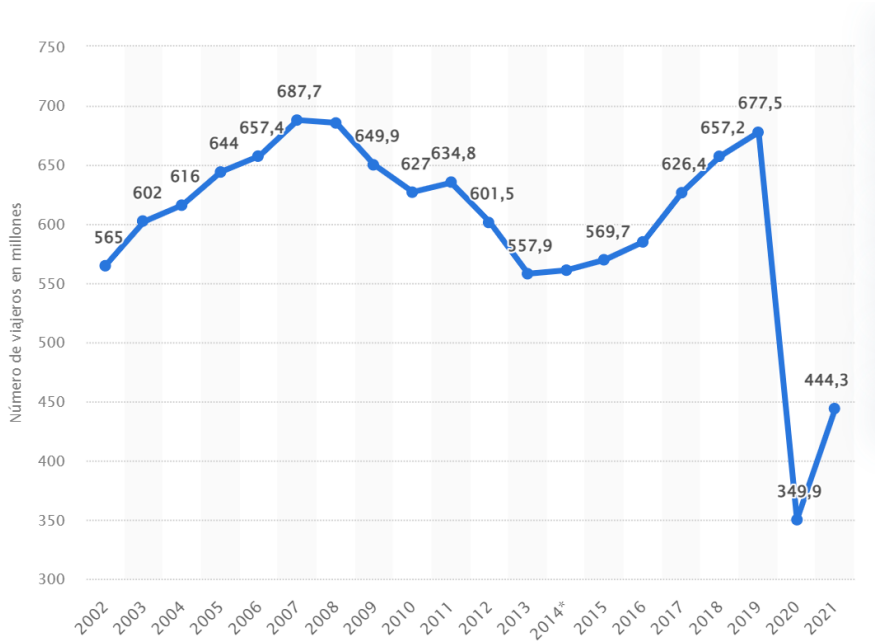


Figura 17. Millones de viajeros anuales 2012-2021.

3.2. Incidentes con viajeros

Con tal cantidad de viajeros diarios son muchos los incidentes por los que la circulación de trenes es detenida, provocando pequeños retrasos e inconvenientes en los intervalos entre trenes y esperas en el andén. Los incidentes que más repercusión tienen en la circulación de trenes son los derivados por caídas de objetos a la vía y/o personas no autorizadas en plataforma de vías como los generados por vandalismos, grafiteros, cruzamiento de vías, agresiones e intentos de suicidio, por los cuales se activan los protocolos de seguridad para

[15] https://www.metromadrid.es/es/nota-de-prensa/2021-07-24/la-comunidad-de-madrid-registro-casi-38-millones-de-viajes-en-metro-en-junio-la-cifra-mas-alta-tras-la-crisis-del-covid-19

evitar accidentes personales y materiales. Si hubiera un accidente generado por alguna de estas causas se suspende el servicio de viajeros en la zona afectada de línea, por un tiempo muy prolongado, hasta que se pueda restablecer el servicio con normalidad.

3.3. Suicidios

Las muertes por suicidio en España van en aumento en los últimos años, causados más aún si cabe también por la crisis sanitaria, como se puede extraer de los datos del INE (Instituto nacional de estadística) [17]

Estas incidencias, las conductas suicidas, son por desgracia bastante comunes en las vías de trenes metropolitanos y en vías ferroviarias. No existen datos que se puedan referenciar ni se comunican en las noticias por seguir las recomendaciones que aconseja la Organización Mundial de la Salud en su informe "Prevención del suicidio: un instrumento para profesionales de los medios de comunicación" [18] para evitar lo que se conoce por "*efecto contagio*", *"efecto llamada" o* "*efecto Werther*[16]". Pero solo hace falta hacer una búsqueda rápida en cualquier motor de búsqueda web con las palabras clave "arrollamientos metro", "suicidios metro" o "accidente con viajeros en metro" y comprobar que los resultados relevantes son abrumadores. Siendo aproximados 50 suicidios o accidentes al año, aproximadamente 1 a la semana, aunque en ciertas épocas del año se incrementan, como en vacaciones estivales o en épocas señaladas como semana santa o navidad.

Es por ello, la idea principal de este trabajo pueda evitar la mayoría de los accidentes e intentos de suicidio, alertando antes de que se produzca el arrollamiento o incluso pudiendo detener los trenes de forma automática.

3.4. Soluciones y herramientas existentes

Son muchos los intentos por paliar este tipo de accidentes que afectan a la sociedad y a todas las empresas e instalaciones ferroviarias, pudiendo separarlas en dos categorías, las arquitectónicas y las proporcionadas por software.

[16] *Efecto Werther:* suicidio imitativo inspirados o inducidos por los medios de comunicación.

3.4.1. Arquitectónicas

En Japón, una de las ciudades con mayor tasa de suicidios del mundo (2013 se quitaron la vida 27.195 personas, muchos de ellos en las instalaciones ferroviarias), lleva años aplicando una controvertida solución para disuadir los actos suicidas, instalar lámparas LED azules en los andenes de las estaciones, como se puede ver en la Figura 18, reduciendo los suicidios hasta en un 84% como demuestra el estudio de Tetsuya Matsubayashia, Yasuyuki Sawadab y Michiko Ueda [19] apoyados en los trabajos de la Universidad de Granada [20] el cual concluye que la luz azul aceleraba los procesos de relajación, la universidad de ciencias médicas de Shiga, que cita "haber una mayor proporción de intentos de suicidio en trenes después de varios días sin luz solar" [21] y el estudio “papel de los colores en la reducción del estrés” [22]

Figura 18**.** Andén de la línea Shinagawa/Shibuya con iluminada con lámparas azules.

Fuente: Damon Coulter

Otra de las soluciones propuestas en Tokio, es la instalación de barreras a la altura del pecho en los andenes, Figura 19, una forma más eficaz de proteger a los viajeros, pero también

más costosas, no pudiendo instalar en todas las estaciones por espacio o porque la fuerza estructural de los andenes no lo permite.

Figura 19. Barreras a la altura del pecho en un andén del metro de Tokio.

Fuentes: Alamy

En otras ciudades como Sevilla, Londres, etc. se ha optado por instalar puertas de andenes. En MetroMadrid S.A. se encuentra en fase de pruebas.

Las puertas de andén son unas mamparas acristaladas instaladas a lo largo de todo el andén, con puertas que abren sincronizadas con las puertas de los trenes, como se pueden ver en Figura 20. Ayudan a la climatización y sonoridad de la estación, aparte de proteger de accidentes y/o por caídas a las vías. Estas tienen sus inconvenientes, como que no en todas las líneas circulan los mismos modelos de trenes, por lo cual, cada modelo de tren tiene las puertas distribuidas a distancia distintas a los otros modelos, también su costosa instalación llegando a costos superiores al millón de euros por estación.

Figura 20**.** Mamparas de seguridad instaladas en fase de prueba en Metro Madrid.

Fuente: https://elpais.com

También existen soluciones con láser o dispositivos LIDAR detección de intrusión (como el REDSCAN RLS-3060SH, Figura 21) diseñados específicamente para detección de eventos críticos, proporcionando una cortina virtual personalizable, capaz de hacer frente a condiciones de iluminación complejas, incluidos los faros del tren, sombras, contraste variable y zonas oscuras. Protegiendo las zonas de entrada a túneles, acceso a plataforma de vías desde los andenes d estaciones de metro, y protección de cruces ferroviarios.

Figura 21. REDSCAN RLS-3060SH de la empresa optex-europe

Fuente: https://www.optex-europe.com/

3.4.2. Proporcionadas por Software

Las soluciones por software son muchas y muy variadas, siendo más baratas y fáciles de instalar, pero a la vez también mucho más genéricas. Se engloban en las llamadas VMS[17] sistemas de administración de videovigilancia, como la de la empresa exacq se puede ver en la Figura 22. Comúnmente son aplicativos de tipo cliente servidor, no suelen tener las características para almacenar videos, pero se integran bien con los servidores dedicados como los NVR/DVR[18], llamando a este conjunto los CMS[19]. Los VMS disponen de módulos par análisis de contenido de video, gestión de alarmas técnicas, gestión de alarmas de seguridad, procesos automatizados según el contexto de las imágenes analizadas, mapas esquemáticos interactivos, gestión de archivos de video, video-vallas o videoswalls.

[17] VMS: por sus siglas en inglés Video Management System, Sistema de gestión de vídeo.
[18] NVR/DVR: por sus siglas en inglés Network Video Recorder o Digital Video Recorder, videograbadores videos digitales.
[19] CMS: por sus siglas en inglés Central Management System, Sistema de gestión central.

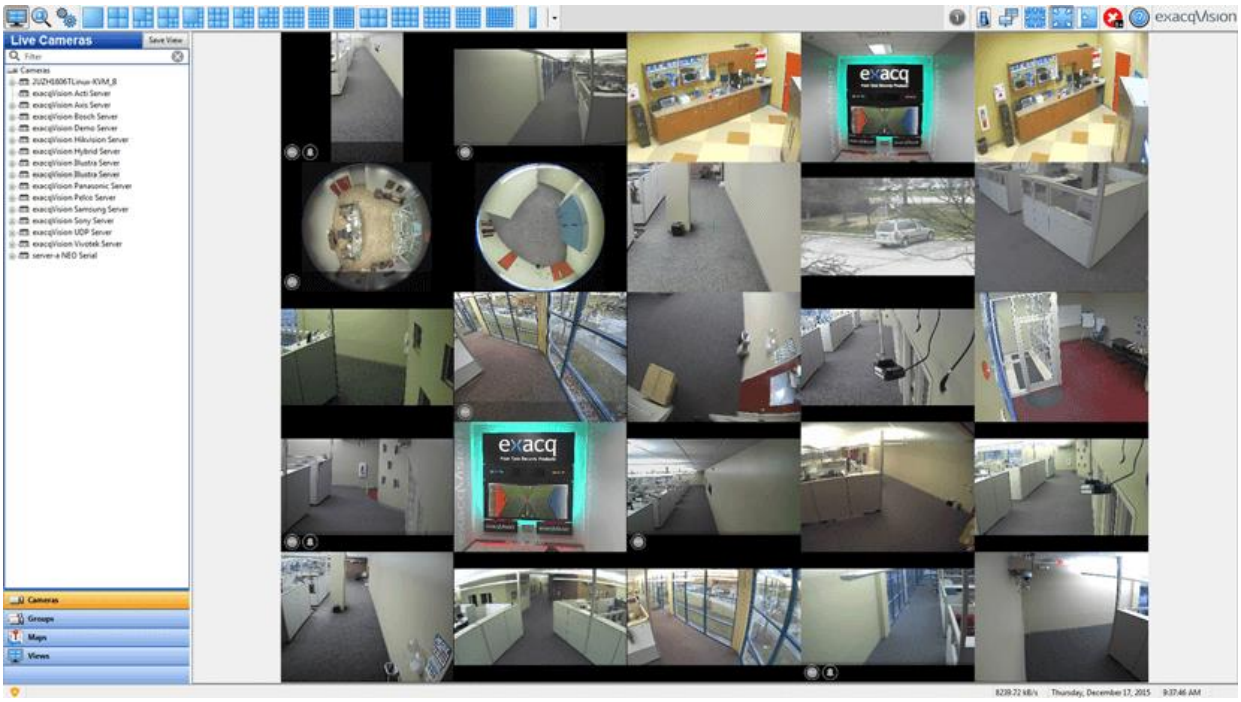


Figura 22**.** Captura de pantalla del VMS de la empresa exacp.

Fuente: https://www.exacq.com/

4. DISEÑO DE LA PROPUESTA

En este capítulo se desarrolla el diseño de la propuesta, explicando en primer lugar la metodología de investigación utilizada, las decisiones tomadas para el diseño de la arquitectura que dan forma al sistema y por ultimo las decisiones en diseño de software para la implementación del código.

4.1. Metodología de investigación

El sistema se divide en dos técnicas de detección, descritas en la sección de Tecnologías base, la sustracción de fondo y la clasificación de objetos en imágenes a través de una red neuronal convolucional (CNN). Para evaluar que técnica de sustracción de fondo se adecua mejor a las necesidades del sistema, se han realizado varias pruebas e iteraciones con las técnicas más avanzadas y actuales comparando sus resultados en términos de precisión y velocidad de procesado de las imágenes por lo cual, se ha elegido el algoritmo MOG2. En la clasificación de imágenes y partiendo del estudio de la literatura existente de las diferentes redes neuronales que desempeñan dicha tarea, en la que se ha elegido YOLOv5. También se compara con respecto a sus versiones anteriores a nivel de precisión y velocidad de procesado de las imágenes.

Para los ejemplos se usa el dataset de video de pruebas del proyecto. En la Figura 23 se muestra un ejemplo de un fotograma del video nº 1 de dicho dataset.

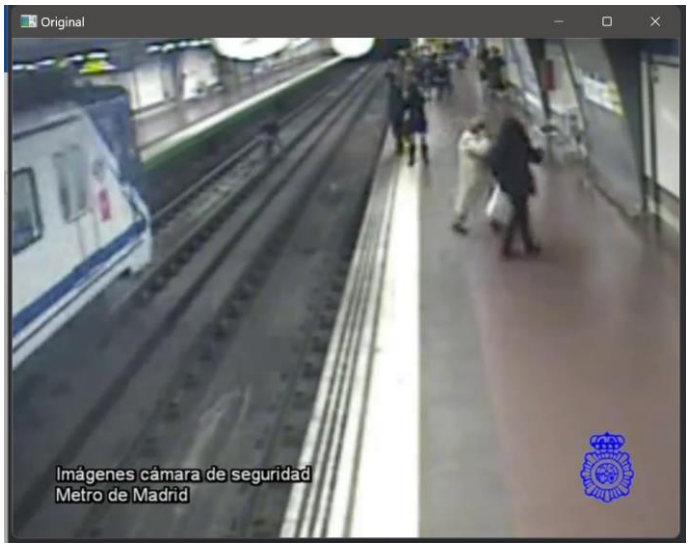


Figura 23. Video del set de pruebas del proyecto.

4.1.1. Evaluar algoritmo para sustracción de fondo

Es importante tener en cuenta que no existe un algoritmo de sustracción de fondo "ideal" que funcione en todos los casos, ya que cada problema es único y puede requerir un enfoque diferente. Por lo tanto, es importante evaluar cuidadosamente las opciones disponibles y seleccionar el algoritmo que mejor se adapte a las necesidades específicas del problema en cuestión.

Para evaluar el algoritmo de sustracción de fondo que mejor desempeño pueda tener en el proyecto existen varias formas, una forma común es medir su precisión, es decir, cuántos errores comete el algoritmo al identificar el fondo en una imagen o video. Esto se hace mediante la comparación de los resultados del algoritmo con una imagen o video de referencia que ya ha sido etiquetado de manera manual para mostrar el fondo correcto. Otro aspecto por evaluar del algoritmo de sustracción de fondo es medir su velocidad de procesamiento, es decir, cuánto tiempo tarda en procesar una imagen o video. Esto es importante en situaciones en las que se requiere un rendimiento rápido, como en aplicaciones en tiempo real como es el caso de este proyecto. Además, también se evalúa la facilidad de uso y la flexibilidad del algoritmo, es decir, su capacidad para adaptarse a diferentes tipos de imágenes y videos.

4.1.2. Evaluar algoritmo para la clasificación de objetos

Para este algoritmo existen comparaciones y evaluaciones con diferentes métricas ya publicadas, como se puede apreciar en el gráfico de curva AUC-ROC[20] de la Figura 24 el modelo YOLOv5x6 con los pesos YOLOv5x6.pt previamente entrenado desde https://pytorch.org/hub, es el que mejor desempeño demuestra en velocidad y mAP[21] [23] en el conjunto de datos COCO[24] de 5000 imágenes en varios tamaños de inferencia de 256 a 1536.

[20] AUC-ROC: representación gráfica de la sensibilidad frente a la especificidad para un sistema clasificador binario según se varía el umbral de discriminación.
[21] AP: Métricas confusas de Precisión Promedio, es un tipo de métricas de rendimiento usado para evaluar un determinado algoritmo como los de detección de objetos, segmentación semántica, segmentación de instancias.

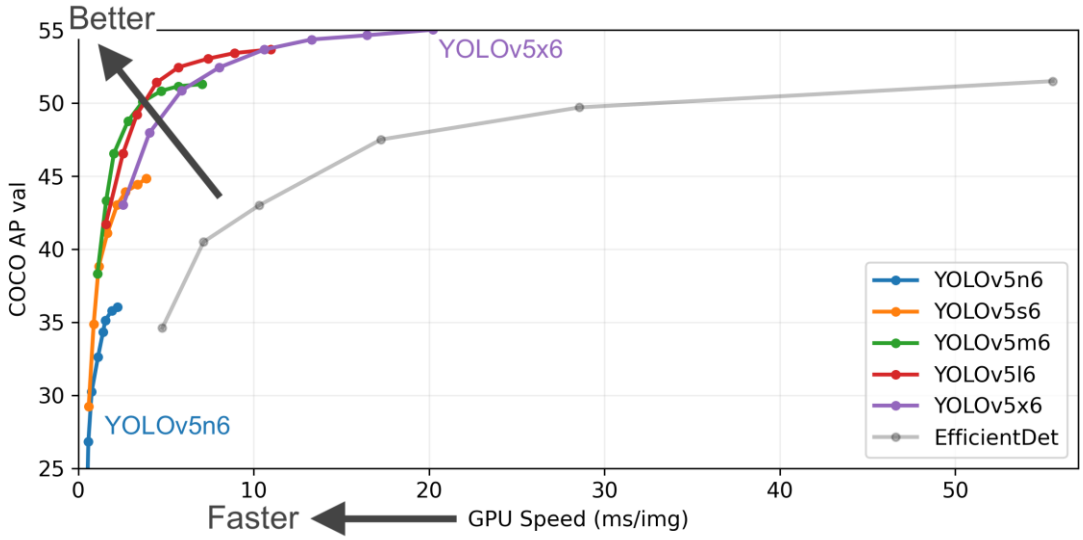


Figura 24. Métrica de diferentes pesos en el conjunto de datos COCO val2017.

Fuente: https://github.com/ultralytics/yolov5

En los últimos meses nuevos sistemas de YOLO están siendo desarrollados, YOLOv6 [25], YOLOv7 [26], YOLOv8, YOLOX, PP-YOLO y PP-YOLOv2, PP-YOLOE, con pequeñas mejoras y por ahora con menos popularidad que YOLOv5.

Se ha elegido YOLOv5 para el proyecto debido a sus ventajas en términos de precisión y velocidad en la detección de objetos.

En comparación con otros algoritmos de detección de objetos, YOLOv5 ha demostrado tener una alta precisión en la detección de objetos en diferentes tipos de imágenes y videos. También tiene una alta velocidad de procesamiento, gracias a implementación en CUDA, lo que lo hace adecuado para aplicaciones en tiempo real, es flexible y fácil de usar. Es por ello por lo que se ha elegido usar en este proyecto.

4.1.3. Métricas empleadas

En esta sección se describen las métricas que se utilizan para evaluar el sistema propuesto. Estas métricas son herramientas fundamentales para medir el rendimiento del sistema y determinar su eficacia en la tarea específica para la cual fue diseñado.

4.1.3.1. Sustracción de fondo

Para la técnica de sustracción de fondo se ha tomado como métrica la cantidad de píxel diferentes de cero por fotograma, la cual nos indica la efectividad del algoritmo en la

detección de objetos en movimiento. Un alto número de pixeles diferentes de cero indica que el algoritmo ha sido capaz de detectar eficazmente los objetos en movimiento y separarlos del fondo estático. Por otro lado, un bajo número de pixeles diferentes de cero indica que el algoritmo ha tenido dificultades para separar los objetos en movimiento del fondo estático. Sin embargo, esta métrica es solo un indicador, ya que puede haber casos en los que se detecten objetos que en realidad no son objetos, especialmente en escenas con mucha iluminación o ruido.

Para evaluar del algoritmo de sustracción de fondo es imprescindible medir su velocidad de procesamiento, ya que son muy diferentes los resultados dependiendo del algoritmo usado.

4.1.3.2. Clasificador de objetos

Para el clasificador de objetos se usa una matriz de confusión comúnmente utilizada en la evaluación de modelos de clasificación, la cual utiliza la siguiente serie de indicadores:

- **TP o True positives (predicción positivos y verdaderos)**: número de fotogramas en los que tanto la **predicción** como la **realidad** están de acuerdo en la presencia de objetos.
- **TN o True negative (predicción negativos y verdaderos)**: número de fotogramas en los que tanto la **predicción** como la **realidad** están de acuerdo en la **no** presencia de objetos.
- **FP o False positives (predicción negativos y falsos)**: Número de fotogramas en los que la **predicción** contiene al menos un objeto mientras que la **realidad** no contiene ningún objeto.
- **FN o False negative (predicción negativos y falsos)**: Número de fotogramas en los que la **realidad** contiene al menos un objeto mientras que la **predicción** no contiene ningún objeto.

Obteniendo dos tipos de errores, error tipo I y error de tipo II como se puede comprobar en la Tabla 5.

Tabla 5. Tipos de error.

		Predicción	
		Positivos	Negativos
Actual Real	Positivos	TP	FN (Error de tipo II)
	Negativos	FP (Error de tipo I)	TN

A partir de la matriz de confusión se extraen las siguientes métricas:

La métrica de **precisión** es el número de veces que el modelo clasifica correctamente un elemento de una clase dividido por el número total de veces que se clasifica esa clase:

$$precisión = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FP}}$$

La métrica de **exactitud** es lo cerca que se está del resultado actual real o de la medición manual, se calcula como la proporción de resultados verdaderos dividido entre el número total de casos:

$$exactitud = \frac{\mathrm{TP} + \mathrm{TN}}{\mathrm{TP} + \mathrm{TN} + \mathrm{FP} + \mathrm{FN}}$$

La métrica de **especificidad** es el número de veces que el modelo clasifica correctamente un elemento que no pertenece a una clase dividido por el número total de elementos que no pertenecen a esa clase.

$$especificidad = \frac{\mathrm{TN}}{\mathrm{TN} + \mathrm{FP}}$$

La métrica de **recall** o **sensibilidad** es el número de veces que el modelo clasifica correctamente un elemento de una clase dividido por el número total de elementos de esa clase en los datos de entrenamiento:

$$recall = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}}$$

La métrica de **F1 score** es una medida de equilibrio entre la precisión y el recall, y se calcula como el promedio armónico entre ambas:

$$F1\ score = \frac{2}{\frac{1}{precisión} + \frac{1}{recall}}$$

Se ha desestimado otros tipos de métricas como Intersección sobre Unión (IoU) y el umbral de confianza, dejando valores fijos altamente aceptados en sistemas similares.

La intersección sobre la Unión mide la similitud entre la clasificación del modelo y la clasificación verdadera utilizando la intersección y unión de las regiones de interés. Esta métrica no es relevante para el sistema propuesto, ya que no interfiere con la lógica de activación de las pre-alarmas y las alarmas en gran medida, por ello se ha dejado un valor umbral ampliamente aceptado de 0,45 como valor elevado de superposiciones de los bboxs.

De la misma forma el umbral de confianza es un valor numérico que se utiliza para determinar qué nivel de certeza se requiere para considerar que un objeto ha sido detectado correctamente. Este valor se utiliza para filtrar las detecciones de objetos que el modelo considera inciertas o con baja confianza. El valor elegido para este filtro es de 0,30.

Establecer un valor umbral de 0,45 para la IoU y un valor de 0,30 para el umbral de confianza, es una forma de ajustar el sistema para adaptarse a las necesidades específicas para evaluar la aplicación, estos valores se pueden variar dependiendo del uso y el contexto del sistema para cada escena.

4.1.3.3. Métricas globales

La métrica para evaluar el rendimiento computacional del sistema completo (el clasificador de objetos y la técnica de sustracción de fondo), es la latencia en velocidad de tiempo de procesado de cada imagen, la cual determina si es apropiada para utilizar en un sistema en tiempo real. Para ello se ha usado varias configuraciones de hardware y la alternancia de uso con CPU y eGPU, el cual se puede consultar en la sección Recursos disponibles .

Para analizar la efectividad y viabilidad general del sistema es a través de la observación directa de los videos del dataset comprobando el funcionamiento del sistema global y verificando el resultado de las alarmas y pre-alarmas con el resultado esperado.

Otra de las formas para evaluar el sistema es el de comprobar en extracto de video con imágenes en régimen normal de explotación, dicho de otro modo, sin incidencias que

afecten a las vías, o que no se active por falsos negativos. Es una prueba que se realiza para asegurar que el sistema no genera alarmas ni pre-alarma falsas o innecesarias. Siendo una medida importante para garantizar la confiabilidad del sistema y evitar la saturación o desensibilización de los usuarios ante las alarmas reales.

4.1.4. Recursos disponibles

Para el desarrollo del proyecto se ha usado el siguiente software y hardware:

4.1.4.1.Software usado para el desarrollo

El software utilizado para la realización del proyecto costa de las siguientes aplicaciones:

- Visual Studio Code, versión 1.72.1
- Microsoft Office 2019 (Word, Excel, OneNote…)
- Acrobat Reader
- Sistema operativo Windows 11 Pro-64 bits

Los framework más destacados utilizados en el proyecto son:

- OpenCV
- PyTorch
- Numpy
- YOLOv5
- SQLite
- SQLAlchemy

4.1.4.2.Hardware usado para el desarrollo

Se cuenta con un ordenador portátil de la marca Lenovo yoga i9 modelo 14ITL5 sus características más reseñables con respecto al proyecto son:

- Procesador Intel 11th Gen Core i7-1185G7
    - Velocidad por Núcleo: 4.8GHz,
    - 4 núcleos
    - 8 hilos
- Controlador gráfico Integrado: Intel Iris Xe Graphics G7 96EUs

- Memoria RAM: 16GB LPDDR4X-4266, timing 20-20-20-45, dual-channel
- Almacenamiento: 512GB SSD. M.2 2280 PCIe 3.0x4 NVMe
- Puertos:
    - 2 Thunderbolt 4, USB-C
    - 1 USB 3.2 Type-A

Por otro lado, se ha usado una tarjeta gráfica externa la Gigabyte RTX 3060 TI Gaming OC 8G, por la baja potencia gráfica del ordenador portátil usado para el proyecto, esta Tarjeta gráfica ayuda a aumentar FPS[22] de forma exponencial en el proceso de inferencia, y aportar potencia de cómputo en el caso de entrenamientos de diferentes redes neuronales convolucionales (CNN), aprovechar el procesamiento con CUDA de Nvidea y el tratamiento de imágenes con las siguientes características más reseñables para el proyecto:

- Frecuencia de reloj 1665 MHz
- CUDA: SI
- 4864 CUDA Cores
- Generación 2ª de RT Cores
- Generación 3ª de Tensor Cores
- Memoria RAM 8GB Type GDDR6 256-bit interfaz de memoria

Conectado al ordenador portátil por un cable USB-C Tecnóloga Thunderbolt 3 de 40 Gbps (5 GB/s).

[22] FPS: Frames per second o Frames por segundo, cantidad de imágenes consecutivas que se muestran en pantalla por cada segundo

4.2. Diseño de la arquitectura

Como diseño de arquitectura se ha optado por una estructura de software monolítica, esta arquitectura consiste en la creación de clases y métodos que contienen todas las funcionalidades necesarias para el programa en una sola aplicación. De esta manera se podrá desplegar una instancia para cada cámara video elegida.

La arquitectura monolítica se ha elegido por su simplicidad y flexibilidad, permitiendo a los usuarios especificar qué operaciones desean realizar en cada ejecución y ejecutarlas de manera sencilla. Además, permite una fácil integración de nuevas funcionalidades en el futuro, simplemente agregando nuevos métodos a las clases existentes.

En definitiva, la arquitectura de software monolítica tiene varias ventajas, entre ellas:

- **Simplicidad y facilidad de uso**: la estructura de una aplicación monolítica es sencilla y fácil de entender, lo que permite a los usuarios especificar qué operaciones desean realizar en una instancia del programa de manera sencilla.
- **Flexibilidad**: la arquitectura monolítica permite agregar nuevas funcionalidades al programa de manera sencilla, simplemente agregando nuevos métodos a las clases existentes.
- **Facilidad de depuración y solución de problemas**: en una aplicación monolítica, todo el código se encuentra en un solo lugar, lo que facilita la depuración y la solución de problemas.
- **Mejor rendimiento**: en general, una aplicación monolítica tiene un mejor rendimiento que una aplicación de arquitectura de otro tipo, ya que todas las operaciones se realizan en un solo lugar y no se requiere la comunicación entre diferentes máquinas.
- **Facilidad de despliegue**: en una aplicación monolítica, sólo es necesario desplegar una única aplicación en el servidor, lo que simplifica el proceso de despliegue y mantenimiento.

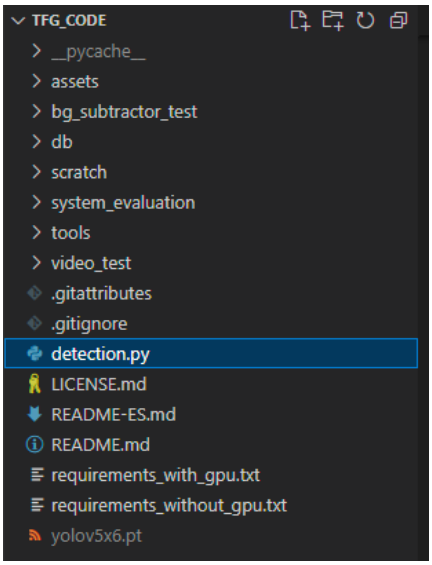


Figura 25. Estructura del directorio del proyecto.

Toda la lógica del programa está contenida en el archivo `detection.py`, como se puede apreciar en la Figura 25, la base de datos se encuentra en la carpeta `/db`, con la idea de en un futuro poder realizar una interfaz gráfica e implementar el patrón arquitectónico MVC Figura 26.

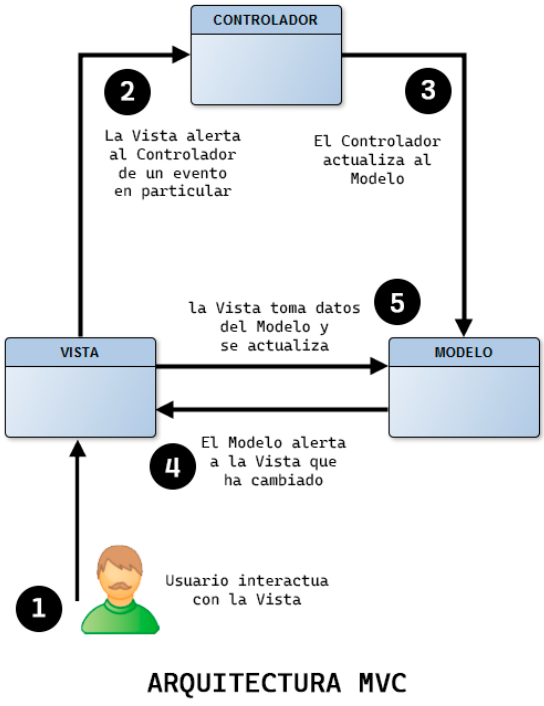


Figura 26. Patrón arquitectónico MVC

fuente: https://www.jc-mouse.net/

4.3. Diseño del software

Para el diseño del software del proyecto se ha usado la metodología, decisiones de código más relevantes y explicación del diseño de la base de datos que se explican a continuación:

4.3.1.Metodología de desarrollo

La metodología usada para el correcto desarrollo de este proyecto ha sido la de investigación científica de método en “V”, elegida en base a conseguir los objetivos generales y los objetivos específicos del proyecto. Dicho modelo es adecuado para solo un único desarrollador del producto de software, siendo así también como lo es el autor del presente documento.

El método en “V” es un enfoque para el desarrollo de software que se divide en cinco etapas del ciclo de vida de un proyecto: Análisis, Diseño, Implementación, Verificación y Mantenimiento. En su representación gráfica Figura 27 se aprecia la fase descendente en la que se desarrolla las necesidades del proyecto, culminando dicha fase en la codificación para más adelante, en la fase ascendente se especifiquen las verificaciones de las necesidades del proyecto, dando paso a las validaciones de todos los pasos.

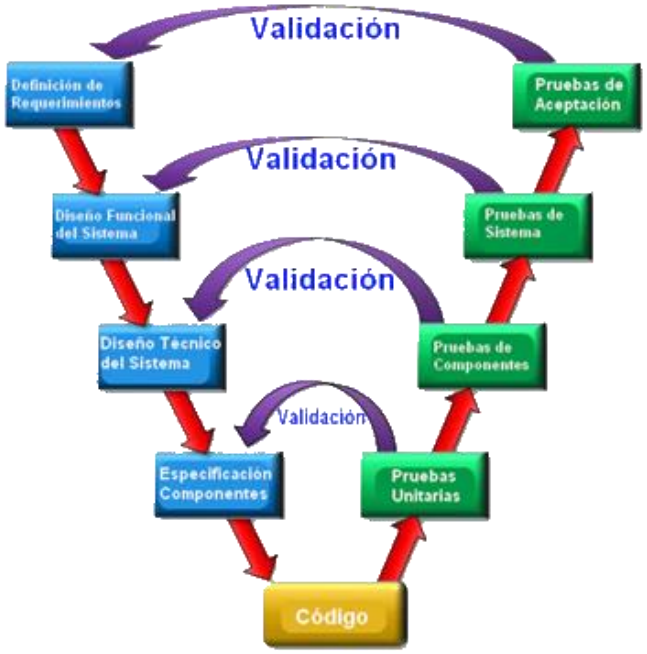


Figura 27. Representación gráfica del método en "V".

fuente: https://www.appvizer.es/

4.3.2.Código

Una vez elegido los algoritmos de sustracción de fondo y de detección de objetos que forman parte del proyecto se toman varias decisiones de implementación de la lógica del programa.

4.3.2.1. Menús y opciones del programa

En primer lugar, se implementa una serie de argumentos en línea de comandos, para la elección de la configuración antes de la ejecución del programa, con las opciones que se muestran en la Tabla 6.

Tabla 6. Argumentos en línea de comandos.

Argumento corto *Argumento largo*	*Descripción*
-h *--help*	Mostrar mensaje de ayuda y salir
-v *--version*	Versión del programa
-info *--information*	Información de las versiones de los paquetes usados
-m *--mask*	Muestra la mascara
-d *--detection*	Muestra las detecciones de objetos
-s *--slicer*	Muestra barra de desplazamiento (consume muchos recursos)
-mm *--mouse*	Muestra por consola las coordenadas de los clics
-c *--process_image*	Parámetro GPU o CPU
-i *--input*	Ruta de video a procesar

Destacando dos de las opciones, la primera, la elección de ejecución con CPU o GPUs, esta última muy recomendable por el alto rendimiento que proporciona para el buen desempeño del programa, la segunda, la posibilidad de generar los puntos de ROI para una nueva escena de captura de video, la cual arroja por consola la línea de código para introducir en la base de datos, mediante una inclusión de una línea en el archivo `/db/input_data.py` tal como se muestra en la Figura 28.

```
8,197,
312,29,
361,30,
234,465,
36,455,
ROI_polygon_XX = controler.ROI_polygon(polygon_id = 'XX', point_11 = '8', point_12 = '197', point_21 = '312', point_22 = '29'
, point_31 = '361', point_32 = '30', point_41 = '234', point_42 = '465', point_51 = '36', point_52 = '455', camera_id = 'XX')
```

Figura 28. Generación de polígono a partir de interacción con las opciones del programa.

Una vez en ejecución podemos modificar cierto comportamiento de las características visuales y sonoras del programa, las opciones son las descritas en la Tabla 7.

Tabla 7. Descripción de las opciones una vez ejecutado el programa.

Tecla	*Descripción*
Esc	Cierra la ejecución del video
p	Parar el video (Cualquier otra tecla para reanudar el video)
c	Captura un fotograma del video y lo guarda en "/capturas/{numero}_img.jpg'"
s	Desactivar / Activar sonería de alarmas

El **OSD**[23] se maneja con las siguientes teclas numéricas descritas en la Tabla 8. La opción 6 “mejora de rendimiento” provoca que se procesen la mitad de las imágenes, obteniendo un rendimiento mayor.

[23] OSD: On screen display, visualización en pantalla.

Tabla 8. Descripción de las opciones de OSD una vez ejecutado el programa.

Tecla	*Descripción*
1	Información, Alarma, FPS, numero de fotograma, Activar Sonido
2	ROI
3	Contornos dentro de ROI por sustracción de fondo
4	Punto pie derecho
5	Rectángulo detección, Contornos en la escena (Personas, trenes, bolsos, carros)
6	Activar mejor rendimiento

4.3.2.2. Alarma por persona en plataforma de vías

Uno de los retos del proyecto, al usar las cámaras de CCTV ya existentes, es determinar si una persona está en el andén o en la vía, dicho de otra manera, si una persona está dentro del área de interés (ROI) o no.

La ubicación de las cámaras está al final de cada andén, con un ángulo que, enfoca en oblicuo al final del andén de enfrente. Por este motivo las cámaras generan unas imágenes con una perspectiva que provoca la oclusión de cierta parte de la plataforma de vías por personas cerca del borde del andén donde se encuentra dicha cámara, como se muestra la Figura 29

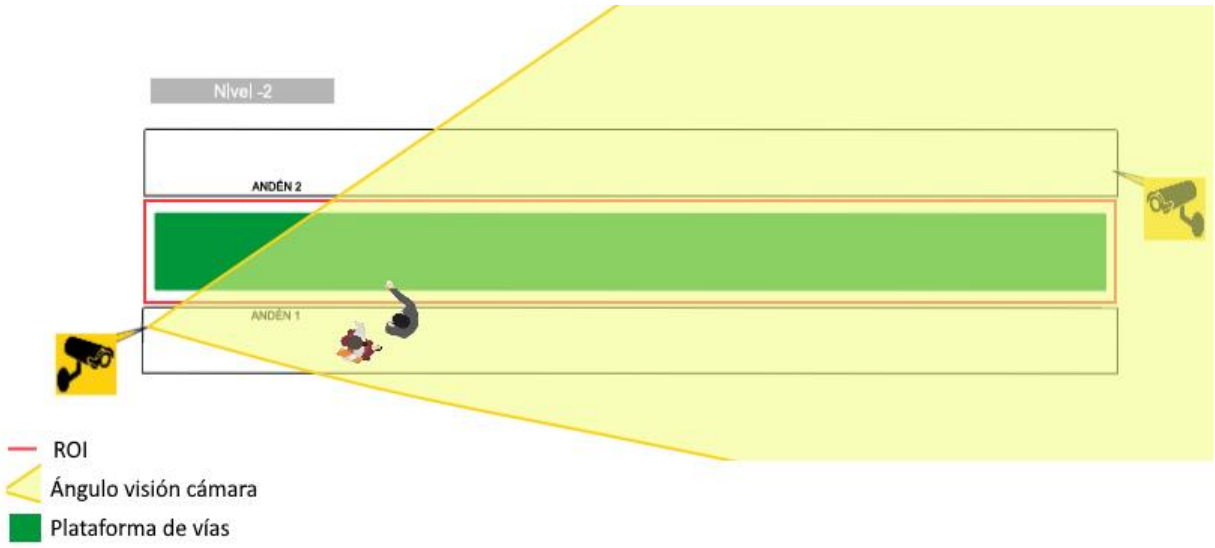


Figura 29. Perspectiva de la cámara con respecto a la plataforma de vías.

Si una persona está situada al borde del andén, el sistema de detección de objetos generará un BBox que traspasará el área de interés o ROI como se muestra en el ejemplo de la Figura 30, siendo esto una falsa alarma, por no estar la persona en plataforma de vía, si no en el andén.

Figura 30. BBox generado por sistema de detección de objetos.

Para evitar la oclusión ocasionada por las personas al borde del andén, con respecto al área de interés o ROI, se ha decidido generar un punto lo más alejado posible, siendo este en la parte inferior derecha de cada BBox, como se muestra en la Figura 31, que corresponde al

pie izquierdo de una persona mirando de frente a la cámara, este punto servirá de referencia para activar la alarma por objeto dentro del ROI, emitiendo un sonido de alerta, explicado de otra manera, si el punto referencia está dentro del polígono que representa el ROI, se activara la alarma por persona en plataforma de vías.

Figura 31. Punto generado en la parte inferior derecha de cada BBox (pie izquierdo de cada persona).

La implementación del código que satisface este requisito se encuentra en la ruta `/tools/utils.py` como un método llamado `point_in_polygon(point, polygon)` en el proyecto y es el siguiente:

```
def point_in_polygon(point, polygon):
    """Ray casting
    Check if a point is inside a polygon

        polygon - List of tuples with the points that form the vertices [(x1,
x2), (x2, y2), ..., (xn, yn)]
    """
    i = 0
    x = point[0]
    y = point[1]
    j = len(polygon) - 1
    output = False

    for i in range(len(polygon)):
        if (polygon[i][1] < y and polygon[j][1] >= y) or (polygon[j][1] < y
and polygon[i][1] >= y):
```

```
            if polygon[i][0] + (y - polygon[i][1]) / (polygon[j][1] -
polygon[i][1]) * (polygon[j][0] - polygon[i][0]) < x:
                output = not output
        j = i
  return output
```

4.3.2.3. Pre-alarmas por movimiento en plataforma de vías

Con ayuda del algoritmo de sustracción de fondo elegido MOG2, se generan las pre-alarmas cuando más de 5 fotogramas seguidos tienen más de 250 pixeles diferentes de 0 en un mismo contorno, la pre-alarma emite un sonido de alerta diferente al de las alarmas. En la Figura 32, se puede observar que solo dentro del ROI es donde se ejecuta la sustracción de fondo y se ignora los movimientos en el andén, para así de esta manera generar alarma de varias índoles, como caída de personas, objetos voluminosos, inundaciones repentinas de la cuenca de vías, desprendimiento de bóveda, etc.

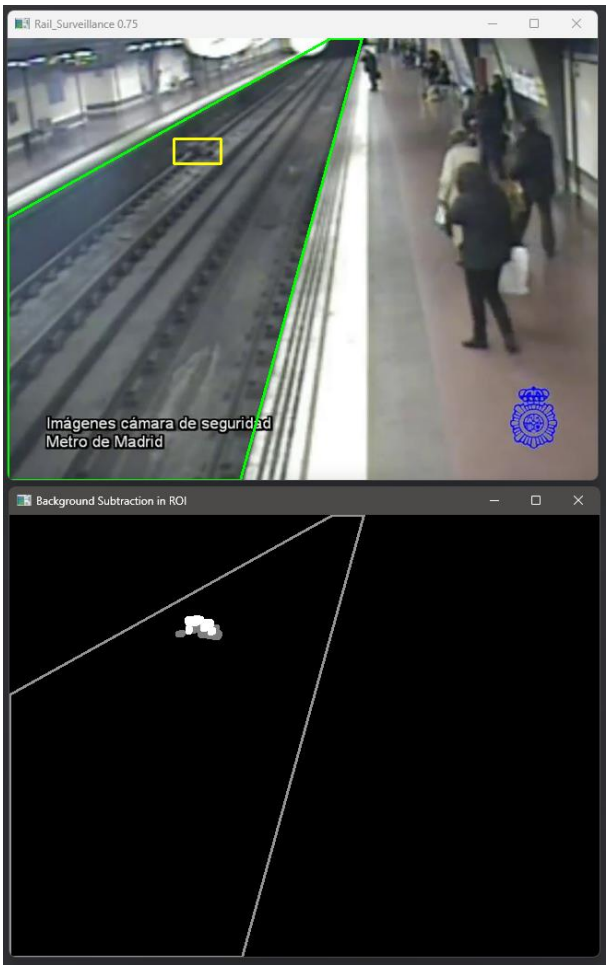


Figura 32. Detección por sustracción de fondo dentro del ROI.

4.3.2.4. Detección a la llegada del tren

Una vez detectada la llegada del tren se desactivan las posibles pre-alarmas, pero no las alarmas generadas por personas en plataforma de vías o lo que es lo mismo, por personas dentro del área de interés o ROI. Este tipo de alarma además de emitir el sonido de alerta mostrara por consola el mensaje “ARRIVAL OF THE TRAIN WITH PERSON ON THE TRAIN TRACK¡¡” como se muestra en la Figura 33.

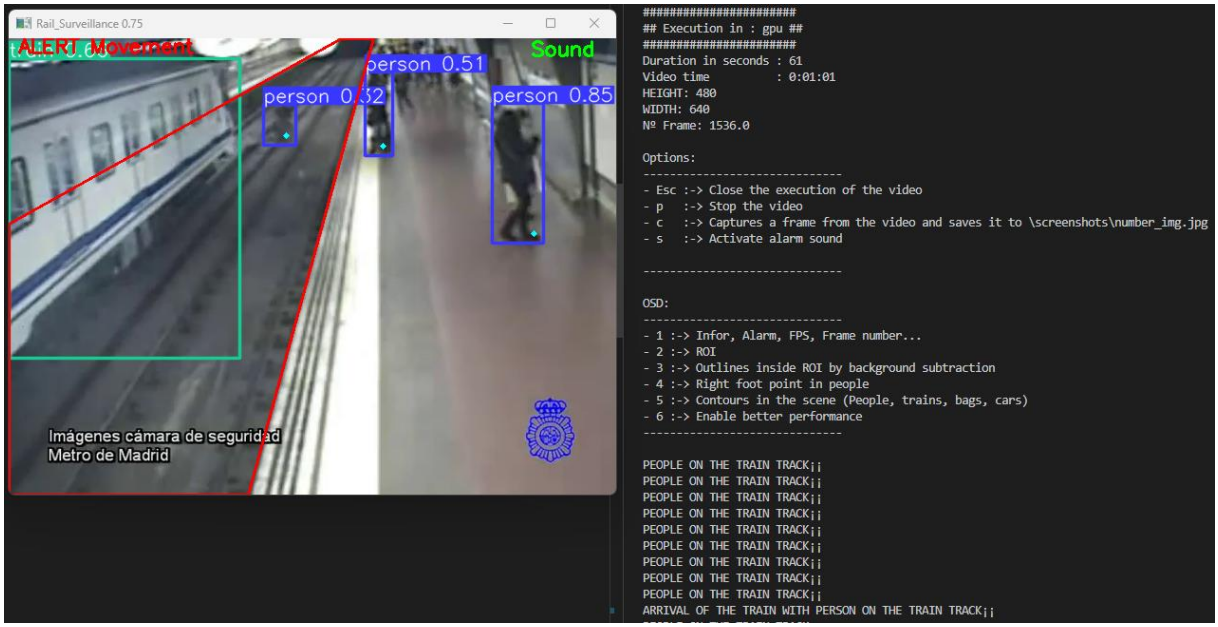


Figura 33. Mensajes mostrados por consola.

4.3.3.Diseño de la base de datos

El diseño de una base de datos es una etapa importante en el desarrollo de un sistema de información, ya que afecta a la eficiencia y a la facilidad de uso de este. Por lo tanto, es importante realizar cuidadosamente el diseño, la especificación de las reglas y las restricciones para asegurar que se adapte a las necesidades del sistema y pueda manejar eficientemente el volumen y la complejidad de los datos, manteniendo así la integridad y coherencia de la información.

Para la implementación de la base de datos se ha optado por usar un ORM[24] que permite trabajar con la base de datos utilizando un lenguaje de programación orientado a objetos en lugar de utilizar sentencias SQL. Esto proporciona varias ventajas como mayor productividad al tener que escribir menos código, portabilidad por ser independiente del motor de base de datos, así como flexibilidad y seguridad, aportando protección contra la inyección de SQL y otros ataques.

Para el diseño de la base de datos se ha tenido en cuenta el proceso de normalización, que elimina dependencias para asegurar que las tablas estén estructuradas de manera eficiente, se mantenga su integridad y sean coherentes. El nivel de normalización que se ha llevado a cabo es la tercera forma normal (3FN) con las siguientes restricciones:

- No tener dependencias transitivas.
- Todos los atributos son dependientes de la clave principal.
- Todos los atributos no clave deben depender directamente de la clave principal.
- No hay transacciones multivaluadas, esto significa que no se permite que una fila tenga más de un valor para un atributo determinado.

Una de las necesidades del proyecto es la de disponer de varias áreas de interés o ROI por cámara, ya que es probable que las cámaras puedan hacer zoom, lo que generaría la necesidad de otra toma de la escena, por lo tanto, otro ROI. La relación de las tablas `Camera` a la tabla `ROI_polygon` es de 1 a varios, siendo la ***Primary Key*** de la tabla `Camera` el atributo `camera_id` y generando una clave ***Foreign Key*** de `camera_id` en cada tupla de `ROI_polygon` como se puede ver en el diagrama de la Figura 34.

[24] ORM: Object Relational Mapping, Modelo de programación que permite mapear las estructuras de una base de datos relacional.

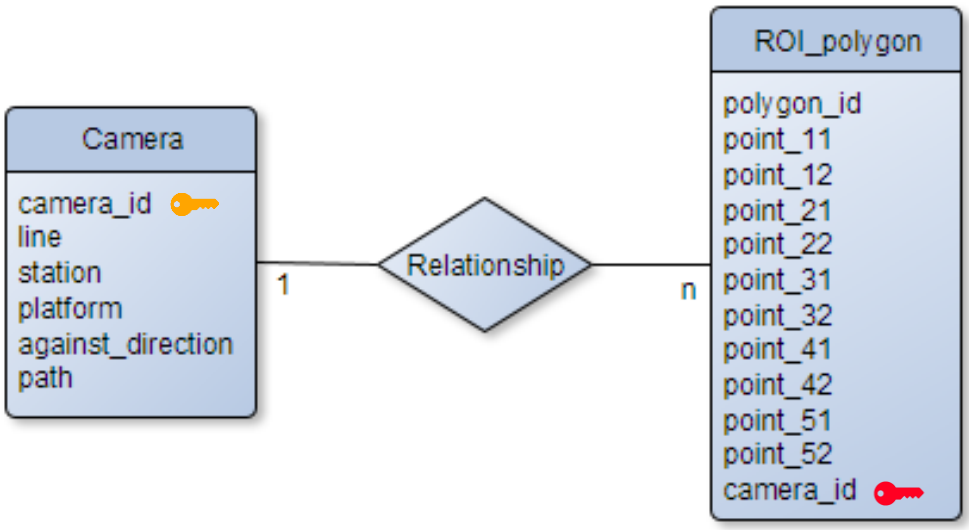


Figura 34. Diagrama UML de la base de datos.

5. EVALUACIÓN Y COMPARACIÓN

En este capítulo se evalúa el rendimiento del sistema propuesto, para más adelante analizar los resultados.

5.1. Evaluación y análisis de resultados

Se evalúa los dos grandes bloques del sistema por separado, sustracción de fondo y la clasificación de objetos, a continuación, se evalúa el sistema completo.

5.1.1. Sustracción de fondo

Para evaluar los algoritmos de sustracción de fondo se compara la precisión de cada uno de los algoritmos como se muestra en la Figura 35, donde se puede apreciar que el algoritmo GMG es el que más artefactos o ruido presenta, con respecto a MOG2, tienen sus propias ventajas y desventajas. Sin embargo, hay algunas diferencias clave entre estos algoritmos que pueden hacer que MOG2 sea mejor que GMG en ciertas situaciones. Una de las principales diferencias entre MOG2 y GMG es que MOG2 utiliza un modelo de mezcla gaussiana adaptativa, mientras que GMG utiliza un modelo de mezcla gaussiana global. Esto significa que MOG2 puede adaptarse dinámicamente a los cambios en el fondo de la escena, lo que le permite una mayor precisión en la identificación del fondo y un mejor desempeño en situaciones con un fondo dinámico. En comparación, GMG utiliza los 120 primeros fotogramas de un video para construir un modelo del fondo, lo que lo hace menos adaptativo a los cambios en la escena. Otra diferencia importante es que MOG2 es capaz de manejar la superposición de objetos en el primer plano y de identificar objetos que se están moviendo lentamente en la escena.

MOG2 en comparativa con KNN, es más fácil de usar y es más flexible, ya que permite ajustar sus parámetros de forma más sencilla. KNN solo tiene en cuenta la distancia entre los píxeles y puede tener un desempeño menos preciso en situaciones con un fondo dinámico. Por lo tanto, MOG2 permite una mayor precisión en la identificación de pequeños objetos.

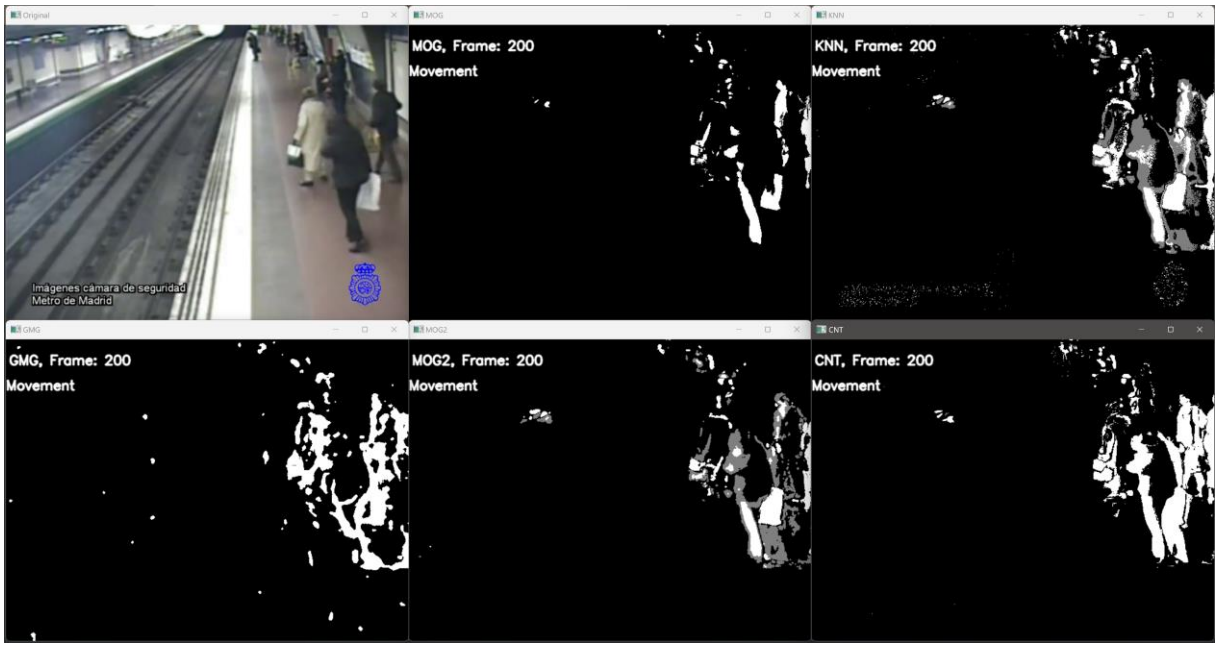
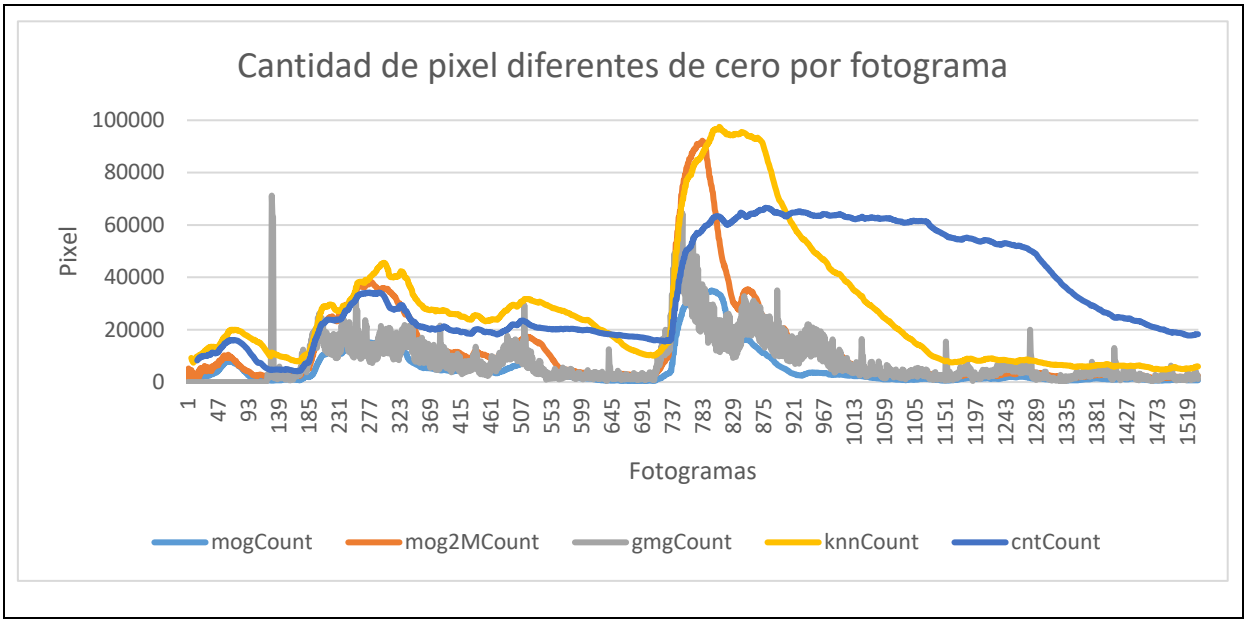


Figura 35. Comparación de algoritmos de sustracción de fondo en el fotograma 200.

Para evaluar la cantidad de pixeles distintos de cero dentro de las máscaras y medir la velocidad de procesamiento por fotograma en todos los algoritmos, se ha implementado el archivo `test_BackgroundSubtractor.py` ubicado en la carpeta `/test_BackgroundSubtractor` el cual genera un archivo `test_BS.csv` (archivo de datos separados por comas). Después de varias ejecuciones con el parámetro predeterminado de sustractores, otras ejecuciones con los parámetros mejorados y en aquellos sustractores que lo permiten, la aplicación de kernel de apertura morfológica que mejora los resultados con respecto a ruidos, se han elaborado varias tablas para comparar los resultados.

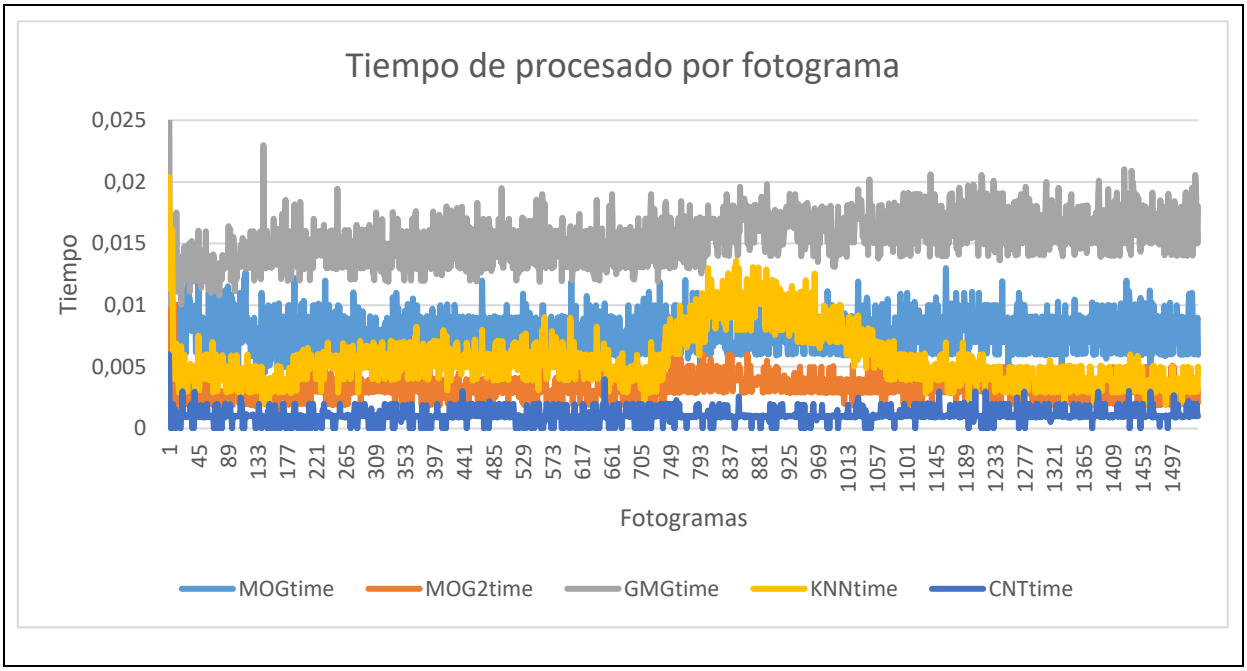
En la Tabla 9 se puede observar en el eje de abscisas (eje x) los fotogramas consecutivos y en el eje de ordenadas (eje y) se aprecia la cantidad de pixeles. En dicha Tabla se muestra los resultados de cada algoritmo, los datos son muy similares en cuanto a cantidad de pixeles, se puede apreciar observando la Tabla el momento de llegada del tren sobre el fotograma 740, los algoritmos CNT y KNN tardan muchos fotogramas en recuperar el estado del fondo, lo que puede ocasionar muchos falsos positivos a la larga, GMG tiene fuertes fluctuaciones constantes que generan mucho ruido en la imagen binaria, se aprecia más estabilidad y rapidez en los cambios de imagen en los algoritmos MOG y MOG2.

Tabla 9. Cantidad de píxel diferentes de cero por fotograma.



En la Tabla 10 se puede observar en el eje de abscisas (eje x) los fotogramas consecutivos y en el eje de ordenadas (eje y) se aprecia el tiempo en el que cada algoritmo tarda en procesar cada fotograma. Se aprecia un mejor desempeño en tiempo en el algoritmo CNT con 0,0010 segundos de tiempo medio, seguido de MOG2, con 0,0032 segundos de tiempo medio, siendo MOG2 más estable que KNN (0,005 segundos), GMG (0,015 segundos), y MOG (0,007 segundos), tienen tiempos que doblan el valor de MOG2.

Tabla 10. Tiempo de procesado por fotograma.



Con todos las pruebas y datos realizados se ha generado la Tabla 11 comparativa, que muestra todos los algoritmos utilizados y las características más destacadas para beneficio del proyecto.

Tabla 11. Tabla comparativa de algoritmos de sustracción de fondo.

	MÉTODO PARA LA DETECCIÓN POR PÍXEL	*CAMBIOS DE ILUMINACIÓN*	*ACELERACION CON CUDA*	*CANTIDAD DE RUIDO*	*RECUPERACIÓN DE ESTADO ANTERIOR*	*VELOCIDAD DE PROCESADO POR FOTOGRAMA*
MOG	Mezcla de distribución gaussianas	No	Si	Moderada	Media	Media
MOG2	Mezcla de distribución gaussianas	Si	Si	Baja	Alta	Baja
GMG	Segmentación bayesiana	No	Si	Alta	Media	Alta
KNN	K-vecinos más cercanos	No	Si	Moderada	Baja	Baja
CNT	Resta conteo	No	No	Alta	Baja	Baja

Después de realizar un análisis detallado de los diferentes algoritmos de sustracción de fondo, sea ha decidido utilizar el algoritmo MOG2 para el proyecto debido a su buen desempeño en comparación con otros algoritmos.

La elección de MOG2 se basa en las gráficas y la tabla comparativa que se ha realizado, donde se puede apreciar que este algoritmo tiene una precisión similar a la de otros algoritmos, pero tiene una mayor velocidad de procesamiento. Además, MOG2 también tiene una mayor adaptabilidad a cambios en la iluminación y el fondo de la escena, lo que puede ser importante para el desarrollo del proyecto.

5.1.2. Clasificador de objetos

Para evaluar el clasificador de objetos se han elegido varios de los videos de los que se dispone en el proyecto (dataset de pruebas), se ha clasificado los fotogramas manualmente, después se ha ejecutado el sistema que guarda los resultados de la predicción del clasificador de objetos automáticamente en el archivo `eval.cvs` de la carpeta

`system_evaluation` para a continuación, con respecto a las métricas descritas en la sección Clasificador de objetos, obtener los resultados de la Tabla 12 y Tabla 15, para el objeto "Train" y "Preson_on_via" que generan respectivas matrices de confusión de la Tabla 13 y Tabla 16.

5.1.2.1. Objeto "Train"

Datos para el objeto "Train":

Tabla 12. Indicadores para la matriz de confusión del objeto "Train".

TP Train	792
TN Train	732
FP Train	8
FN Train	4

Tabla 13. Matriz de confusión del objeto "Train".

Objeto: **TRAIN**		**Predicción**	
		Positivos	Negativos
Real	Positivos	792	4 (Error de tipo II)
	Negativos	8 (Error de tipo I)	732

El archivo `/system_evaluation/eval (used in the Document).xlsx` contiene las fórmulas para el cálculo automático de los porcentajes de las métricas.

Tabla 14. Métricas "Train".

La métrica precisión para el objeto "Train"	99%
La métrica exactitud para el objeto "Train"	99,21%
La métrica especificidad para el objeto "Train"	98,91%
La métrica Recall o sensibilidad para el objeto "Train"	99,49%
La métrica F1 score para el objeto "Train"	99,24%

Como se puede observar en la Tabla 14 los valores de las métricas presentadas indican un buen rendimiento del modelo para el objeto "Train" en particular:

- La precisión del 99% indica que el modelo es capaz de identificar correctamente el 99% de las veces cuando un objeto es de la clase "Train".
- La exactitud del 99,21% indica que el modelo es capaz de identificar correctamente el 99,21% de las veces tanto los objetos de la clase "Train" como los de otras clases.
- La especificidad del 98,91% indica que el modelo es capaz de identificar correctamente el 98,91% de las veces cuando un objeto no es de la clase "Train".
- El recall del 99,49% indica que el modelo es capaz de encontrar el 99,49% de los objetos realmente de la clase "Train" entre los objetos predichos como de esa clase.
- El puntaje F1 score del 99,24% indica que el modelo tiene un buen balance entre precisión y recall, lo que significa que tiene un buen rendimiento tanto en identificar correctamente los objetos de la clase "Train" como en encontrar la mayoría de los objetos realmente de esa clase.

En general, estos valores indican que el modelo tiene un buen rendimiento en la clasificación de objetos de la clase "Train", sin embargo, es importante tener en cuenta que estos resultados se basan en un conjunto de datos específico y que es posible que el rendimiento del modelo varíe en diferentes conjuntos de datos o en situaciones diferentes.

5.1.2.2. Objeto “Person on via”

Datos para el objeto “Person on via”:

Tabla 15. Indicadores para la matriz de confusión del objeto "Person on via".

TP Person on via	29
TN Person on via	1003
FP Person on via	0
FN Person on via	504

Tabla 16. Matriz de confusión del objeto "Person on via".

Objeto: **Person on via**		**Predicción**	
		Positivos	Negativos
Real	Positivos	29	504 (Error de tipo II)
	Negativos	0 (Error de tipo I)	1003

Tabla 17. Métrica “Person on via”.

La métrica precisión para el objeto “Person on via”	100%
La métrica exactitud para el objeto “Person on via”	67,18%
La métrica especificidad para el objeto “Person on via”	100%
La métrica Recall o sensibilidad para el objeto “Person on via”	0,05%
La métrica F1 score para el objeto “Person on via”	0,10%

Los valores de las métricas presentadas en la Tabla 17, indican un rendimiento limitado del modelo para el objeto "Person on via", en particular:

- La precisión del 100% indica que el modelo es capaz de identificar correctamente el 100% de las veces cuando un objeto es de la clase "Person on via".
- La exactitud del 67,18% indica que el modelo es capaz de identificar correctamente solo el 67,18% de las veces tanto los objetos de la clase "Person on via" como los de otras clases.

- La especificidad del 100% indica que el modelo es capaz de identificar correctamente el 100% de las veces cuando un objeto no es de la clase "Person on via".
- El recall del 0,05% indica que el modelo solo es capaz de encontrar el 0,05% de los objetos realmente de la clase "Person on via" entre los objetos predichos como de esa clase.
- El puntaje F1 del 0,10% indica que el modelo tiene un rendimiento muy limitado tanto en precisión como en recall, lo que significa que tiene dificultades tanto en identificar correctamente los objetos de la clase "Person on via" como en encontrar la mayoría de los objetos realmente de esa clase.

En general, estos valores indican que el modelo tiene un rendimiento limitado en la clasificación de objetos de la clase "Person on via". Es posible que el modelo esté sobre ajustado a la clase "Person on via", lo que resulta en una precisión alta, pero en un rendimiento limitado en la generalización. Es importante investigar las causas de este comportamiento y considerar posibles soluciones, como recopilar más datos, utilizar una técnica de regularización o hacer ajustes en el proceso de entrenamiento. El cual se realiza en el Capítulo de CONCLUSIONES.

5.1.3. Evaluación global del sistema.

La primera evaluación es la de latencia de tiempo total de fotograma por inferencia y se realiza en la grabación `01.mp4` video default del dataset de pruebas del proyecto, tanto en la eGPU como en la CPU de la se dispone.

Tabla 18. Latencia de inferencia por fotograma con eGPU.

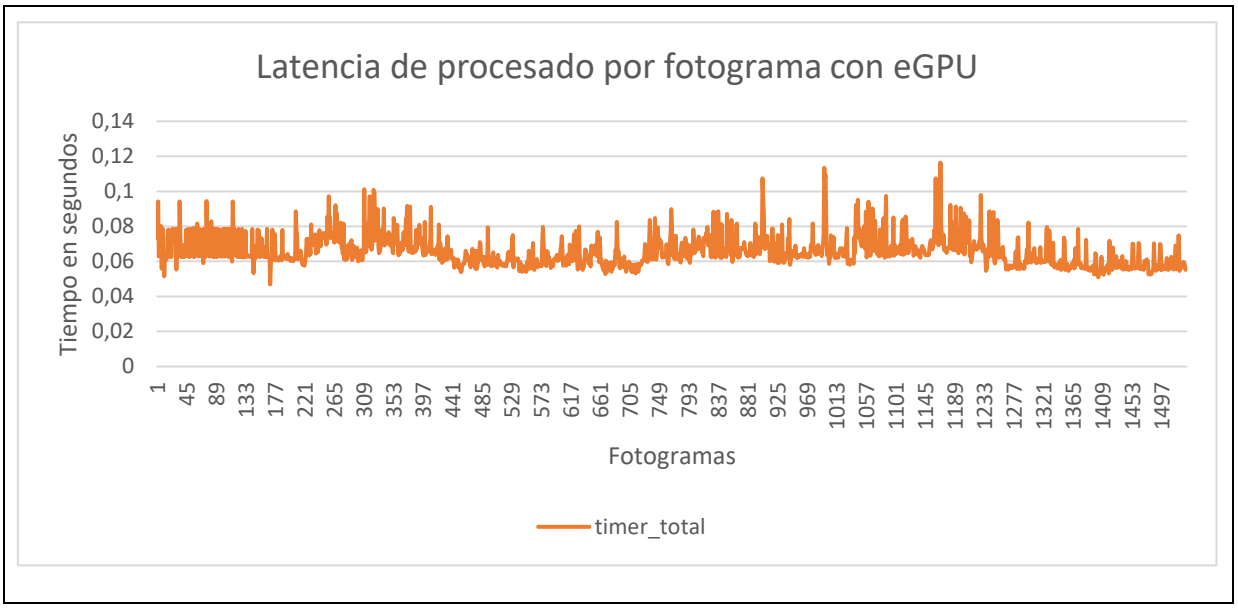
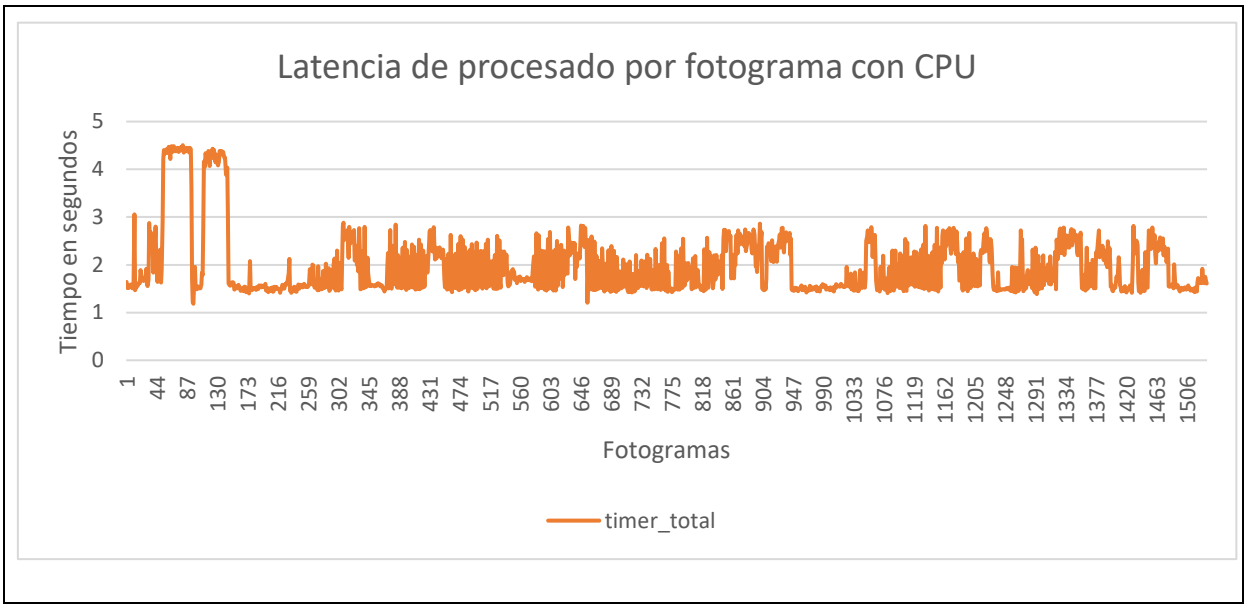


Tabla 19. Latencia de inferencia por fotograma con CPU.



Como se puede observar hay una gran diferencia en el tiempo de latencia por procesado de fotograma dando como tiempo medio en el procesado por CPU de 1,98436515 segundos, comparado con el tiempo medio por procesado en eGPU de 0,546870555 segundos, siendo un 26,15 veces más rápida y obteniendo menos fluctuaciones.

Tabla 20. FPS alcanzadas con eGPU.

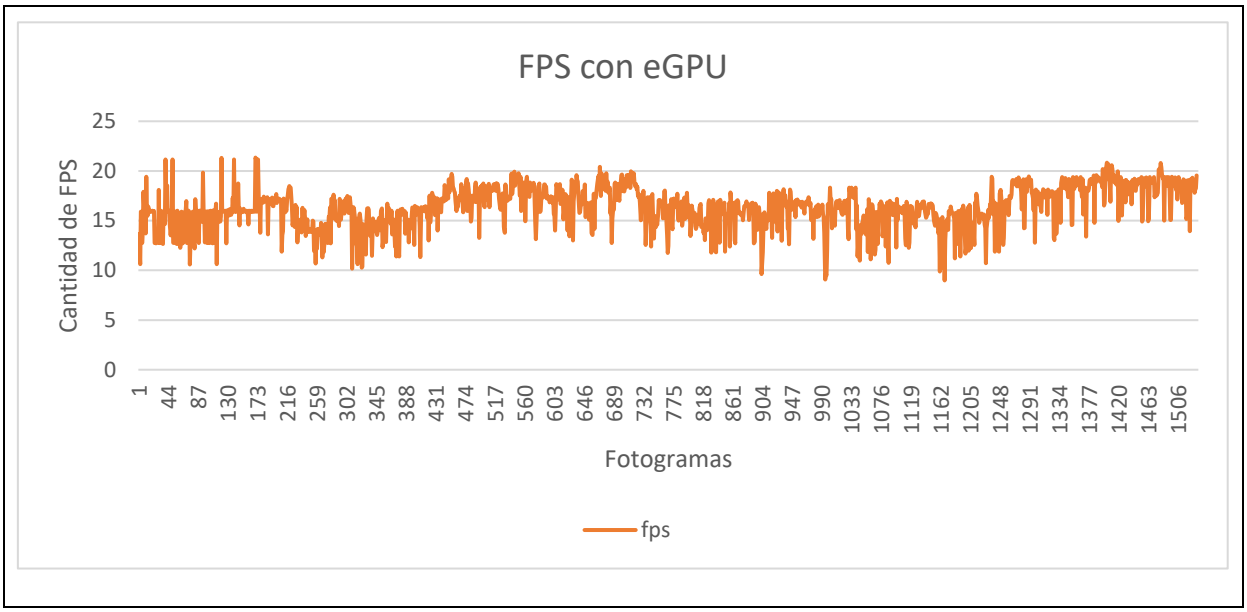
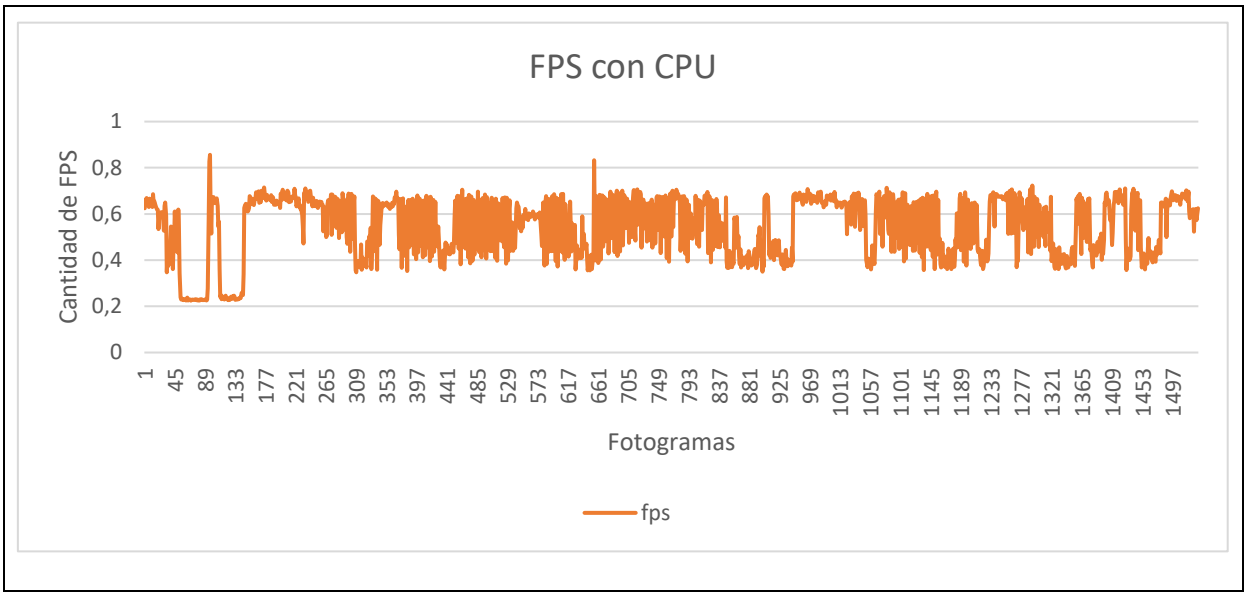


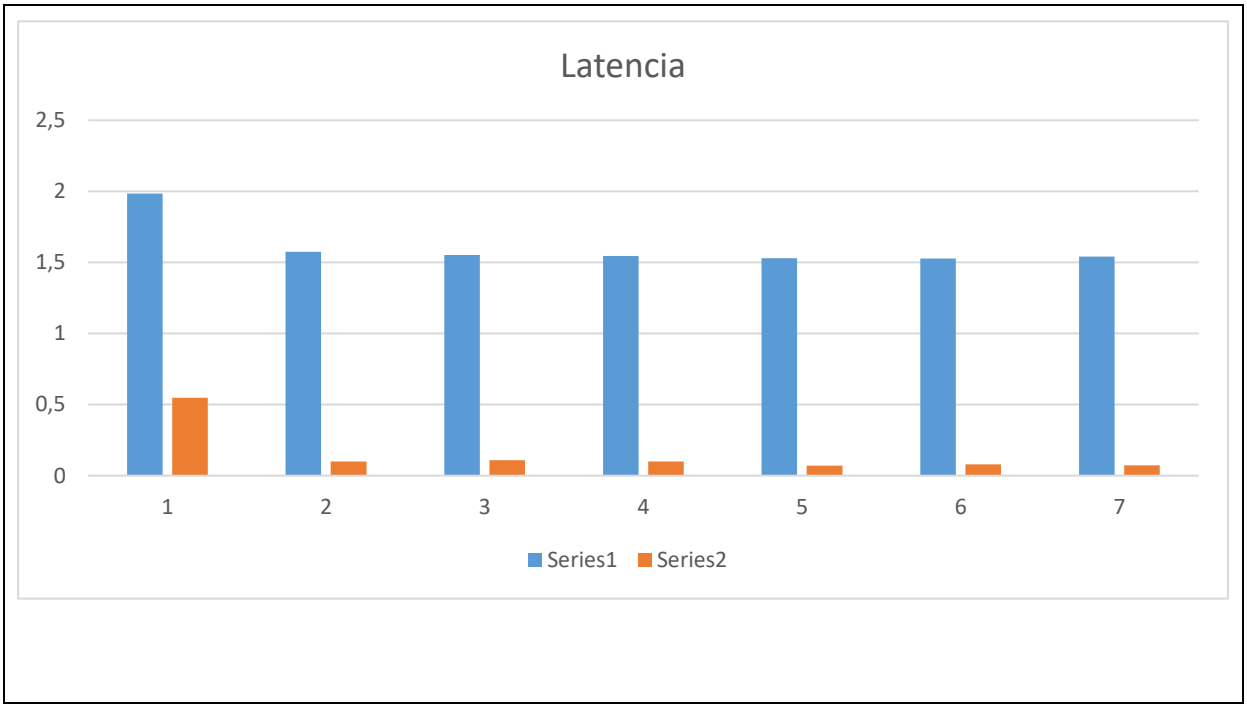
Tabla 21.FPS alcanzadas con CPU.



Los FPS que se han alcanzado de media con la CPU es de 0,546870555 FPS, siendo los de eGPU de 15,24492244 FPS, un 26,87% más rápida.

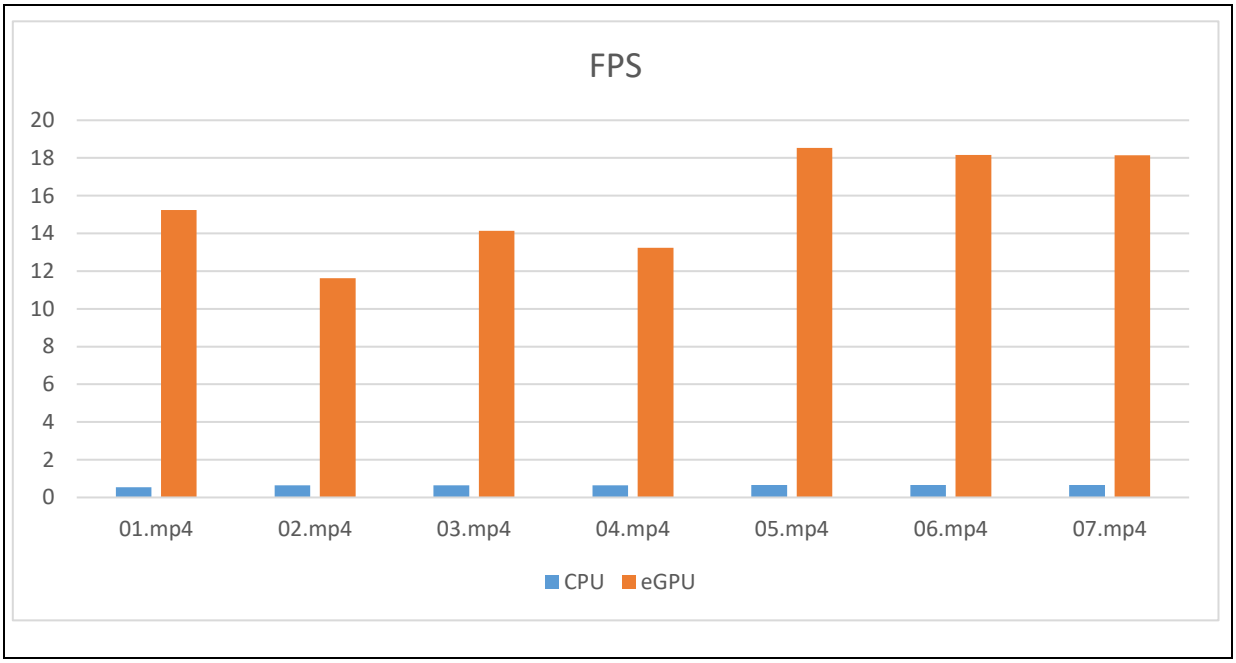
En las siguientes tablas, la Tabla 22 y Tabla 23 se comparan las métricas de latencia y FPS en todos los videos del dataset de pruebas.

Tabla 22. Comparativa de latencia de inferencia por fotograma en los videos del dataset de pruebas.



Los resultados indican que la latencia de procesamiento por fotograma varía entre los diferentes videos. En general, se observa que la latencia es significativamente mayor en la CPU que en la eGPU. Esto sugiere que la eGPU es más eficiente en el procesamiento de vídeo en comparación con el CPU para el procesamiento de fotogramas en tiempo real. Esto es debido a que la eGPU tiene una arquitectura diseñada específicamente para realizar tareas de procesamiento gráfico y paralelo, lo que la hace más eficiente para este tipo de tareas. Además, la GPU cuenta con una mayor cantidad de núcleos y memoria dedicada, lo que también contribuye a su mayor rendimiento. En resumen, se puede concluir que la eGPU es una mejor opción para sistemas en tiempo real debido a su mayor rendimiento y eficiencia.

Tabla 23. Comparativa de FPS en los videos del dataset de pruebas



Los resultados de los FPS (cuadros por segundo) también muestran que la eGPU es significativamente más rápida que la CPU en términos de rendimiento. Se puede observar que la GPU es capaz de mostrar una cantidad significativamente mayor de FPS en comparación con la CPU. Esto se debe a que la eGPU tiene una mayor capacidad de procesamiento y paralelismo, lo que la hace más adecuada para tareas de tratamiento gráficos en tiempo real. Además, la eGPU tiene una mayor cantidad de núcleos y memoria dedicada, lo que también contribuye a su mayor rendimiento. En resumen, se puede concluir que la GPU es una mejor opción para sistemas y aplicaciones en tiempo real debido a su mayor rendimiento y eficiencia en términos de FPS.

El funcionamiento global del sistema se demuestra por observación directa de los videos del dataset de pruebas, arrojando los datos que se pueden observar en la Tabla 24.

Tabla 24. Análisis de efectividad y viabilidad del sistema con anormalidad de la explotación.

Input	Localización	Resultado del sistema global	Alarma	Pre-Alarma
1	Puerta del Ángel	Funcionando correctamente	Coincide con el resultado esperado	Coincide con el resultado esperado
2	Marqués de Vadillo	Funcionando correctamente	Coincide con el resultado esperado	Coincide con el resultado esperado
3	Lavapiés	Funcionando correctamente	Coincide con el resultado esperado	Coincide con el resultado esperado
4	Estrecho	Mejoras necesarias	Difiere del resultado esperado	Coincide con el resultado esperado
5	Chueca	Mejoras necesarias	Difiere del resultado esperado	Difiere del resultado esperado
6	Opera	Mejoras necesarias	Difiere del resultado esperado	Coincide con el resultado esperado
7	Sol	Funcionando correctamente	Coincide con el resultado esperado	Coincide con el resultado esperado

Los resultados de la tabla indican que el sistema está funcionando correctamente en la mayoría de los fragmentos de video analizados. En los fragmentos 1, fragmentos 2 y en el fragmento 3 del dataset de prueba, el sistema está funcionando adecuadamente y los resultados de las alarmas y pre-alarmas coinciden con lo esperado. Sin embargo, en los fragmentos 4 y fragmentos 5, fragmentos 6, fragmentos 7, se han detectado problemas en el sistema de alarma y pre-alarma que requieren una explicación.

En concreto se han detectado problemas de falta de nitidez en los videos, de la mitad del andén hasta el final, lo que dificulta la detección de objetos. Así como distancias muy largas que hacen a los objetos muy difuminados e irreconocibles. Esto a su vez dificulta la detección de alarmas y pre-alarmas en esas zonas. Dichos problemas pueden deberse a una mala configuración de las cámaras o a una mala iluminación en las zonas de detección.

También es necesario mencionar que cada estación dispone de al menos dos cámaras, una en cada hastial de la bocana del túnel y contrapeadas, las cuales cubren zonas compartidas. Es importante realizar un análisis detallado de estas zonas para identificar las causas de los problemas y tomar medidas para solucionarlos. Hay que recordar que el sistema es probado con una configuración de valor umbral de 0,45 para la IoU y un valor de 0,30 para el umbral de confianza, los cuales se pueden modificar para un mejor rendimiento dependiendo del uso y el contexto del sistema para cada escena.

Por último, se evalúa el sistema en varios fragmentos de video en los cuales se observa un régimen normal de explotación, los resultados se pueden comprobar en la Tabla 25

Tabla 25. Análisis de efectividad y viabilidad del sistema en régimen de normalidad de la explotación.

Input	Localización	Resultado del sistema global
1	Puerta del Ángel	Coincide con el resultado esperado
2	Marqués de Vadillo	Coincide con el resultado esperado

En ambos fragmentos, el resultado del sistema global se indica como "Coincide con el resultado esperado". Esto significa que el sistema está funcionando adecuadamente y cumple con las expectativas en cuanto a su rendimiento y capacidad de detección de eventos.

Es importante destacar que, aunque solo se presentan dos localizaciones, por la baja cantidad de fragmentos del dataset de pruebas, es necesario hacer la evaluación del sistema en diferentes escenarios y condiciones para garantizar su eficacia y fiabilidad en otras situaciones.

5.2. Comparación

En comparación con otros trabajos, en el campo de la detección de personas y objetos en plataformas ferroviarias estudiados en la sección Estado de la cuestión, mi contribución se enfoca en la combinación de las técnicas más modernas de la actualidad, la sustracción de fondo MOG2 y redes neuronales con YOLOv5. Con ello obtenemos beneficio de las nuevas contribuciones de dichas técnicas, que mejoran la efectividad y eficiencia con respecto a otros sistemas similares.

El trabajo propuesto por Changmu Lee hace uso de múltiples cámaras que combinan información de diferencias de fotogramas, umbralización, etiquetado y fusión con detección de sensores láser. En este sentido, mi trabajo difiere del citado, al no tener que precisar la instalación de nuevos elementos, haciendo uso de solo, una cámara de CCTV por andén ya instaladas.

Por otro lado, los trabajos de T. Xiao. y A. D. Petrović tienen enfoques diferentes en la detección de obstáculos en la vía. T. Xiao proponiendo un método multisensorial sin

contacto, con cámaras y un dispositivo LiDAR y A. D. Petrović proponiendo una combinación de redes neuronales profundas y métodos de detección de bordes, con cámaras. Las dos propuestas emplean para la detección de obstáculos en la vía, dispositivos On-Board. Basado en este sentido, mi trabajo se enfoca en la detección de objetos y personas desde cada andén.

Por último, el trabajo de Haixia Pan & Tian se enfoca en la detección de señales ferroviarias y el problema de la línea de vía bloqueada. Se diferencia de mi contribución en su enfoque, en la detección de líneas de seguimiento y su combinación de redes de aprendizaje multitarea.

En general, mi contribución es más flexible en cuanto a configuración y simplicidad a la hora de generar alarmas y pre-alarmas. Ello posibilita implementar fácilmente nuevas alarmas en la lógica del programa, es escalable, pudiendo lanzar múltiples instancias por cada cámara.

En resumen, mi contribución se enfoca en aspectos específicos de la seguridad ferroviaria, en concreto la seguridad de las personas. Se diferencia de otros trabajos por su enfoque en la detección de objetos/personas, en plataformas de vía ferroviarias. Pudiendo confirmar que nuestra investigación da un salto en cuanto a calidad, mayor configuración, sencillez, precisión y rapidez en el procesado de fotogramas.

6. CONCLUSIONES

La supervisión mediante sistemas de visión artificial permite efectivamente automatizar y mejorar la eficiencia en la supervisión de sistemas de seguridad. Nuestro sistema utiliza cámaras y algoritmos de visión artificial para detectar y analizar patrones en imágenes y videos, lo que permite realizar tareas de supervisión con mayor precisión y repetibilidad que el actual proceso realizado por vista humana. Además, la supervisión mediante nuestro sistema de visión artificial es más sistemática y eficaz que la supervisión humana, ya que pueden operar de manera continua.

En este proyecto se ha desarrollado y evaluado un sistema para la detección de personas y/u objetos en vías ferroviarias a tiempo real usando modernas técnicas de procesamiento de imágenes, clasificación de objetos y sustracción de fondo, con una arquitectura basada en YOLOv5 y el algoritmo MOG2, respectivamente.

El algoritmo MOG2 ha generado unas métricas en velocidad de procesado de 0,0032 segundos de tiempo medio por fotograma, con muy buena precisión, que muestra pocos artefactos o ruido y gran recuperación gracias a adaptarse dinámicamente a los cambios en el fondo de la escena, por la técnica que utiliza de mezcla gaussiana adaptativa. MOG2 permite una mayor precisión en la identificación de pequeños objetos y por la cantidad de pixeles distintos de cero se aprecia más estabilidad y rapidez en los cambios de imagen.

Con respecto a la clasificación de imágenes la métrica precisión para el objeto “Person on via”es del 100%. La métrica exactitud para el objeto “Person on via” es del 67,18% y la métrica precisión para el objeto “Train” es del 99%, la métrica exactitud para el objeto “Train” es del 99,21%, lo que significa que el modelo está prediciendo correctamente cada vez que detecta esos objetos. Sin embargo, la exactitud es del 67,18% para el objeto "Person on via" y del 99,21% para el objeto "Train", lo que significa que el modelo está prediciendo correctamente en un porcentaje menor de casos en general, independientemente de la clase específica. Estas limitaciones pueden ser debidas a que el modelo esté sobre ajustado a la clase "Person on via", alguna de las soluciones sería recopilar más datos, utilizar una técnica de regularización o hacer ajustes en el proceso de entrenamiento, o realizar un entrenamiento dirigido específicamente a los dos objetos necesarios en este proyecto. Es

preciso recordar que el modelo usado para este proyecto es un modelo entrenado con PyTorch hub, con el conjunto de datos COCO.

Hemos de tener en cuenta también la certificación de los sistemas en una explotación ferroviaria, la certificación se basa en un conjunto de estándares internacionales para los sistemas de control y protección en el ferrocarril, conocido como EN-50126, EN-50128 y EN-50129 (RAMS[25]). Estos estándares establecen los requisitos para la planificación, diseño, construcción, instalación, configuración y mantenimiento de los sistemas de control y protección, ya que nunca una IA ha alcanzado nivel SIL 4[26].

La coherencia de los resultados obtenidos nos demuestra la viabilidad y el buen rendimiento del sistema propuesto, cumpliendo los objetivos específicos marcados al desarrollar un software o sistema de videovigilancia automática de las vías ferroviarias que sea viable, eficaz y económico, validado la solución propuesta con las diferentes evaluaciones realizadas.

6.1. Líneas futuras

Se proponen las siguientes líneas futuras con las que continuar y completar el proyecto:

- Entrenar una nueva red neuronal convolucional (CNN) con imágenes reales de las CCTV para las dos categorías necesarias para el proyecto, reentrenar a través del método transferencia de conocimiento o aprendizaje por transferencia de las CNN con los pesos ya disponibles.
- Realizar una interfaz gráfica para controlar los elementos del sistema que ahora se controlan por la línea de comandos e implementar el patrón arquitectónico MVC.
- Automatizar la detección de ROIs e introducción en la base de datos.

[25] RAMS: Especificación y demostración de la fiabilidad, la disponibilidad, la mantenibilidad y la seguridad.
[26] SIL: Nivel de integridad de seguridad (Safety Integrity Level) en seguridad ferroviaria es un sistema de evaluación de la seguridad de los sistemas de control y protección en el ferrocarril. El objetivo de la certificación es garantizar que los sistemas de control y protección cumplen con los niveles requeridos de fiabilidad y seguridad.

- Implementar nuevas alarmas en la lógica del programa, como olvido de objetos en los andenes, inundaciones repentinas, desprendimiento de bóveda, globos aerostáticos de helio, etc.
- Evaluar el sistema en plataformas en la nube como, Roboflow, o framework OpenVino (Open Visual Inference and Neural Network Optimization).

Estas son solo algunas posibles líneas de investigación futura para mejorar el sistema de detección de personas y objetos de vías ferroviarias en tiempo real. Con el uso de tecnologías avanzadas y metodologías de investigación, se podría mejorar significativamente el rendimiento y la precisión del sistema.

6.2. Valoraciones personales

Este ha sido un proyecto muy ambicioso, por abarcar mucho de lo estudiado en el grado de ingeniería informática, he tenido que lidiar con muchos contratiempos, sin embargo, a pesar de los desafíos y obstáculos que he enfrentado, he aprendido mucho a lo largo del camino. He adquirido habilidades y técnicas importantes en programación, análisis de datos y diseño de sistemas, además de haber mejorado mi capacidad para trabajar en el ámbito de la informática y la de resolver diversos problemas.

El proyecto me ha permitido poner en práctica lo aprendido en el aula y me ha dado una visión más completa de cómo se aplican las teorías en la vida real. Además, ha sido una gran oportunidad para desarrollar mi creatividad y pensamiento crítico, lo que me ayudará en mi carrera profesional. Estoy muy satisfecho con el resultado final.

A lo largo del camino de investigación me encontré que el creador de YOLO Joseph Redmon, abandonó el desarrollo por las implicaciones militares y de privacidad, escribió este Tweet:

“I stopped doing CV research because I saw the impact my work was having. I loved the work, but the military applications and privacy concerns eventually became impossible to ignore.”

Joseph Redmon

Lo cual, me hizo reflexionar sobre ello, es nuestra responsabilidad como desarrolladores e investigadores ser conscientes de las posibles consecuencias de nuestras acciones y trabajar para minimizar cualquier efecto negativo, considerando también los aspectos éticos y morales para tener en cuenta cómo pueden afectar a la sociedad en su conjunto. Sin embargo, es importante notar que, también existen aplicaciones positivas de la IA, como proyectos como este que pueden salvar vidas.

Este proyecto no termina aquí, seguiré su desarrollo hasta su implantación.

Bibliografía

[1] Á. M. Caballero, J. Navor, M. Cassia, y A. P. Montes, «Síndrome de Fatiga ocular y su relación con el medio laboral», 2017.

[2] S. Oh, S. Park, y C. Lee, «A platform surveillance monitoring system using image processing for passenger safety in railway station», en *ICCAS 2007 - International Conference on Control, Automation and Systems*, 2007, pp. 394-398. doi: 10.1109/ICCAS.2007.4406975.

[3] T. Xiao, Y. Xu, y H. Yu, «Research on Obstacle Detection Method of Urban Rail Transit Based on Multisensor Technology», *Journal of Artificial Intelligence and Technology*, vol. 1, n.º 1, pp. 61-67, ene. 2021, doi: 10.37965/jait.2020.0027.

[4] A. D. Petrović *et al.*, «Integration of Computer Vision and Convolutional Neural Networks in the System for Detection of Rail Track and Signals on the Railway», *Applied Sciences (Switzerland)*, vol. 12, n.º 12, jun. 2022, doi: 10.3390/app12126045.

[5] H. Pan, Y. Li, H. Wang, y X. Tian, «Railway Obstacle Intrusion Detection Based on Convolution Neural Network Multitask Learning», *Electronics (Basel)*, vol. 11, n.º 17, p. 2697, ago. 2022, doi: 10.3390/electronics11172697.

[6] K. Matt Nolan, «L'intelligence artificielle au service de la sécurité sur les voies de la ligne Docklands Light Railway du métro londonien», *MOBILITÉ URBAINE, FRANCE, INTELLIGENCE ARTIFICIELLE, SÉCURITÉ*, 7 de marzo de 2022.

[7] P. Kaewtrakulpong y R. Bowden, «An Improved Adaptive Background Mixture Model for Real-time Tracking with Shadow Detection», Kluwer Academic Publishers.

[8] Z. Zivkovic, «Improved adaptive Gaussian mixture model for background subtraction», en *Proceedings - International Conference on Pattern Recognition*, 2004, vol. 2, pp. 28-31. doi: 10.1109/icpr.2004.1333992.

[9] A. B. Godbehere, A. Matsukawa, y K. Goldberg, «Visual Tracking of Human Visitors under Variable-Lighting Conditions for a Responsive Audio Art Installation».

[10] Z. Zivkovic y F. van der Heijden, «Efficient adaptive density estimation per image pixel for the task of background subtraction», *Pattern Recognit Lett*, vol. 27, n.º 7, pp. 773-780, may 2006, doi: 10.1016/j.patrec.2005.11.005.

[11] S. Zeevi, «BackgroundSubtractorCNT: A Fast Background Subtraction Algorithm», dic. 2016, doi: 10.5281/ZENODO.4267853.

[12] N. Srivastava, G. Hinton, A. Krizhevsky, y R. Salakhutdinov, «Dropout: A Simple Way to Prevent Neural Networks from Overfitting», 2014.

[13] J. Redmon, S. Divvala, R. Girshick, y A. Farhadi, «You Only Look Once: Unified, Real-Time Object Detection», jun. 2015, [En línea]. Disponible en: http://arxiv.org/abs/1506.02640

[14] J. Redmon, S. Divvala, R. Girshick, y A. Farhadi, «You Only Look Once: Unified, Real-Time Object Detection». [En línea]. Disponible en: http://pjreddie.com/yolo/

[15] A. Bochkovskiy, C.-Y. Wang, y H.-Y. M. Liao, «YOLOv4: Optimal Speed and Accuracy of Object Detection». [En línea]. Disponible en: https://github.com/AlexeyAB/darknet.

[16] E. T. S. I. Telecomunicación, D. Saúl, y R. Raneros, «UNIVERSIDAD DE VALLADOLID Estudio de la arquitectura YOLO para la detección de objetos mediante deep learning».

[17] Instituto Nacional de Estadística, «Estadística de defunciones según la causa de muerte», *https://www.ine.es/dynt3/inebase/index.htm?padre=8272&capsel=8277*, 8 de febrero de 2022.

[18] Organización Mundial de la Salud, «Prevención del suicidio: un instrumento para profesionales de los medios», 2000.

[19] T. Matsubayashi, Y. Sawada, y M. Ueda, «Does the installation of blue lights on train platforms prevent suicide? A before-and-after observational study from Japan», *J Affect Disord*, vol. 147, n.º 1-3, pp. 385-388, may 2013, doi: 10.1016/J.JAD.2012.08.018.

[20] J. Minguillon, M. A. Lopez-Gordo, D. A. Renedo-Criado, M. J. Sanchez-Carrion, y F. Pelayo, «Blue lighting accelerates post-stress relaxation: Results of a preliminary study», *PLoS One*, vol. 12, n.º 10, oct. 2017, doi: 10.1371/journal.pone.0186399.

[21] H. Kadotani, Y. Nagai, y T. Sozu, «Railway suicide attempts are associated with amount of sunlight in recent days», *J Affect Disord*, vol. 152-154, n.º 1, pp. 162-168, ene. 2014, doi: 10.1016/J.JAD.2013.08.040.

[22] L. C. Lubos, «The Role of Colors in Stress Reduction», *Liceo Journal of Higher Education Research*, vol. 5, n.º 2, dic. 2008, doi: 10.7828/ljher.v5i2.39.

[23] B. Farnham, S. Tokyo, B. Boston, F. Sebastopol, y T. Beijing, «Hands-on Machine Learning with Scikit-Learn, Keras, and TensorFlow Concepts, Tools, and Techniques to Build Intelligent Systems SECOND EDITION».

[24] T.-Y. Lin *et al.*, «LNCS 8693 - Microsoft COCO: Common Objects in Context», 2014.

[25] C. Li *et al.*, «YOLOv6: A Single-Stage Object Detection Framework for Industrial Applications», sep. 2022, [En línea]. Disponible en: http://arxiv.org/abs/2209.02976

[26] C.-Y. Wang, A. Bochkovskiy, y H.-Y. M. Liao, «YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors», jul. 2022, [En línea]. Disponible en: http://arxiv.org/abs/2207.02696

Recursos online consultados

AIA. (s.f.). *AIA*. Obtenido de Según la Automated Imaging Association.

Andrés, G. (21 de mayo de 2022). *https://www.metropoliabierta.com/*. Obtenido de https://www.metropoliabierta.com/informacion-municipal/sucesos/mas-30-personas-se-tiran-vias-tren-barcelona-suicidarse_55064_102.html

Bagnato, J. I. (2022). *https://www.aprendemachinelearning.com/*. Obtenido de https://www.aprendemachinelearning.com/como-funcionan-las-convolutional-neural-networks-vision-por-ordenador/

Baraniuk, C. (23 de enero de 2019). *https://www.bbc.com*. Obtenido de https://www.bbc.com/future/article/20190122-can-blue-lights-prevent-suicide-at-train-stations

Barrios, D. J. (s.f.). *https://www.juanbarrios.com/*. Obtenido de https://www.juanbarrios.com/la-matriz-de-confusion-y-sus-metricas/

Carreres, A. (29 de marzo de 2016). *https://www.vice.com/*. Obtenido de https://www.vice.com/es/article/yv9bvg/suicidio-metro-2903

Cuartas, J. (30 de 1 de 2021). *https://josecuartas.medium.com*. Obtenido de https://josecuartas.medium.com/el-concepto-de-la-convoluci%C3%B3n-en-gr%C3%A1ficos-para-comprender-las-convolutional-neural-networks-cnn-519d2eee009c

Ian Goodfellow, Y. B. (2016). Deep learning. London: The MIT Press.

Jiménez, J. (9 de agosto de 2019). *https://www.xataka.com*. Obtenido de https://www.xataka.com/medicina-y-salud/luz-azul-al-final-tunel-como-influye-luminosidad-suicidios

Liu, Y. (s.f.). *https://yanfengliux.medium.com/*. Obtenido de https://yanfengliux.medium.com/the-confusing-metrics-of-ap-and-map-for-object-detection-3113ba0386ef

Madrid, C. d. (2022). *http://www.madrid.org*. Obtenido de http://www.madrid.org/iestadis/fijas/estructu/general/anuario/ianucap09.htm

Sanchis, A. (2 de septiembre de 2022). *https://magnet.xataka.com/*. Obtenido de https://magnet.xataka.com/preguntas-no-tan-frecuentes/japon-lleva-anos-luchando-suicidios-luces-azules-metro-les-ha-funcionado

Índice de acrónimos

3FN	3º Forma Normal.
AIA	Automated Imaging Association, o asociación de imágenes automatizadas.
API	Interfaz de programación de aplicaciones.
ATO	Automatic Train Operation, u operación automática del tren.
AUC-ROC	Receiver Operating Characteristic, o característica operativa del receptor.
BBox	Bounding box, o cuadro delimitador.
C	Nombre de un lenguaje de programación de propósito general.
C++	Nombre de un lenguaje de programación de propósito general.
CCTV	Closed Circuit Television, o circuito cerrado de televisión.
CMS	Central Management System, o sistema de gestión central.
CNN	Convolutional Neural Networks, o redes neuronales convolucionales.
COCO	Common Objects in Context, u objetos comunes en contexto.
COVID-19	coronavirus disease, o enfermedad del coronavirus 19.
CPU	Central Processing Unit, o unidad central de procesamiento.
CUDA	Compute Unified Device Architecture, o arquitectura unificada de dispositivos de cómputo.
DL	Deep learning, o aprendizaje profundo.
DLR	Docklands Light Railway, o sistema de tren ligero.
eGPU	External Graphics Processing Unit, o unidad de procesamiento gráfico externa.

FAER	Facebook AI Research.
FPS	Frames per second, o frames por segundo.
GPU	Graphics Processing Unit, o unidad de procesamiento gráfico.
IA	Inteligencia artificial.
INE	Instituto Nacional de Estadística.
IoU	Intersection over Union, o interseción bajo la union.
LED	light-emitting diode, o Diodo de emisor de luz.
LiDAR	Light Detection and Ranging, o Laser Imaging Detection and Ranging.
ML	Machine learning, o aprendizaje profundo.
MVC	Model View Controller, o modelo vista controlador.
NVR/DVR	Network Video Recorder, o Digital Video Recorder, videograbadores videos digitales.
ORM	Object-Relational mapping, o mapeo objeto-relacional.
OSD	On screen display, o visualización en pantalla.
RAMS	Fiabilidad (Reliability), Disponibilidad (Availability), Mantenibilidad (Maintainability) y Seguridad (Safety).
ROI	Region of Interest, o área o región de una imagen.
SIL	Safety Integrity Level, o nivel de integridad de seguridad.

Anexo A: Instalación de OpenCV habilitada para GPU

OpenCV en su versión compilada descargable de github.com no viene habilitada para GPU de Nvidia, para poder usar esta librería habilitada con GPU se tiene que compilar de nuevo a partir de su código fuente, para ello debemos tener instalada previamente varias herramientas ***Cmake, VC16 2019.***

Cmake es herramientas que usa el compilador GCC de c++, se utiliza para controlar el proceso de compilación del software usando ficheros de configuración, que veremos más adelante.

https://cmake.org/download/

VC16 2019 es un IDE (entorno de desarrollo integrado) utilizado para el desarrollo de software.

https://visualstudio.microsoft.com/downloads/

Por otro lado, debemos tener instalado CUDA y la versión compatible de cuDNN según la GPU que se valla a usar, para saber la arquitectura binaria de la GPU tenemos que ir a la página

https://docs.nvidia.com/cuda/cuda-installation-guide-microsoft-windows/index.html o https://en.wikipedia.org/wiki/CUDA y para la cuDNN la dirección web https://developer.nvidia.com/rdp/cudnn-archive

Como para este proyecto hace falta tener instalado PyTorch también hará falta su versión compatible con CUDA y por consiguientes con su cuDNN.

Un ejemplo para la tarjeta graficas Nvidea 3060 Ti usada en este proyecto seria:

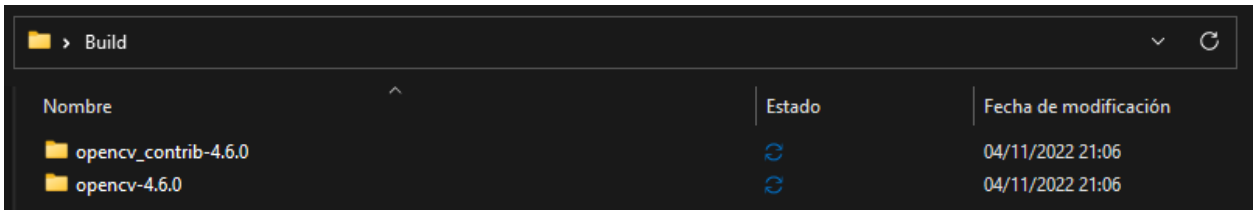
- CUDA de Nvidea 11.3 "CUDA Driver Version / Runtime Version 11.5.50"
- CUDA Capability Major/Minor version number: 8.6
- cuDNN v8.0 (guardar en la carpeta correspondiente normalmente "C:\Program Files\NVIDIA GPU Computing Toolkit\CUDA\v11.3\bin")
- PyTorch 1.11.0: install torch torchvision torchaudio --extra-index-url https://download.pytorch.org/whl/cu113

1. El primer paso será descargar los códigos fuentes de los proyectos `opencv` y los paquetes de módulos adicionales `opencv-contrib`

https://github.com/opencv/opencv/archive/refs/tags/4.6.0.zip

https://github.com/opencv/opencv_contrib/archive/refs/tags/4.6.0.zip

descomprimimos en una carpeta aparte llamada por ejemplo **Build**, que será la carpeta donde trabajaremos.

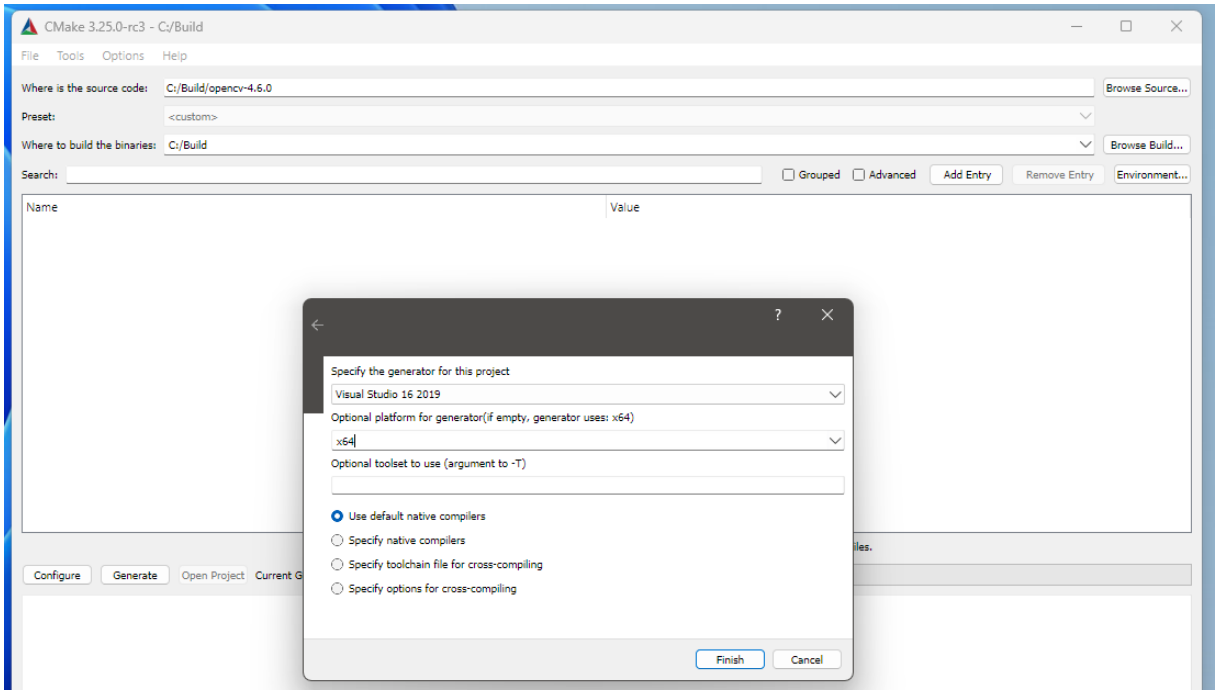


2. Instalamos el paquete numpy con: `pip install numpy`

3. Modificamos el archivo "OpenCVDetectPython.cmake" de la carpeta "\Build\opencv-4.6.0\cmake\" ya que para construir el proyecto CMake usará por defecto Python 2 en vez de Python 3, quedando así:

```
if(PYTHON_DEFAULT_EXECUTABLE)
    set(PYTHON_DEFAULT_AVAILABLE "TRUE")
elseif(PYTHON3INTERP_FOUND)
 # Use Python 3 as default Python interpreter
    set(PYTHON_DEFAULT_AVAILABLE "TRUE")
    set(PYTHON_DEFAULT_EXECUTABLE "${PYTHON3_EXECUTABLE}")
elseif(PYTHON2INTERP_FOUND)
    # Use Python 2 as fallback Python interpreter (if there is no Python 3)
    set(PYTHON_DEFAULT_AVAILABLE "TRUE")
    set(PYTHON_DEFAULT_EXECUTABLE "${PYTHON2_EXECUTABLE}")
endif()
```

4. Configuramos Cmake con la ruta de código fuente y la de la carpeta ***Bulid*** donde se guardarán los binarios compilados, presionamos el botón "*Configure*" para seleccionar la plantadora x64 como se muestra en la siguiente figura:



5. Al darle a botón "*Finish*" CMake empezará la configuración, una vez terminada chequearemos los siguientes ítems:

WITH_CUDA

OPENCV_DNN_CUDA

ENABLE_FAST_MATH

OPENCV_EXTRA_MODULES_PATH "\opencv-contrib-4.6.0\opencv-contrib-4.6.0\modules"

6. Volvemos a presionar "*Configure*", cuando termine buscaremos y chequearemos los siguientes ítems:

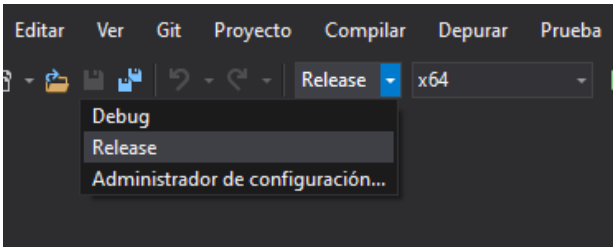
CUDA_FAST_MATH

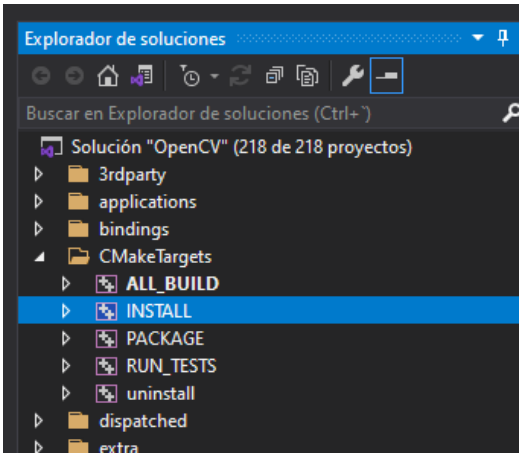
CUDA_ARCH_BIN — 7.5 (Este número dependerá de la arquitectura de cada GPU usada).

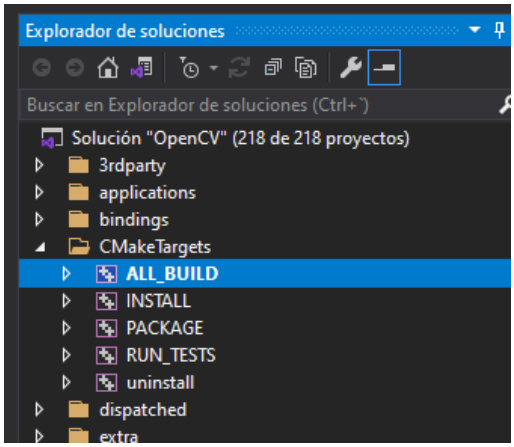
7. Presionamos de nuevo "*Configure*" y *"Generate"* esperamos una salida satisfactoria:

```
------------------------------------------------------------------

Configuring done
Generating done
```

8. Ahora abriremos el archivo “OpenCV.sln” de la carpeta Build con el programa Visual Code 16 2019, cambiamos en la barra de herramientas de “debug” a “release”, en la parte derecha saldrá un árbol con las carpetas del proyecto, buscamos la carpeta llamada “*CMakeTargets*” y la expandimos, en el ítem “*ALL_Build*” desplegamos el menú contextual con botón derecho del ratón y elegimos la opción *Compilar* cuando termine haremos lo mismo con el ítem “*INSTALL*”





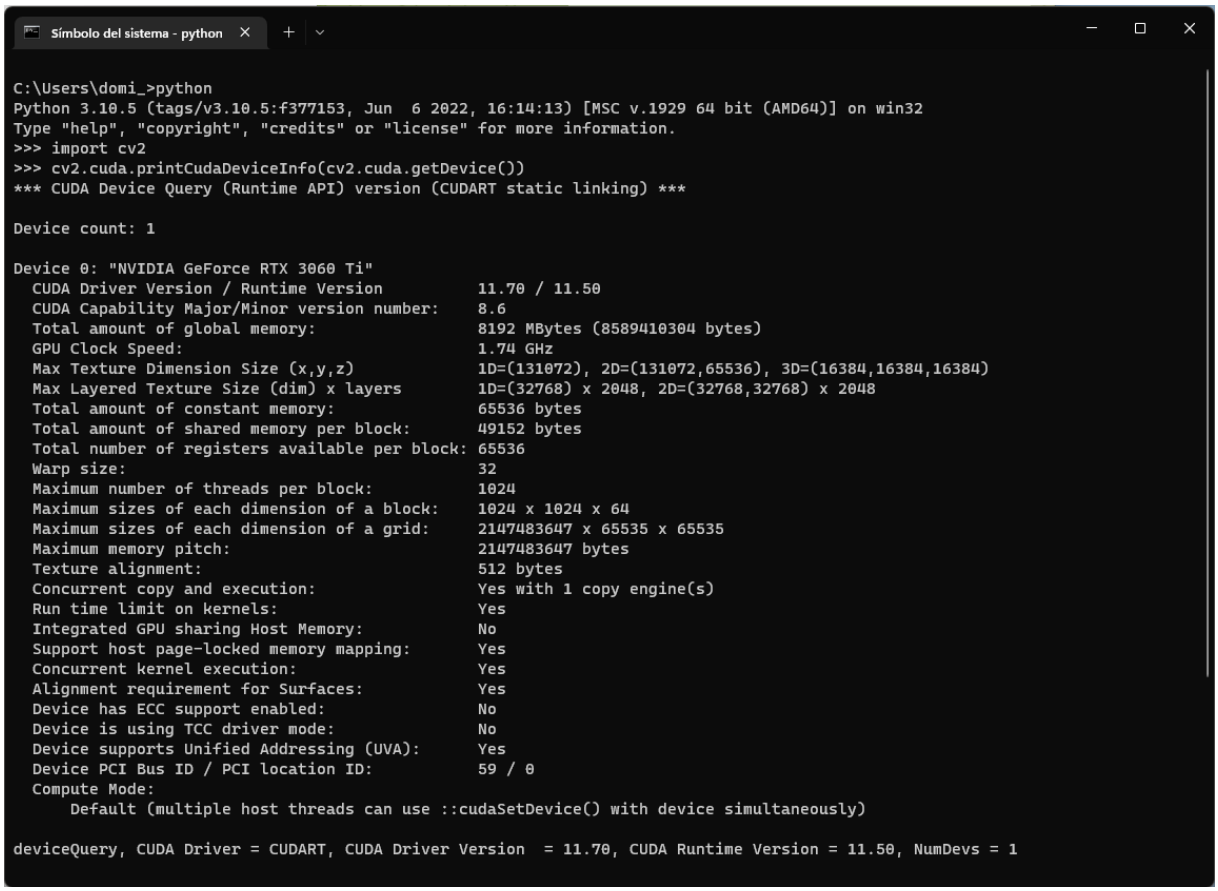


9. Para comprobar el éxito de la instalación podemos probar el siguiente comando:

import cv2

cv2.__version__

cv2.cuda.printCudaDeviceInfo(cv2.cuda.getDevice())



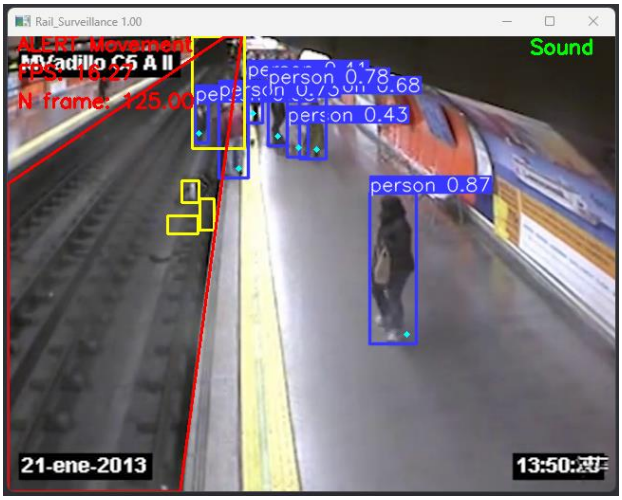
Anexo B: Creación de la Base de Datos

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
# ------------------------------------------------------------------------------
# Created By  : Domingo Martínez Núñez
# Created Date: abril 2022
# version ='1.0'
# https://github.com/Domy5/Rail_Surveillance/
# ------------------------------------------------------------------------------
""" Detention of people and/or objects on the railway platform in real time
"""
# ------------------------------------------------------------------------------
# https://www.sqlalchemy.org/


import sqlite3 as sql
import sqlalchemy
from sqlalchemy.ext.declarative import declarative_base
from sqlalchemy import Column, Integer, String, Table, ForeignKey
from sqlalchemy import create_engine

from sqlalchemy.orm import relationship
from sqlalchemy.orm import sessionmaker
from sqlalchemy.exc import IntegrityError as exc


# engine = create_engine('sqlite:///db/Camaras.sqlite', echo=True)
engine = create_engine('sqlite:///db/Cameras.sqlite')
base = declarative_base()

class Camera(base):
    __tablename__ = 'camera'

    camera_id = Column(Integer(), primary_key=True)
    line = Column(String(3), nullable=False, unique=False)
    station = Column(String(50), nullable=False, unique=False)
    platform = Column(Integer(), nullable=False, unique=False)
    against_direction = Column(Integer(), nullable=False, unique=False)
    path = Column(String(100), nullable=True, unique=True)
    ROI_polygon = relationship('ROI_polygon')

    def __str__(self):
        return self.camera_id

class ROI_polygon(base):
```

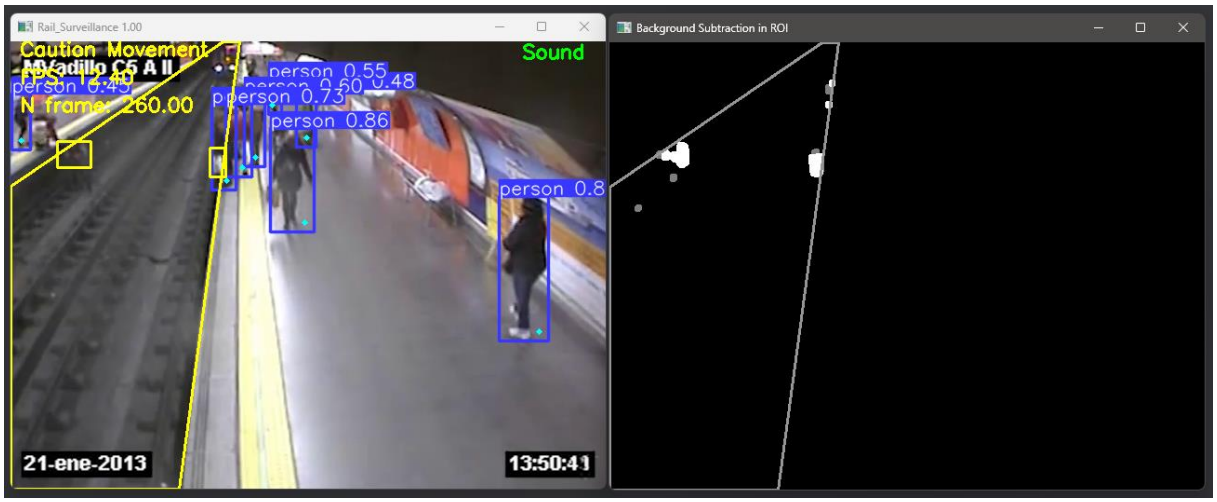
```
    __tablename__ = 'ROI_polygon'

    polygon_id = Column(Integer, primary_key=True)
    point_11    = Column(Integer, nullable=False, unique=False)
    point_12    = Column(Integer, nullable=False, unique=False)
    point_21    = Column(Integer, nullable=False, unique=False)
    point_22    = Column(Integer, nullable=False, unique=False)
    point_31    = Column(Integer, nullable=False, unique=False)
    point_32    = Column(Integer, nullable=False, unique=False)
    point_41    = Column(Integer, nullable=False, unique=False)
    point_42    = Column(Integer, nullable=False, unique=False)
    point_51    = Column(Integer, nullable=False, unique=False)
    point_52    = Column(Integer, nullable=False, unique=False)
    camera_id = Column(Integer, ForeignKey("camera.camera_id"))

    def __str__(self):
        return self.polygon_id

def get_video_path(camera_id):

    Session = sessionmaker(bind=engine)
    session = Session()

    camera = session.query(Camera).filter(Camera.camera_id == camera_id).all()

    session.close()

    return camera[0].path

def get_video_ROI(camera_id):

    Session = sessionmaker(bind=engine)
    session = Session()

    polygon = session.query(ROI_polygon).filter(ROI_polygon.camera_id ==
camera_id).all()

    session.close()

    polygon =
[[polygon[0].point_11,polygon[0].point_12],[polygon[0].point_21,polygon[0].poi
nt_22],[polygon[0].point_31,polygon[0].point_32],[polygon[0].point_41,polygon[
0].point_42],[polygon[0].point_51,polygon[0].point_52]]

    return polygon
```

Anexo C: Secuencias de ejecución del sistema

Fotograma de fragmento de video en línea 5 estación de Marque Vadillo andén II:



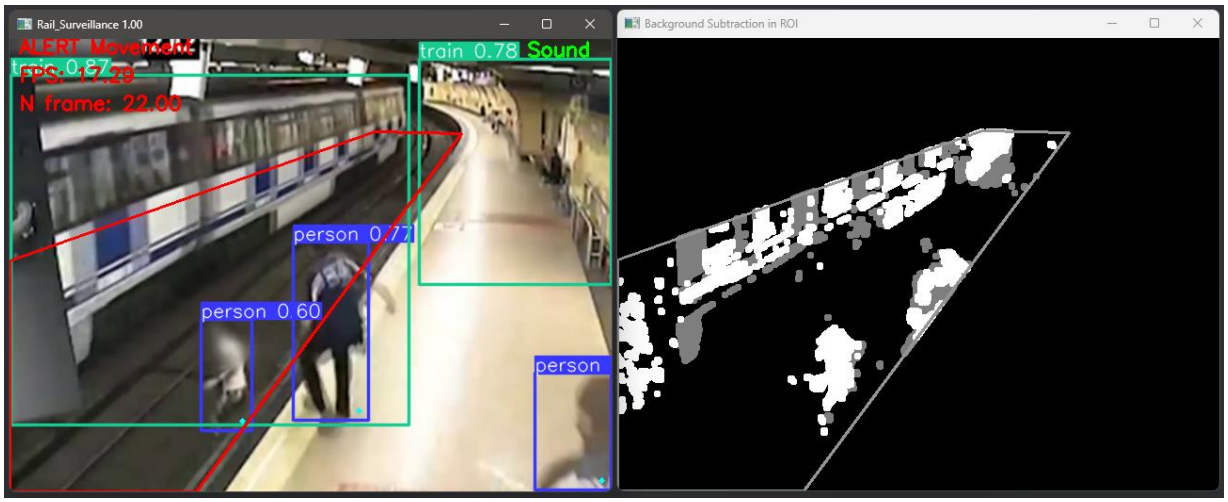
Alarma



Pre-Alarma

Fotograma de fragmento de video en línea 3 estación de Lavapiés andén II:

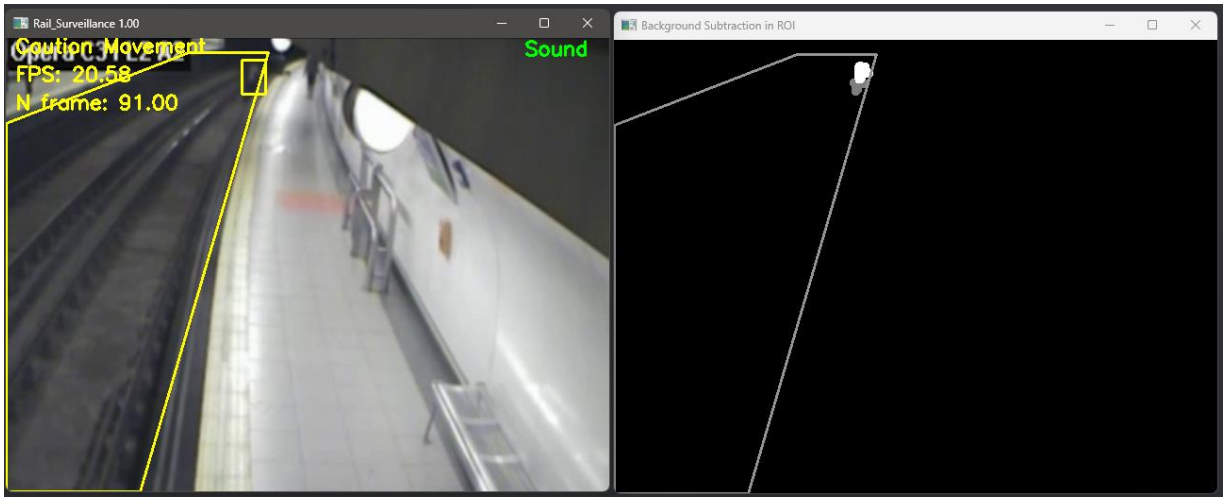
Alarma



Alarma

Fotograma de fragmento de video en línea 2 estación de Opera andén II:

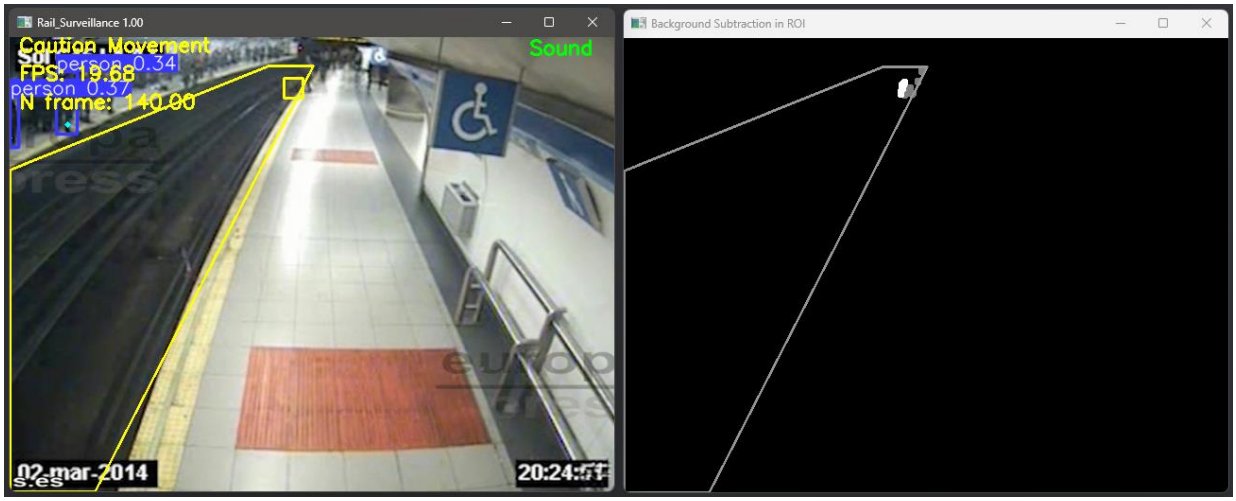
Pre-Alarma



Pre-Alarma

Fotograma de fragmento de video en línea 1 estación de Sol andén I:

Alarma



Pre-Alarma

Anexo D: Propuesta para publicación de artículos académicos

Detection of people and/or objects on railway tracks

D. Domingo Martínez Núñez
School of Engineering and Technology
The International University of La Rioja
Madrid, Spain
domi.mtz@gmail.com

Dr. Fernando Carlos López Hernández
Mathematical Analysis and Applied Mathematics Department
University Complutense of Madrid
Madrid, Spain
fclh@ucm.es

Dr. J. Javier Rainer Granados
School of Engineering and Technology
The International University of La Rioja
Madrid, Spain
javier.rainer@unir.net

***Abstract*— This paper describes the development and evaluation of a system for the detection of people and/or objects on railway tracks in real time. The paper describes several background subtraction techniques us with different CNNs using YOLO. The results show a novel strategy to mitigate the occlusion caused by the perspective of the cameras used and the conjunction of the two techniques that generate alarms and pre-alarms.**

To evaluate it, we have implemented an automated control and notification system with the fusion of current computer vision techniques with metrics of 15 FPS average per video and latency per frame processing of 0.54 seconds, meeting the expectations in terms of performance and with respect to the analysis of effectiveness and feasibility of the system in normal/abnormal regime.

The system matches the expected result, with improvement over current systems, anticipating accidents and providing more information to the operator. As a result, the proposed approach is a more effective, and economical alternative to the other approaches described in the paper.

Key words*— *Artificial vision, computer vision, artificial intelligence, machine learning, safety.

I. INTRODUCTION

The effort of this work is the research and implementation of a software tool that allows the detection of people and/or objects in areas delimited as dangerous in railway tracks, through the CCTV cameras already existing in the facilities, generating an automatic alarm that can stop the circulation of trains or speed up the detection of such incidents, notifying the agents in charge of security. For this purpose, the latest computer vision techniques, such as object detection and background subtraction in images, are implemented and evaluated.

Our research offers a significant contribution to the field of railway security, as it combines two innovative techniques to achieve accurate and efficient object detection, generating real-time alarms and pre-alarms with a single camera previously installed. In comparison with other works, investigated in the state-of-the-art chapter, which use multiple cameras, devices or additional sensors.

This work seeks to contribute to minimize the impact of this type of accidents, due to falls, of people on the railway tracks, as well as the fall of objects on the tracks that can generate a dangerous situation and the violation of railway rules during service hours (suicides, graffiti, vandalism, crossing of tracks, aggressions, etc.). All this disturbs the smooth running of railway lines.

The automatic processing of the camera images allows us to monitor in a more continuous and systematic way than a human could visualize (avoiding visual fatigue [1] and being able to attend to other job functions), classifying objects and people without interruptions. This system should make it possible to determine different dangerous situations and alert the decision-makers more quickly so that they can take the appropriate decisions.

With Artificial Intelligence (AI) assistance, particularly with two computer vision (VxC) techniques, object detection and background subtraction, we will be able to detect autonomously, the unauthorized presence on tracks with train traffic, in order to alert security personnel, station staff or traffic controllers, who will be able to stop the trains. This would avoid injuries, fatalities and material accidents, and would also minimize the loss of time for travelers until normal service is restored.

II. STATE OF THE ART

In the literature there are different techniques to detect people or objects on railway track platforms, such as the platform surveillance system using image processing technology proposed by Changmu Lee of EMU Advanced Research Team, Korea Railway Research Institute, Uiwang-City, Gyeonggi-Do, Korea [2] proposes by using multiple cameras perpendicular along the track to monitor almost the entire length of the track line on the platform.

Changmu Lee, divides the line monitoring process into two steps: detection of train status and detection of objects and people on the track.

Other algorithms [3], [4], that incorporate different sensing modalities, such as LiDAR, laser or remote sensing data, have been developed to detect pathways.

In the paper by T. Xiao et al. entitled "Multisensor Technology-Based Urban Rail Transit Obstacle Detection Method of Railway Traffic" [5], a non-contact multisensor technology detection method is proposed. For this purpose, an On-Board obstacle detection device based on a camera and a LiDAR device is installed, allowing to detect obstacles in the orbit area in real time, and to determine whether the train should brake automatically, or the engineer should brake manually.

The team of Sukruti et al. [6] has gone one step further by suggesting the implementation of a multisensory barrier composed of infrared (IR) and ultrasonic (US) sensors, in addition to a vision system, in order to alert the surveillance system about the presence of obstacles on the train track and thus prevent possible accidents.

In recent years, various methods have been developed to extract the rails. For example, F. Kaleli and Y. S. Akgul [7] proposed an algorithm based on dynamic programming to extract the trajectory and track ahead of the train in railway environments. This method consists of three steps: first, the Sobel operator is used to calculate the gradient of the input image, then a Hough transform process is applied to the binary image to detect the railroad line, finally dynamic programming is used to extract the tracks.

There is a neural network called RailNet created by Y. Wang, et al. [8], it is a segmentation network for track detection, trained with a self-constructed dataset (RSDS). This algorithm consists of a feature extraction network and a segmentation network.

However, in the paper by Juangwon Kang et al. Track Point Extraction and Association Network for Rail Path Proposal Generation [9], they propose a rail path extraction process in which the pixels of the left-right rails of each path are extracted and associated using a convolutional architecture called TPE-Net. Which has two different regression branches to return the locations of the center points of each rail route, generate the possible train routes (called "ego-routes"), the experimental results indicate that this technique has a high accuracy and recovery of true positives, reached in this approach the state of the art.

Recently (2022) A. D. Petrović et al. [9] have proposed a combination of deep neural networks (YOLO-v5 with CNN) for railway signal detection and the tracks themselves with VxC with Canny edge detection method and Hough transform.

Another technique Mask RCNN [10] employs a pyramidal structure to obtain high performance in the task of instance segmentation and human pose estimation on the COCO dataset.

With similarities in our approach, it differs in that our research does not focus on signals and track bifurcation, but on falling objects and people onto the tracks.

Haixia Pan & Tian [11] propose a multi-task learning network that detects, classifies tracking lines and performs line segmentation. This approach makes improvements over other multitask networks such as paying more attention to accuracy than recall. Segmentation aids classification results. The tracking line detection algorithm in the multitask network can effectively deal with the blocked track line problem; it avoids the disadvantage of segmentation network in detecting tracking lines with thinner and smaller pixel ratio. Its anchorless designs avoid the problem that the size of the anchor frame a priori is not suitable for small targets.

Recently at London Underground's Docklands Light Railway (DLR), experimentation is underway with the development of the CCTV AI Trial project [12]. The alert system has been deployed to prevent accidents on the network lines and ensure that they are not false alarms. A that this project is in the testing phase.

To the best of our knowledge, there are very few approaches using learning algorithms for detecting people and/or objects on railroad tracks, and there is little similar research that can be directly comparable to our work.

III. Materials y Methods

The proposed system is divided into two detection techniques, background subtraction and classification of objects in images through a convolutional neural network (CNN).

To evaluate which background subtraction technique best suits the needs of the system, several tests and iterations have been performed with the most advanced and current techniques, MOG2 (Gaussian Mixture Based Background/First Plane Segmentation Algorithm), GMG (Combination of statistical estimation of the background image and Bayesian segmentation per pixel), KNN (Nearest Neighbors), CNT (Count Based), comparing their results in terms of accuracy and speed of image processing for which, the MOG2 algorithm has been chosen. In image classification and based on the study of the existing literature of the different neural networks that perform this task, in which YOLO-v5 [13] has been chosen. It is also compared with respect to its previous versions at the level of accuracy and speed of image processing.

To evaluate the background subtraction algorithm that can perform better in the project there are several ways, a common way is to measure its accuracy, i.e., how many errors the algorithm makes when identifying the background in an image or video. This is done by comparing the results of the algorithm with a reference image or video that has already been manually labeled to show the correct background. Another aspect to evaluate the background subtraction algorithm is to measure its processing speed, i.e., how long it takes to process an image or video. This is important in situations where fast performance is required, such as in real-time applications as is the case in this project. In addition, the ease of use and flexibility of the algorithm, i.e., its ability to adapt to different types of images and videos, is also evaluated.

For this algorithm there are comparisons and evaluations with different metrics already published, as can be seen in

the AUC-ROC curve plot in Figure 1, the YOLOv5x6 model with the YOLOv5x6.pt weights previously trained from pytorch.org/hub, is the one that shows the best performance in speed and mAP [14] on the COCO [15] dataset of 5000 images at various inference sizes from 256 to 1536.

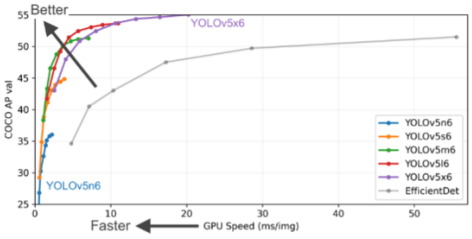


Figure 1. Metrics of different weights in the COCO val2017 dataset.

Source: https://github.com/ultralytics/yolov5

YOLOv5 has been chosen for the project due to its advantages in terms of accuracy and speed in object detection.

Compared to other object detection algorithms, YOLOv5 has proven to have high accuracy in object detection in different types of images and videos. It also has a high processing speed, thanks to CUDA implementation, which makes it suitable for real-time applications, flexible and easy to use.

A. Metrics used Background subtraction.

For the background subtraction technique, the number of non-zero pixels per frame has been taken as a metric, which indicates the effectiveness of the algorithm in detecting moving objects. A high number of non-zero pixels indicates that the algorithm has been able to effectively detect moving objects and separate them from the static background. On the other hand, a low number of pixels different from zero indicates that the algorithm has had difficulty separating the moving objects from the static background. However, this metric is only an indicator, as there may be cases where objects that are not actually objects are detected, especially in scenes with high illumination or noise.

To evaluate the background subtraction algorithm, it is essential to measure its processing speed, since the results are very different depending on the algorithm used.

B. Metrics used Object classifier.

A confusion matrix commonly employed in the evaluation of classification models is used for the object classifier, which uses the following set of indicators. TP (True Positive), TN (True Negative), FP (False Positive), FN (False Negative).

C. Global metrics.

The metric to evaluate the computational performance of the complete system (the object classifier and the background subtraction technique), is the latency in processing time speed of each image, which determines if it is appropriate for use in a real-time system.

To analyze the overall effectiveness and viability of the system is through direct observation of the dataset videos by checking the overall system performance and verifying the result of the alarms and pre-alarms with the expected result.

Another way to evaluate the system is to check the video extract with images in normal operation, in other words, without incidents that affect the tracks, or that are not activated by false negatives. This test is performed to ensure that the system does not generate false or unnecessary alarms or pre-alarms. It is an important measure to guarantee the reliability of the system and to avoid saturation or desensitization of users to real alarms.

IV. SOFTWARE DESIGN

Program logic implementation decisions for alarm generation.

A. Alarm per person on track platform.

One of the challenges of the project, using the existing CCTV cameras, is to determine whether a person is on the platform or in the roadway, in other words, whether a person is within the ROI or not.

The location of the cameras is at the end of each platform, with an angle that focuses at an oblique angle to the end of the opposite platform. For this reason, the cameras generate images with a perspective that causes the occlusion of a certain part of the track platform by people near the edge of the platform where the camera is located, as shown in Figure 2.

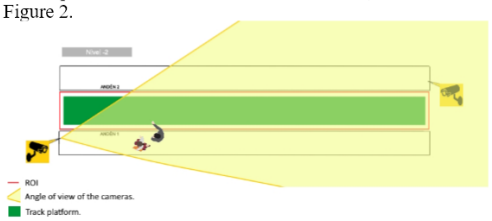


Figure 2. Perspective of the camera with respect to the track platform.

If a person is located at the edge of the platform, the object detection system will generate a BBox that will cross the area of interest or ROI as shown in the example in Figure 3, being a false alarm, as the person is not on the track platform, but on the platform.

Figure 3. BBox generated by object detection system.

To avoid the occlusion caused by people at the edge of the platform, with respect to the area of interest or ROI, it has been decided to generate a point as far away as possible, this being at the bottom right of each BBox, which corresponds to the left foot of a person facing the camera as shown in Figure 4, This point will serve as a reference to activate the alarm per object inside the ROI, emitting an alert sound, explained in another way, if the reference point is inside the polygon that represents the ROI, the alarm will be activated per person on the track platform.

Figure 4. Point generated at the bottom right of each BBox (left foot of each person).

B. Pre-alarmas por movimiento en plataforma de vías.

With the help of the chosen background subtraction algorithm MOG2, pre-alarms are generated when more than 5 frames in a row have more than 250 pixels different from 0 in the same contour, the pre-alarm emits an alert sound different from that of the alarms. It can be observed that only within the ROI is where the background subtraction, Figure 5, is executed and the movements on the platform are ignored, thus generating alarms of various kinds, such as falling people, bulky objects, flash flooding of the track basin, vault detachment, etc.

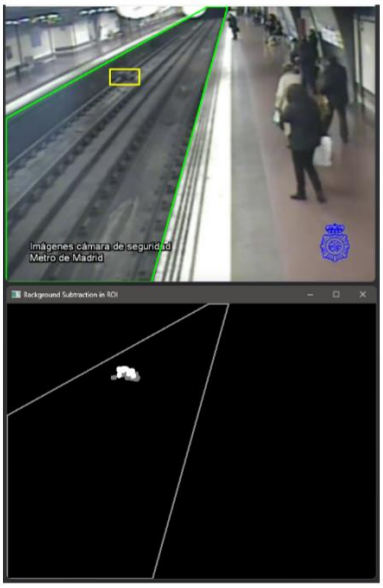


Figure 5. Detection by background subtraction within the ROI.

C. Detection upon train arrival.

Once the arrival of the train is detected, the possible pre-alarms are deactivated, but not the alarms generated by persons on the track platform or, in other words, by persons within the area of interest or ROI. This type of alarm, in addition to emitting the alert sound, will display the message "ARRIVAL OF THE TRAIN WITH PERSON ON THE TRAIN TRACK" on the console, as shown in Figure 6.

Figure 6. Messages displayed by console.

V. RESULTS

The two large blocks of the system are evaluated separately, background subtraction and object classification, then the complete system is evaluated.

A. Evaluation of the background subtraction algorithms.

To evaluate the background subtraction algorithms, the accuracy of each of the algorithms is compared, where it can be seen that the GMG algorithm has the most artifacts or noise, with respect to MOG2, they have their own advantages and disadvantages. However, there are some key differences between these algorithms that may make MOG2 better than GMG in certain situations. One of the main differences between MOG2 and GMG is that MOG2 uses an adaptive Gaussian mixture model, while GMG uses a global Gaussian mixture model. This means that MOG2 can dynamically adapt to changes in the scene background, which allows for greater accuracy in background identification and better performance in situations with a dynamic background. In comparison, GMG uses the first 120 frames of a video to build a model of the background, making it less adaptive to changes in the scene. Another important difference is that MOG2 is able to handle the overlapping of objects in the foreground and to identify objects that are moving slowly in the scene.

MOG2, compared to KNN, is easier to use and is more flexible, as it allows to adjust its parameters more easily. KNN only takes into account the distance between pixels and may perform less accurately in situations with a dynamic background. Therefore, MOG2 allows for higher accuracy in identifying small objects.

After several runs with the default subtractor parameter, other runs with improved parameters and in those subtractors that allow it, the application of morphological aperture kernel improves the results with respect to noise.

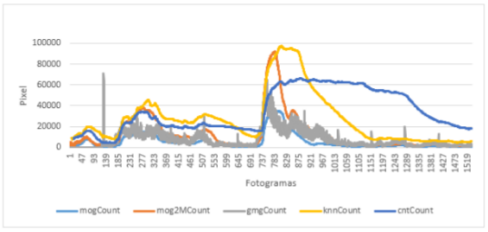


Figure 7. Number of non-zero pixels per frame.

The data are very similar in terms of number of pixels, the Figure 7 shows the time of arrival of the train on frame 740, CNT and KNN algorithms take many frames to recover the background state, which can cause many false positives in the long run, GMG has strong constant fluctuations that generate a lot of noise in the binary image, more stability and speed in the image changes can be seen in the MOG and MOG2 algorithms.

The Figure 8 shows the time each algorithm takes to process each frame. A better performance in time is seen in the CNT algorithm with 0.0010 seconds average time, followed by MOG2, with 0.0032 seconds average time, being MOG2 more stable than KNN (0.005 seconds), GMG (0.015 seconds), and MOG (0.007 seconds), have times that double the value of MOG2.

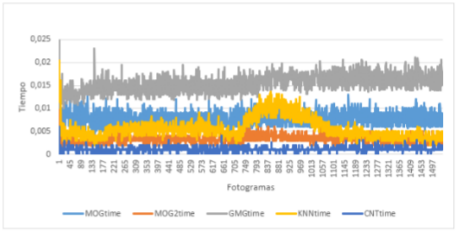


Figure 8. Processing time per frame.

B. Object classifier.

To evaluate the object classifier, several of the videos available in the test dataset were chosen, the frames were classified manually, then the system was run to save the results of the object classifier prediction automatically in a file and then, with respect to the metrics described above, to obtain the results for the "Train" and "Preson_on_via" object to generate the confusion matrices.

Table 1. Confusion matrix of the "Train" object.

Object: TRAIN		Prediction	
		Positives	Negatives
Real	Positives	792	4 (Type II error)
	Negatives	8 (Type I error)	732

Next, several metrics are extracted from the confusion matrix for the object "Train" Table 1, the precision metric is 99%, the accuracy metric reaches 99.21%, the specificity metric is 98.91%, the Recall or sensitivity metric for "Train" 99.49%, the F1 score metric is 99.24%.

Overall, these values indicate that the model performs well in class "Train" object classification, however, it is important to keep in mind that these results are based on a specific dataset and it is possible that the model performance may vary in different datasets or in different situations.

Table 2. Confusion matrix of the object "Person on via".

Object: Person on via		Prediction	
		Positives	Negatives
Real	Positives	29	504 (Type II error)
	Negatives	0 (Type I error)	1003

The metrics for the object "Person on via" extracted from the confusion matrix Table 2, are as follows, the precision metric is 100%, the accuracy metric reaches 67.18%, the specificity metric is 100%, the Recall or sensitivity metric for "Train" 0.05%, the F1 score metric is 0.10%.

Overall, these values indicate that the model has limited performance in classifying objects of the "Person on via" class. It is possible that the model is over-fitted to the "Person on via" class, resulting in high accuracy, but limited generalization performance.

C. Overall evaluation of the system.

The first evaluation is that of total frame time latency by Figure 9 inference and is performed on the Project test dataset recordings.

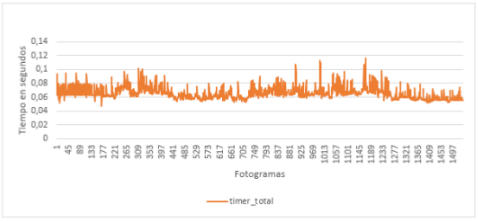


Figure 9. Per-frame inference latency with eGPU.

The Figure 10 shows the FPS data achieved by the system.

Table 3. FPS achieved with eGPU.

Figure 10. FPS achieved with eGPU.

The results indicate that the system is operating correctly in most of the video fragments analyzed.

Finally, the system is evaluated in several video fragments in which a normal operating regime is observed, in these fragments, the overall system result matches the expected result. This means that the system is working properly and meets the expectations in terms of performance and event detection capability.

VI. COMPARACIÓN

The contribution of our research focuses on the combination of state-of-the-art techniques, MOG2 background subtraction and neural networks with YOLO-v5, to benefit from the new contributions of these techniques, which improve the effectiveness and efficiency over other similar systems.

The work proposed by Changmu Lee makes use of multiple cameras combining frame difference information, thresholding, labeling and fusion with laser sensor detection. In this sense, the work differs from the aforementioned, as it does not require the installation of new elements, making use of only one CCTV camera per platform already installed.

On the other hand, the works of T. Xiao. and A. D. Petrović have different approaches to the detection of obstacles on the track. T. Xiao proposing a non-contact multisensor method, with cameras and a LiDAR device and A. D. Petrović proposing a combination of deep neural networks and edge detection methods, with cameras. Both proposals employ for the detection of obstacles on the road, On-Board devices. Based on this, our research focuses on object and person detection from each platform.

Finally, the work of Haixia Pan & Tian focuses on railway signal detection and the blocked track line problem. It differs from your contribution in its focus on track line detection and its combination of multi-task learning networks.

In general, our research is more flexible in terms of configuration and simplicity in generating alarms and pre-alarms. This makes it possible to easily implement new alarms in the program logic, it is scalable, being able to launch multiple instances for each camera.

In summary, our research focuses on specific aspects of railway safety, specifically the safety of people. It differs from other works by its focus on object/person detection on railway track platforms. Being able to confirm that our research gives a leap in terms of quality, greater configuration, simplicity, accuracy and speed in the processing of frames.

VII. DISCUSSION AND CONCLUSIONS

Supervision by means of artificial vision systems effectively automates and improves efficiency in the supervision of security systems. Our system uses cameras and machine vision algorithms to detect and analyze patterns in images and videos, enabling monitoring tasks to be performed with greater accuracy and repeatability than the current process performed by human eyesight. In addition, supervision by our machine vision system is more systematic and efficient than human supervision, as they can operate continuously.

In this project, we have presented a system for real-time detection of people and/or objects on railway tracks using modern image processing, object classification and background subtraction techniques, with an architecture based on YOLO-v5 and the MOG2 algorithm, respectively.

The MOG2 algorithm has generated processing speed metrics of 0.0032 seconds average time per frame, with very good accuracy, showing little artifacts or noise and great recovery by dynamically adapting to changes in the scene background, due to the adaptive Gaussian blending technique used. MOG2 allows greater precision in the identification of small objects and, due to the number of non-

zero pixels, more stability and speed in image changes can be appreciated.

With respect to image classification, the accuracy metric for the object "Person on via" is 100%. The accuracy metric for the "Person on via" object is 67.18% and the precision metric for the "Train" object is 99%, the accuracy metric for the "Train" object is 99.21%, which means that the model is predicting correctly every time it detects those objects. However, the accuracy is 67.18% for the "Person on via" object and 99.21% for the "Train" object, which means that the model is predicting correctly in a lower percentage of cases in general, regardless of the specific class.

REFERENCES

[1] Prado Montes, A., Morales Caballero, Á., & Molle Cassia, J. N. «Síndrome de Fatiga ocular y su relación con el medio laboral.» Medicina y seguridad del trabajo, 63(249), 345-361, dic. 2017.

[2] S. Oh, S. Park, y C. Lee, «A platform surveillance monitoring system using image processing for passenger safety in railway station», en ICCAS 2007 - International Conference on Control, Automation and Systems, pp. 394-398, ene. 2007.

[3] M. Arastounia, «Automated recognition of railroad infrastructure in rural areas from LIDAR data» Remote Sens., vol. 7, no. 11, ene. 2015.

[4] B. Le Saux, A. Beaupère, A. Boulch, J. Brossard, A. Manier, and G. Villemin, «Railway detection: From ltering to segmentation networks» in Proc. IEEE Int. Geosci. Remote Sens. Symp., Valencia, Spain, Jul. 2018.

[5] T. Xiao, Y. Xu, y H. Yu, «Research on Obstacle Detection Method of Urban Rail Transit Based on Multisensor Technology», Journal of Artificial Intelligence and Technology, vol. 1, n.o 1, pp. 61-67, ene. 2021.

[6] Taori, S., Kakade, V., Gandhi, S., & Jadhav. «Multi Sensor Obstacle Detection on Railway Tracks». International Research Journal of Engineering and Technology, mar. 2021.

[7] F. Kaleli and Y. S. Akgul, «Vision-based railroad track extraction using dynamic programming» in Proc. 12th Int. IEEE Conf. Intell. Transp. Syst., St. Louis, MO, USA, oct. 2009.

[8] Y. Wang, L. Wang, Y. H. Hu, and J. Qiu, «RailNet: A Segmentation Network for Railroad Detection» IEEE Access, vol. 7, pp. 143772–143779, oct. 2019.

[9] Kang, J., Ghorbanalivakili, M., Sohn, G., Beach, D., & Marin, «TPE-Net: Track Point Extraction and Association Network for Rail Path Proposal Generation». http://arxiv.org/abs/2302.05803, feb. 2023.

[10] A. D. Petrović et al., «Integration of Computer Vision and Convolutional Neural Networks in the System for Detection of Rail Track and Signals on the Railway», Applied Sciences (Switzerland), vol. 12, n.o 12, jun. 2022.

[11] K. He, G. Gkioxari, P. Dollár, and R. Girshick, ``Mask R-CNN,'' in Proc. IEEE Int. Conf. Comput. Vis. (ICCV), pp. 2980_2988. oct. 2017.

[12] H. Pan, Y. Li, H. Wang, y X. Tian, «Railway Obstacle Intrusion Detection Based on Convolution Neural Network Multitask Learning», Electronics (Basel), vol. 11, n.o 17, p. 2697, ago. 2022.

[13] K. Matt Nolan, «L'intelligence artificielle au service de la sécurité sur les voies de la ligne Docklands Light Railway du métro londonien», Mobilité Urbaine, France, Intelligence Artificielle, Sécurité, mar. 2022.

[14] J. Redmon, S. Divvala, R. Girshick, y A. Farhadi, «You Only Look Once: Unified, Real-Time Object Detection», jun. 2015.

[15] B. Farnham, S. Tokyo, B. Boston, F. Sebastopol, y T. Beijing, «Hands-on Machine Learning with Scikit-Learn, Keras, and TensorFlow Concepts, Tools, and Techniques to Build Intelligent Systems Second Edition».

[16] T.-Y. Lin et al., «LNCS 8693 - Microsoft COCO: Common Objects in Context», 2014