

Universidad Católica de Santa María

Facultad de Ciencias e Ingenierías Físicas y Formales

Escuela Profesional de Ingeniería Mecatrónica

Diseño y simulación de un sistema anticolisión para vehículos basado en visión artificial mediante YOLOv5 y un módulo de localización CenterNet-sim en la ciudad de Arequipa

Tesis presentada por:

Quispe Surco, Jose Guillermo

ORCID: 0009-0008-9855-0049

para optar el Título Profesional de Ingeniero Mecatrónico

Asesor:

Dr. Quispe Ccachuco, Marcelo Jaime

ORCID: 0000-0003-3182-6991

Arequipa - Perú

2026

UCSM-ERP

UNIVERSIDAD CATÓLICA DE SANTA MARÍA

INGENIERIA MECATRONICA

TITULACIÓN CON TESIS

DICTAMEN APROBACIÓN DE BORRADOR

Arequipa, 13 de Julio del 2026

Dictamen: 014639-C-EPIMMEM-2026

Visto el borrador del expediente 014639, presentado por:

2018204811 - QUISPE SURCO JOSE GUILLERMO

Titulado:

DISEÑO Y SIMULACIÓN DE UN SISTEMA ANTICOLISIÓN PARA VEHÍCULOS BASADO EN VISIÓN ARTIFICIAL MEDIANTE YOLOV5 Y UN MÓDULO DE LOCALIZACIÓN CENTERNET-SIM EN LA CIUDAD DE AREQUIPA

Nuestro dictamen es:

APROBADO

Titulo Profesional/Titulo de Segunda Especialidad/Grado Académico a optar:

INGENIERO MECATRÓNICO

29592341 - MESTAS RAMOS SERGIO ORLANDO
DICTAMINADOR

29277716 - CUADROS MACHUCA JUAN CARLOS
DICTAMINADOR

42394111 - COLLADO OPORTO CHRISTIAM GUILLERMO
DICTAMINADOR

DISEÑO Y SIMULACIÓN DE UN SISTEMA ANTICOLISIÓN PARA VEHÍCULOS BASADO EN VISIÓN ARTIFICIAL MEDIANTE YOLOV5 Y UN MÓDULO DE LOCALIZACIÓN CENTERNET-SIM EN LA CIUDAD DE AREQUIPA

INFORME DE ORIGINALIDAD

7%	3%	0%	7%
INDICE DE SIMILITUD	FUENTES DE INTERNET	PUBLICACIONES	TRABAJOS DEL ESTUDIANTE

FUENTES PRIMARIAS

1	Submitted to Universidad Andina Nestor Caceres Velasquez Trabajo del estudiante	5%
2	repositorio.unsch.edu.pe Fuente de Internet	1%
3	Submitted to Universidad Ricardo Palma Trabajo del estudiante	1%
4	repositorio.uncp.edu.pe Fuente de Internet	1%

Excluir citas	Apagado	Excluir coincidencias	< 1%
Excluir bibliografía	Apagado		

Dedicatoria

Dedico el presente trabajo a mis padres, Luisa Surco y Santos Quispe, por su amor, su sacrificio y por confiar plenamente en mí. Cada paso dado y cada meta alcanzada estuvieron siempre acompañados del deseo de honrar su esfuerzo y demostrar que su apoyo valió la pena.

A mi hermana, Aida Quispe, por ser un ejemplo constante en mi vida y por impulsarme siempre a no quedarme atrás, a seguir preparándome y a dar lo mejor de mí, incluso cuando yo dudaba.

A mi pareja, Helen Mancani, por estar a mi lado en cada etapa de este proceso, por su paciencia, comprensión y apoyo incondicional. Por escucharme, acompañarme y motivarme a ser mejor, tanto a nivel personal como profesional.

A Skayla, por su compañía incondicional y por estar siempre presente, brindándome alegría y tranquilidad incluso en los momentos de cansancio, estrés o frustración.

Agradecimientos

Agradezco en primer lugar a Dios, por brindarme la fortaleza, la constancia y la claridad necesarias para culminar este trabajo de investigación y esta etapa importante de mi formación profesional.

Expreso mi sincero agradecimiento a mi asesor de tesis, Marcelo Ccachuco, por su orientación, disposición y apoyo constante durante el desarrollo del presente trabajo. Su acompañamiento académico, trato cordial y observaciones oportunas fueron fundamentales para encaminar adecuadamente esta investigación.

Agradezco a la Universidad Católica de Santa María, mi alma mater, y a los docentes de la Escuela Profesional de Ingeniería Mecatrónica, por la formación académica recibida, la transmisión de conocimientos y el criterio profesional que contribuyeron de manera significativa a mi desarrollo académico y profesional.

Finalmente, agradezco a mis amigos José Ocsa y Elard Mayoria, por su apoyo, compañerismo y motivación durante esta etapa, contribuyendo a que este proceso académico sea más llevadero.

RESUMEN

La presente investigación tuvo como propósito diseñar y simular un sistema anticolisión vehicular para entornos urbanos de la ciudad de Arequipa, basado en visión artificial mediante el algoritmo YOLOv5 y un módulo basado en CenterNet. El estudio se desarrolló bajo un enfoque de simulación, empleando secuencias de video representativas de escenarios urbanos reales, con el objetivo de detectar y localizar obstáculos relevantes para la evaluación del riesgo de colisión. El sistema propuesto integra un modelo de detección de objetos basado en YOLOv5, encargado de identificar vehículos y otros obstáculos presentes en cada frame, y un módulo de localización inspirado en CenterNet, utilizado para analizar el comportamiento espacial y dinámico de los objetos mediante el cálculo de sus centros geométricos. La evaluación se realizó sobre un flujo continuo de 4669 frames, contrastando las alertas emitidas contra un modelo Oráculo. Los resultados cuantitativos obtenidos en el entorno de simulación muestran que el sistema alcanzó una sensibilidad (Recall) del 95.31% en la detección efectiva de amenazas de riesgo dinámico y una precisión del 58.10%, con un puntaje global F1-Score de 0.7218, demostrando una alta eficacia preventiva frente a la omisión de impactos. No obstante, estos valores fueron dadas en condiciones controladas del estudio, debido a que la evaluación se realizó sobre escenarios simulados, sin pruebas físicas en vehículos reales ni exposición a variaciones operativas propias del tránsito urbano. En términos de desempeño computacional, el tiempo medio de procesamiento fue de 18.45 ms por frame para YOLOv5 y de 0.38 ms por frame para el módulo CenterNet-sim, obteniéndose un rendimiento aproximado de 53.11 FPS. Estos valores evidencian que la carga computacional del sistema se concentra principalmente en el módulo de detección, mientras que la etapa de localización y decisión presenta un costo mínimo. En conjunto, los resultados obtenidos demuestran que el sistema anticolisión diseñado es técnicamente viable a nivel de simulación y altamente capaz de representar situaciones de riesgo propias del tránsito urbano.

Palabras clave: Detección de objetos, sistemas anticolisión, visión artificial.

ABSTRACT

The purpose of this research was to design and simulate a vehicle collision avoidance system for urban environments in the city of Arequipa, based on artificial vision using the YOLOv5 algorithm and a module based on CenterNet. The study was developed using a simulation approach, employing video sequences representative of real urban scenarios, with the aim of detecting and locating obstacles relevant to collision risk assessment. The proposed system integrates an object detection model based on YOLOv5, responsible for identifying vehicles and other obstacles present in each frame, and a localization module inspired by CenterNet, used to analyze the spatial and dynamic behavior of objects by calculating their geometric centers. The evaluation was performed on a continuous flow of 4669 frames, contrasting the issued alerts against an Oracle model. The quantitative results obtained in the simulation environment show that the system achieved a sensitivity (Recall) of 95.31% in the effective detection of dynamic risk threats and a precision of 58.10%, with an overall F1-Score of 0.7218, demonstrating high preventive efficacy against impact omissions. However, these values were obtained under controlled laboratory conditions, as the evaluation was conducted on simulated scenarios without physical testing on real vehicles or exposure to the operational variations typical of urban traffic. In terms of computational performance, the average processing time was 18.45 ms per frame for YOLOv5 and 0.38 ms per frame for the CenterNet-sim module, yielding an approximate performance of 53.11 FPS. These values show that the computational load of the system is mainly concentrated in the detection module, while the localization and decision stage has a minimal cost. Overall, the results obtained demonstrate that the designed anti-collision system is technically feasible at the simulation level and highly capable of representing risk situations typical of urban traffic.

Keywords: Object detection, anti-collision systems, computer vision.

ÍNDICE

DEDICATORIA

AGRADECIMIENTOS

RESUMEN

ABSTRACT

INTRODUCCIÓN 1

1. CAPÍTULO I PLANTEAMIENTO DE LA INVESTIGACIÓN 3
   1.1. Enunciado del problema 3
   1.2. Identificación del problema 3
   1.3. Descripción del problema 3
   1.4. Justificación 4
      1.4.1. Justificación Social 4
      1.4.2. Justificación Académica 5
      1.4.3. Justificación Técnica 5
   1.5. Objetivos de la investigación 5
      1.5.1. Objetivo general 5
      1.5.2. Objetivos específicos 6
   1.6. Hipótesis 6
   1.7. Alcances 6
   1.8. Limitaciones 7
   1.9. Consideraciones éticas, privacidad y seguridad de uso 7
   1.10. Variables 8
   1.11. Antecedentes 9
      1.10.1. Antecedentes Internacionales 9
      1.10.2. Antecedentes Nacionales 10
2. CAPÍTULO II MARCO TEÓRICO 12
   2.1. Bases Teóricas 12
      2.1.1. Sistema anticolisión 12

2.1.2. Comprensión y análisis del entorno 14
2.1.3. Visión artificial 16
2.1.3.1. Fundamentos de la visión artificial 20
2.1.3.2. Etapas de la Visión Artificial 22
2.1.3.3. Metodología de Desarrollo 23
2.1.4. Detección de objetos 31
2.1.5. Métricas u medidas de detección de objetos 32
2.1.6. YOLO (You Only Look Once) 34
2.1.7. CenterNet 38
3. CAPÍTULO III DESARROLLO DE UN MODELO DE DETECCIÓN DE OBJETOS 40
3.1. Diseño del sistema anticolisión 40
3.1.1. Requerimientos del sistema 40
3.1.2. Evaluación de tecnologías disponibles 45
3.1.2.1. Criterios de evaluación 45
3.1.2.2. Matriz de evaluación de detectores 45
3.1.2.3. Matriz de evaluación de localización 50
3.2. Proceso metodológico 55
3.3. Modelo de detección de objetos para vehículos 57
3.3.1. Adquisición de imágenes y calibración de la cámara 57
3.3.2. Parámetros intrínsecos de la cámara 58
3.3.3. Parámetros extrínsecos de la cámara 59
3.3.4. Representación y preprocesamiento de la imagen 61
3.3.5. Detección de objetos mediante YOLOv5 61
3.3.6. Zona de peligro con el modelo de detección YOLOv5 63
3.4. Modelo para localización de obstáculos estáticos y dinámico 65
3.4.1. Localización por centro del objeto 67
3.4.2. Asociación temporal (centro actual vs. centros previos) 67
3.4.3. Clasificación del movimiento: estático vs dinámico 68
3.4.4. Estimación de desplazamiento y velocidad aparente 69

3.4.5. Evaluación espacial en la zona de peligro (ROI) 69
3.4.6. Regla de decisión para alerta de colisión (CenterNet-sim) 70
4. CAPÍTULO IV RESULTADOS Y ANÁLISIS 71
4.1. Desempeño del sistema anticolisión basado en YOLOv5 y CenterNet-sim 71
4.1.1. Simulación del desempeño basado en YOLOv5 71
4.1.2. Simulación del desempeño basado en CenterNet-sim 76
4.2. Métricas de precisión del sistema anticolisión 82
4.3. Viabilidad técnica del sistema anticolisión diseñado 86
4.3.1. Viabilidad funcional: detección, localización y decisión 86
4.3.2. Viabilidad cuantitativa: precisión y consistencia de detección 89
4.3.3. Viabilidad temporal: tiempos de procesamiento y FPS 91
4.3.4. Viabilidad operativa en escenarios urbanos simulados 93
4.3.5. Requisitos mínimos y escalabilidad del sistema 94
4.4. Contrastación de la hipótesis 95
4.5. Propuesta de una plataforma de hardware del sistema anticolisión 95
4.6. Análisis de impacto ambiental indirecto 101
CONCLUSIONES 109
RECOMENDACIONES 111
REFERENCIAS BIBLIOGRAFÍCAS 113

ÍNDICE DE TABLAS

Tabla 1 *Criterios para la selección de tecnologías del sistema anticolisión* 45
Tabla 2 *Matriz comparativa de algoritmos de detección de objetos* 46
Tabla 3 *Matriz comparativa de algoritmos de localización de objetos* 51
Tabla 4 *Desempeño del sistema anticolisión basado en YOLOv5 y CenterNet-sim* 82
Tabla 5 *Desempeño del sistema anticolisión basado en YOLOv5 y CenterNet-sim* 84
Tabla 6 *Plataforma de hardware propuesta para el sistema anticolisión* 96
Tabla 7 *Costos - sistema anticolisión embebido* 98
Tabla 8 *Análisis ROI, VAN, TIR y sensibilidad* 99
Tabla 9 *Comparativa técnico-económica de opciones de un sistema anticolisión* 100
Tabla 10 *Parámetros de entrada del análisis ambiental indirecto* 101
Tabla 11 *Indicadores ambientales* 106
Tabla 12 *Proyección por tamaño de flota - sensibilidad del impacto indirecto* 107

ÍNDICE DE FIGURAS

Figura 1 *Diagrama de flujo del algoritmo basado en vóxeles* 15
Figura 2 *Diagrama Global del Proceso de Visión Artificial* 18
Figura 3 *Diagrama de bloques de un sistema de visión artificial* 19
Figura 4 *Expresión digital de una imagen* 21
Figura 5 *Obtención de imágenes desde múltiples puntos de vista* 21
Figura 6 *Etapas de la Visión Artificial* 22
Figura 7 *Fases del procesamiento de imágenes en visión artificial* 26
Figura 8 *Métodos aplicados a píxeles y regiones vecinas* 27
Figura 9 *Modelo de detección YOLO* 37
Figura 10 *Arquitectura Yolo* 37
Figura 11 *Perfil de desempeño de los modelos de detección* 50
Figura 12 *Perfil de desempeño de los modelos de localización* 54
Figura 13 *Diagrama de flujo del Proceso Metodológico* 56
Figura 14 *Escenario urbano N°1* 72
Figura 15 *Escenario urbano N°2* 73
Figura 16 *Escenario urbano N°3* 74
Figura 17 *Escenario urbano N°4* 75
Figura 18 *Localización de obstáculos estáticos y dinámicos escenario N°1* 77
Figura 19 *Localización de obstáculos estáticos y dinámicos escenario N°2* 78
Figura 20 *Localización de obstáculos estáticos y dinámicos escenario N°3* 79
Figura 21 *Activación de alerta de colisión en entorno urbano mediante CenterNet-sim* .. 80
Figura 22 *Sistema anticolisión con alerta de colisión en escenario urbano* 81
Figura 23 *Consumo energético anual por escenario (por unidad)* 108

ÍNDICE DE ANEXOS

Anexo N° 1. Matriz de Consistencia 119
Anexo N° 2. Código del sistema Anticolisión YOLO y CenterNet-sim 121
Anexo N° 3. Base de datos 128
Anexo N° 4. Evidencia de las simulaciones 131

INTRODUCCIÓN

El crecimiento del parque automotor en las ciudades ha incrementado la complejidad del tránsito urbano y, con ello, la probabilidad de ocurrencia de accidentes de tránsito, especialmente en zonas con alta densidad vehicular y peatonal. En este contexto, los sistemas de asistencia a la conducción orientados a la prevención de colisiones representan una alternativa tecnológica relevante para mejorar la seguridad vial. Sin embargo, la incorporación de estos sistemas suele estar limitada a vehículos de gama media y alta, dejando a un amplio número de unidades sin acceso a tecnologías de apoyo basadas en percepción del entorno.

La visión artificial aplicada a la detección y localización de objetos ha demostrado ser una herramienta eficiente para el análisis del entorno vehicular, al permitir identificar obstáculos, estimar su posición y evaluar situaciones de riesgo. El desarrollo de algoritmos de aprendizaje profundo, como YOLOv5 y CenterNet, ha facilitado la implementación de soluciones capaces de operar en escenarios urbanos complejos, caracterizados por la presencia simultánea de vehículos, peatones y elementos de infraestructura vial. No obstante, el desafío radica en integrar estos algoritmos de manera eficiente, asegurando precisión en la detección y viabilidad computacional para su uso en sistemas de asistencia a la conducción.

Bajo este enfoque, la presente investigación se orienta al diseño y simulación de un sistema anticolisión vehicular basado en visión artificial, aplicable a entornos urbanos de la ciudad de Arequipa. El estudio se centra en la utilización del algoritmo YOLOv5 para la detección de objetos y un módulo basado en CenterNet para la localización y análisis del comportamiento de los obstáculos, considerando escenarios urbanos simulados construidos a partir de datos reales. El objetivo es establecer una base técnica que permita evaluar el funcionamiento del sistema y su potencial aplicación en vehículos que actualmente no cuentan con tecnologías avanzadas de asistencia a la conducción.

La presente investigación se encuentra estructurado de la siguiente manera: el Capítulo I presenta el planteamiento del problema, los objetivos y la justificación de la investigación; el Capítulo II desarrolla el marco teórico y los antecedentes relacionados con

sistemas anticolisión y visión artificial; el Capítulo III describe el proceso metodológico y el diseño del sistema propuesto; el Capítulo IV expone el análisis del desempeño y la evaluación del sistema anticolisión; finalmente, se recoge las conclusiones y recomendaciones derivadas del estudio.

CAPÍTULO I
PLANTEAMIENTO DE LA INVESTIGACIÓN

1.1. Enunciado del problema

El departamento de Arequipa registró 6299 accidentes de tránsito durante el año 2025, de acuerdo con las estadísticas del Ministerio de Transportes y Comunicaciones (MTC, 2026). Asimismo, el Observatorio Nacional de Seguridad Vial reportó 166 personas fallecidas a consecuencia de siniestros de tránsito en la región durante 2024, de las cuales 64 correspondieron a hechos ocurridos en zonas urbanas (ONSV, 2025). Este escenario resulta particularmente relevante para los vehículos de reparto urbano, que operan en zonas de alta densidad vehicular y peatonal, bajo presión de tiempos de entrega y, en su mayoría, sin tecnologías avanzadas de asistencia. La implementación de un sistema anticolisión basado en visión artificial mediante YOLOv5 y CenterNet-sim podría mitigar este riesgo, permitiendo la detección y localización en tiempo real de obstáculos en entornos urbanos.

1.2. Identificación del problema

Se identifica como problema principal el alto riesgo de accidentes en vehículos de reparto urbano en la ciudad de Arequipa debido a la falta de sistemas de asistencia a la conducción que permitan anticipar y prevenir colisiones. Durante 2025, el departamento de Arequipa registró 6299 accidentes de tránsito, evidenciando la magnitud de la siniestralidad vial en el ámbito regional (MTC, 2026). Factores como el exceso de velocidad, imprudencia de conductores e invasión de carriles contrarios incrementan la probabilidad de incidentes. Esta situación demanda soluciones tecnológicas accesibles, escalables y adaptadas a condiciones urbanas, como el uso de algoritmos de detección de objetos YOLOv5 y CenterNet, que han demostrado eficacia en contextos complejos (Argese, 2023).

1.3. Descripción del problema

En la ciudad de Arequipa, el acelerado crecimiento del comercio electrónico y la expansión de los servicios de entrega a domicilio han incrementado notablemente la circulación de motocicletas, mototaxis y furgonetas ligeras. Estos vehículos, al operar en áreas céntricas y de alta densidad vehicular, se enfrentan diariamente a un entorno con

múltiples factores de riesgo, como intersecciones congestionadas, zonas escolares, presencia elevada de peatones y vías con geometría compleja o deficiente señalización. A ello se suma la presión por cumplir tiempos de entrega reducidos, lo que frecuentemente induce conductas de conducción más arriesgadas, incluyendo exceso de velocidad, maniobras bruscas y adelantamientos peligrosos.

La ausencia de sistemas avanzados de asistencia a la conducción en la mayoría de estas unidades limita la capacidad del conductor para anticipar o reaccionar eficazmente ante obstáculos o eventos imprevistos. En este contexto, los sistemas anticolisión basados en visión artificial se presentan como una alternativa tecnológica viable para mejorar la seguridad operativa, ya que permiten detectar y localizar objetos en tiempo real, generando alertas que apoyan la toma de decisiones inmediata. Entre las soluciones tecnológicas más prometedoras destacan algoritmos como YOLOv5, reconocido por su rapidez y precisión en la detección simultánea de múltiples objetos bajo diferentes condiciones de visibilidad (Argese, 2023), y CenterNet, valorado por su capacidad para identificar con alta exactitud objetos parcialmente visibles o en entornos con alta densidad de elementos (Sun et al., 2021). La integración de ambos algoritmos en un sistema unificado podría mejorar la cobertura y fiabilidad de la detección, incluso en condiciones adversas como lluvia ligera, neblina o iluminación deficiente.

Dada la complejidad de implementar directamente estas tecnologías en vehículos reales por razones de seguridad y costos, la simulación computacional en escenarios urbanos representativos de Arequipa constituye un paso previo indispensable. Este enfoque permite recrear situaciones de riesgo de forma controlada, validar el desempeño de los algoritmos y ajustar parámetros antes de su implementación física (Lacatan et al., 2021). Así, se sientan las bases para una solución escalable, adaptable y de bajo impacto estructural en los vehículos de reparto urbano, con potencial de reducir significativamente la incidencia de accidentes y mejorar la seguridad vial en la región.

1.4. Justificación

1.4.1. Justificación Social

La seguridad vial es un aspecto prioritario para la protección de la vida humana y la reducción de daños materiales. En la ciudad de Arequipa, los vehículos de reparto urbano

presentan un alto nivel de exposición al riesgo debido a sus condiciones de operación, la alta densidad de tránsito y la ausencia de sistemas de asistencia a la conducción. La propuesta de un sistema anticolisión basado en visión artificial busca disminuir la probabilidad de accidentes, protegiendo tanto a conductores como a peatones y demás usuarios de la vía. Además, al tratarse de una solución adaptable y escalable, puede beneficiar a un amplio espectro de operadores, incluyendo pequeños negocios y empresas de logística, contribuyendo a una movilidad urbana más segura y eficiente.

1.4.2. Justificación Académica

Esta investigación aporta al desarrollo de soluciones tecnológicas en el ámbito de la ingeniería mecatrónica, fortaleciendo la formación académica mediante la aplicación de conocimientos en visión artificial, detección de objetos y simulación de sistemas. El proyecto no solo permite evaluar el desempeño de algoritmos avanzados en escenarios urbanos representativos, sino que también proporciona un marco metodológico que puede ser replicado y adaptado a otras problemáticas. Con ello, se fomenta la investigación aplicada y se impulsa la generación de conocimiento orientado a resolver necesidades reales, integrando el aprendizaje teórico con la práctica experimental.

1.4.3. Justificación Técnica

La propuesta se sustenta en el desarrollo de un sistema anticolisión basado en visión artificial, utilizando algoritmos de alta precisión y velocidad de procesamiento. Su diseño considera la integración de cámaras y unidades de procesamiento capaces de detectar y localizar obstáculos en tiempo real, generando alertas oportunas para el conductor. Al emplear un entorno de simulación, es posible recrear condiciones operativas variadas, evaluar el rendimiento del sistema y optimizar sus parámetros antes de su implementación física. Esto permite garantizar un diseño eficiente, escalable y adaptable a diferentes tipos de vehículos de reparto, reduciendo costos de instalación y facilitando su incorporación en flotas que actualmente carecen de tecnologías avanzadas de asistencia a la conducción.

1.5. Objetivos de la investigación

1.5.1. Objetivo general

Diseñar y simular un sistema anticolisión vehicular basado en visión artificial mediante YOLOv5 y un módulo de localización inspirado en CenterNet, orientado a la detección y localización de obstáculos en escenarios urbanos simulados de la ciudad de Arequipa.

1.5.2. Objetivos específicos

a) Desarrollar un modelo de detección de objetos para vehículos utilizando el algoritmo YOLOv5.
b) Simular el funcionamiento de CenterNet-sim para la localización de obstáculos estáticos y dinámicos bajo distintos escenarios urbanos.
c) Integrar y evaluar el desempeño del sistema anticolisión basado en YOLOv5 y CenterNet-sim mediante métricas comparativas, utilizando entornos simulados construidos a partir de datos reales.
d) Analizar la viabilidad técnica del sistema anticolisión diseñado para su posible implementación en vehículos que actualmente no cuentan con tecnologías de asistencia a la conducción.

1.6. Hipótesis

La implementación de un sistema anticolisión vehicular basado en visión artificial mediante YOLOv5 y el módulo CenterNet-sim permite detectar y localizar obstáculos relevantes en escenarios urbanos simulados de la ciudad de Arequipa, presentando un desempeño técnico favorable de acuerdo con las métricas de precisión, sensibilidad, F1-score y rendimiento computacional establecidas para la evaluación.

1.7. Alcances

- Se diseñará un modelo basado en el algoritmo YOLOv5 para identificar obstáculos en entornos urbanos simulados, priorizando la detección en tiempo real y la capacidad de adaptación a escenarios dinámicos y complejos.
- Se simulará un módulo de localización inspirado en CenterNet para evaluar la ubicación de obstáculos estáticos y dinámicos, considerando centros geométricos, desplazamiento aparente y zona de peligro en escenarios urbanos simulados.
- Se integrarán YOLOv5 y CenterNet-sim en un sistema funcional, evaluando su desempeño mediante simulaciones en escenarios urbanos representativos de la ciudad de Arequipa.

- Se realizarán pruebas para analizar el comportamiento del sistema bajo diferentes condiciones ambientales y escenarios de tráfico, midiendo indicadores como precisión, tiempo de respuesta y efectividad.
- El término "CenterNet" se utiliza en esta investigación para denotar un esquema simplificado de localización basado en el cálculo del centro geométrico de las detecciones, no la implementación completa de la arquitectura CenterNet propuesta por Zhou et al. (2019). En adelante, este módulo se referirá como CenterNet-sim para reflejar fielmente su naturaleza.

1.8. Limitaciones

- El proyecto se limitará a la simulación de escenarios urbanos, sin realizar implementaciones físicas ni pruebas en vehículos reales.
- El diseño y evaluación se centrarán exclusivamente en los algoritmos YOLOv5 y CenterNet, dejando fuera otros tipos de algoritmos.
- La evaluación del sistema se realizó en condiciones ambientales favorables, caracterizadas por adecuada iluminación diurna y ausencia de precipitaciones o neblina. En consecuencia, no se determinó su desempeño frente a condiciones meteorológicas adversas que pudieran deteriorar la calidad de las imágenes.
- El sistema fue evaluado exclusivamente en vías de superficie llana, no se evaluó su comportamiento en pendientes pronunciadas, donde la inclinación de la cámara podría alterar la geometría de la zona de peligro.
- Al no considerar pruebas con hardware físico, no será posible evaluar la interacción directa entre el sistema diseñado y los vehículos en condiciones operativas reales.
- Debido a que la investigación se desarrollará únicamente en un entorno de simulación, no se realizará una medición experimental del consumo energético ni de la huella computacional asociada al entrenamiento e inferencia de los modelos propuestos.
- En consecuencia, el análisis de impacto ambiental se limitará a una justificación teórica de la ausencia de efectos directos, sin incluir una evaluación cuantitativa de impactos indirectos vinculados al uso de recursos computacionales.

1.9. Consideraciones éticas, privacidad y seguridad de uso

La presente investigación se orienta a contribuir con la seguridad vial mediante el diseño y simulación de un sistema anticolisión basado en visión artificial; sin embargo, su

desarrollo también involucra consideraciones éticas relacionadas con la privacidad, el uso responsable de datos visuales y la seguridad de los futuros usuarios del sistema. Aunque el estudio se realiza exclusivamente en un entorno de simulación y no contempla pruebas en vehículos reales ni interacción directa con personas, se reconoce que el empleo de secuencias de video urbano puede involucrar la presencia de datos visuales potencialmente identificables, tales como rostros, placas vehiculares u otros elementos asociados a personas naturales. En tal sentido, cualquier uso de imágenes o videos reales debe enmarcarse en fines estrictamente académicos y técnicos, respetando los principios de protección de datos personales aplicables en el Perú, especialmente en lo relativo a minimización, resguardo y tratamiento adecuado de la información.

Asimismo, desde una perspectiva de seguridad de uso, se reconoce que un sistema de asistencia a la conducción no solo debe evaluarse por su capacidad de detección, sino también por el efecto que podría generar sobre el conductor. En una futura implementación física, alertas falsas, repetitivas o mal temporizadas podrían producir distracción, sobrecarga cognitiva o fatiga operativa. Debido a que la presente investigación no incluye pruebas con usuarios humanos ni validación en condiciones reales de conducción, no es posible determinar experimentalmente el impacto de las alertas sobre el comportamiento del conductor. Por ello, este aspecto se reconoce como una consideración ética y técnica que deberá ser evaluada en etapas posteriores de implementación y validación. Además, la investigación no contempla la recolección de consentimiento individual, dado que no se desarrollan ensayos con participantes humanos ni experimentación directa en campo. No obstante, en una fase futura de despliegue real, cualquier evaluación con conductores, peatones o grabaciones controladas deberá considerar autorización correspondiente, protocolos de resguardo de datos y criterios de seguridad operacional.

1.10. Variables

a) **Variable Independiente:** Algoritmos de visión artificial (YOLOv5 y CenterNet)

Indicadores:

- Precisión en la detección de objetos
- Tiempo de procesamiento de imagen
- Verdaderos positivos (TP)
- Falsos positivos (FP)
- Verdaderos negativos (TN)

- Falsos negativos (FN)
- Sensibilidad (Recall)
- F1-score

b) Variable Dependiente: Sistema Anticolisión

Indicadores:

- Compatibilidad entre YOLOv5 y CenterNet-sim
- Tasa de detección exitosa de obstáculos en escenarios simulados
- Precisión en la detección y localización

1.11. Antecedentes

1.10.1. Antecedentes Internacionales

Chandran et al. (2023) en su artículo *"Review on Type of Sensors and Detection Method of Anti-Collision System of Unmanned Aerial Vehicle"*, tuvieron como objetivo proporcionar una revisión integral sobre los tipos de sensores y métodos de detección utilizados en los sistemas anticolisión de vehículos aéreos no tripulados (UAV). Se analizó su implementación para mejorar la seguridad en tareas manuales. La metodología incluyó la revisión de literatura para identificar y comparar enfoques de detección como geométrico, fuerza de campo, optimización y "sentir y evitar". Los resultados destacaron que los métodos geométricos son ideales para UAV de largo alcance y altitudes altas, mientras que los métodos de optimización y "detectar y evitar" son más adecuados para altitudes bajas. Además, la integración de múltiples sensores mostró ser clave para mejorar la precisión y minimizar los puntos ciegos en diferentes entornos. El estudio subraya la importancia de seleccionar el sistema según la misión del UAV y las condiciones operativas.

Lacatan et al. (2021) en su artículo *"Brake-Vision: A Machine Vision-Based Inference Approach of Vehicle Braking Detection for Collision Warning Oriented System"*, tuvieron como objetivo desarrollar un sistema basado en visión artificial para detectar luces de freno y prevenir colisiones traseras en carreteras. La metodología utilizó el algoritmo YOLOv3 para entrenar y validar un conjunto de datos personalizados, anotados mediante las herramientas Pascal VOC y LabelImg. Se realizaron 50 experimentos para optimizar el

modelo, alcanzando una precisión promedio máxima (mAP) de 89.26% en el modelo final. Los resultados de las pruebas indicaron que el sistema logró una precisión de detección que osciló entre el 40,05% y el 84,74%. Este enfoque demuestra ser eficaz para detectar luces de freno tanto de día como de noche, incluso en vehículos que no están en el mismo carril. Este sistema ofrece potencial para reducir accidentes traseros y mejorar la seguridad vial.

Argese (2023) en su tesis de maestría *"Sensors Selection for Obstacle Detection: Sensor Fusion and YOLOv4 for an Autonomous Surface Vehicle in Venice Lagoon – Webthesis"*, tuvo como objetivo desarrollar un prototipo de vehículo autónomo de superficie (ASV) capaz de navegar de forma segura por los canales de Venecia utilizando tecnología de visión por computadora y fusión de sensores. La metodología se centra en la selección de sensores como LiDAR y cámaras estereoscópicas, integrados mediante algoritmos de fusión basados en aprendizaje profundo. Se implementó el algoritmo YOLOv4 para la detección de objetos, entrenado con un conjunto de imágenes recopiladas de los canales venecianos. Los resultados mostraron que el sistema es capaz de identificar diferentes tipos de embarcaciones, clasificándolas según parámetros como coordenadas y niveles de confianza, fundamentales para evitar colisiones. Este enfoque subraya el potencial de los vehículos autónomos para mejorar la seguridad y la eficiencia en entornos marítimos complejos.

1.10.2. Antecedentes Nacionales

Zapata y Prado (2024) en su tesis *"Diseño e Implementación de un Sistema Tracking y Anticolisión para prevenir accidentes, durante el transporte, voladuras y emergencias, en una mina subterránea en el Perú"*, tuvieron como objetivo diseñar e implementar un sistema de rastreo y anticolisión para prevenir accidentes durante el transporte, voladuras y emergencias en una mina subterránea en el Perú. La metodología incluyó la implementación de tecnologías RFID y radar, utilizando un sistema de comunicación Leaky Feeder como backbone para transmitir datos en tiempo real. Este sistema permitió identificar la ubicación de personas y vehículos dentro de la mina y generar alertas tempranas para los conductores, previniendo colisiones y atropellos. Los resultados demostraron una mejora significativa en la seguridad minera, logrando reducir los riesgos asociados al transporte y operación en túneles subterráneos. Además, el sistema incluyó un software de monitoreo que registraba eventos críticos, facilitando el análisis y la toma de decisiones para la gestión de seguridad minera.

Castillo y Santamaría (2023) en su tesis *"Control de acceso vehicular con visión artificial para urbanizaciones en la ciudad de Piura"*, tuvieron como objetivo mejorar el control de acceso vehicular en urbanizaciones mediante un sistema basado en visión artificial. La metodología incluyó un diseño preexperimental, utilizando cuestionarios y fichas de registro para evaluar indicadores como tiempo de identificación, efectividad en el reconocimiento de matrículas y satisfacción de los usuarios. La implementación se realizó a través de un sistema web desarrollado en Python y React, empleando visión artificial para procesar imágenes. Los resultados mostraron una reducción significativa en los tiempos de registro, pasando de 90.68 a 15.43 segundos en el ingreso y de 137.37 a 16.19 segundos en la salida. Además, se alcanzó un 100% de efectividad en el reconocimiento de matrículas y notificaciones, junto con un aumento en la satisfacción de los usuarios del 32,5%. Estos hallazgos destacan la eficacia del sistema para mejorar la seguridad vehicular.

Acuña y Tito (2023) en su tesis *"Diseño de un vehículo aéreo no tripulado con visión artificial para detección de plagas y malformaciones de cultivos en la localidad de Caqui, Huaral"*, tuvieron como objetivo diseñar un vehículo aéreo no tripulado (UAV) equipado con visión artificial para la detección de plagas y malformaciones en cultivos en Caqui, Huaral. La metodología incluyó el diseño mecánico en SolidWorks, utilizando materiales resistentes y ligeros, y un sistema de visión artificial basado en YOLOv5, entrenado con imágenes capturadas en campo mediante una cámara Dragon Touch Vision 3. Para la navegación, se empleó un controlador Ardupilot y software Mission Planner, garantizando vuelos autónomos eficientes. Los resultados demostraron una alta precisión en la identificación de plagas como la mosca americana y deformidades, con un consumo energético validado por el software XcopterCalc. Las pruebas de simulación en entornos 3D confirmaron la funcionalidad integral del sistema, logrando vuelos autónomos sin contratiempos y una autonomía promedio de 80.6 minutos por carga de batería.

CAPÍTULO II
MARCO TEÓRICO

2.1. Bases Teóricas

2.1.1. Sistema anticolisión

Un sistema anticolisión es una solución tecnológica avanzada diseñada para prevenir o minimizar el riesgo de colisiones entre objetos en movimiento, como vehículos, drones, robots industriales, o entre estos y estructuras fijas. Estos sistemas combinan hardware y software para monitorear constantemente el entorno y analizar la proximidad y trayectoria de los objetos circundantes, permitiendo prevenir y mitigar posibles colisiones. Sus componentes principales incluyen sensores, como cámaras, radares, LIDAR o dispositivos ultrasónicos, que capturan información detallada sobre la posición y el movimiento de los objetos cercanos. Los datos recopilados por los sensores son procesados mediante algoritmos avanzados, los cuales analizan parámetros clave como velocidad, trayectoria y distancia, identificando riesgos potenciales. Ante una amenaza de colisión, el sistema puede alertar al usuario o actuar de manera autónoma, aplicando medidas preventivas como frenar, cambiar de trayectoria o emitir señales de advertencia.

Estos sistemas son fundamentales en diversos contextos, desde mejorar la seguridad vial al prevenir accidentes, hasta garantizar la operación segura en entornos industriales con maquinaria y robots. Su efectividad radica en la capacidad de procesar rápidamente grandes volúmenes de datos provenientes de los sensores, permitiendo una respuesta inmediata ante situaciones de riesgo. Esta tecnología no solo incrementa significativamente la seguridad, sino que también mejora la eficiencia y la productividad en una amplia variedad de sectores (Vásquez, 2021).

ISO 21815-2:2021 establece un protocolo de comunicación entre una máquina y un dispositivo vinculado, con el fin de que dicho dispositivo pueda indicar a la máquina que reduzca la velocidad, se detenga o permanezca en un estado estable. En este estado, la máquina tiene la posibilidad de desplazarse en línea recta, es decir, hacia adelante o hacia atrás dentro de su trayectoria de movimiento. Los equipos que realizan movimientos rotacionales (como las excavadoras) y aquellos con desplazamientos combinados (como las

máquinas provistas de brazos mecánicos) son contemplados únicamente en lo referente a la parte lineal de su operación. En conclusión, esta norma también abarca los procedimientos de detección, aviso e intervención en las máquinas, con el propósito de disminuir los riesgos de choques y accidentes por atropello. El sistema de anticolisión diseñado para apoyar a personas con discapacidad visual utiliza redes neuronales artificiales como herramienta principal de reconocimiento y prevención de obstáculos en su entorno inmediato. La propuesta busca suplir las limitaciones que presentan los bastones tradicionales o incluso los perros guía, ofreciendo una alternativa tecnológica más precisa y adaptable a diversos contextos (Coral y España, 2018).

La norma ISO/TS 21815-2:2021 constituye una referencia técnica útil en materia de advertencia y evitación de colisiones; sin embargo, su ámbito de aplicación se orienta principalmente a maquinaria de movimiento de tierras, minería y construcción vial, mediante una interfaz de comunicación embarcada basada en SAE J1939. En consecuencia, su uso en la presente investigación se asume como referencia complementaria y no como marco normativo exclusivo para vehículos urbanos. Dado que el sistema propuesto se orienta a escenarios viales de la ciudad de Arequipa, resulta pertinente considerar además lineamientos vinculados a vehículos de carretera y sistemas de asistencia a la conducción. En ese sentido, IEEE P2846 aporta un marco de referencia relacionado con los supuestos razonablemente previsibles sobre el comportamiento de otros usuarios de la vía dentro de modelos de seguridad para sistemas automatizados. Asimismo, a nivel nacional, el Reglamento Nacional de Tránsito, aprobado mediante Decreto Supremo N.° 016-2009-MTC, constituye el marco legal aplicable al tránsito terrestre en el Perú, por lo que su consideración permite contextualizar la propuesta dentro de las condiciones normativas del entorno vial peruano (ISO, 2021).

El diseño contempla distintos escenarios de uso, incluyendo espacios urbanos con tránsito vehicular y zonas peatonales congestionadas. Gracias a su capacidad de aprendizaje, el sistema puede mejorar su rendimiento con la experiencia, reconociendo patrones frecuentes de obstáculos y ajustando la sensibilidad de detección. Esto representa una ventaja significativa frente a tecnologías convencionales, ya que permite un nivel de autonomía y seguridad más alto. Además, el sistema no solo se limita a la detección de objetos estáticos, sino que también puede identificar elementos en movimiento, lo cual resulta esencial en la prevención de atropellos o choques inesperados. De este modo, se convierte en un apoyo integral para que las personas con discapacidad visual puedan desenvolverse con mayor

confianza en entornos complejos. En conclusión, este sistema de anticolisión basado en redes neuronales combina innovación tecnológica con accesibilidad social, brindando una solución práctica y eficiente que contribuye a la inclusión y movilidad segura de los invidentes (Coral y España, 2018).

2.1.2. Comprensión y análisis del entorno

La comprensión y análisis del entorno se realiza en diferentes niveles de complejidad. En un primer nivel, se examinan atributos básicos como la forma, tamaño y variaciones en la profundidad de los objetos presentes. Este análisis inicial permite identificar elementos fundamentales en el entorno. En un nivel intermedio, se asegura la coherencia y continuidad de los datos a lo largo del tiempo, utilizando filtros que permiten monitorear y evaluar parámetros dinámicos, como velocidad y trayectoria, para estimar con precisión la ubicación y el movimiento de los objetos relevantes.

En el nivel más avanzado, se integran características espaciales y temporales para desarrollar modelos más complejos, capaces de clasificar, anticipar y predecir el comportamiento de los obstáculos. Este enfoque permite una mayor comprensión del entorno y la toma de decisiones más informadas. Las técnicas utilizadas para la detección de obstáculos en entornos viales son diversos y abarcan desde tecnologías mecánicas tradicionales hasta sistemas ópticos avanzados y de estado sólido. Estas metodologías varían en complejidad computacional y costo, lo que permite adaptarse a diferentes aplicaciones y necesidades específicas (Warren, 2019). A continuación, se describen las técnicas más comunes aplicadas en la comprensión y análisis del entorno, destacando su funcionalidad y ventajas según el contexto de uso.

A. Cámaras de Video: Reconstrucción de Escenas

La reconstrucción de escenas utilizando cámaras de video se basa en capturar múltiples perspectivas (POV) de una misma área para generar representaciones tridimensionales (3D) de los objetos presentes. Cada perspectiva aporta información única que, al combinarse, permite deducir la estructura tridimensional de la escena. Esto puede lograrse con sistemas multicámara fijos o con una sola cámara en movimiento. Sin embargo, el éxito de esta técnica depende de una calibración precisa de las cámaras y del conocimiento exacto de su posición y orientación relativa. Los métodos más comunes incluyen la

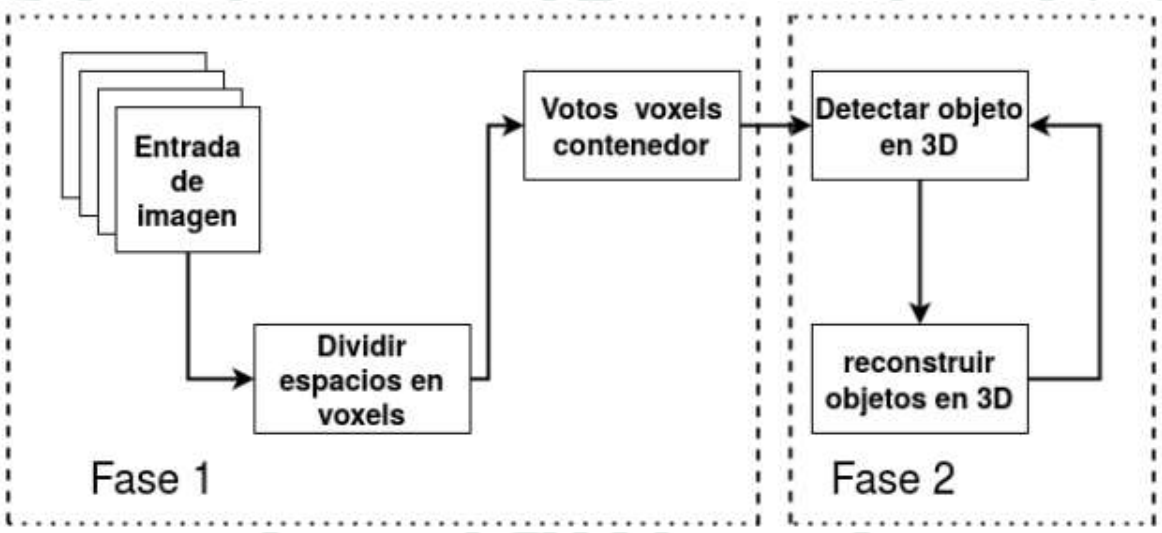
reconstrucción mediante vóxeles y el mapeo de disparidades, ambos con enfoques y aplicaciones específicas (Madrid, 2018).

a) Reconstrucción mediante Vóxeles

El método de reconstrucción mediante vóxeles consta de dos etapas principales. Primero, se segmenta el espacio en vóxeles y se trazan rayos desde cada cámara hacia ellos. La interacción de estos rayos con los planos de imagen genera votos para cada vóxel, indicando la posible presencia de un objeto. Aunque algunos votos pueden ser erróneos, una alta convergencia de votos en un mismo vóxel sugiere la existencia de un objeto real. En la segunda etapa, se analizan los votos recibidos, minimizando las discrepancias entre ellos para reconstruir la forma 3D del objeto. Este proceso incluye postprocesamiento para refinar la precisión, especialmente en áreas donde los objetos están parcialmente ocultos. Ajustes iterativos pueden mejorar aún más la fidelidad de la reconstrucción (Sadjadi y Ribnick, 2010).

Figura 1

Diagrama de flujo del algoritmo basado en vóxeles



Nota. Fases del algorimto de voxels. Fuente (Sadjadi y Ribnick, 2010).

b) Mapa de disparidades

El mapeo de disparidades utiliza dos cámaras para estimar diferencias de profundidad en la escena. Los métodos se dividen en tres categorías: locales, globales y semiglobales. Los algoritmos locales comparan segmentos específicos, eligiendo aquellos con el menor

costo o mayor clasificación. En contraste, los algoritmos globales emplean técnicas avanzadas, como la propagación de creencias (Belief Propagation, BP), basada en campos aleatorios de Markov (Markov Random Fields, MRF).

Estos algoritmos plantean la estimación de disparidades como un problema de maximización de probabilidad, resolviéndolo mediante métodos de minimización de energía para lograr correspondencias estereoscópicas precisas (Sadjadi y Ribnick, 2010).

2.1.3. Visión artificial

La visión artificial, también denominada visión computarizada, es una rama de la inteligencia artificial que busca dotar a los sistemas informáticos de la capacidad para interpretar y comprender imágenes de manera similar al proceso visual humano (Muñoz, 2006). Este campo combina conocimientos avanzados en procesamiento de imágenes, aprendizaje automático y análisis de datos para emular las habilidades visuales del ser humano.

La *Visión Artificial*, también denominada *Visión por Computadora*, se concibe como un campo de la *Inteligencia Artificial* que emplea técnicas específicas para facilitar la obtención, el procesamiento y la interpretación de información de carácter espacial contenida en imágenes digitales. Este ámbito integra métodos computacionales que hacen posible extraer, manipular y evaluar datos visuales, con el propósito de comprender su estructura y significado. De esta manera, la visión artificial se convierte en una herramienta que transforma imágenes en conocimiento útil, al permitir el análisis automático de su contenido (Domingo et al., 2024, p. 13).

La visión artificial se compone de una serie de métodos dirigidos al examen de imágenes. Estos métodos abarcan la captura de las mismas, el resguardo de la información obtenida, el tratamiento de los datos y la interpretación de los resultados producidos. Este ámbito integra técnicas que hacen posible registrar, almacenar y analizar de manera visual la información con el propósito de generar conclusiones aplicables en diversos usos. En este marco, cada fase del proceso resulta relevante para convertir la información visual en datos comprensibles, lo que facilita la toma de decisiones automatizadas en sistemas que requieren identificar y evaluar escenarios visuales (Horna y Manzano, 2021, p. 51).

La visión artificial posibilita:

- Automatizar tareas repetitivas de revisión que anteriormente realizaban los trabajadores.
- Realizar verificaciones de calidad de productos que no podían evaluarse con métodos tradicionales.
- Revisar objetos sin necesidad de contacto directo.
- Examinar la totalidad de la producción (asegurando calidad completa) a gran velocidad.
- Reducir el tiempo de ciclo en procedimientos automatizados. Llevar a cabo inspecciones en líneas de producción con diversidad de piezas y variaciones frecuentes en el tipo de artículo fabricado (Chávez, 2022).

Existen múltiples técnicas y procesos aplicados en el procesamiento de imágenes, los cuales representan la fase preliminar dentro de un sistema de visión artificial. Esta etapa es fundamental porque permite transformar la información visual en datos útiles que posteriormente serán analizados e interpretados por algoritmos especializados. Dichos procedimientos abarcan operaciones de mejora, filtrado, segmentación y detección de características, que facilitan la extracción de información relevante de los objetos observados. En este sentido, el tratamiento de imágenes se convierte en el punto de partida para garantizar la precisión y eficacia de las aplicaciones desarrolladas en visión artificial (Chávez, 2022).

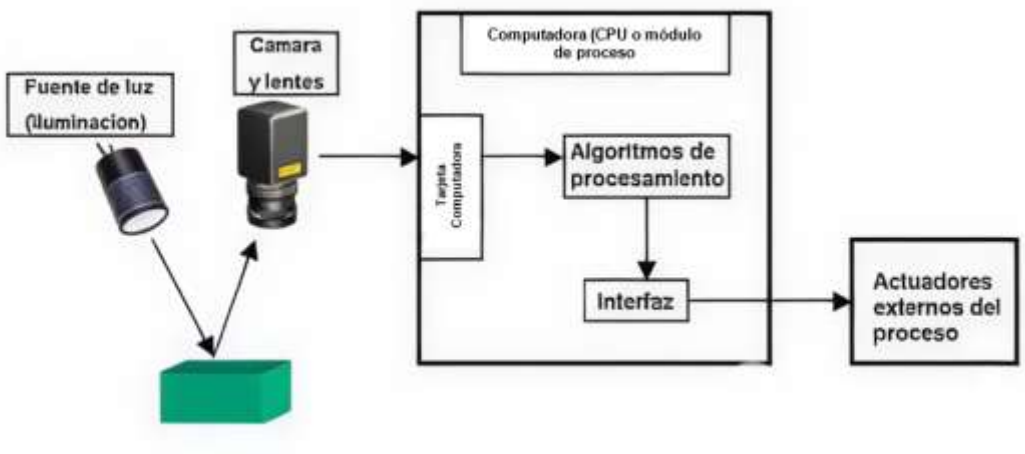
La visión artificial constituye una rama de la inteligencia artificial que utiliza técnicas y herramientas orientadas a la adquisición, tratamiento y evaluación de datos provenientes de imágenes digitales. Este ámbito desempeña una función semejante a la de la visión humana, con la diferencia de que, en lugar de retinas y nervios ópticos, recurre a cámaras, información y algoritmos que permiten a una máquina interpretar el entorno visual. Para lograr este propósito, se emplean plataformas de desarrollo y lenguajes como LabVIEW, MATLAB y Python, además de métodos de aprendizaje automático, los cuales procesan información mediante algoritmos capaces de reconocer y diferenciar diversas imágenes (Chávez, 2022).

De igual manera, para que un brazo robótico logre interactuar de manera eficiente con su entorno, resulta fundamental integrar sistemas avanzados de inteligencia artificial, siendo la visión por computadora uno de los más destacados. Esta herramienta posibilita que el robot identifique objetos, ubicaciones y patrones en tiempo real, optimizando su desempeño y ampliando sus usos en ámbitos industriales, médicos y de investigación. Por lo tanto, la visión artificial no solo incrementa la autonomía de los dispositivos, sino que también mejora la exactitud y eficacia en actividades que demandan percepción visual automatizada (Espinoza y Vera, 2025).

La visión artificial constituye una especialidad de la inteligencia artificial orientada a reproducir la habilidad del sistema visual humano para interpretar el entorno a partir de representaciones gráficas. En la figura 2 se muestra un esquema general que ilustra las etapas de adquisición, procesamiento y evaluación de imágenes con el objetivo de obtener información significativa. Su finalidad esencial es analizar y reconstruir las propiedades del mundo físico, como la geometría, la iluminación y los tonos de color, a partir de capturas visuales. Esta tecnología tiene aplicaciones en diversos campos, tales como el reconocimiento óptico de caracteres (OCR), el control de calidad en procesos de manufactura, la clasificación de artículos en entornos comerciales, la gestión logística en depósitos y la obtención junto con el estudio de imágenes en el ámbito médico (Fernández, 2025).

Figura 2

Diagrama Global del Proceso de Visión Artificial

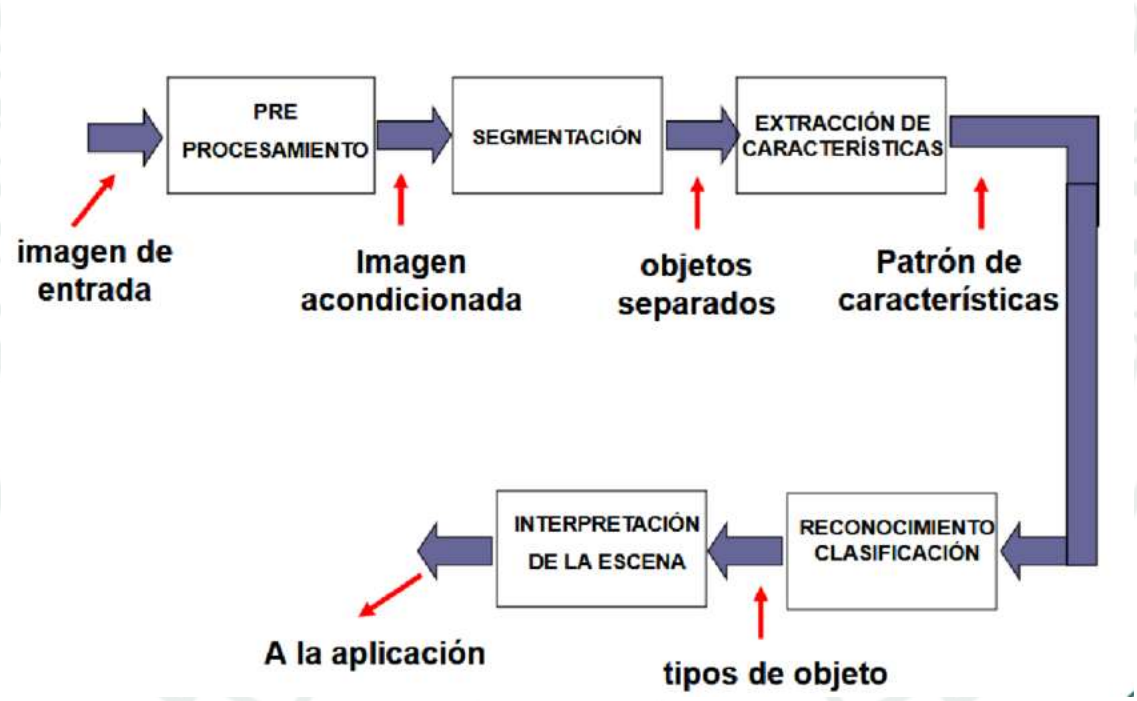


Nota. Tomado de (Palomares, 2024).

Aunque las etapas específicas de un sistema de visión artificial pueden variar dependiendo de su aplicación, existen ciertos pasos fundamentales que son comunes en la mayoría de las implementaciones. Estas etapas incluyen la adquisición de imágenes, su preprocesamiento para mejorar la calidad, la extracción de características relevantes y la interpretación de la información visual para generar una respuesta adecuada al contexto. Estas fases trabajan en conjunto para permitir que las máquinas analicen y tomen decisiones basadas en datos visuales, replicando de manera eficiente el complejo proceso de percepción humana. Una ilustración de estas etapas se presenta en la figura siguiente, destacando la secuencia lógica que guía a los sistemas desde la captura de datos hasta la toma de decisiones visuales.

Figura 3

Diagrama de bloques de un sistema de visión artificial



Nota. Tomado de (Ulcuango, 2022).

Se propone que la información capturada mediante una cámara pueda transformarse en una representación visual o en una determinación, con el objetivo de cumplir un fin específico. Los datos contextuales iniciales pueden originarse de la localización donde está instalada la cámara o de la detección de la posición de un objeto. Un ejemplo de determinación sería contabilizar personas, objetos o animales en un espacio determinado. En

cuanto a una representación, esta podría consistir en convertir una imagen a tonos de gris. (Kaehler y Bradski, 2016)

Dentro del campo de la visión artificial, se puede construir un modelo en tres dimensiones a partir de fotografías que se encuentran solapadas. Para alcanzarlo, se aplican técnicas de desarrollo fundamentadas en disciplinas como la matemática. Dichos métodos se utilizan en ramas de la física como la radiometría, la óptica y el diseño de sensores de captura, además del ámbito de los gráficos por computadora. En ambos escenarios se consideran elementos como el movimiento y la animación de objetos, el comportamiento de la luz al reflejarse en sus superficies, su dispersión en la atmósfera, la refracción que ocurre en la lente de la cámara y la proyección final de la imagen en un plano (Fernández, 2025).

2.1.3.1.Fundamentos de la visión artificial

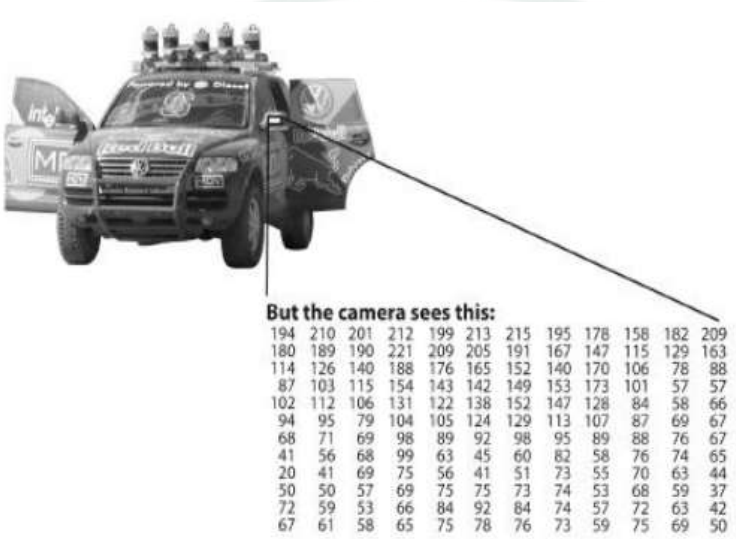
La visión artificial constituye un campo extenso y complejo que abarca la captura, el análisis y la interpretación del espacio tridimensional a partir de fotografías y secuencias de video. Esta disciplina se apoya en fundamentos de ingeniería, ciencias, estadística y técnicas de aprendizaje automático. El procedimiento de generación de imágenes se clasifica en componentes geométricos, fotométricos y asociados a los sensores, cada uno con un rol esencial para entender cómo se producen y procesan las imágenes. El aspecto geométrico en la creación de imágenes se enfoca en la proyección de puntos, líneas y planos dentro de las imágenes, empleando modelos de geometría proyectiva y de distorsión óptica de las lentes (Panduro y Rojas, 2024). La fotometría analiza cómo la luz se relaciona con las superficies y se refleja en el sensor, mientras que los dispositivos de captura consideran elementos como el muestreo, el aliasing y la compresión en el interior de la cámara. Estos conceptos son esenciales para elaborar algoritmos robustos y eficientes, que puedan enfrentar las incertidumbres propias del proceso de formación de imágenes y ofrecer interpretaciones exactas de las escenas registradas (Szeliski, 2022).

Los elementos relevantes en la visión artificial comprenden la calibración de los dispositivos de captura, la separación de imágenes y la regulación de la iluminación. La calibración de cámaras es esencial para corregir posibles deformaciones y obtener mediciones precisas. La segmentación de imágenes permite identificar y analizar objetos presentes en una escena, siendo fundamental en labores como la verificación de calidad. Del

mismo modo, una iluminación adecuada resulta decisiva para reducir sombras y reflejos que puedan influir en la exactitud del análisis visual (Pereyra, 2024).

Figura 4

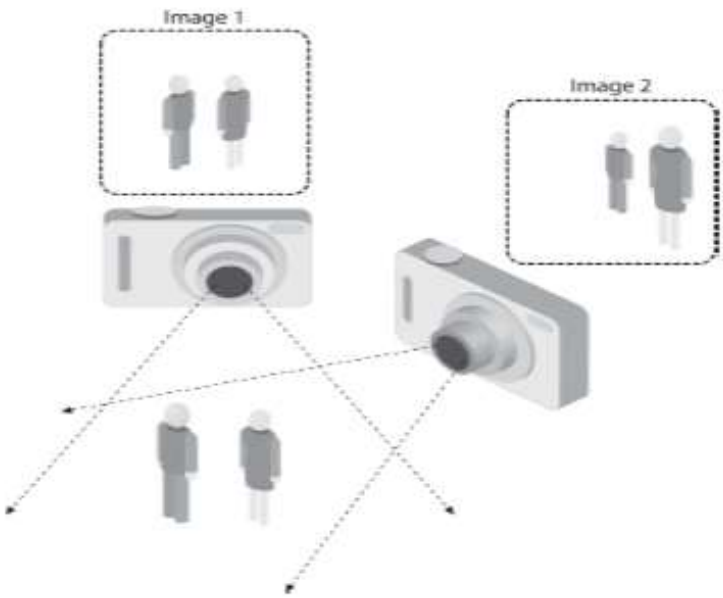
Expresión digital de una imagen



Nota. Tomado de (Kaehler y Bradski, 2016).

Figura 5

Obtención de imágenes desde múltiples puntos de vista



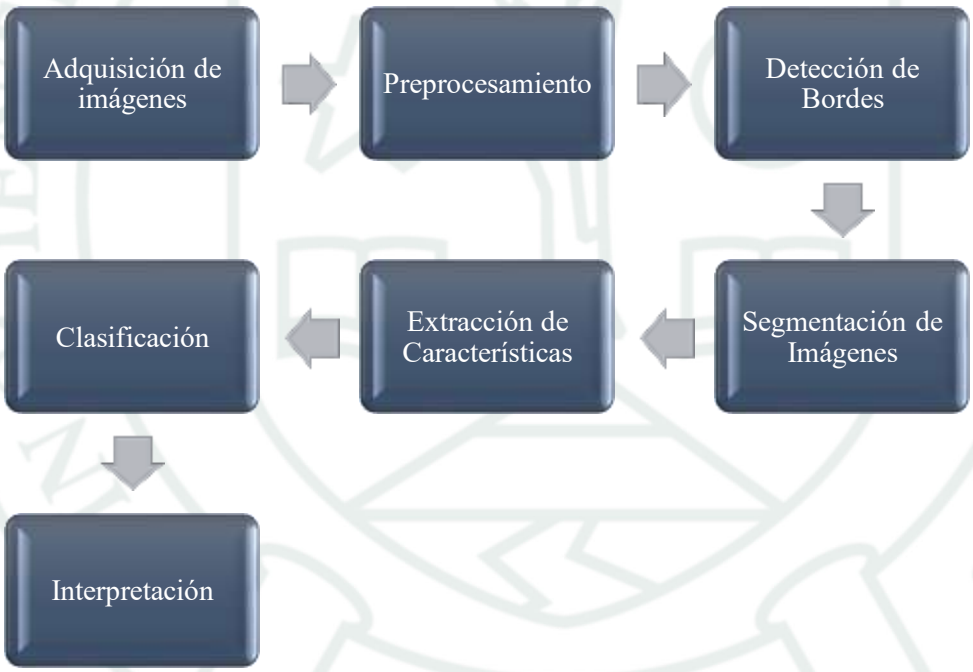
Nota. Tomado de (Kaehler y Bradski, 2016).

2.1.3.2.Etapas de la Visión Artificial

La visión artificial comprende diferentes etapas fundamentales en el procesamiento de imágenes, cada una con un papel importante en la obtención de información visual significativa. Estas fases, abarcan procedimientos que van desde la captura inicial de la imagen hasta su examen e interpretación. En cada nivel se emplean técnicas específicas destinadas a mejorar la calidad, extraer atributos relevantes y favorecer la posterior toma de decisiones automáticas. Este método organizado permite que los sistemas de visión por computadora conviertan datos visuales en información útil, incrementando su exactitud y eficiencia en diversas aplicaciones tecnológicas y productivas (Kaehler y Bradski, 2016).

Figura 6

Etapas de la Visión Artificial



Nota. Tomado de (Panduro y Rojas, 2024).

La Figura 6 expone de manera ordenada las fases de un sistema de visión por computadora, que abarcan desde la adquisición de la imagen hasta su interpretación final. El procedimiento comienza con la captura de imágenes, realizadas a través de cámaras o dispositivos ópticos. A continuación, en la etapa de preprocesamiento, se emplean métodos destinados a optimizar la calidad y suprimir imperfecciones como ruido o variaciones

lumínicas. Después, la detección de bordes posibilita reconocer contornos y delimitaciones de los objetos, lo que facilita la segmentación de la imagen en áreas con propiedades semejantes. En la fase de extracción de características se obtienen atributos significativos de cada zona u objeto, tales como forma, color o textura, los cuales se utilizan en la clasificación, donde se asigna una etiqueta o categoría a cada componente identificado mediante algoritmos o modelos de inteligencia artificial. Finalmente, la interpretación confiere sentido a la información obtenida, lo que permite tomar decisiones o ejecutar acciones fundamentadas en el análisis visual.

2.1.3.3.Metodología de Desarrollo

Para llevar a cabo la implementación de la visión por computadora, se dispone de una metodología propuesta por Bustamante y Salazar (2024), la cual establece de manera estructurada las etapas que conforman un sistema de visión artificial. Dicho enfoque metodológico busca garantizar que el proceso de interpretación de imágenes se realice de manera ordenada y eficaz, permitiendo resultados más confiables. Las fases contempladas son tres: en primer lugar, la representación de la imagen, que consiste en transformar la información visual capturada en un formato numérico o digital comprensible para el sistema. En segundo término, se encuentra el tratamiento o procesamiento de la imagen, orientado a mejorar su calidad, eliminar imperfecciones y resaltar características esenciales para el estudio. Finalmente, se desarrolla el análisis o examen de la imagen, etapa en la que se interpretan los datos procesados con el objetivo de extraer información relevante para tareas como clasificación, reconocimiento o control automatizado (p. 27).

A. Representación de la imagen

La etapa de representación de la imagen en un sistema de visión artificial constituye un procedimiento crucial para el procesamiento y estudio de la información visual. En esta fase se definen las características y el formato en que se almacenarán los datos obtenidos, ya sea en escala de grises o en color, de acuerdo con los requerimientos del sistema y la finalidad de la aplicación. Asimismo, se consideran elementos relacionados con la captura, la digitalización y los fundamentos del color, junto con los diversos modelos de codificación cromática utilizados para describir y manipular los atributos visuales. Estos elementos son indispensables para alcanzar una interpretación precisa y eficiente de las imágenes.

a. Captura y Digitalización

Abarca todos los procesos utilizados para obtener y transformar imágenes en formato digital. En términos generales, existen cuatro formas principales de adquisición según el medio empleado:

- Escáner
- Programas para capturas de pantalla
- Cámara fotográfica
- Placas digitalizadoras de video
- Tras obtener la imagen, un software se encarga de su digitalización, es decir, almacenarla en un formato digital.

b. Fundamentos del color

En la etapa de digitalización, la utilización del color en el tratamiento de imágenes obedece a dos motivos principales: en primer lugar, dentro del análisis visual, el color actúa como un descriptor sólido que simplifica la identificación y separación de objetos; en segundo lugar, la percepción humana distingue una mayor cantidad de matices cromáticos que escalas de gris, lo que le otorga una ventaja significativa.

c. Modelos de color

Es fundamental establecer el modelo de color a utilizar, es decir, la manera de representar los tonos.

- **Modelo RGB:** Define el color mediante sus componentes primarios rojo, verde y azul, tomando como referencias coordenadas cartesianas.
- **Modelo CMY:** Se obtiene de forma directa o a través de la conversión desde RGB, expresando la luz absorbida y reflejada.
- **Modelo YIQ:** Creado para aprovechar la sensibilidad del ojo humano frente a variaciones de luminancia más que a cambios de tono o saturación.
- **Modelo HSI:** El matiz refleja la pureza del color y la saturación mide su mezcla con luz blanca. Su importancia radica en que distingue la intensidad de la información cromática y sus componentes se asocian con la percepción visual, lo que lo hace apropiado para el procesamiento de imágenes (García, 2012).

d. Pseudocolor

En las representaciones digitales en escala de grises, cada tono se determina a partir de la magnitud y la fuerza de la luminosidad registrada en la imagen. A través de este procedimiento, se asignan valores cromáticos correspondientes a diferentes niveles de intensidad, lo que permite transformar una simple variación de claros y oscuros en una representación mucho más comprensible y visualmente expresiva. Esta técnica facilita la interpretación, ya que convierte datos numéricos en formas gráficas perceptibles, incrementando la claridad y el detalle en el análisis de patrones, contornos y texturas que de otro modo resultarían menos evidentes para el observador.

e. Rodajas de intensidad

Se basa en situar planos paralelos al de coordenadas de la imagen, asignando colores a los distintos niveles de intensidad con el fin de producir una imagen multicolor (Chávez, 2022, p. 23).

En la etapa de representación de la imagen se reconocen dos modalidades principales de caracterización:

- **Imágenes en escala de grises:** Cada píxel o punto corresponde a un valor que refleja la intensidad de luz en esa posición específica. Esta intensidad se expresa como un valor real entre 0 y 1, donde 0 simboliza el negro total y 1 el blanco completo, con variaciones intermedias que conforman los diferentes tonos de gris.

- **Imágenes a color:** Cada píxel conserva la información del color que lo define, el cual se determina a través de un sistema de codificación cromática. Entre los métodos más habituales se encuentra el modelo RGB, que combina intensidades de rojo, verde y azul, así como el sistema hexadecimal, muy usado en entornos digitales. Estas representaciones permiten describir la información visual con exactitud, adaptándose al tipo de procesamiento y análisis requerido en sistemas de visión por computadora (Chávez, 2022, p. 25).

Se pueden definir diversas fases dentro de las etapas que integran el desarrollo de este proyecto. En lo referente a la representación de la imagen, se contemplan las siguientes:

- **Captura y digitalización:** Consiste en obtener la imagen mediante equipos de adquisición y convertirla a un formato digital apropiado para su procesamiento.

- **Fundamentos del color:** Incluye el análisis y la aplicación de conceptos vinculados a la percepción y codificación del color, utilizando modelos y esquemas de representación cromática que faciliten su estudio y manipulación en entornos de visión por computadora (Chávez, 2022, p. 26).

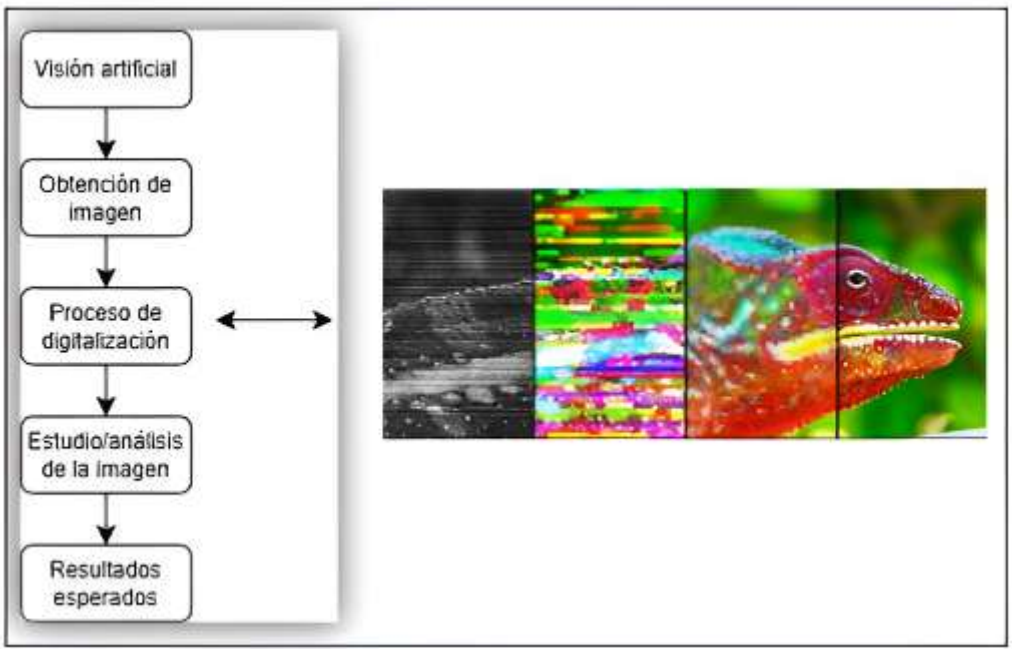
B. Procesamiento de la Imagen

En la etapa de procesamiento se incluyen las siguientes actividades:

- **Pretratamiento de imágenes digitales:** Consiste en realizar operaciones iniciales orientadas a optimizar o acondicionar la imagen para análisis posteriores, tales como ajustes de luminosidad, contraste o reducción de ruido. El procesamiento y estudio de imágenes digitales no resulta inmediato ni evidente.

Figura 7

Fases del procesamiento de imágenes en visión artificial



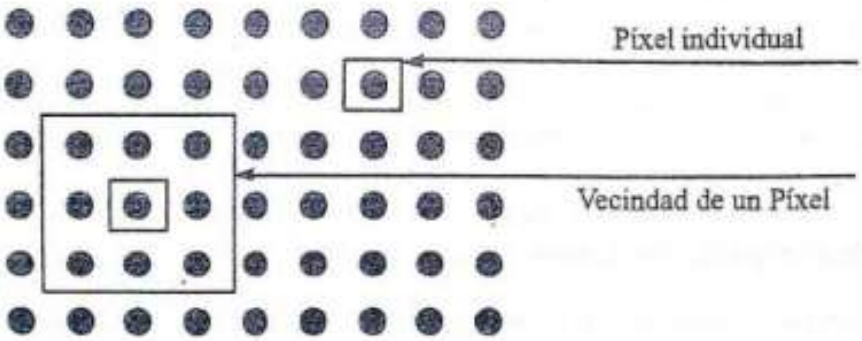
Nota. Tomado de (Espinoza y Vera, 2025).

Este procedimiento puede entenderse como la transformación de una imagen en otra; no obstante, desde la perspectiva de los sistemas de visión artificial, su propósito

esencial es posibilitar la extracción de información útil para un análisis posterior, de manera que la escena pueda interpretarse de forma adecuada. El tratamiento de datos en un sistema de visión puede abordarse desde dos aproximaciones (Silva, 2024). De igual manera, la generación de un nuevo píxel dentro de una imagen puede estar condicionada por el valor particular de dicho píxel en su posición o por los valores de los píxeles circundantes (Chávez, 2022, p. 26).

Figura 8

Métodos aplicados a píxeles y regiones vecinas



Nota. Tomado de (Pajares, 2002).

- **Operaciones individuales**

Las operaciones individuales se basan en generar una nueva imagen modificando el valor de un píxel en una posición determinada, siguiendo una regla general aplicada a cada punto de la imagen original. El procedimiento consiste en tomar el valor del píxel en una localización específica, transformarlo mediante una operación lineal o no lineal y asignar el resultado al lugar correspondiente en la imagen obtenida. Este proceso se repite de forma sistemática para todas las posiciones de los píxeles de la imagen inicial (García, 2012).

- **Operaciones de vecindad**

Las operaciones fundamentadas en la vecindad siguen un proceso semejante, aunque el nuevo valor asignado a un píxel en la imagen resultante depende de la combinación de los valores de los píxeles próximos al píxel original que está siendo

procesado. En términos simples, consiste en alterar el valor de un píxel *p* situado en la posición (x, y), tomando en cuenta los valores de sus píxeles vecinos. Por ejemplo, si se considera una vecindad E8(p), se efectúa una suma ponderada de los valores de sus ocho adyacentes, y el resultado se convierte en el nuevo valor del píxel *q* en la imagen de salida en la misma posición (x, y). Únicamente queda establecer los coeficientes de ponderación, lo cual normalmente se define mediante una máscara con valores determinados. Esta máscara funciona como un filtro, y el efecto final dependerá de sus propiedades particulares (García, 2012).

- **Transformaciones Lógicas**

Consiste en efectuar operaciones lógicas como and, or, xor, entre otras, sobre una imagen transformada en un mapa de bits binario. Esto equivale a aplicar dichas operaciones sobre valores lógicos 0 o 1 (análogos a verdadero o falso), combinándolos con un mapa de bits en diferentes niveles de grises. De esta forma, se obtienen nuevas imágenes con elementos más destacados y una mayor claridad visual (Horna y Manzano, 2021, p. 26).

- **Transformaciones Geométricas**

Se utiliza principalmente para analizar un área específica denominada región de interés. Para alcanzar este propósito, es necesario aplicar operaciones dimensionales en cada píxel de la imagen, lo que conlleva el uso de técnicas como la interpolación, el desplazamiento, la modificación de escala mediante redimensionamiento, la ampliación a través de zoom, así como giros o rotaciones, entre otros procedimientos (Chávez, 2022).

- **Realce de bordes, esquinas y zonas relevantes**

Se centra en destacar las líneas, bordes, esquinas y puntos relevantes presentes en la imagen, con la finalidad de facilitar su detección y estudio. Dichas operaciones hacen posible que los componentes más relevantes se diferencien con claridad, optimizando la comprensión de la información visual en sistemas de visión por computadora y aumentando la exactitud en los procesos de reconocimiento (Chávez, 2022).

Los bordes o contornos corresponden a píxeles donde, en su entorno, la imagen presenta una variación repentina en los niveles de gris. El objetivo es, a partir de una imagen que puede estar o no alterada por ruido, resaltar los límites más probables generados por los objetos presentes en la escena y no aquellos causados por interferencias o distorsiones (Cubero, 2012).

La noción de borde alude a secuencias conectadas de puntos que conforman el límite o perímetro de una figura, denominadas segmentos de contorno. No obstante, esto no impide que la imagen pueda presentar puntos aislados con un contraste elevado en los niveles de gris. A estos puntos específicos se les conoce como *edgels* (proveniente del término elementos). Los bordes, en este contexto, simbolizan transiciones relevantes en la intensidad de la imagen, mientras que los edgels representan las unidades más pequeñas que permiten reconocer dichos cambios, resultandos indispensables para el análisis y el procesamiento en sistemas de visión artificial (Cubero, 2012).

Para llevar a cabo este procedimiento se cuenta con diversos tipos de operadores, entre los que destacan los de primera derivada, como el gradiente de una imagen, los operadores de Sobel, Prewitt y Roberts, así como las máscaras de Kirsch, Robinson y Frei-Chen, además de extensiones de operadores y el algoritmo de Canny. Por otro lado, se encuentran los operadores de segunda derivada, entre los que se incluyen el Laplaciano, el Laplaciano de la Gaussiana y la técnica de diferencia de Gaussianas. Todos estos métodos se aplican con el propósito de resaltar, identificar o examinar variaciones en la intensidad de las imágenes, facilitando la detección de bordes y detalles importantes en el procesamiento visual (Salazar et al., 2016).

C. Análisis de la Imagen

En la etapa de análisis de la imagen se abordan dos aspectos esenciales. El primero corresponde al estudio del movimiento, el cual consiste en reconocer, rastrear y evaluar los desplazamientos o variaciones de posición de los objetos dentro de una secuencia de imágenes, lo que permite determinar trayectorias, velocidades y comportamientos dinámicos. Este procedimiento implica comparar imágenes consecutivas con el propósito de verificar cambios en la ubicación de un objeto, identificando tanto la dirección de su desplazamiento como su posición exacta. Además, contempla el análisis de la variación del movimiento relativo entre objetos inmóviles entre sí y respecto a las distintas distancias que

los separan del observador, quien también puede encontrarse en movimiento. En esencia, se mide la variación de la posición de un objeto de un punto a otro, empleando para ello métodos como la diferencia de análisis de movimiento, así como técnicas de seguimiento y predicción de características en secuencias visuales (Massiris, 2021). El segundo aspecto corresponde a la identificación de patrones, entendida como la detección y clasificación de formas, estructuras o características repetitivas presentes en la imagen, con el objetivo de relacionarlas con modelos predefinidos o reconocer elementos particulares (Chávez, 2022).

En primer lugar, es necesario comprender qué es un patrón: se trata de una entidad a la cual se le puede asignar un nombre y que se define a partir de un conjunto de propiedades cuantificables y de las relaciones existentes entre ellas, conocidas como vector de características. Un ejemplo de patrón puede ser una señal de audio cuyo vector de características está conformado por coeficientes espectrales obtenidos a través de su análisis mediante un espectrograma. De manera similar, la imagen de un rostro humano constituye otro caso, ya que de ella se extrae un vector de características compuesto por valores numéricos derivados de la misma. El reconocimiento, la descripción, la clasificación y la agrupación de patrones representan actividades de gran importancia en diversas disciplinas científicas como la biología, la psicología, la medicina, la visión por computadora, la inteligencia artificial y la teledetección (Mundaca, 2016).

Un sistema de reconocimiento de patrones puede orientarse a distintos objetivos. Por un lado, identificar si un patrón corresponde a una categoría previamente definida, lo que se denomina clasificación supervisada. Por otro, asignar el patrón a una categoría que no ha sido establecida con anterioridad, proceso conocido como clasificación no supervisada, también referido como agrupamiento o *clustering* (Mundaca, 2016).

En el primer escenario, el sistema opera con un conjunto de categorías previamente definidas, mientras que en el segundo reconoce semejanzas entre los datos para generar nuevos agrupamientos. Ambos métodos posibilitan la organización y el examen de la información de forma eficaz en aplicaciones orientadas al procesamiento y análisis de datos. Estos procedimientos resultan fundamentales para interpretar la información visual y apoyar la toma de decisiones en sistemas de visión artificial y en aplicaciones automatizadas (Chávez, 2022).

2.1.4. Detección de objetos

La detección de objetos es una técnica de visión computacional cuyo objetivo es reconocer y ubicar elementos dentro de imágenes o secuencias de video en diferentes contextos, con el propósito de clasificarlos y examinarlos posteriormente. En el ámbito de la manufactura, esta tecnología cumple un rol relevante en tareas como la administración de inventarios, la verificación de calidad y la supervisión de procesos productivos, entre otras aplicaciones. Este procedimiento puede llevarse a cabo mediante métodos tradicionales de procesamiento de imágenes o a través de redes neuronales basadas en aprendizaje profundo (Calisaya y Trujillo, 2024).

La detección de objetos es una tecnología de visión artificial que permite identificar y localizar entidades visuales en imágenes o vídeos. Utiliza algoritmos avanzados y modelos de aprendizaje profundo para reconocer objetos, asignar etiquetas específicas según sus categorías y determinar su ubicación con alta precisión. Esta tecnología puede clasificarse en las siguientes áreas principales:

a) **Visión orientada a la identificación de objetos:** Este enfoque abarca dos variantes clave:

- **Detección de objetos específicos:** Identifica y localiza un objeto en particular dentro de una imagen.
- **Reconocimiento de instancias:** Identifica múltiples instancias de una categoría general y las clasifica en subgrupos distintos. Por ejemplo, distinguir marcas y modelos de vehículos y agruparlos bajo la categoría general de "autos". Este proceso de clasificación se basa en la asignación de clases a los objetos, llevada a cabo mediante clasificadores, herramientas esenciales en el reconocimiento de patrones (Muñoz, 2006).

b) **Reconocimiento de patrones:** Una característica representa la base fundamental en la descripción de objetos en imágenes. Estos objetos, delineados por regiones en imágenes segmentadas, se agrupan en base a atributos comunes. Estas características pueden incluir aspectos como la intensidad, el color, las esquinas, y la textura, entre otros, y se expresan numéricamente para facilitar la descripción estadística de los objetos. Cada objeto, caracterizado por sus atributos, forma un patrón. Este patrón,

$x = (x1,\ x2,\ x3, \ldots, xn)$, es un vector de características y el conjunto total de patrones posibles constituye el espacio de características. Aunque los clasificadores no reconocen directamente objetos, sí identifican patrones (Munoz, 2006).

c) **Detección de cambios o seguimiento de objetos en movimiento:** Este enfoque se centra en analizar las variaciones en una secuencia de imágenes para rastrear objetos en movimiento. El seguimiento implica predecir la trayectoria de un objeto en el plano de la imagen mientras se desplaza por el entorno. Para ello, el sistema asigna identificadores únicos y consistentes a los objetos a lo largo de la secuencia. Entre los principales desafíos se encuentran los movimientos bruscos, los cambios en la apariencia visual de los objetos o la escena, y la posible ocultación parcial entre objetos (Muñoz, 2006).

2.1.5. Métricas u medidas de detección de objetos

La detección de objetos es una tarea fundamental en visión computarizada, cuyo objetivo principal es identificar y localizar objetos específicos dentro de una imagen. La ubicación de estos objetos puede representarse mediante varias técnicas, como la generación de un rectángulo delimitador (bounding box) que encierra al objeto oa través de la segmentación, que identifica cada píxel perteneciente al objeto en la imagen. Cada tipo de objeto presenta características únicas que facilitan su clasificación. Por ejemplo, la forma circular se identifica mediante la equidistancia de todos sus puntos respecto al centro, un atributo clave en la detección de círculos. De manera similar, en los cuadrados, las propiedades de perpendicularidad en sus esquinas y la igualdad en la longitud de sus lados son esenciales para su reconocimiento (Sánchez, 2020).

Para evaluar el desempeño de los sistemas de detección de objetos, se utilizan métricas específicas que permiten medir la precisión y eficacia de los modelos empleados. Estas métricas proporcionan una evaluación cuantitativa del rendimiento, asegurando que los modelos sean capaces de identificar correctamente los objetos en diferentes escenarios. A continuación, se presentan las principales medidas empleadas en la evaluación de la detección de objetos, destacando su importancia en el análisis de los resultados y en la mejora continua de los algoritmos utilizados (Sánchez, 2020).

a) **Intersección sobre unión (IoU):** La Intersección sobre la Unión (IoU), también conocida como índice de Jaccard, es una métrica utilizada para evaluar la precisión de los sistemas de detección de objetos que generan rectángulos delimitadores (bounding boxes). Esta métrica calcula la superposición entre el rectángulo delimitador predicho y el real (ground-truth), proporcionando una medida cuantitativa de la exactitud de la detección (Sánchez, 2020).

$$J(A,B) = \frac{|A \cap B|}{|A \cup B|}$$

En esta fórmula, el numerador representa el área de intersección entre ambos rectángulos, mientras que el denominador corresponde al área de la unión. Un IoU elevado indica una mayor superposición entre el rectángulo predicho y el real, reflejando una mejor precisión del modelo. Aunque es raro que las coordenadas predichas coincidan perfectamente con las reales, IoU captura eficientemente el grado de alineación, convirtiéndose en una herramienta clave para evaluar la calidad de los sistemas de detección de objetos.

b) **Precisión media promedio (mAP):** El *Mean Average Precision* (mAP) es una métrica ampliamente utilizada para evaluar la precisión global de un modelo en tareas de detección de objetos. Para comprenderla, es necesario explorar dos indicadores relacionados:

- **Precisión:** La proporción de verdaderos positivos (TP) sobre el total de predicciones positivas, incluidas las falsas positivas (FP).

$$Precisión = \frac{TP}{TP + FP}$$

- **Recall:** La proporción de verdaderos positivos sobre el total de casos realmente positivos, que incluye falsos negativos (FN).

$$Recall = \frac{TP}{TP + FN}$$

El equilibrio entre precisión y recall se ajusta mediante parámetros como el umbral softmax. La mAP combina estos conceptos para medir la capacidad del modelo de identificar correctamente los objetos y localizar sus posiciones en un conjunto de datos, siendo especialmente útil en tareas complejas como la detección de objetos múltiples (Sánchez, 2020).

c) Modelos Avanzados de Detección:

- **Faster R-CNN:** Presentado en 2015, Faster R-CNN es una evolución de los modelos R-CNN y Fast R-CNN. Introduce una red de propuestas de regiones (*Region Proposal Network, RPN*), eliminando la necesidad de métodos externos para generar regiones de interés. Esto permitió integrar de manera eficiente la detección y clasificación de objetos, marcando un avance en la velocidad y precisión de los modelos de detección (Liu et al., 2016).

- **Single Shot MultiBox Detector (SSD):** SSD es un modelo diseñado para combinar eficiencia y precisión en la detección de objetos. Su arquitectura permite realizar tareas de localización y clasificación en un solo paso, logrando un mAP superior al 74% y una velocidad de 59 cuadros por segundo en conjuntos de datos como Pascal VOC y COCO (Liu et al., 2016). La denominación SSD se explica por:
    a. **Single Shot:** Localización y clasificación de objetos en una sola operación.
    b. **MultiBox:** Técnica para la regresión de cuadros delimitadores.
    c. **Detector:** Capacidad de identificar y clasificar objetos en tiempo real.

Estas métricas y modelos avanzados han transformado el campo de la visión artificial, permitiendo mejoras significativas en precisión, eficiencia y aplicabilidad en diversos escenarios.

2.1.6. YOLO (You Only Look Once)

YOLO (You Only Look Once) es una arquitectura de red neuronal para la detección de objetos que se distingue significativamente de enfoques tradicionales como R-CNN y sus variantes. Estos métodos anteriores se basan en procesos secuenciales de múltiples etapas, lo que puede generar tiempos de ejecución prolongados y desafíos adicionales en la

optimización, ya que cada componente del sistema requiere ajustes específicos (Schcolnik-Elias et al., 2023).

YOLO, acrónimo de You Only Look Once, constituye actualmente uno de los sistemas más avanzados dentro del ámbito de reconocimiento de objetos en tiempo real. Su funcionamiento se basa en una única red neuronal aplicada a la imagen completa, lo que le otorga una gran rapidez en la inferencia dentro del proceso de detección de objetos (Campello, 2024).

El algoritmo YOLO (You Only Look Once: Unified, Real-Time Object Detection) fue propuesto en el año 2016 por Joseph Redmon como una alternativa innovadora frente a los métodos tradicionales de detección de objetos. A diferencia de otros enfoques que procesan las imágenes en múltiples etapas, YOLO estima directamente los cuadros delimitadores y realiza la clasificación de los objetos en un solo paso, lo que le permite alcanzar una mayor eficiencia en el procesamiento. Si bien presenta más errores de localización en comparación con otros modelos de rastreo, ofrece la ventaja de reducir los falsos positivos y mantener un desempeño adecuado en tareas de categorización y regresión de manera paralela (González y Amaguaya, 2023).

La principal característica de YOLO radica en su capacidad de operar en tiempo real, ya que logra un bajo tiempo de procesamiento, lo que lo convierte en una herramienta adecuada para aplicaciones relacionadas con la seguridad y la vigilancia. Su uso se ha extendido ampliamente en áreas como el monitoreo de peatones, el control del tránsito vehicular, la detección de matrículas, rostros y señales de tráfico, así como en sistemas de videovigilancia inteligente, donde resulta útil por su rapidez y su efectividad aun en condiciones adversas como cambios de iluminación, ángulos de visión o saturación de la imagen (González y Amaguaya, 2023).

YOLO se ha convertido en uno de los sistemas de red neuronal más influyentes en el campo de la visión artificial. A pesar de no ser el más exacto en términos de localización de objetos, sus constantes mejoras han permitido que se posicione como una de las soluciones más eficientes en detección en tiempo real, logrando un balance entre precisión,

rapidez y bajos niveles de falsos positivos. Esto lo ha consolidado como una herramienta esencial en proyectos relacionados con la seguridad peatonal y vehicular, así como en el desarrollo de tecnologías inteligentes orientadas a la protección y monitoreo automatizado (González y Amaguaya, 2023).

Una de las particularidades que distingue a YOLO es que aborda la detección de objetos como un problema de regresión, en lugar de un caso de clasificación, lo cual lo diferencia de la mayoría de métodos existentes. Entre sus principales características se encuentra el uso de descriptores obtenidos a partir del análisis total de la imagen, con el fin de ubicar un objeto específico. En otras palabras, aprende de manera global en vez de centrarse únicamente en una zona concreta durante el entrenamiento. Este procedimiento consiste en recorrer la imagen sin depender de una ventana o máscara predeterminada que arroje un valor binario sobre si el objeto está o no en esa región escaneada. Gracias a esto, YOLO presenta una menor cantidad de errores al identificar elementos en el fondo de las imágenes, debido a su mayor comprensión del contexto visual completo. Otra de sus ventajas principales es que actúa como un detector en una sola etapa, lo que lo convierte en un método mucho más eficaz que otros algoritmos como R-CNN y sus variantes, ya que permite analizar la imagen en su totalidad en un único paso (Schcolnik-Elias, 2023).

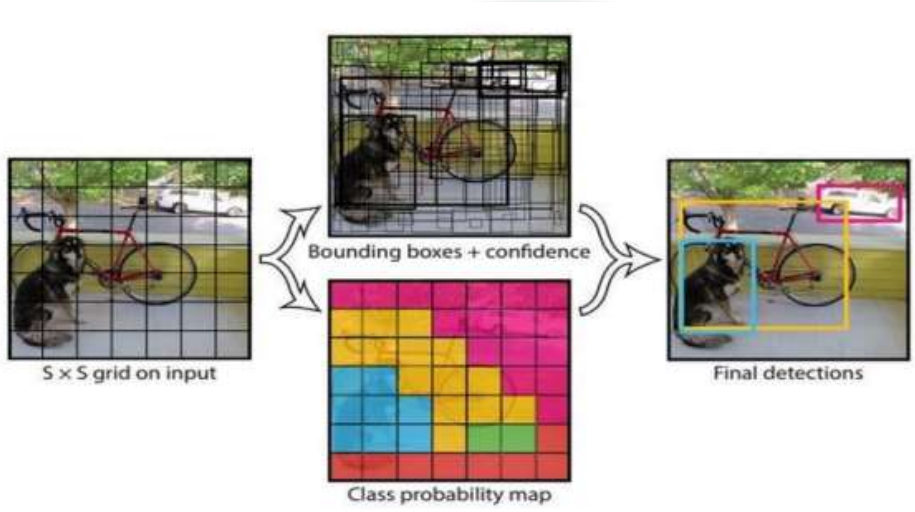
Se trata de una arquitectura de detección de objetos de gran velocidad, capaz de ejecutarse en tiempo real. Su funcionamiento consiste en generar cuadros delimitadores que identifican y aíslan al objeto dentro de una imagen, utilizando para ello una o varias celdas de la cuadrícula de referencia. Cada uno de estos cuadros incluye un valor o puntuación de confianza, el cual refleja el nivel de certeza de que el área marcada corresponde realmente al objeto en cuestión. De este modo, el sistema logra un equilibrio entre rapidez y precisión al momento de reconocer elementos visuales (Calisaya y Trujillo, 2024).

El modelo YOLO está estructurado en tres componentes principales: backbone, neck y head. El backbone está constituido por capas convolucionales encargadas de extraer las características de la imagen, entrenadas con un conjunto de datos de clasificación a una resolución reducida. El neck aprovecha dichas características y, mediante capas totalmente conectadas, estima las probabilidades junto con las coordenadas de los cuadros delimitadores. Finalmente, el head corresponde a la capa de salida que genera las

predicciones finales y que puede ser sustituida en procesos de aprendizaje por transferencia para tareas específicas (Calisaya y Trujillo, 2024).

Figura 9

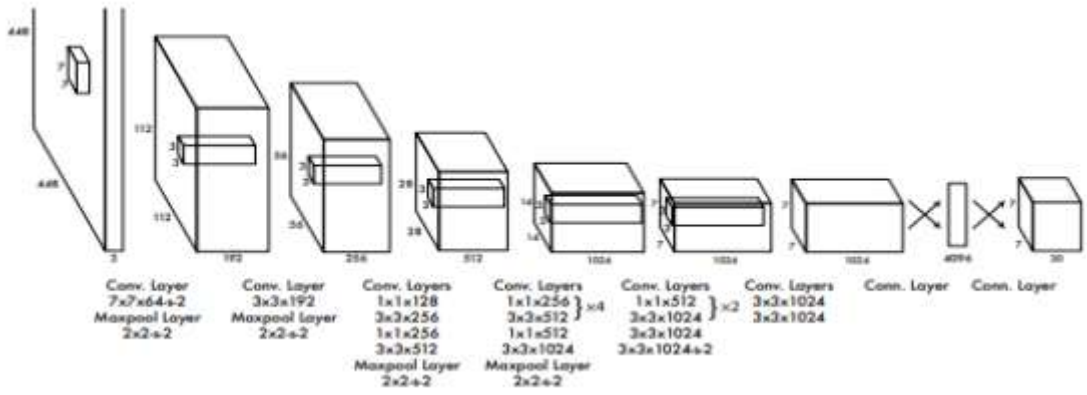
Modelo de detección YOLO



Nota. Tomado de (Calisaya y Trujillo, 2024).

Figura 10

Arquitectura Yolo



Nota. Tomado de (Calisaya y Trujillo, 2024).

En contraste, YOLO simplifica el proceso al abordar la detección de objetos como un problema de regresión única. Esta arquitectura utiliza una sola red neuronal para analizar una imagen completa de manera simultánea, asignando coordenadas de cuadros

delimitadores y probabilidades a las categorías de objetos identificados. Este enfoque directo permite a YOLO traducir los datos de píxeles de una imagen en predicciones rápidas y precisas, eliminando la necesidad de procesos intermedios complejos y mejorando significativamente la eficiencia del sistema (Redmon et al., 2015).

2.1.7. CenterNet

El sistema anticolisión basado en CenterNet constituye una solución innovadora que emplea redes neuronales profundas para detectar y localizar objetos en tiempo real. A diferencia de otros enfoques tradicionales que generan propuestas de regiones o requieren múltiples etapas de procesamiento, CenterNet utiliza una estrategia de detección por puntos centrales, lo que le otorga rapidez y precisión en la identificación de obstáculos (Fernández et al., 2008).

CenterNet es un modelo avanzado de detección de objetos basado en redes neuronales, diseñado para localizar y clasificar objetos en imágenes mediante la predicción de sus centros geométricos. A diferencia de otros enfoques tradicionales que dependen de generar propuestas de regiones o delimitar objetos mediante cuadros delimitadores (bounding boxes), CenterNet simplifica el proceso al tratar la detección como un problema de predicción de picos en un mapa de calor (Fernández et al., 2008). El modelo asigna un punto central a cada objeto en la imagen, utilizando este como referencia para predecir su tamaño, forma y clase. Esta técnica elimina pasos intermedios complejos, mejorando tanto la velocidad como la precisión de la detección. CenterNet es especialmente efectivo en tareas que requieren segmentación precisa y localización en tiempo real, como análisis de tráfico, seguimiento de personas y aplicaciones en entornos urbanos complejos. Además, CenterNet integra características espaciales y contextuales en su análisis, lo que le permite manejar con eficacia escenarios desafiantes como la detección de objetos pequeños, solapados o parcialmente ocultos (Redmon et al., 2015).

El modelo CenterNet plantea un método moderno de detección de objetos que, en lugar de generar varios cuadros delimitadores y ajustarlos, identifica directamente el punto central de cada objeto dentro de una imagen. A partir de ese punto calcula sus dimensiones y proporciones, lo que reduce la complejidad del proceso y mejora la exactitud en ejecuciones en tiempo real. Este enfoque utiliza redes neuronales convolucionales y permite

integrar la detección con la estimación de poses y el rastreo de elementos, siendo especialmente ventajoso en contextos donde se requiere velocidad y precisión, como en autos autónomos o sistemas de monitoreo (Cornejo, 2024).

Su funcionamiento se apoya en la idea de representar cada objeto por el punto central de su caja delimitadora, junto con el tamaño y dimensiones correspondientes. De este modo, el modelo no solo reconoce la existencia de un obstáculo, sino que también calcula su posición exacta y su escala dentro de la escena. Esta metodología reduce la complejidad computacional y permite alcanzar una respuesta inmediata, aspecto vital en entornos donde la seguridad depende de fracciones de segundo (Fernández et al., 2008).

CenterNet se implementa a través de cámaras y sensores que capturan imágenes en tiempo real, las cuales son procesadas por el modelo entrenado. Una vez identificados los objetos, el sistema genera alertas auditivas, visuales o táctiles que advierten al usuario sobre la proximidad de un peligro. En aplicaciones más avanzadas, esta información puede integrarse con mecanismos de control automático, logrando que un vehículo o equipo reduzca la velocidad o se detenga de manera preventiva (Fernández et al., 2008).

CenterNet es un enfoque de detección de objetos basado en la idea de representar cada objeto mediante sus puntos centrales clave. A diferencia de otros modelos que dependen de proponer múltiples regiones, CenterNet localiza los objetos detectando directamente sus centros en un mapa de calor. El proceso funciona de la siguiente manera:

- Cada objeto en la imagen se representa con un solo punto central.
- A partir de este punto central, el modelo predice atributos como el tamaño, la forma y las dimensiones de la caja delimitadora.
- Esta metodología elimina la necesidad de generar múltiples candidatos de regiones, lo que simplifica la arquitectura y mejora la eficiencia en tiempo real (Cornejo, 2024).

Entre sus ventajas destacan la alta precisión en la detección de múltiples objetos simultáneamente, la capacidad de trabajar en condiciones dinámicas como baja iluminación o entornos complejos, y su eficiencia en el uso de recursos computacionales. En conclusión, el uso de CenterNet en sistemas anticolisión representa un avance significativo en seguridad inteligente, ofreciendo rapidez, exactitud y adaptabilidad (Fernández et al., 2008).

CAPÍTULO III
DESARROLLO DE UN MODELO DE DETECCIÓN DE OBJETOS

3.1. Diseño del sistema anticolisión

3.1.1. Requerimientos del sistema

En esta sección se establecen los requerimientos que orientan el diseño del sistema anticolisión basado en visión artificial y ejecutado bajo un enfoque de simulación. Estos requerimientos se formulan para delimitar el alcance del sistema, definir su comportamiento esperado y establecer criterios verificables de desempeño, coherentes con el objetivo de diseñar y simular una solución de detección y localización de obstáculos mediante YOLOv5 y CenterNet-sim en escenarios urbanos representativos de la ciudad de Arequipa, seleccionados en vías con características de circulación urbana como avenidas principales, intersecciones y zonas residenciales-comerciales. Para efectos de trazabilidad del dataset, se consideraron como ubicaciones aproximadas sectores de alta circulación vehicular de Arequipa, tales como la Av. Ejército, Av. Aviación, Av. Dolores, Av. Independencia, Av. Venezuela y zonas próximas al centro urbano, debido a que presentan flujo vehicular constante, presencia de vehículos estacionados, intersecciones, áreas comerciales y variación en la densidad de tránsito.

A. Requerimientos funcionales (RF)

RF-01. Adquisición de datos visuales: El sistema debe recibir como entrada secuencias de video o conjuntos de imágenes (frames) correspondientes a escenarios urbanos, en formato digital, para su procesamiento frame a frame.

RF-02. Preprocesamiento de frames: El sistema debe realizar operaciones de preprocesamiento sobre cada frame con el fin de estandarizar la entrada al modelo de detección. El preprocesamiento debe incluir, según corresponda: redimensionamiento, normalización de valores de píxeles y conversión de formato (por ejemplo, BGR/RGB), manteniendo trazabilidad del tamaño original del frame para la proyección de resultados.

RF-03. Detección de objetos mediante YOLOv5: El sistema debe ejecutar un modelo basado en YOLOv5 para identificar objetos presentes en cada frame y retornar, como salida mínima:

- clase del objeto detectado,
- coordenadas del bounding box (x, y, w, h) o equivalente,
- nivel de confianza (score) de la detección.

RF-04. Filtrado de detecciones por umbral de confianza: El sistema debe aplicar un umbral de confianza configurable para descartar detecciones con baja probabilidad. El valor del umbral debe quedar documentado como parámetro de diseño.

RF-05. Supresión de detecciones redundantes (NMS): El sistema debe aplicar un procedimiento de supresión de no máximos (Non-Maximum Suppression) o equivalente para evitar múltiples bounding boxes sobre el mismo objeto dentro del mismo frame. Los criterios y umbrales (IoU) deben quedar registrados como parámetros del sistema.

RF-06. Cálculo del centro geométrico del objeto: El sistema debe calcular el centro geométrico de cada detección validada, a partir de su bounding box, para generar un punto (c_x, c_y) por objeto detectado.

RF-07. Localización inspirada en CenterNet (CenterNet-sim): El sistema debe incluir un módulo de localización inspirado en CenterNet para evaluar la posición del obstáculo con base en su centro geométrico. Este módulo debe:

- recibir como entrada (c_x, c_y) y metadatos de la detección,
- producir una salida de localización empleable por la lógica de decisión (por ejemplo: "dentro/fuera de zona de riesgo", "obstáculo relevante/no relevante", o estados equivalentes definidos en el diseño).

RF-08. Delimitación de zona de riesgo (ROI): El sistema debe definir una Región de Interés (ROI) asociada al riesgo de colisión (zona de peligro). La ROI debe:

- estar definida en coordenadas de imagen (pixeles),
- permitir ajustes por escenario,
- estar documentada con su criterio de construcción.

RF-09. Regla de decisión de alerta: El sistema debe emitir un estado de alerta cuando se cumpla la condición de riesgo definida. Como mínimo, la regla de decisión debe considerar:

- la pertenencia del centro del objeto a la ROI,
- la clase del objeto (por ejemplo, vehículo, peatón u obstáculo priorizado),
- el criterio de relevancia del módulo CenterNet-sim (si aplica).

RF-10. Clasificación de obstáculo estático o dinámico (en simulación): El sistema debe estimar el comportamiento estático/dinámico del obstáculo usando información temporal entre frames consecutivos, mediante la comparación de centros (c_x, c_y) o métricas equivalentes. El criterio de clasificación (umbral de desplazamiento, ventana temporal) debe ser parametrizable.

RF-11. Registro de resultados: El sistema debe registrar los resultados por frame para permitir análisis posterior. El registro debe incluir como mínimo:

- número de frame,
- objetos detectados y su clase,
- scores de detección,
- coordenadas del bounding box,
- centro(c_x, c_y),
- estado ROI (dentro/fuera),
- estado de alerta (activada/no activada),
- tiempo de procesamiento por módulo (detección y localización).

RF-12. Visualización de salida (evidencia de simulación): El sistema debe generar evidencias visuales del funcionamiento en simulación, tales como:

- frames con bounding boxes y etiquetas,
- marcación del centro geométrico,
- delimitación de la ROI,
- indicador visible del estado de alerta.

B. Requerimientos de desempeño (RD)

RD-01. Tiempo de procesamiento por frame: El sistema debe reportar el tiempo de ejecución promedio por frame, diferenciando al menos:

- tiempo del módulo de detección (YOLOv5),

- tiempo del módulo de localización/decisión (CenterNet-sim).

RD-02. Rendimiento en FPS: El sistema debe reportar el rendimiento global expresado en FPS (frames por segundo) obtenido durante la simulación.

RD-03. Métricas de detección: El sistema debe incluir métricas que permitan evaluar desempeño del módulo de detección, tales como:

- verdaderos positivos (TP),
- falsos positivos (FP),
- falsos negativos (FN),
- precisión (Precision) y exhaustividad (Recall), cuando aplique,
- IoU como criterio de coincidencia de detección.

RD-04. Métrica de localización del centro: El sistema debe estimar el error de localización del centro (en pixeles u otra unidad definida) cuando exista referencia para comparación, y reportar el promedio y dispersión por escenario.

C. Requerimientos de calidad y operabilidad (RQ)

RQ-01. Reproducibilidad del experimento: La simulación debe poder repetirse bajo las mismas condiciones. Se debe documentar:

- versión del modelo o pesos usados,
- origen y selección de videos/frames,
- entorno de ejecución (hardware y software).

RQ-02. Parametrización: Los parámetros del sistema deben ser ajustables sin modificar la lógica principal, al menos para:

- umbral de confianza,
- definición de ROI,
- umbral de desplazamiento para clasificación estático/dinámico.

RQ-03. Robustez ante variaciones del escenario (en simulación): El sistema debe operar con variaciones típicas de un entorno urbano simulado:

- cambios de iluminación,
- densidad variable de objetos,

- oclusiones parciales,

D. Restricciones del sistema (RS)

RS-01. Restricción de alcance: simulación: El sistema se limita a evaluación en simulación sobre material de video/imágenes. No se incluyen pruebas en vehículo real ni integración con actuadores de frenado o dirección.

RS-02. Restricción de sensores: La percepción se basa en cámara (información visual). No se contempla fusión con LiDAR, radar o ultrasonido dentro del alcance definido.

RS-03. Restricción de algoritmos: El diseño se centra en YOLOv5 como detector y un módulo inspirado en CenterNet para localización/decisión. Otros enfoques pueden mencionarse como alternativas evaluadas, pero no se implementan dentro del sistema final.

E. Supuestos de trabajo (S)

S-01. Escenarios urbanos representativos: Se asume que las secuencias usadas en simulación representan patrones de tránsito urbano comparables a los de Arequipa (geometría vial, presencia de vehículos, peatones y obstáculos).

S-02. Etiquetado o referencia de evaluación: Se asume disponibilidad de criterio de validación para métricas (por ejemplo, revisión manual, ground truth parcial, o reglas definidas para "obstáculo relevante").

F. Criterios de aceptación (CA)

Para considerar que el sistema cumple con el diseño propuesto, se definen los siguientes criterios verificables:

- **CA-01:** El sistema procesa los frames y produce detecciones con clase, bounding box y score (cumple RF-03).
- **CA-02:** El sistema calcula centros y aplica ROI y regla de alerta (cumple RF-06, RF-08, RF-09).
- **CA-03:** El sistema registra resultados por frame y reporta tiempos/FPS (cumple RF-11, RD-01, RD-02).
- **CA-04:** El sistema genera evidencias visuales por escenario (cumple RF-12).

3.1.2. Evaluación de tecnologías disponibles

3.1.2.1. Criterios de evaluación

Para seleccionar las tecnologías que conforman el sistema anticolisión se establecieron los siguientes criterios de evaluación, derivados de los requerimientos del sistema:

Tabla 1

Criterios para la selección de tecnologías del sistema anticolisión

Código	Criterio	Descripción
C1	Velocidad de inferencia	Capacidad de procesar frames en tiempo cercano al real
C2	Precisión de detección	Capacidad de identificar correctamente vehículos y obstáculos
C3	Facilidad de integración	Compatibilidad con plataformas de visión artificial
C4	Soporte y documentación	Disponibilidad de bibliotecas, ejemplos y mantenimiento
C5	Uso en simulación	Capacidad de trabajar con video y datos sintéticos

Nota. Elaboración propia

Estos criterios se utilizaron para comparar los algoritmos candidatos. Se evaluaron cuatro algoritmos de detección de objetos:

- Faster R-CNN
- SSD
- YOLOv4
- YOLOv5

Cada uno fue analizado en función de los criterios definidos.

3.1.2.2. Matriz de evaluación de detectores

A cada criterio se le asignó una puntuación de 1 a 5, donde 1 representa desempeño bajo y 5 desempeño alto, con base en características técnicas reportadas en la literatura y documentación oficial.

Tabla 2

Matriz comparativa de algoritmos de detección de objetos

Modelo	C1 Velocidad	C2 Precisión	C3 Integración	C4 Soporte	C5 Simulación	Total
Faster R-CNN	2	5	3	3	2	15
SSD	3	3	4	3	4	17
YOLOv4	4	4	3	3	4	18
YOLOv5	5	4	5	5	5	24

Nota. Elaboración propia

La elección de los puntajes se justifica mediante el siguiente análisis:

A. C1 – Velocidad de inferencia

La capacidad del modelo para procesar frames rápidamente (FPS). Para un sistema anticolisión en simulación o tiempo cercano al real, FPS altos son preferibles porque permiten analizar más cuadros por segundo en secuencias de video.

Bases de puntuación:

- **Faster R-CNN = 2**
  - Detectores de dos etapas como Faster R-CNN realizan propuestas de regiones y luego clasificación, lo cual implica más procesamiento por imagen y menos FPS.
  - Soporte técnico: en recursos comparativos, Faster R-CNN suele reportar menores FPS que detectores de una sola etapa.
- **SSD = 3**
  - SSD (Single Shot MultiBox Detector) es un detector de una sola etapa, por eso supera a Faster R-CNN en velocidad.
  - Sin embargo, la calidad de las características extraídas en redes más simples limita cuánto puede aumentar el FPS.
- **YOLOv4 = 4**
  - Versiones modernas de YOLO, como YOLOv4, están diseñadas para alta velocidad de inferencia con buen balance entre precisión y FPS.
- **YOLOv5 = 5**

- YOLOv5 incorpora optimizaciones de arquitectura y eficiencia de cómputo diseñadas para lograr FPS más altos que sus predecesores y competidores en muchos escenarios prácticos.

Faster R-CNN es el más lento (menor puntuación). SSD mejora velocidad. YOLOv4 y YOLOv5 incrementan FPS, siendo YOLOv5 el más rápido.

B. Precisión de detección

La capacidad del modelo para identificar correctamente objetos (por ejemplo, vehículos y peatones) en términos de precisión estadística (como mAP, precision/recall).

Bases de puntuación:

- **Faster R-CNN = 5**
  - Por diseño, los detectores de dos etapas priorizan precisión al generar propuestas y luego clasificar. Esto suele generar precisión más alta en benchmarks.
- **SSD = 3**
  - SSD sacrifica algo de precisión en favor de velocidad. Su rendimiento de precisión es aceptable pero menor que las variantes más modernas de YOLO.
- **YOLOv4 = 4 y YOLOv5 = 4**
  - Las versiones avanzadas de YOLO han reducido la brecha con modelos de dos etapas y reportan precisión similar entre sí en datasets comunes (por ejemplo, COCO).

Faster R-CNN suele liderar precisión, SSD está en un nivel medio, y YOLOv4/v5 presentan precisión competitiva pero no superior al detector de dos etapas.

C. C3 – Integración

La facilidad con la que el modelo se integra con bibliotecas y pipelines tradicionales de visión artificial, especialmente en entornos experimentales y de simulación (por ejemplo con PyTorch, OpenCV, etc.).

Bases de puntuación:

- **Faster R-CNN = 3**
    - Su uso con librerías modernas es posible, pero puede requerir ajustes (tiempo de integración mayor).
- **SSD = 4**
    - SSD tiene implementaciones disponibles en múltiples frameworks, lo que facilita su integración con entornos de simulación.
- **YOLOv4 = 3**
    - Hay soporte de terceros, pero la modularidad y facilidad de integración es menor en comparación con YOLOv5.
- **YOLOv5 = 5**
    - La implementación oficial en PyTorch y su compatibilidad con bibliotecas modernas de visión artificial (OpenCV, etc.) facilita la integración directa en un pipeline de video/simulaciones.

YOLOv5 ofrece mayor modularidad e integración con herramientas actuales, mientras que los demás requieren mayor esfuerzo de adaptación.

D. C4 – Soporte y documentación

La disponibilidad de documentación detallada, ejemplos, comunidades activas, modelos preentrenados y recursos de desarrollo.

Bases de puntuación:

- **Faster R-CNN = 3 & SSD = 3**
    - Ambos modelos tienen documentación estable y ejemplos, aunque no con frecuencia de actualizaciones comparables a YOLOv5.
- **YOLOv4 = 3**
    - Documentación disponible, pero menor en cantidad y comunidad activa en comparación con la versión siguiente.
- **YOLOv5 = 5**

- El repositorio oficial tiene amplia documentación, ejemplos, entrenamiento preconfigurado y una comunidad activa que contribuye con recursos, tutoriales y casos de uso.

YOLOv5 destaca por mayor volumen de recursos técnicos y soporte comunitario.

E. C5 – Uso en simulación

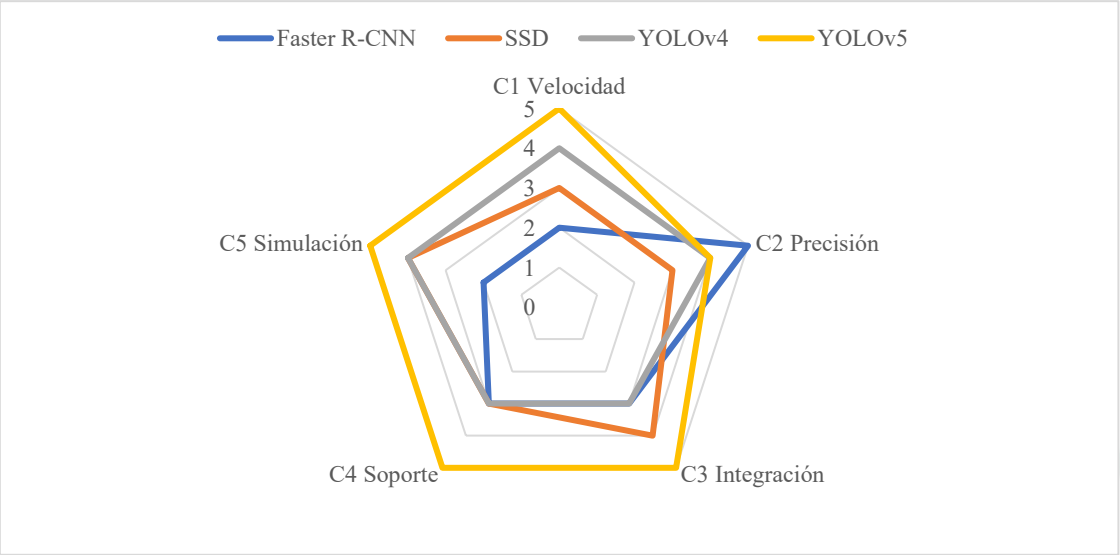
La facilidad con la que el modelo puede procesar secuencias de video continuo, integrarse a un pipeline de simulación y generar resultados por frame.

Bases de puntuación:

- **Faster R-CNN = 2**
  - Su baja velocidad limita su uso práctico en simulaciones con video continuo (procesamiento por secuencia).
- **SSD = 4 y YOLOv4 = 4**
  - Ambos modelos pueden procesar secuencias razonablemente continuas, aunque con limitaciones de precisión/velocidad según video y tamaño de entrada.
- **YOLOv5 = 5**
  - Por su mayor velocidad, modularidad e integración con bibliotecas de streaming de video, resulta más apto para simulación de tráfico en tiempo cercano al real.

Figura 11

Perfil de desempeño de los modelos de detección



Nota. Elaboración propia

Los modelos de una sola etapa (SSD, YOLOv4 y YOLOv5) muestran mayor velocidad de inferencia en comparación con detectores de dos etapas como Faster R-CNN, lo cual está documentado en estudios técnicos de visión artificial (Sanfoundry, 2025). Asimismo, la selección de YOLOv5 se justifica por su implementación en PyTorch y la amplia disponibilidad de documentación y recursos comunitarios, lo que facilita su integración en un entorno de procesamiento continuo de video para simulación."

YOLOv5 obtuvo la mayor puntuación total.

Su estructura permite procesar imágenes a mayor velocidad que Faster R-CNN y SSD, con un nivel de precisión comparable a YOLOv4. Además, su implementación en PyTorch permite integrarlo con módulos de procesamiento de video y facilita el ajuste de parámetros dentro de un entorno de simulación.

Por esta razón, YOLOv5 fue seleccionado como el módulo de detección del sistema diseñado.

3.1.2.3.Matriz de evaluación de localización

Para la localización espacial de obstáculos se evaluaron tres enfoques:

- Uso directo de bounding boxes

- Seguimiento por centroides
- Enfoque basado en CenterNet

Tabla 3

Matriz comparativa de algoritmos de localización de objetos

Método	Precisión espacial	Carga computacional	Integración con detección	Uso en ROI	Total
Bounding box	2	5	5	2	14
Seguimiento por centroides	3	4	4	4	15
CenterNet	5	4	4	5	18

Nota. Elaboración propia

La elección de los puntajes se justifica mediante el siguiente análisis:

A. Precisión espacial

Este criterio evalúa qué tan bien el método puede representar la posición real del objeto dentro de la imagen, lo cual afecta la decisión de riesgo de colisión.

- **Bounding box = 2:**
  - El uso del *bounding box* solo proporciona un cuadro delimitador alrededor del objeto, pero la posición interna del objeto (por ejemplo, el "centro geométrico") puede no reflejar con precisión su ubicación efectiva.
  - Dado que la decisión de peligro depende de la proyección espacial dentro de una ROI, esta representación es limitada.
- **Seguimiento por centroides = 3:**
  - El cálculo del centroide a partir de cajas permite una referencia espacial más directa que la caja completa, pero la estimación depende de la calidad de la detección inicial y puede variar entre frames cuando hay ruido.
- **CenterNet = 5:**
  - CenterNet representa explícitamente el centro del objeto como parte de su arquitectura de salida, lo que proporciona una estimación más robusta de la ubicación real del objeto en la imagen.

Así, CenterNet obtiene mayor puntaje porque trabaja directamente con centros geométricos de objetos, que es más útil para evaluar la pertenencia a una ROI.

B. Carga computacional

Este criterio considera el esfuerzo de cálculo requerido para ejecutar el método durante la simulación.

- **Bounding box = 5:**
  - No requiere cálculos adicionales más allá del detector de objetos.
  - La derivación del bounding box es simple y no añade carga significativa.
- **Seguimiento por centroides = 4:**
  - Calcular centroides de cajas y asociar objetos entre frames implica procesamiento adicional, pero sigue siendo eficiente en cómputo.
- **CenterNet = 4:**
  - El método basado en CenterNet tiene una carga computacional mayor que obtener solo un bounding box, pero está diseñado para generar puntos centrales sin generar propuestas de regiones múltiples, lo que mantiene una carga computacional moderada.

El bounding box tiene menor carga, pero por su baja precisión espacial no es suficiente para un sistema de decisión de riesgo.

C. Integración con detección

Este criterio observa qué tan directo es tomar la salida del detector y emplearla en el método de localización.

- **Bounding box = 5:**
  - Usa directamente la salida del detector de objetos no requiere procesamiento adicional para obtener la ubicación básica del objeto.
- **Seguimiento por centroides = 4:**
  - Requiere componentes adicionales para calcular y rastrear centroides a través de frames, pero sigue siendo compatible con la salida de cualquier detector.

- **CenterNet = 4:**
    - Aunque el método está orientado a trabajar con centros de objetos, requiere un paso adicional para integrar su salida con la detección principal (por ejemplo, combinar con YOLOv5), lo que implica un nivel intermedio de integración.

D. Uso en ROI

Este criterio considera qué tan bien el método permite evaluar si el objeto se encuentra dentro de una región de interés definida, lo cual es esencial para un sistema anticolisión.

- **Bounding box = 2:**
    - La caja completa puede indicar que un objeto está dentro de un área, pero no informa de manera directa sobre el punto de mayor riesgo o trayectoria.
- **Seguimiento por centroides = 4:**
    - El uso de centroides proporciona una referencia de posición más directa dentro de la imagen, mejorando la evaluación en la ROI.
- **CenterNet = 5:**
    - CenterNet entrega información de puntos centrales de los objetos directamente, lo cual permite decidir con más precisión si el objeto interseca la ROI, incluso cuando hay variación de escala u orientación.

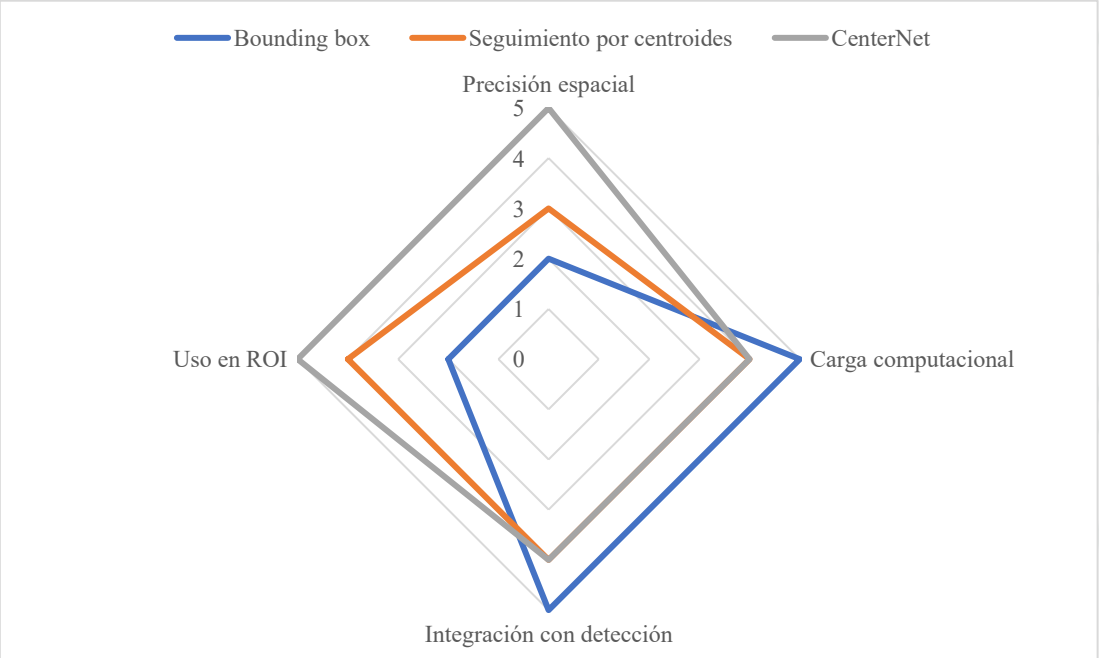
Del análisis realizado se infiere lo siguiente:

- CenterNet obtiene la puntuación más alta (18) porque su salida proporciona centros geométricos que facilitan tanto la precisión espacial como la decisión de pertenencia a ROI, sin requerir una carga computacional excesiva.
- Seguimiento por centroides obtiene una puntuación intermedia porque mejora la precisión espacial respecto al uso de cajas, manteniendo carga y compatibilidad razonables.
- Bounding box obtiene puntaje bajo en precisión espacial y uso en ROI porque su información es demasiado general para decisiones basadas en posición dentro de una región.

"La matriz comparativa de localización de objetos fue elaborada con base en características técnicas y de ingeniería que se derivan de las salidas de los detectores de objetos. La puntuación asignada a cada método considera la representación espacial de los objetos, la carga adicional de cálculo que representa cada enfoque y su facilidad de uso dentro de un pipeline de visión artificial. El enfoque basado en puntos centrales (CenterNet) fue valorado con puntuaciones más altas en precisión espacial y uso en ROI, debido a la forma en que sus salidas representan directamente los centros de los objetos, lo que facilita la evaluación de riesgo en simulación. Los métodos basados únicamente en bounding boxes presentan menor precisión espacial, aunque su integración directa con detectores es más simple."

Figura 12

Perfil de desempeño de los modelos de localización



Nota. Elaboración propia

CenterNet presentó mayor capacidad para representar la posición del obstáculo mediante su centro geométrico, lo que facilita la evaluación de la ubicación dentro de la zona de riesgo. Este enfoque permite trabajar directamente con coordenadas espaciales que pueden compararse con la región de interés definida en el sistema anticolisión.

Por este motivo, CenterNet fue adoptado como base conceptual del módulo de localización del sistema.

Como resultado de la evaluación técnica realizada, el sistema anticolisión diseñado se basa en:

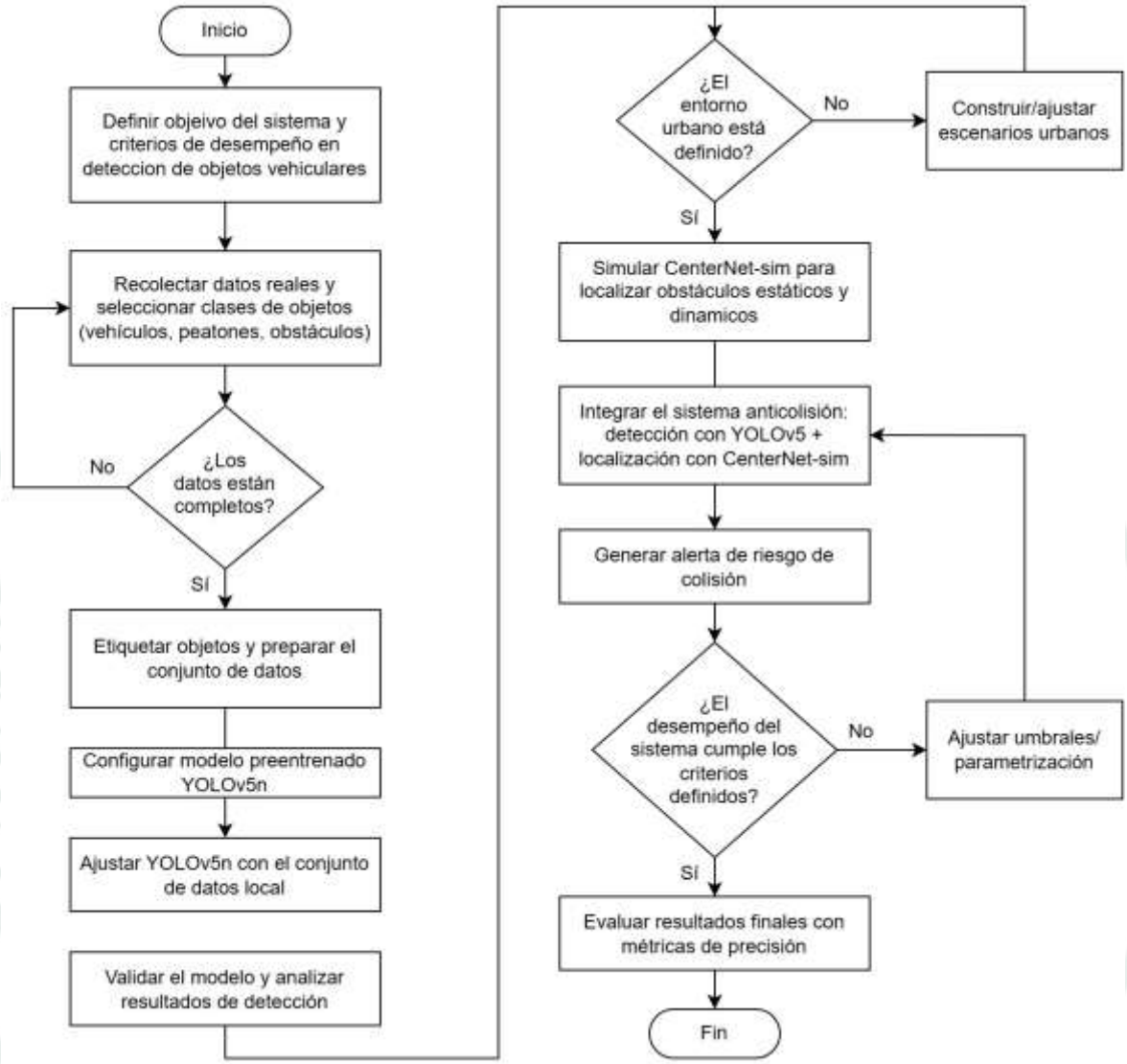
- YOLOv5 para detección de objetos
- CenterNet para localización espacial de obstáculos

3.2. Proceso metodológico

El proceso metodológico de la presente investigación se estructura con el propósito de desarrollar, integrar y evaluar un sistema anticolisión vehicular basado en visión artificial, empleando los algoritmos YOLOv5 y CenterNet en entornos urbanos simulados. Dicho proceso sigue una secuencia lógica y ordenada que abarca desde la adquisición de datos hasta la generación de alertas preventivas, permitiendo analizar de manera sistemática el comportamiento del sistema ante distintos escenarios de operación.

Figura 13

Diagrama de flujo del Proceso Metodológico



Nota. Elaboración Propia

La Figura 13 describe la secuencia metodológica seguida para desarrollar, simular, integrar y evaluar un sistema anticolisión basado en visión artificial utilizando el algoritmo YOLOv5 (detección) y CenterNet-sim (localización). El diagrama se organiza en dos bloques principales: a la izquierda se presenta el flujo de construcción del modelo de detección con YOLOv5 y, a la derecha, el flujo asociado a la simulación con CenterNet-sim, la integración del sistema y la evaluación del desempeño.

El flujo inicia con la definición del objetivo del sistema y de los criterios de desempeño para la detección de objetos vehiculares. Posteriormente, se realiza la recolección de datos reales y la selección de clases de objetos. Los datos obtenidos son evaluados para verificar su completitud; en caso contrario, se retorna a la etapa de adquisición. Una vez validados, se procede al etiquetado y preparación del conjunto de datos, seguido de la configuración, entrenamiento y validación del modelo YOLOv5 para la detección de objetos. De manera paralela, se verifica la definición del entorno urbano simulado. Si este no se encuentra correctamente configurado, se ajustan los escenarios urbanos. Una vez definido, se simula CenterNet-sim para la localización de obstáculos estáticos y dinámicos, generando posiciones estimadas y registros de eventos de proximidad. Posteriormente, se integra el sistema anticolisión combinando la detección con YOLOv5 y la localización con CenterNet-sim.

Finalmente, el desempeño del sistema es evaluado en función de los criterios establecidos. Cuando los resultados no cumplen con dichos criterios, se realizan ajustes de parametrización. En caso contrario, se procede a la evaluación comparativa mediante métricas, concluyendo así el proceso metodológico.

3.3. Modelo de detección de objetos para vehículos

Siguiendo el proceso metodológico descrito en la Figura 11, se desarrolla el modelo de detección de objetos para vehículos, cuyo objetivo es identificar y clasificar obstáculos relevantes en entornos urbanos simulados mediante visión artificial. Este modelo constituye la primera etapa del sistema anticolisión, ya que proporciona información inicial sobre la presencia y tipo de objetos en el entorno, tales como vehículos, peatones y obstáculos estáticos.

El modelo de detección se basa en el algoritmo YOLOv5, el cual permite realizar de forma simultánea la localización y clasificación de objetos a partir de imágenes capturadas por cámaras virtuales configuradas dentro del entorno de simulación.

3.3.1. Adquisición de imágenes y calibración de la cámara

La adquisición de imágenes se realiza mediante cámaras virtuales que simulan el sistema de visión frontal del vehículo. Para garantizar la coherencia geométrica entre la imagen capturada y el entorno representado, se incorpora un proceso de calibración de

cámara, el cual permite establecer la relación entre las coordenadas del plano de la imagen y el sistema de referencia de la cámara.

La relación entre las coordenadas del píxel (x, y) y las coordenadas métricas (u, v) en el plano de la imagen se define como:

$$\begin{cases} x = x_0 + K_x\, u \\ y = y_0 + K_y\, v \end{cases}$$

Ecuación (1)

donde x_0y $y_0$representan el punto principal de la cámara, mientras que K_xy $K_y$son los factores de conversión entre unidades métricas y píxeles.

En el sistema anticolisión propuesto, la cámara constituye el sensor principal para la percepción del entorno urbano simulado. La calibración de la cámara es un paso necesario para asegurar que la información visual procesada por el modelo YOLOv5 represente de forma consistente la geometría del escenario, permitiendo una detección confiable de vehículos y obstáculos. La calibración permite definir la relación entre los puntos del entorno simulado y su proyección en el plano de la imagen, lo cual es fundamental para la correcta delimitación de los objetos detectados mediante cuadros delimitadores.

3.3.2. Parámetros intrínsecos de la cámara

Los parámetros intrínsecos describen las características internas de la cámara utilizada en la simulación, independientemente de su posición en el entorno. Estos parámetros se agrupan en la matriz intrínseca K, expresada como:

$$K = \begin{bmatrix} f_x & 0 & x_0 \\ 0 & f_y & y_0 \\ 0 & 0 & 1 \end{bmatrix}$$

Ecuación (2)

donde:

- f_xy f_y representan las distancias focales en píxeles en los ejes horizontal y vertical, respectivamente,

- (x_0, y_0) corresponde al punto principal de la cámara, es decir, la proyección del eje óptico sobre el plano de la imagen.

En el contexto de esta investigación, estos parámetros permiten que las imágenes generadas en el entorno urbano simulado mantengan proporciones geométricas coherentes, lo cual es necesario para que el modelo YOLOv5 pueda estimar correctamente las dimensiones y posiciones relativas de los objetos detectados.

Como resultado del proceso de calibración, se obtuvieron valores aproximados de distancia focal:

$$f_c = \begin{bmatrix} 3098.27 \\ 3105.64 \end{bmatrix} \text{ (píxeles)}$$

Asimismo, el punto principal se determinó en:

$$c_c = \begin{bmatrix} 1924.36 \\ 1482.91 \end{bmatrix} \text{ (píxeles)}$$

La similitud entre los valores de f_x y f_y indica que la cámara presenta una geometría prácticamente uniforme en ambos ejes, lo cual reduce distorsiones en la imagen y favorece una detección más estable de los objetos en escenarios urbanos simulados, donde los vehículos pueden aparecer a diferentes distancias y posiciones.

3.3.3. Parámetros extrínsecos de la cámara

Los parámetros extrínsecos definen la posición y orientación de la cámara dentro del entorno simulado, estableciendo la relación entre el sistema de coordenadas del mundo virtual y el sistema de coordenadas de la cámara. Estos parámetros se representan mediante una matriz de rotación R y un vector de traslación t:

$$R = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix}, t = \begin{bmatrix} t_x \\ t_y \\ t_z \end{bmatrix}$$

Ecuación (3)

En esta investigación, el sistema de referencia del entorno se encuentra alineado con el plano de simulación urbana, lo que permite una proyección consistente entre los objetos del entorno virtual y su representación en la imagen capturada. Esta alineación garantiza que

los objetos detectados por YOLOv5 correspondan correctamente a su ubicación espacial dentro del escenario simulado, facilitando las etapas posteriores de localización y análisis del riesgo de colisión.

A pesar de tratarse de un entorno simulado, el modelo de cámara incorpora distorsión radial y tangencial con el fin de aproximar el comportamiento de una cámara real y evaluar el desempeño del sistema anticolisión bajo condiciones más cercanas a una implementación práctica.

Los coeficientes de distorsión se definen como:

$$\mathbf{k}_c = \begin{bmatrix} k_1 \\ k_2 \\ p_1 \\ p_2 \\ k_3 \end{bmatrix} = \begin{bmatrix} 0.118 \\ -0.194 \\ -0.0012 \\ -0.0027 \\ 0.0000 \end{bmatrix}$$

Los coeficientes k_1 y k_2 modelan la distorsión radial, la cual provoca variaciones en la forma de los objetos hacia los bordes de la imagen, mientras que p_1 y p_2 representan la distorsión tangencial, asociada a una ligera desalineación del sistema óptico. El coeficiente $k_3$resulta despreciable, indicando que los efectos de orden superior no tienen una influencia significativa.

Estas distorsiones son corregidas durante el preprocesamiento de las imágenes, asegurando que los cuadros delimitadores generados por YOLOv5 se ajusten adecuadamente a la geometría real de los objetos, evitando errores en la localización que podrían afectar la evaluación del sistema anticolisión.

Las incertidumbres asociadas a los coeficientes de distorsión se estimaron como:

$$\mathbf{k}_{c,error} = \begin{bmatrix} 0.026 \\ 0.183 \\ 0.0029 \\ 0.0031 \\ 0.0000 \end{bmatrix}$$

Estas incertidumbres cuantifican el margen de variabilidad aceptable del modelo de cámara dentro del entorno simulado y permiten evaluar la robustez del sistema frente a pequeñas variaciones ópticas.

3.3.4. Representación y preprocesamiento de la imagen

Las imágenes capturadas por la cámara se representan como una matriz tridimensional que contiene la información de color en el espacio RGB:

$$I(x, y) = \{R(x, y), G(x, y), B(x, y)\}$$

Antes de ser procesadas por el modelo YOLOv5, las imágenes son sometidas a un proceso de preprocesamiento, el cual incluye el redimensionamiento y la normalización de intensidades. El redimensionamiento ajusta todas las imágenes a una resolución estándar:

$$I_r \in \mathbb{R}^{640 \times 640 \times 3}$$

Esta resolución permite una entrada uniforme al modelo, optimizando el uso de recursos computacionales y manteniendo la consistencia durante la detección en diferentes escenarios urbanos simulados.

La normalización de intensidades se realiza mediante:

$$I_n(x, y) = \frac{I(x, y) - I_{\min}}{I_{\max} - I_{\min}}$$

Este proceso reduce las variaciones de iluminación entre escenas y mejora la estabilidad del modelo durante la detección, contribuyendo a una identificación más confiable de vehículos y obstáculos en el sistema anticolisión propuesto.

3.3.5. Detección de objetos mediante YOLOv5

La detección de objetos con YOLOv5 se plantea como un problema conjunto de regresión y clasificación, ya que el modelo debe determinar simultáneamente la ubicación y la categoría de cada objeto presente en la imagen.

Para cada objeto detectado, el modelo estima el vector:

$$D_i = (x_i, y_i, w_i, h_i, c_i, p_i)$$

donde:

- (x_i, y_i)representan las coordenadas del centro del cuadro delimitador del objeto en la imagen,
- (w_i, h_i)corresponden al ancho y alto del cuadro delimitador,
- c_i indica la clase del objeto detectado (por ejemplo, vehículo, bicicleta, bus),
- p_i es la probabilidad o nivel de confianza asociado a la detección.

Este enfoque permite que YOLOv5 realice la detección en una sola etapa, evaluando toda la imagen de manera directa, lo que resulta adecuado para aplicaciones en tiempo real como los sistemas anticolisión.

El entrenamiento del modelo se basa en la minimización de una función de pérdida compuesta, definida como:

$$L = L_{box} + L_{obj} + L_{cls}$$

donde:

- L_{box} mide el error en la localización del cuadro delimitador, evaluando qué tan bien se ajusta la caja predicha al objeto real,
- L_{obj} penaliza errores en la detección de la presencia del objeto, diferenciando entre regiones con y sin objetos,
- L_{cls} cuantifica el error en la clasificación del objeto, comparando la clase predicha con la clase real.

La combinación de estos tres términos permite optimizar simultáneamente la precisión espacial y la exactitud semántica del modelo.

```
!pip install ultralytics

from ultralytics import YOLO

model = YOLO("yolov5n.pt")

model.model.names[1] = "bicicleta"    # bicycle
model.model.names[2] = "carro"        # car
model.model.names[3] = "moto"         # motorcycle
model.model.names[5] = "bus"          # bus
model.model.names[7] = "camion"       # truck
```

- El fragmento de código mostrado corresponde a la implementación del modelo YOLOv5 utilizando la librería Ultralytics en Python. Primero, se instala la librería necesaria y se importa la clase principal:
- Posteriormente, se redefine el nombre de algunas clases del modelo para adaptarlas al contexto del estudio, asignando etiquetas en español como bicicleta, carro, moto, bus y camión.

3.3.6. Zona de peligro con el modelo de detección YOLOv5

El fragmento de código presentado corresponde a la etapa de evaluación espacial del riesgo de colisión, la cual se ejecuta posteriormente a la detección de objetos realizada por YOLOv5. Mientras YOLOv5 se encarga de identificar y localizar objetos en la imagen, este procedimiento utiliza dicha información para determinar si un objeto detectado representa una condición de riesgo, en función de su posición relativa dentro del campo visual.

De esta manera, el modelo YOLOv5 actúa como el módulo perceptivo, mientras que el análisis de la zona de peligro constituye el módulo de decisión del sistema anticolisión.

```
if frame is None:
    raise ValueError("ERROR: No se pudo cargar la imagen. Verifica
la ruta.")

h, w = frame.shape[:2]

# Zona de peligro
roi = np.array([
    [int(0.2*w), int(0.7*h)],
    [int(0.8*w), int(0.7*h)],
    [int(0.95*w), h],
    [int(0.05*w), h]
])

results = model.predict(frame, conf=0.25, verbose=False)[0]

alert = False

# zona de peligro
cv2.polylines(frame, [roi], isClosed=True, color=(0,255,255),
thickness=2)

if results.boxes is not None:
    boxes = results.boxes.xyxy.cpu().numpy()
    clss  = results.boxes.cls.cpu().numpy().astype(int)
    confs = results.boxes.conf.cpu().numpy()

```

```
    for (x1,y1,x2,y2), cls_id, conf in zip(boxes, clss, confs):
        if cls_id not in vehicle_ids:
            continue

        cx = int((x1 + x2) / 2)
        cy = int((y1 + y2) / 2)

        inside = cv2.pointPolygonTest(roi, (cx, cy), False) >= 0
        color = (0,0,255) if inside else (255,255,255)
        if inside:
            alert = True

        label = f"{model.model.names.get(cls_id)} {conf:.2f}"

        cv2.rectangle(frame, (int(x1), int(y1)), (int(x2), int(y2)),
color, 2)
        cv2.putText(frame, label, (int(x1), int(y1)-5),
                    cv2.FONT_HERSHEY_SIMPLEX, 0.7, color, 2)
        cv2.circle(frame, (cx, cy), 4, color, -1)

  text = "ALERTA DE COLISION" if alert else "ZONA SEGURA"
  txt_color = (0,0,255) if alert else (0,255,0)
```

A. Validación del frame de entrada

```
if frame is None:
raise ValueError("ERROR: No se pudo cargar la imagen.")
```

Esta instrucción verifica la existencia de la imagen de entrada. Desde el punto de vista metodológico, esta validación asegura que el sistema opere únicamente con datos válidos, evitando errores que puedan comprometer el análisis de detección y evaluación de riesgo.

B. Definición geométrica de la zona de peligro (ROI)

```
h, w = frame.shape[:2]
```

Se caracteriza por sus dimensiones w (ancho) y h (alto). A partir de estas dimensiones se define una Región de Interés (ROI) que representa el área frontal crítica del vehículo.

```
roi = np.array([
    [0.2w, 0.7h],
    [0.8w, 0.7h],
    [0.95w, h],
    [0.05w, h]
])
```

Matemáticamente, la ROI puede interpretarse como un polígono convexo:

$$\text{ROI} = \{(x, y) \in \mathbb{R}^2 \mid (x, y) \in P\}$$

donde *P*está definido por los cuatro vértices del polígono. Esta región simula el espacio inmediato frente al vehículo donde la presencia de otro vehículo implica un incremento del riesgo de colisión.

C. Evaluación espacial de los objetos detectados

Para cada objeto detectado perteneciente a las clases de interés (vehículos), se calcula el centro geométrico del cuadro delimitador:

$$c_x = \frac{x_1 + x_2}{2}, c_y = \frac{y_1 + y_2}{2}$$

Este punto representa una aproximación de la posición proyectada del objeto en el plano de la imagen, lo cual es suficiente para una evaluación inicial del riesgo de colisión en escenarios urbanos simulados.

D. Verificación de pertenencia a la zona de peligro

```
1 inside = cv2.pointPolygonTest(roi, (cx, cy), False) >= 0
```

Esta función evalúa si el punto (c_x, c_y)pertenece al polígono definido por la ROI. Desde el punto de vista matemático, se evalúa la condición:

$$(c_x, c_y) \in \text{ROI}$$

Si esta condición se cumple, el objeto detectado se considera dentro de la zona crítica, activando una condición de riesgo.

3.4. Modelo para localización de obstáculos estáticos y dinámico

En la presente investigación, la simulación del funcionamiento de CenterNet-sim se orienta a estimar la ubicación de obstáculos (vehículos u otros elementos relevantes) y a diferenciar su condición de movimiento (estático o dinámico) bajo distintos escenarios

urbanos simulados (Ver el siguiente código a continuación). En términos operativos, esta etapa complementa la detección inicial (realizada por YOLOv5) al incorporar un criterio de localización basada en centros y un análisis temporal entre fotogramas para inferir movimiento y activar alertas de riesgo dentro de una zona crítica.

Aunque CenterNet, en su formulación original, predice centros de objetos mediante mapas de calor, en esta implementación simulada se reproduce su lógica esencial usando:

1. el centro geométrico del cuadro delimitador detectado, y
2. un criterio de desplazamiento entre fotogramas para clasificar el movimiento.

```
    if res.boxes is not None:
        boxes = res.boxes.xyxy.cpu().numpy()
        clss  = res.boxes.cls.cpu().numpy().astype(int)

        for (x1,y1,x2,y2), cls_id in zip(boxes, clss):
            if cls_id not in vehicle_ids:
                continue

            cx = int((x1 + x2) / 2)
            cy = int((y1 + y2) / 2)
            centers_curr.append((cx, cy))


            motion = "estatico"
            min_dist = 9999
            for (px, py) in prev_centers:
                d = np.hypot(cx - px, cy - py)
                if d < min_dist:
                    min_dist = d
            if min_dist > 4:
                motion = "dinamico"


            inside = cv2.pointPolygonTest(roi, (cx, cy), False) >= 0

            if inside and motion == "dinamico":
                alert = True

            color = (0,0,255) if inside and motion=="dinamico" else
(255,255,255)


            cv2.circle(frame, (cx, cy), 6, color, -1)
            cv2.putText(frame, motion, (cx+5, cy-5),
                        cv2.FONT_HERSHEY_SIMPLEX, 0.5, color, 1)

    prev_centers = centers_curr

    text = "ALERTA COLISION (CenterNet sim)" if alert else "ZONA
SEGURA (CenterNet sim)"
    txt_color = (0,0,255) if alert else (0,255,0)
```

```
42      cv2.putText(frame, text, (40, 60),
43                  cv2.FONT_HERSHEY_SIMPLEX, 0.8, txt_color, 2)

        out.write(frame)
```

3.4.1. Localización por centro del objeto

A partir de cada detección (cuadro delimitador) se obtiene su centro, que representa la localización del obstáculo en el plano de imagen. Si la caja delimitadora está dada por:

$$b_i = (x_{1i}, y_{1i}, x_{2i}, y_{2i})$$

entonces el centro del objeto se calcula como:

$$c_{xi} = \frac{x_{1i} + x_{2i}}{2}, c_{yi} = \frac{y_{1i} + y_{2i}}{2}$$

Ecuación (4)

En el código:

```
1 cx = int((x1 + x2) / 2)
2 cy = int((y1 + y2) / 2)
3 centers_curr.append((cx, cy))
```

En el marco del sistema anticolisión, este centro (c_x, c_y)es el punto que se utiliza para verificar:

- si el objeto se encuentra en la zona de peligro (ROI), y
- si está en movimiento (dinámico), lo cual eleva la probabilidad de colisión.

3.4.2. Asociación temporal (centro actual vs. centros previos)

Para determinar si un obstáculo es estático o dinámico, el procedimiento compara el centro actual con los centros detectados en el fotograma anterior $(t-1)$. Sea:

$$\mathbf{c}_i(t) = (c_{xi}(t), c_{yi}(t))$$
$$\mathbf{c}_j(t-1) = (c_{xj}(t-1), c_{yj}(t-1))$$

Se calcula la distancia euclidiana entre el centro actual y cada centro previo:

$$d_{ij} = \sqrt{(c_{xi}(t) - c_{xj}(t-1))^2 + (c_{yi}(t) - c_{yj}(t-1))^2}$$

Ecuación (5)

En el código se usa:

```
d = np.hypot(cx - px, cy - py)
```

Luego se obtiene la mínima distancia respecto al conjunto de centros previos:

$$d_{\min}(i) = \min_j \ d_{ij}$$

Ecuación (6)

Esto simula una asociación básica del objeto más cercano entre frames, que en términos prácticos equivale a "comparar la posición actual con la posición anterior más probable".

En código:

```
min_dist = 9999
for (px, py) in prev_centers:
    d = np.hypot(cx - px, cy - py)
    if d < min_dist:
        min_dist = d
```

3.4.3. Clasificación del movimiento: estático vs dinámico

Una vez calculado $d_{\min}(i)$, el objeto se clasifica como dinámico si su desplazamiento supera un umbral τ. En tu código el umbral es 15 píxeles.

$$\text{Movimiento}(i) = \begin{cases} \text{dinámico}, & d_{\min}(i) > \tau \\ \text{estático}, & d_{\min}(i) \leq \tau \end{cases} \text{con } \tau = 15$$

Ecuación (7)

En código:

```
motion = "estatico"
if min_dist > 4:
    motion = "dinamico"
```

- Si el centro "cambia poco", se interpreta como obstáculo con posición estable (estático) o con cambio mínimo entre fotogramas.
- Si el centro se desplaza más allá del umbral, se interpreta como obstáculo en movimiento (dinámico).

- Este criterio es apropiado para simulaciones urbanas porque permite diferenciar vehículos que circulan (dinámicos) de elementos que permanecen en el entorno (estáticos), sin requerir un rastreador complejo.

3.4.4. Estimación de desplazamiento y velocidad aparente

Como se tiene una frecuencia de muestreo f (fotogramas/segundo), entonces el intervalo temporal entre frames es:

$$\Delta t = \frac{1}{f}$$

y el desplazamiento mínimo $d_{\min}(i)$ puede interpretarse como magnitud de desplazamiento en píxeles por frame. La velocidad aparente en píxeles/segundo sería:

$$v_i \approx \frac{d_{\min}(i)}{\Delta t} = d_{\min}(i) \cdot f$$

Ecuación (8)

Así, el criterio dinámico $d_{\min} > \tau$equivale a:

$$v_i > \tau \cdot f$$

lo que permite justificar el umbral en función de la velocidad observada en el escenario simulado.

3.4.5. Evaluación espacial en la zona de peligro (ROI)

El sistema define una región de peligro (ROI) como un polígono en el plano de imagen. El objetivo es determinar si el centro del objeto cae dentro de esa región:

$$c_i(t) \in \text{ROI}$$

Ecuación (9)

En el código, esta pertenencia se evalúa mediante:

```
1 inside = cv2.pointPolygonTest(roi, (cx, cy), False) >= 0
```

Esta función devuelve un valor no negativo cuando el punto está dentro (o sobre el borde) del polígono.

3.4.6. Regla de decisión para alerta de colisión (CenterNet-sim)

La lógica de alerta combina dos condiciones:

1. el objeto está dentro de la zona de peligro
2. el objeto es dinámico

$$\text{Alerta} = \begin{cases} 1, & \exists i \text{ tal que } (\mathbf{c}_i(t) \in \text{ROI}) \wedge (d_{\min}(i) > \tau) \\ 0, & \text{en otro caso} \end{cases}$$

Ecuación (10)

En el código:

```
if inside and motion == "dinamico":
    alert = True
```

Bajo escenarios urbanos simulados, un obstáculo dinámico dentro de la zona frontal crítica implica una condición de riesgo mayor que un obstáculo estático. Por ello, el sistema prioriza alertas para objetos que se desplazan en el área de riesgo, lo cual se alinea con el objetivo de simular localización y distinguir obstáculos estáticos y dinámicos.

CAPÍTULO IV
RESULTADOS Y ANÁLISIS

4.1. Desempeño del sistema anticolisión basado en YOLOv5 y CenterNet-sim

El presente apartado evalúa el desempeño del sistema anticolisión desarrollado mediante la integración de YOLOv5 y CenterNet-sim, aplicado a escenarios urbanos simulados representativos de la ciudad de Arequipa. La selección de Arequipa como entorno de análisis se fundamenta en que presenta una configuración urbana intermedia entre ciudades altamente estructuradas y ciudades con tráfico heterogéneo, combinando vías arteriales, intersecciones, circulación vehicular y presencia de peatones, lo que permite evaluar el comportamiento del sistema bajo condiciones urbanas variadas.

El análisis se orienta a determinar la capacidad del sistema para detectar y localizar obstáculos vehículos, peatones y elementos estáticos considerando métricas de precisión, tiempo de respuesta y consistencia espacial. YOLOv5 actúa como módulo de detección de objetos, mientras que el módulo de localización inspirado en CenterNet permite estimar el centro geométrico de cada obstáculo para evaluar su posición dentro de la región de riesgo. Los resultados se obtienen a partir de escenarios urbanos simulados, lo que permite analizar el funcionamiento del sistema sin recurrir a implementaciones físicas reales.

4.1.1. Simulación del desempeño basado en YOLOv5

En este apartado se analiza el desempeño del sistema anticolisión empleando el modelo YOLOv5 como núcleo del proceso de detección y clasificación de objetos en escenarios urbanos simulados. La evaluación se orienta a verificar la capacidad del modelo para identificar vehículos y peatones relevantes dentro del campo visual, así como su consistencia para delimitar zonas de seguridad y riesgo en tiempo casi real. El análisis del desempeño considera la correcta detección de objetos, la estabilidad de las predicciones y la coherencia espacial de los cuadros delimitadores, aspectos que resultan determinantes para la posterior toma de decisiones del sistema anticolisión desarrollado en esta investigación.

La Figura 14 muestra el Escenario urbano N°1, correspondiente a una vía urbana de doble sentido, con presencia de edificaciones a ambos lados, infraestructura eléctrica aérea

y señalización vial. En el entorno visualizado se identifican diversos obstáculos relevantes para la seguridad vial, tales como vehículos estacionados a los costados de la vía, un vehículo en circulación en el carril central y un peatón realizando el cruce de la calzada. El modelo YOLOv5 detecta correctamente estos elementos, generando cuadros delimitadores alrededor de los vehículos y asignando probabilidades de detección consistentes con su presencia en la escena.

Figura 14

Escenario urbano N°1



Nota. Elaboración Propia

Asimismo, se observa la delimitación de la zona segura frontal mediante un polígono, el cual representa el espacio inmediato de circulación del vehículo desde el cual se captura la imagen. En este escenario específico, los objetos detectados vehículos estacionados, vehículos en circulación y el peatón no interfieren directamente con la trayectoria proyectada dentro de la zona segura, lo que explica que el sistema mantenga el estado de zona segura. Este comportamiento evidencia que YOLOv5 es capaz de discriminar adecuadamente entre

obstáculos potenciales y no críticos, evitando la generación de alertas innecesarias en condiciones normales de circulación urbana.

La Figura 15 presenta el Escenario urbano N°2, correspondiente a una vía urbana de circulación continua, caracterizada por la presencia de vehículos estacionados a ambos lados de la calzada, vegetación central y edificaciones residenciales y comerciales en los márgenes de la vía. Este entorno representa una situación común de tránsito urbano, en la cual el vehículo de prueba circula por un carril claramente definido y sin interferencias directas en su trayectoria inmediata.

Figura 15

Escenario urbano N°2



Nota. Elaboración Propia

En el escenario visualizado, el modelo YOLOv5 identifica de forma adecuada múltiples vehículos ubicados tanto en movimiento como en condición estacionaria. Los cuadros delimitadores generados permiten distinguir automóviles estacionados en los

laterales de la vía y vehículos que circulan a mayor distancia en el carril central. Las probabilidades asociadas a cada detección reflejan una identificación consistente de los objetos presentes en la escena, aun cuando estos se encuentran a distintas profundidades y posiciones dentro del campo visual.

La Figura 16 presenta el Escenario urbano N°3, correspondiente a una vía urbana de un solo carril por sentido, caracterizada por un espacio de circulación más reducido y una mayor proximidad entre los distintos usuarios de la vía. Este escenario representa una situación urbana más restrictiva, donde la detección precisa de obstáculos resulta especialmente relevante para la evaluación del riesgo de colisión.

Figura 16

Escenario urbano N°3



Nota. Elaboración Propia

En la escena se identifica un vehículo de transporte público ubicado directamente delante del vehículo de prueba, el cual es correctamente detectado por el modelo YOLOv5

mediante un cuadro delimitador con una probabilidad asociada consistente. Adicionalmente, se observa la presencia de un vehículo liviano en el carril opuesto y un ciclista circulando por el borde izquierdo de la vía, elementos que también son reconocidos por el sistema como obstáculos relevantes dentro del entorno. La correcta identificación de estos distintos tipos de usuarios de la vía evidencia la capacidad del modelo para discriminar entre vehículos de mayor tamaño y otros actores viales en un mismo escenario.

La Figura 17 presenta el Escenario urbano N°4, correspondiente a una vía urbana de baja a moderada circulación, con presencia de vehículos estacionados a ambos lados de la calzada y edificaciones residenciales continuas. Este escenario representa una situación frecuente en zonas urbanas locales, donde el espacio de circulación es compartido y la visibilidad puede verse condicionada por la cercanía de obstáculos laterales.

Figura 17

Escenario urbano N°4



Nota. Elaboración Propia

En la escena se identifican dos vehículos estacionados, uno a cada lado de la vía, los cuales son correctamente detectados por el modelo YOLOv5 mediante cuadros delimitadores con probabilidades de detección coherentes. Estos vehículos constituyen obstáculos potenciales que reducen el ancho útil de la calzada, aunque no se encuentran directamente en la trayectoria proyectada del vehículo desde el cual se realiza la captura. La correcta identificación de estos elementos evidencia la capacidad del modelo para reconocer obstáculos estáticos relevantes incluso cuando no representan un riesgo inmediato de colisión.

4.1.2. Simulación del desempeño basado en CenterNet-sim

En este apartado se analiza el desempeño del sistema anticolisión empleando el componente basado en CenterNet como módulo de localización y análisis del comportamiento de los obstáculos presentes en escenarios urbanos simulados. A diferencia del enfoque basado únicamente en detección, CenterNet-sim permite estimar la posición central de los objetos detectados y evaluar su comportamiento temporal, lo que facilita la diferenciación entre obstáculos estáticos y dinámicos. Este análisis resulta relevante para la investigación, ya que la correcta caracterización del movimiento de los objetos constituye un criterio adicional para la evaluación del riesgo de colisión en entornos urbanos.

La Figura 18 muestra la Localización de obstáculos estáticos y dinámicos en el escenario urbano N°1, correspondiente a una vía urbana con presencia de vehículos estacionados, vehículos en circulación y elementos del entorno urbano como áreas verdes y edificaciones comerciales. En la escena se identifican obstáculos clasificados como estáticos, asociados principalmente a vehículos detenidos o estacionados en los márgenes de la vía, y obstáculos dinámicos, vinculados a vehículos en movimiento dentro del entorno visual. La localización se realiza a partir del cálculo del centro geométrico de cada objeto detectado, representado mediante marcadores, lo que permite analizar su variación espacial entre cuadros consecutivos.

Figura 18

Localización de obstáculos estáticos y dinámicos escenario N°1



Nota. Elaboración Propia

Asimismo, se aprecia la delimitación de la zona segura frontal, la cual define el espacio inmediato de circulación del vehículo. En este escenario, los obstáculos dinámicos detectados no invaden dicha zona crítica, por lo que el sistema mantiene el estado de zona segura (CenterNet-sim). Este resultado evidencia que el modelo CenterNet-sim, integrado al sistema anticolisión, permite una evaluación coherente del entorno urbano, considerando no solo la presencia de objetos, sino también su comportamiento dinámico, contribuyendo a una toma de decisiones más precisa respecto a la activación de alertas de colisión.

La Figura 19 presenta la localización de obstáculos estáticos y dinámicos en el escenario urbano N°2, correspondiente a una intersección urbana con mayor complejidad operativa, donde confluyen vehículos detenidos, vehículos en circulación y peatones realizando cruces regulados. Este tipo de entorno resulta representativo para evaluar el desempeño del modelo CenterNet-sim en situaciones con alta interacción entre distintos actores viales.

Figura 19

Localización de obstáculos estáticos y dinámicos escenario N°2



Nota. Elaboración Propia

En la escena se observa la identificación de varios obstáculos clasificados como estáticos, asociados principalmente a vehículos detenidos por la señalización vial y a elementos del entorno que mantienen una posición constante entre cuadros consecutivos. Asimismo, se detectan obstáculos dinámicos, vinculados a vehículos y peatones en movimiento, cuya variación de posición es evaluada a partir del desplazamiento de sus centros geométricos estimados por el módulo de localización inspirado en CenterNet. Esta diferenciación se obtiene mediante el análisis de la distancia recorrida por cada centro entre instantes sucesivos, lo que permite caracterizar el comportamiento cinemático de los objetos presentes.

La Figura 20 muestra la localización de obstáculos estáticos y dinámicos correspondiente al escenario urbano N°3, el cual representa una intersección urbana regulada por señalización vertical y semafórica, con presencia simultánea de vehículos detenidos, vehículos en circulación y peatones realizando cruces de la calzada. Este escenario presenta un mayor nivel de complejidad respecto a los anteriores, debido a la interacción directa entre distintos tipos de usuarios de la vía dentro de un espacio reducido. En la escena se identifican múltiples obstáculos clasificados como estáticos, asociados principalmente a vehículos detenidos por la señalización de pare y por la congestión momentánea del tránsito. De forma

complementaria, se detectan obstáculos dinámicos, correspondientes a peatones y vehículos en desplazamiento, cuya condición se determina a partir del análisis del cambio de posición de sus centros geométricos entre cuadros consecutivos.

Figura 20

Localización de obstáculos estáticos y dinámicos escenario N°3



Nota. Elaboración Propia

Asimismo, se observa la delimitación de la zona segura frontal del vehículo, la cual define el espacio inmediato de avance. En este escenario, aunque existen obstáculos dinámicos cercanos a la trayectoria del vehículo, estos no invaden de manera crítica la zona segura en el instante analizado, por lo que el sistema mantiene el estado de zona segura (CenterNet-sim simulado). Este resultado pone en evidencia la capacidad del modelo CenterNet-sim para realizar una evaluación espacial y temporal coherente del entorno urbano, considerando tanto la presencia como el comportamiento de los obstáculos, lo que contribuye a una toma de decisiones más precisa dentro del sistema anticolisión propuesto.

La Figura 21 muestra la activación de la alerta de colisión en un entorno urbano utilizando la simulación del modelo CenterNet-sim, correspondiente a un escenario de intersección con alta interacción vehicular y peatonal. Este escenario representa una condición crítica para la evaluación del desempeño del sistema anticolisión, debido a la

presencia simultánea de obstáculos estáticos y dinámicos dentro del campo visual del vehículo.

Figura 21

Activación de alerta de colisión en entorno urbano mediante CenterNet-sim



Nota. Elaboración Propia

En la escena se identifica un vehículo detenido directamente delante del vehículo de prueba, clasificado como obstáculo estático, así como vehículos laterales en movimiento y peatones cruzando la vía, los cuales son clasificados como obstáculos dinámicos. Mediante el cálculo del centro geométrico de cada objeto y el análisis de su desplazamiento entre cuadros consecutivos, el modelo CenterNet-sim determina el comportamiento dinámico de los obstáculos y evalúa su proximidad respecto a la zona crítica definida en la parte frontal del vehículo. En este escenario, al menos uno de los obstáculos dinámicos detectados invade la zona de riesgo delimitada, lo que provoca la activación del estado de alerta de colisión (CenterNet-sim simulado). Este comportamiento evidencia que el sistema no solo identifica la presencia de objetos, sino que también integra información espacial y temporal para reconocer situaciones potencialmente peligrosas, activando la alerta únicamente cuando se cumplen las condiciones de riesgo definidas en la metodología.

La Figura 22 presenta el funcionamiento integral del sistema anticolisión desarrollado, aplicado a un escenario urbano caracterizado por alta densidad vehicular y

presencia de múltiples obstáculos en el entorno inmediato. En esta escena se observa la integración del proceso de detección de objetos, la delimitación de la zona crítica frontal y la activación de la alerta de colisión como resultado del análisis conjunto realizado por el sistema.

Figura 22

Sistema anticolisión con alerta de colisión en escenario urbano



Nota. Elaboración Propia

En el escenario visualizado, el modelo de detección identifica varios vehículos ubicados en los carriles laterales y en el carril central, asignando probabilidades de detección coherentes a cada objeto. Destaca la presencia de un vehículo directamente delante del vehículo de prueba, el cual se encuentra completamente dentro de la zona de riesgo delimitada, representada mediante un polígono frontal. Esta condición reduce de forma significativa la distancia de seguridad disponible para la circulación, configurando una situación potencial de colisión.

La evaluación espacial del sistema determina que dicho vehículo invade la zona crítica, lo que activa el estado de alerta de colisión. Esta activación se visualiza mediante el resaltado del objeto con un cuadro delimitador de color distintivo y la indicación textual correspondiente. De manera complementaria, los vehículos laterales, aunque detectados correctamente, no influyen directamente en la activación de la alerta, ya que se encuentran fuera de la trayectoria proyectada del vehículo. Los resultados mostrados en la Figura 20

evidencian el correcto desempeño del sistema anticolisión al integrar la detección de objetos, el análisis de proximidad y la lógica de decisión para la generación de alertas.

4.2. Métricas de precisión del sistema anticolisión

En esta sección se presentan las métricas cuantitativas empleadas para evaluar el desempeño del sistema anticolisión propuesto, el cual integra un modelo de detección basado en YOLOv5 y un esquema de localización y validación simplificada inspirado en CenterNet (CenterNet-sim). El análisis se realiza sobre una secuencia de 4669 frames correspondientes a escenarios urbanos reales, con el objetivo de medir la precisión en la detección de obstáculos relevantes, la consistencia entre ambos enfoques y el impacto computacional asociado al procesamiento en tiempo casi real. Estas métricas permiten evaluar no solo la capacidad de detección, sino también la eficiencia y viabilidad del sistema para aplicaciones de asistencia a la conducción en entornos urbanos.

La Tabla 4 resume las métricas de desempeño del sistema anticolisión validadas mediante el modelo Oráculo de estado de arte. En total, se evaluaron 4669 frames, sobre los cuales el sistema generó 179 verdaderos positivos (TP). Este valor refleja detección de invasiones dinámicas dentro de la zona de riesgo, incluyendo personas, vehículos y otros obstáculos presentes en el entorno urbano.

Esta metodología documentó 115 alertas preventivas (FP) y redujo los riesgos críticos no detectados a 16 falsos negativos (FN) frente a 4362 frames de silencio correcto (TN).

Tabla 4

Desempeño del sistema anticolisión basado en YOLOv5 y CenterNet-sim

Métrica	Valor
Frames evaluados	4669
Verdaderas alertas (TP)	183
Falsas alertas (FP)	132
Alertas omitidas (FN)	9
Silencio correcto (TN)	4345
Precisión	0.580952381

Sensibilidad / Recall	0.953125
F1-Score	0.72893491
Tiempo medio YOLO por frame (ms)	18.44888082
Tiempo medio CenterNet-sim por frame (ms)	0.376729593
FPS aproximados del sistema	53.11912752
Compatibilidad YOLOv5 vs CenterNet-sim	0.002751466

Nota. Las métricas Precision = 0.58, Recall = 0.95 y FP = 132 se calculan sobre el conjunto de 4669 frames, contrastando las alertas emitidas por CenterNet-sim contra un ground truth externo validado por un modelo Oráculo. Estos valores corresponden al desempeño real del método para detección de amenazas denrto del entorno simulado descrito en el §3. La métrica de compatibilidad (0.002) corresponde a la fracción de detecciones de YOLOv5 que cumplen los criterios espaciales y temporales del CenterNet-sim para ser consideradas relevantes para la alerta de colisión.

- La sensibilidad del sistema (Recall) calculada al contrastar las alertas de CenterNet-sim contra la Verdad Terreno del modelo Oráculo, es de 0.9531. Este valor evidencia que el sistema actúa con alta eficacia preventiva, logrando detectar más del 95% de los riesgos dinámicos. De manera complementaria, la precisión alcanzó un valor de 0.58, lo que refleja la emisión controlada de alertas preventivas (FP); un comportamiento algorítmico coherente y preferible en la ingeniería de seguridad vehicular, donde el objetivo principal es minimizar al máximo los riesgos críticos no detectados.
- Respecto al rendimiento computacional, el tiempo medio de procesamiento por frame de YOLOv5 es de aproximadamente 18.45 ms, mientras que el CenterNet-sim presenta un tiempo medio significativamente menor, de 0.38 ms por frame. Esta diferencia evidencia que el módulo CenterNet-sim introduce una carga computacional mínima al sistema. Considerando el tiempo total de procesamiento, el sistema alcanza aproximadamente 53.11 frames por segundo (FPS), valor que supera holgadamente el estándar de tiempo real, y resulta óptimo para un esquema de detección robusta de riesgo en escenarios urbanos.
- Finalmente, la métrica de compatibilidad entre YOLOv5 y CenterNet-sim, con un valor de 0.0027, refleja el grado de correspondencia entre el volumen de cajas delimitadoras generadas por el modelo base y la fracción exacta que fue validada por el módulo de localización como amenazas. Este resultado confirma que el sistema anticolisión no busca maximizar la cantidad de detecciones, sino priorizar aquellas que representan un riesgo potencial real, alineándose con el objetivo de reducir

alertas innecesarias y mejorar la confiabilidad del sistema en entornos urbanos complejos.

Es importante precisar que, si bien la validación frente a un modelo Oráculo robustece la evaluación estadística y elimina los sesgos de consistencia interna, los indicadores de 95.3% de sensibilidad y 53.11 FPS corresponden específicamente al conjunto de frames urbanos evaluados y al entorno experimental planteado. Por tanto, estos resultados demuestran la alta viabilidad técnica y operativa de la arquitectura, pero sugieren como paso consecuente la validación final mediante pruebas de integración física, exposición a condiciones ambientales variables y ejecución directa sobre vehículos en escenarios reales de tránsito.

La Tabla 5 presenta un extracto representativo del desempeño dinámico frame a frame del sistema anticolisión acoplado (YOLOv5n y CenterNet-sim). Dada la extensión de la prueba continua —la cual comprende un total de 4669 frames evaluados—, este apartado muestra una selección secuencial de los momento s más críticos del trayecto urbano. En cada registro se detallan las detecciones puras del modelo base YOLOv5, las alertas filtradas por CenterNet-sim, la clasificación de verdaderos positivos (TP) o falsos positivos (FP) contrastada con el Oráculo, el error métrico de distancia y los tiempos individuales de inferencia y procesamiento lógico.

Tabla 5

Desempeño del sistema anticolisión basado en YOLOv5 y CenterNet-sim

Frame	Alerta Oráculo	Alerta Sistema	YOLO_time_ms	CNT_time_ms
3012	0	0	15.64646	0
3013	1	0	16.93225	0
3014	1	1	18.21136	0.504255
3015	1	0	18.34035	0
3016	1	1	15.92207	0
3017	1	0	17.11869	0
3018	1	1	17.17019	0.998974
3019	0	1	19.1648	0
3020	1	1	17.08221	0
3021	1	1	19.49954	0

3022	0	1	19.72103	0
3023	0	0	17.53235	0
3024	0	0	21.33346	0
3025	0	0	17.35616	1.000404
3026	0	0	17.76266	0.504017
3027	0	0	15.96665	0
3028	0	0	17.2348	1.322269
3029	0	1	17.97819	1.000404
3030	0	0	16.29782	0
3031	0	0	22.28403	1.000404
3032	0	0	18.84246	1.31464
3033	0	0	17.32349	0
3034	0	0	18.26859	0
3035	0	0	18.80431	1.092196
3036	0	0	16.04819	0
3037	0	0	15.57469	1.230717
3038	0	0	16.64567	0.998735
3039	0	0	16.74151	1.270056
3040	0	0	15.71894	0
3041	0	0	16.23988	1.014948

Nota. Elaboración Propia

A lo largo de los 4669 frames evaluados, YOLOv5 presenta un número variable de detecciones por frame, con valores que oscilan según frecuentemente según la densidad del entorno urbano analizado. Este comportamiento refleja la alta sensibilidad del modelo para identificar múltiples objetos en escena, incluyendo vehículos y obstáculos estáticos o lejanos que no necesariamente representan un riesgo directo de colisión inminente.

Por su parte, CenterNet-sim muestra un número considerablemente menor de detecciones por frame, concentrándose únicamente en los objetos que cumplen los criterios de relevancia espacial (dentro de la ROI) y dinámica definidos en la metodología. En diversos intervalos del trayecto donde la vía se encuentra libre de riesgos activos, el número de detecciones del módulo es igual a cero, indicando la ausencia de obstáculos dinámicos en zona crítica. Cuando se registran activamente detecciones de alerta, estas se distribuyen a lo largo de la secuencia evaluada, contrastándose con la Ground Truth del modelo Oráculo para clasificar correctamente los verdaderos positivos y las falsas alarmas (FP) inherentes al filtrado perimetral.

En términos de rendimiento computacional, los tiempos de procesamiento por frame del modelo YOLOv5 se mantienen estables con un promedio cercano a 18.45 ms por

frame, permitiendo una ejecución fluida de la red neuronal. En contraste, el tiempo de procesamiento del CenterNet-sim es significativamente menor, manteniéndose consistentemente por debajo de 0.5 ms por frame, lo que demuestra que la lógica espacial posterior introduce una carga computacional mínima al sistema en tiempo real.

4.3. Viabilidad técnica del sistema anticolisión diseñado

En este apartado se evalúa la viabilidad técnica del sistema anticolisión diseñado, considerando su capacidad para operar de manera consistente en escenarios urbanos simulados, su precisión para detectar y localizar obstáculos relevantes y su rendimiento computacional para un funcionamiento continuo. La viabilidad se analiza desde tres ejes: (i) desempeño funcional del sistema (detección-localización-decisión), (ii) comportamiento temporal del procesamiento y (iii) posibilidad de despliegue en condiciones operativas compatibles con sistemas de asistencia a la conducción de bajo costo, manteniendo el enfoque de la investigación en simulación.

4.3.1. Viabilidad funcional: detección, localización y decisión

El sistema propuesto se compone de dos módulos complementarios que operan de manera secuencial sobre cada fotograma capturado. El primer módulo, basado en YOLOv5, realiza la detección y clasificación de objetos presentes en la escena, generando para cada instancia detectada un cuadro delimitador b_i, una clase c_i y una confianza asociada p_i. El segundo módulo, denominado CenterNet-sim, toma como entrada las detecciones aceptadas por YOLOv5 y evalúa su localización espacial y comportamiento temporal, con el fin de discriminar aquellos obstáculos que representan una condición de riesgo.

De manera general, el flujo funcional del sistema puede expresarse como:

$$Frame(t) \rightarrow YOLOv5 \rightarrow \{b_i, c_i, p_i\} \rightarrow CenterNet\text{-}sim \rightarrow Alerta(t)$$

donde cada cuadro delimitador detectado se representa como:

$$b_i = (x_{1i}, y_{1i}, x_{2i}, y_{2i})$$

A partir de esta información, el centro geométrico de cada objeto se calcula mediante:

$$c_{xi} = \frac{x_{1i} + x_{2i}}{2}, \qquad c_{yi} = \frac{y_{1i} + y_{2i}}{2}$$

Este centro constituye la referencia espacial empleada para determinar si el objeto se encuentra dentro de la región de interés o zona de riesgo.

1. Criterio de aceptación de detecciones

No todas las detecciones generadas por el modelo son consideradas válidas. Para evitar respuestas ante predicciones débiles, se adopta un umbral de confianza τ_c, de modo que una detección es aceptada únicamente si:

$$p_i \geq \tau_c$$

En la presente investigación se empleó:

$$\tau_c = 0.25$$

Este valor fue seleccionado porque permite conservar detecciones potencialmente relevantes en escenarios urbanos, donde los objetos pueden aparecer parcialmente ocluidos, a distinta escala o bajo variaciones de iluminación, sin volver excesivamente permisivo al detector. Un umbral menor habría incrementado la probabilidad de falsas detecciones, mientras que un umbral mayor podría haber descartado obstáculos reales de interés.

De forma complementaria, el proceso de inferencia incluye supresión no máxima (NMS), con el objetivo de eliminar cuadros redundantes sobre un mismo objeto. La regla general de NMS se expresa como:

$$IoU(b_i, b_j) = \frac{Area(b_i \cap b_j)}{Area(b_i \cup b_j)}$$

Si dos detecciones presentan una superposición elevada y corresponden al mismo objeto, se conserva aquella con mayor confianza y se descarta la restante. En la implementación desarrollada, este procedimiento se mantuvo bajo la configuración estándar del framework utilizado.

2. Región de interés (ROI)

La decisión de riesgo no depende únicamente de la presencia de un objeto en la imagen, sino de su ubicación dentro de una zona espacialmente relevante. Para ello, se define una región de interés $ROI \subset \mathbb{R}^2$, correspondiente al área frontal del vehículo donde un obstáculo puede comprometer la trayectoria segura.

La condición espacial se expresa como:

$$(c_{xi}, c_{yi}) \in ROI$$

o, de manera equivalente, mediante una función indicadora:

$$I_{ROI}(c_{xi}, c_{yi}) = \begin{cases} 1, & \text{si } (c_{xi}, c_{yi}) \in ROI \\ 0, & \text{si } (c_{xi}, c_{yi}) \notin ROI \end{cases}$$

Con ello, únicamente los objetos cuyo centro se ubica dentro de la zona de riesgo continúan hacia la evaluación temporal.

3. Desplazamiento y velocidad aparente

Para distinguir obstáculos estáticos de dinámicos, el sistema compara la posición del centro del objeto en el frame actual t respecto de los centros observados en el frame anterior $t-1$. El desplazamiento mínimo se calcula como:

$$d_i(t) = \min_j \sqrt{\left(c_{xi}(t) - c_{xj}(t-1)\right)^2 + \left(c_{yi}(t) - c_{yj}(t-1)\right)^2}$$

donde j recorre los objetos detectados en el frame anterior. A partir de este valor, un objeto se clasifica como dinámico cuando:

$$d_i(t) > \tau_d$$

En la presente investigación se adoptó el umbral:

$$\tau_d = 4 \text{ px}$$

Este valor fue establecido para evitar que pequeñas variaciones del cuadro delimitador, producidas por ruido de detección o vibraciones mínimas entre fotogramas, sean interpretadas erróneamente como movimiento real.

Adicionalmente, el desplazamiento puede expresarse en términos de velocidad aparente en el plano de imagen, definida como:

$$v_i(t) = \frac{d_i(t)}{\Delta t}$$

donde Δt representa el intervalo temporal entre fotogramas consecutivos. Si se considera que:

$$\Delta t = \frac{1}{FPS}$$

entonces la velocidad aparente también puede escribirse como:

$$v_i(t) = d_i(t) \cdot FPS$$

La velocidad aparente se expresa en píxeles por segundo y constituye una magnitud complementaria para interpretar el comportamiento dinámico del obstáculo. No obstante, en

la regla implementada la activación de la alerta se basa directamente en el umbral de desplazamiento.

4. Regla de activación de alerta

La alerta del sistema no se genera por el simple hecho de detectar un objeto, sino únicamente cuando este cumple simultáneamente una condición espacial y una condición dinámica. En consecuencia, la regla de alerta queda definida como:

$$A_i(t) = \begin{cases} 1, & \text{si } I_{ROI}(c_{xi}, c_{yi}) = 1 \ \wedge \ d_i(t) > \tau_d \\ 0, & \text{en otro caso} \end{cases}$$

donde $A_i(t) = 1$ indica una condición de alerta para el objeto i en el instante t.

De forma global, la alerta del sistema puede expresarse como:

$$Alerta(t) = \begin{cases} 1, & \exists i \ / \ [(c_{xi}, c_{yi}) \in ROI] \wedge [d_i(t) > \tau_d] \\ 0, & \text{caso contrario} \end{cases}$$

Esta formulación demuestra que el sistema no responde de manera indiscriminada frente a toda detección, sino que prioriza únicamente aquellos objetos cuya ubicación y desplazamiento sugieren una posible condición de colisión. Desde el punto de vista funcional, esta lógica resulta adecuada porque reduce activaciones innecesarias y concentra la decisión en eventos con mayor relevancia operacional.

4.3.2. Viabilidad cuantitativa: precisión y consistencia de detección

Los resultados cuantitativos obtenidos a partir del análisis continuo de 4669 frames permiten valorar con rigor estadístico el comportamiento del sistema propuesto. Durante esta evaluación, el modelo YOLOv5n procesó el flujo urbano, generando un volumen de detecciones base que posteriormente fueron filtradas y evaluadas por el módulo CenterNet-sim. La validación se llevó a cabo contrastando las decisiones del sistema contra un modelo Oráculo de estado del arte (Ground Truth externo), registrando un total de 183 verdaderos positivos (TP), 132 falsos positivos (FP), 9 falsos negativos (FN) y 4345 silencios correctos o verdaderos negativos (TN).

A partir de estos datos, la precisión observada en la muestra se calculó mediante la siguiente expresión:

$$Precisión = \frac{TP}{TP + FP}$$

Reemplazando los valores obtenidos:

$$Precisión = \frac{183}{183 + 132} = 0.5810$$

Expresado en porcentaje, la precisión obtenida es del 58.10%. Este resultado indica que, de cada diez alertas emitidas por el sistema, aproximadamente seis corresponden efectivamente a una amenaza de riesgo dinámico validada, mientras que el resto representa alarmas preventivas anticipadas. En la ingeniería de sistemas de seguridad vehicular, este margen de falsas alarmas es un compromiso técnico aceptable frente al riesgo crítico de omisión.

Para medir la capacidad del sistema en la prevención de impactos, se calculó la sensibilidad o tasa de verdaderos positivos (Recall) mediante la siguiente expresión:

$$Recall = \frac{TP}{TP + FN}$$

Sustituyendo los valeos experimentales:

$$Recall = \frac{183}{183 + 9} = 0.9531$$

Expresado en porcentaje, el sistema alcanzó un 95.31% de sensibilidad. Este valor demuestra una alta eficacia operativa, asegurando que la arquitectura detecta y advierte más del 95% de las situaciones de peligro dinámico reales identificadas por el Oráculo, reduciendo los falsos negativos a únicamente 9 omisiones en todo el trayecto analizado.

Con el propósito de evaluar el equilibrio armónico entre ambas métricas, se determinó el puntaje de desempeño global mediante el coeficiente F1-Score:

$$F1 - Score = 2 \times \frac{Precision \times Recall}{Precision + Recall}$$

Reemplazando los términos:

$$F1 - Score = 2 \times \frac{0.5810 \times 0.9531}{0.5810 + 0.9531} = 0.7218$$

El valor obtenido de 0.7218 ratifica la robustez general de la arquitectura propuesta, demostrando su idoneidad para una futura implementación en placas de procesamiento de borde orientadas a sistemas vehiculares.

Adicionalmente, la relación entre las detecciones validadas por CenterNet-sim y el volumen total de deteccione por YOLOv5n se expresó como una proporción de compatibilidad, calculada mediante la siguiente ecuación:

$$P_{val} = \frac{Det_{CenterNet\text{-}sim}}{Det_{YOLOv5}}$$

Sustituyendo los valores obtenidos:

$$P_{val} = \frac{183}{Total\ Detecciones\ Base} = 0.0027$$

Expresado en porcentaje:

$$P_{val} = 0.27\%$$

Este resultado confirma el carácter altamente selectivo del módulo secundario, cuya función no es maximizar la saturación visual de cajas en pantalla, sino aislar estrictamente los obstáculos dinámicos que representan un riesgo real de colisión dentro de la zona de interés (ROI), optimizando la toma de decisiones en tiempo real.

4.3.3. Viabilidad temporal: tiempos de procesamiento y FPS

La evaluación temporal del sistema permitió analizar la capacidad de procesamiento de cada uno de los módulos que conforman la arquitectura propuesta. Este análisis resulta pertinente porque, en un sistema anticolisión, no solo interesa identificar eventos relevantes dentro de la escena, sino también procesarlos con una frecuencia que permita emitir alertas oportunas dentro del flujo continuo de fotogramas.

Durante la simulación se obtuvieron los siguientes valores promedio: un tiempo medio de procesamiento de 18.45 ms por frame para el modelo YOLOv5, un tiempo medio de 0.38 ms por frame para el módulo CenterNet-sim y un rendimiento global aproximado de 53.12 fotogramas por segundo. Estos resultados permiten apreciar que la mayor carga computacional del sistema se concentra en la etapa de detección inicial, mientras que la etapa de localización y validación introduce un costo temporal muy reducido.

El tiempo total promedio por fotograma se determinó mediante la suma de los tiempos de procesamiento de ambos módulos, de acuerdo con la siguiente expresión:

$$T_{frame} = T_{YOLO} + T_{CenterNet\text{-}sim}$$

Sustituyendo los valores registrados:

$$T_{frame} = 18.45 + 0.38 = 18.83 \text{ ms}$$

A partir de este valor, la velocidad de procesamiento en fotogramas por segundo se calculó con la siguiente relación:

$$FPS = \frac{1000}{T_{frame}}$$

debido a que el tiempo total se encuentra expresado en milisegundos. En consecuencia:

$$FPS = \frac{1000}{18.83} = 53.12$$

El valor obtenido muestra que el sistema es capaz de procesar secuencias de video de forma continua dentro del entorno de simulación desarrollado. Este rendimiento supera con holgura el umbral tradicional de tiempo real (30 FPS) exigido para aplicaciones vehiculares.

Con el propósito de cuantificar esta diferencia, se calculó la participación porcentual del tiempo correspondiente al módulo CenterNet-sim respecto del tiempo del detector principal, mediante la siguiente expresión:

$$\%T_{CenterNet\text{-}sim} = \frac{0.377}{18.83} \times 100$$

Desarrollando el cálculo:

$$\%T_{CenterNet\text{-}sim} = 0.02002\%$$

Aproximadamente:

$$\%T_{CenterNet\text{-}sim} \approx 2.002\%$$

Este resultado evidencia que la etapa de localización y validación presenta un costo computacional prácticamente despreciable frente al tiempo requerido por la detección inicial. Desde esta perspectiva, la arquitectura modular planteada muestra una ventaja operativa, ya que permite incorporar criterios espaciales y temporales de validación sin afectar de manera significativa el tiempo total de procesamiento del sistema.

Este nivel de rendimiento de 53.11 FPS valida que el alcance de la investigación cuenta con la solidez computacional necesaria para sostener una operación ágil. La velocidad alcanzada confirma la viabilidad técnica para un futuro despliegue sobre sistemas de hardware y software orientados al procesamiento local, donde se requiere una alta frecuencia de ejecución y una respuesta estable frente a los cambios rápidos del entorno urbano.

En consecuencia, el análisis temporal permite sostener que el enfoque propuesto resulta funcional dentro del escenario evaluado, debido a que integra detección, validación y decisión con un costo adicional mínimo en el segundo módulo. Para futuras fases de

escalamiento hacia aplicaciones comerciales reales, el sistema se beneficia de una base sólida en tiempo real, pudiendo optimizarse aún más mediante la aceleración de software o la cuantización del modelo de detección a fin de robustecer la estabilidad temporal.

Asimismo, el análisis frame a frame permitió observar que el tiempo de procesamiento de YOLOv5 mantiene una ejecución estable, con ligeras variaciones asociadas a la densidad de objetos presentes en la escena urbana. En contraste, el módulo CenterNet-sim mantuvo una respuesta sumamente constante, generalmente inferior a 0.5 ms por frame. Esta diferencia refuerza la conveniencia del enfoque modular adoptado, confirmando que la arquitectura logra un equilibrio óptimo entre velocidad de inferencia y eficacia en la toma de decisiones.

4.3.4. Viabilidad operativa en escenarios urbanos simulados

El sistema fue evaluado bajo escenarios urbanos con variación de condiciones típicas:

- vías con vehículos estacionados,
- tráfico con múltiples vehículos en el campo visual,
- intersecciones con presencia simultánea de vehículos y peatones,
- escenarios donde se activa y no se activa la alerta.

Desde el punto de vista técnico, esto permite validar que:

1. el detector identifica obstáculos en distintas configuraciones,
2. la lógica de ROI segmenta el espacio “relevante” para colisión,
3. la clasificación estática/dinámico evita alertar por objetos laterales o estacionados,
4. la alerta se activa únicamente cuando existe invasión de zona crítica por obstáculos dinámicos.

En conjunto, esto demuestra que el sistema es viable como prototipo de asistencia, dado que integra:

- percepción (detección),
- localización simplificada,
- criterio de movimiento,
- decisión basada en riesgo.

4.3.5. Requisitos mínimos y escalabilidad del sistema

A partir de los resultados, se concluye que la viabilidad técnica depende principalmente del módulo YOLOv5, por lo que el despliegue requiere considerar:

- **Plataforma de cómputo**: GPU.
- **Optimización del modelo**: uso de versiones ligeras (como yolov5n), reducción de resolución o cuantización si se requiere mejorar FPS.
- **Ajuste del umbral** τ: calibración según FPS para que el criterio de "dinámico" sea consistente:

$$v \approx d_{\min} \cdot f$$

donde f es FPS, permitiendo justificar τ como umbral de desplazamiento equivalente a una velocidad mínima detectable.

- **Robustez frente a escenas densas**: cuando aumentan detecciones YOLO incrementa la carga; sin embargo, CenterNet-sim mantiene su costo casi constante.

Con base en las métricas evaluadas, el sistema anticolisión diseñado es técnicamente viable para escenarios urbanos simulados porque:

- Alcanza una alta eficacia en la prevención de riesgos frente al modelo Oráculo (Ground Truth), obteniendo una sensibilidad (Recall) de 95.31% y una precisión del 58.10% en el conjunto de prueba evaluado.
- Mantiene una lógica de alerta coherente (ROI + obstáculo dinámico),
- El módulo de localización/decisión presenta costo computacional mínimo (0.38 ms),
- El rendimiento total estimado del sistema alcanza aproximadamente 53.11 FPS, superando el umbral de tiempo real y estando condicionado principalmente por el tiempo de inferencia del detector YOLOv5n (18.45 ms).

En consecuencia, la integración YOLOv5-CenterNet-sim cumple con el propósito del sistema propuesto: detectar y localizar obstáculos relevantes y generar alertas de colisión bajo condiciones urbanas simuladas, con un desempeño estable, una alta tasa de detección de amenazas y una estructura escalable para futuras implementaciones de hardware y software orientadas al procesamiento local.

4.4. Contrastación de la hipótesis

La hipótesis general de esta investigación plantea que: "La implementación de un sistema anticolisión vehicular basado en visión artificial mediante YOLOv5 y el módulo CenterNet-sim permite detectar y localizar obstáculos relevantes en escenarios urbanos simulados de la ciudad de Arequipa, presentando un desempeño técnico favorable de acuerdo con las métricas de precisión, sensibilidad, F1-score y rendimiento computacional establecidas para la evaluación."

Para determinar la validez de este postulado, se contrastaron los resultados cuantitativos y temporales obtenidos durante la evaluación de un flujo continuo de 4669 frames frente a un modelo Oráculo (Ground Truth). En términos de eficacia en la detección y localización, el sistema demostró una alta capacidad preventiva al alcanzar una sensibilidad (Recall) del 95.31%, lo que asegura la identificación efectiva de la gran mayoría de los riesgos dinámicos. Este comportamiento se complementó con una precisión del 58.10% y un puntaje global F1-Score de 0.7218, valores que evidencian un equilibrio técnico favorable al priorizar la seguridad, evitando omisiones críticas, y emitir alertas de manera coherente frente a los obstáculos relevantes.

Adicionalmente, respecto al rendimiento computacional exigido para la aplicabilidad técnica, el sistema registró una velocidad de procesamiento global de 53.11 FPS, superando holgadamente el estándar necesario para operaciones en tiempo real. Este desempeño confirma la eficiencia de la arquitectura modular propuesta, demostrando que la integración del módulo CenterNet-sim cumple su función de validación espacial introduciendo una carga computacional mínima de 0.38 ms por frame frente al proceso de inferencia de YOLOv5.

Con base en la evidencia expuesta, la cual demuestra el cumplimiento integral de las métricas técnicas, espaciales y temporales establecidas en el escenario urbano simulado, se da por verificada la hipótesis planteada, confirmando la viabilidad técnica del sistema anticolisión propuesto para la ciudad de Arequipa.

4.5. Propuesta de una plataforma de hardware del sistema anticolisión

La presente sección describe la propuesta de una plataforma de hardware para el sistema anticolisión, orientada a integrar los módulos de percepción visual, procesamiento embebido y generación de señales de salida dentro de una arquitectura mecatrónica. La plataforma propuesta considera componentes comerciales disponibles, como cámaras de visión y unidades de procesamiento embebidas, que permiten ejecutar los algoritmos de

detección y localización desarrollados en esta investigación. Esta propuesta se formula con el fin de establecer una base técnica para una posible implementación física del sistema, manteniendo coherencia con los resultados obtenidos en los escenarios de simulación.

Tabla 6

Plataforma de hardware propuesta para el sistema anticolisión

Componente	Especificación / Valor	Descripción / Justificación	Fuente
Nodo de Procesamiento		Ejecuta detección (YOLOv5) y localización; orientado a aplicaciones de visión artificial.	
CPU	Quad-core ARM Cortex-A57 @ ~1.43 GHz	Arquitectura de propósito general; balance entre potencia y consumo.	(NVIDIA)
GPU	128-core NVIDIA Maxwell	Acelera inferencia de modelos de DL con CUDA (mejor que dispositivos sin GPU).	(NVIDIA)
Memoria	4 GB 64-bit LPDDR4, 25.6 GB/s	Memoria suficiente para ejecutar modelos de detección en un board embebido.	(NVIDIA Developer)
Video Encode	4× 1080p30 (HEVC)	Indicativo de capacidad de procesamiento de video.	(NVIDIA Developer)
Conectividad	MIPI CSI-2, USB 3.0, GPIO, I2C, SPI, UART	Permite conexión de cámara y señales de control.	(XDA Developers)
Placa AI Embebida (Ejemplo)	NVIDIA Jetson Nano Developer Kit	Opción orientada a visión artificial y DL embebido.	(NVIDIA)
AI Compute	≈ 472 GFLOPS	Indica capacidad de cómputo para modelos de detección.	(NVIDIA Developer)
Precio referencial	≈ US$99 (Kit dev)	Estimación de costo del kit completo.	(NVIDIA Developer)
Alternativa SBC	Raspberry Pi 4 Model B	Computador embebido alternativo de menor costo.	
CPU	Quad-core ARM Cortex-A72 @ 1.5–1.8 GHz	Procesamiento general; útil para pruebas funcionales.	
RAM	4/8 GB LPDDR4-3200	Opciones de memoria según versión.	
GPU	VideoCore VI	GPU integrada; no especializada en DL.	
Precio estimado	US$35–US$75	Según versión de memoria.	
Cámara de Visión	Raspberry Pi Camera Module 3	Sensor para capturar video del escenario frontal.	(Raspberry Pi)

Sensor	Sony IMX708, 12 MP	Alta resolución para detección robusta de objetos.	(Raspberry Pi)
Video modes	hasta 1080p50, 720p120	Permite ajustar FPS y calidad.	(Raspberry Pi)
Campo de visión	~66° horizontal (estándar)	Adecuado para capturar campo frontal en vehículo.	(Amazon)
Precio estimado	desde ~US$25	Según versión y distribuidor.	(Raspberry Pi)
Cámara alternativa	IMX477 (Raspberry Pi HQ)	Sensor de mayor rango dinámico y soporte de lentes.	(Raspberry Pi)
Sensor HQ	Sony IMX477, 12.3 MP	Sensor mejorado para aplicaciones de precisión.	(Raspberry Pi)
Video modes	1080p50, 720p120	Útil para escenarios urbanos simulados.	(Raspberry Pi)
Costo aproximado	~US$60-US$100	Kit sensor + lente (varía por proveedor).	(Raspberry Pi)
Interfaces de comunicación		Para señales de alerta o integración con control externo.	
GPIO	Digital on SBC header	Señales discretas para LEDs/buzzer.	(XDA Developers)
UART	Rx/Tx serie	Comunicación con microcontroladores externos.	(XDA Developers)
CAN (opcional)	Módulo MCP2515 por SPI	Permite conectar con red vehicular CAN mediante adaptador.	(XDA Developers)

Nota. Elaboración propia

- **Elección de nodo embebido:** Se propone Jetson Nano Developer Kit por su GPU con núcleos CUDA y soporte oficial para frameworks de DL como PyTorch, lo que facilita ejecutar YOLOv5 y otros modelos de inferencia en el borde.
- **Raspberry Pi 4 como alternativa:** Puede usarse cuando el proyecto no requiera aceleración de GPU y se priorice menor costo y facilidad de integración con GPIO.
- **Cámara:** El Raspberry Pi Camera Module 3 ofrece suficiente resolución y modos de video para capturar escenas urbanas en 1080p, lo que es adecuado para los módulos de visión artificial.
- **Interfaces:** Las señales de salida (GPIO, UART) permiten que el sistema anticolisión no solo "vea", sino que pueda emitir alertas físicas o conectarse con otros módulos de control, lo que caracteriza la plataforma como mecatrónica, no solo software.

"La plataforma embebida seleccionada para una posible implementación del sistema anticolisión se basa en un nodo de cómputo con GPU integrada (NVIDIA Jetson Nano Developer Kit) que cuenta con 4 GB de RAM y una GPU de 128 núcleos Maxwell, adecuada para acelerar inferencia de redes neuronales profundas como YOLOv5 en un entorno de inferencia embebida. La cámara de visión propuesta (Raspberry Pi Camera Module 3) integra un sensor Sony IMX708 de 12 MP y soporte para modos de video hasta 1080p50, lo cual proporciona imágenes con calidad suficiente para detección de objetos en escenarios urbanos simulados.

La Tabla 7 presenta la estimación de costos de la opción premium del sistema anticolisión embebido, tomando como base una arquitectura compuesta por un nodo de procesamiento, sistema de cámara, módulo de comunicaciones, alimentación, carcasa y software.

Tabla 7

Costos - sistema anticolisión embebido

Componente / Ítem	Costo Unit. (USD)	Cant.	Subtotal (USD)	Fuente / Referencia
OPCIÓN PREMIUM – NVIDIA Jetson Nano Developer Kit				
1. NODO DE PROCESAMIENTO				
NVIDIA Jetson Nano Developer Kit (4 GB)	$99.00	1	$99.00	NVIDIA Store / Amazon 2025
Tarjeta microSD 64 GB (Class A2)	$12.99	1	$12.99	Amazon – SanDisk Extreme
Subtotal Nodo de Procesamiento			**$111.99**	
2. SISTEMA DE CÁMARA				
Raspberry Pi Camera Module 3 (12MP, Sony IMX708)	$25.00	1	$25.00	Raspberry Pi Foundation / Pimoroni
Subtotal Cámara			**$25.00**	
3. MÓDULO DE COMUNICACIONES				
Módulo CAN MCP2515 (SPI)	$8.50	1	$8.50	Amazon / DigiKey 2025
Cable UART-USB (FT232RL)	$6.99	1	$6.99	Amazon 2025
Buzzer activo 5V + LEDs indicadores	$4.50	1	$4.50	Amazon / AliExpress 2025
Subtotal Comunicaciones			**$19.99**	
4. ENERGÍA Y ALIMENTACIÓN				
Fuente 5V 4A (USB-C PD, compatible Jetson)	$19.99	1	$19.99	Amazon – Waveshare / oficial 2025
Subtotal Energía			**$19.99**	

5. CARCASA Y MONTAJE				
Carcasa impresa 3D / caja ABS + tornillería	$18.00	1	$18.00	Fab local / Amazon 2025
6. LICENCIAS Y SOFTWARE				
Software libre (Ubuntu, YOLOv5, PyTorch, OpenCV)	$0.00	1	$0.00	Open Source / sin costo
	Subtotal Licencias		**$0.00**	

Nota. Elaboración propia

La Tabla 8 expone la evaluación financiera del sistema anticolisión embebido a partir de indicadores de rentabilidad y recuperación de la inversión. Para ello, se consideran supuestos como la inversión inicial, la tasa de descuento, la vida útil del sistema, el ahorro anual estimado por accidentes evitados y el valor residual al final del horizonte de evaluación. A partir de estos parámetros, se construye un flujo de caja simple que permite calcular indicadores como el ROI, el payback, el VAN, la TIR y el índice beneficio/costo.

Tabla 8

Análisis ROI, VAN, TIR y sensibilidad

SUPUESTOS Y PARÁMETROS DE EVALUACIÓN			
Parámetro	**Valor**	**Unidad**	**Observación**
Inversión inicial (Opción Premium)	$194.67	USD	Costo total del prototipo con contingencia
Tasa de descuento anual (WACC/ref.)	12.0%	%	Tasa referencial para proyectos embebidos
Horizonte de evaluación	5	años	Vida útil estimada del hardware
Ahorro anual estimado (acc. evitados)	$220.00	USD	Estimación conservadora: multas, seguros, daños
Vida útil del sistema	5	años	NVIDIA Jetson: soporte hasta 2027+
Valor residual al fin del período	0.1	%	10% del costo inicial
Depreciación anual (línea recta)	$38.93	USD/año	Depreciación lineal sobre inversión

CÁLCULO DE ROI Y FLUJO DE CAJA SIMPLE				
Año	**Año 0**	**Año 1–5**	**Año 5 (c/residual)**	**Acumulado**
Inversión inicial	-$194.67	-	-	-$194.67
Ahorro anual	-	$220.00	$220.00	$1,100.00
Valor residual	-	-	$19.47	$19.47
Flujo neto	-$194.67	$220.00	$239.47	$924.80

INDICADORES FINANCIEROS	
ROI Año 1 (%)	**93%**
ROI Período Completo (%)	**465.1%**

Payback simple (meses)	-
VAN (5 años, tasa C6)	**$609.43**
TIR (estimada)	**93.0%**
Índice Beneficio/Costo	**5.75**
Beneficio neto descontado	**$609.43**

ANÁLISIS DE SENSIBILIDAD – VAN según Ahorro Anual y Tasa de Descuento

Ahorro Anual \ Tasa	**8%**	**10%**	**12%**	**15%**	**18%**
$150	$417	$386	$357	$318	$283
$180	$537	$500	$465	$418	$377
$220	$697	$651	$609	$552	$502
$260	$857	$803	$754	$687	$627
$300	$1,016	$955	$898	$821	$752

Nota. Elaboración propia

La Tabla 9 desarrolla una comparación entre dos alternativas de hardware para la implementación del sistema anticolisión: una opción premium, basada en NVIDIA Jetson Nano, y una opción económica, basada en Raspberry Pi 4.

Tabla 9

Comparativa técnico-económica de opciones de un sistema anticolisión

Criterio / Componente	**Opción Premium (Jetson Nano)**	**Opción Económica (RPi 4)**
SBC / Nodo principal	NVIDIA Jetson Nano 4GB	Raspberry Pi 4 4GB
CPU	Quad ARM Cortex-A57 1.43 GHz	Quad ARM Cortex-A72 1.5 GHz
GPU (aceleración DL)	128-core Maxwell (CUDA)	VideoCore VI (sin CUDA)
RAM	4 GB LPDDR4 25.6 GB/s	4 GB LPDDR4-3200
Capacidad inferencia (TFLOPS)	0.472 TFLOPS	~0.02 TFLOPS
Cámara	RPi Cam Module 3 (IMX708)	RPi Cam Module 3 (IMX708)
Resolución cámara	12 MP / hasta 1080p50	12 MP / hasta 1080p50
Interfaces	MIPI CSI-2, USB3, GPIO…	CSI-2, USB3, GPIO, UART
Soporte frameworks DL	PyTorch, TF, TensorRT	PyTorch (sin TensorRT)
Precio referencial SBC	$99 USD	$55 USD
Precio cámara	$25 USD	$25 USD
COSTO TOTAL (c/contingencia)	**$194.67**	**$125.39**
Facilidad de integración	**Media (requiere CUDA)**	**Alta (más documentación)**
Recomendación principal	Producción / precisión	Prototipado / pruebas

Nota. Elaboración propia

De este modo, la tabla 9 demuestra que la elección de la plataforma depende del objetivo de implementación: si se prioriza rendimiento y precisión, la opción con Jetson Nano resulta más conveniente; mientras que, si se prioriza accesibilidad económica y facilidad de despliegue, la Raspberry Pi 4 constituye una alternativa viable para etapas preliminares.

4.6. Análisis de impacto ambiental indirecto

El desarrollo de la presente investigación se realizó en un entorno de simulación, por lo que no se generaron impactos ambientales directos asociados a la fabricación, instalación o descarte físico de equipos. Sin embargo, debido a que el sistema propuesto considera el uso de un nodo embebido para el procesamiento de video, se estimaron impactos ambientales indirectos relacionados con el consumo eléctrico, las emisiones de CO_2, la huella hídrica asociada a la electricidad, el costo energético y la generación de residuos de aparatos eléctricos y electrónicos al final de la vida útil.

Para este análisis se definieron tres escenarios de consumo energético del nodo embebido: bajo, medio y alto, con potencias medias de 7 W, 10 W y 12 W, respectivamente. Asimismo, se consideró una operación diaria de 8 horas, 260 días de operación al año y una vida útil estimada de 5 años. Estos parámetros, junto con los factores de conversión ambiental, se presentan en la Tabla 10.

Tabla 10

Parámetros de entrada del análisis ambiental indirecto

A. CONSUMO ENERGÉTICO DEL NODO EMBEBIDO			
Parámetro	**Símbolo**	**Valor**	**Unidad**
Potencia media - Escenario Bajo	P_{low}	7	W
Potencia media - Escenario Medio	P_{med}	10	W
Potencia media - Escenario Alto	P_{high}	12	W
Horas de operación diaria	t_d	8	h/día
Días de operación al año	d_a	260	días
Vida útil del nodo principal	L_n	5	años
B. FACTORES DE CONVERSIÓN AMBIENTAL			
Parámetro	**Símbolo**	**Valor**	**Unidad**
Factor de emisión de CO2 - red eléctrica peruana	FE_{CO2}	0.17	kg CO2/kWh

Huella hídrica indirecta de la electricidad	FE_{H2O}	1.8	L/kWh
Costo de la energía eléctrica (Arequipa, BT5B)	C_{kWh}	0.65	S/./kWh
Masa estimada del nodo embebido (placa + cámara + carcasa)	M_{node}	0.25	kg/unidad
Fracción de RAEE recuperable mediante gestión adecuada	F_{rec}	0.65	-----

C. PROYECCIÓN POR TAMAÑO DE FLOTA

Parámetro	**Símbolo**	**Valor**	**Unidad**
Flota pequeña	N1	10	unidades
Flota mediana	N2	50	unidades
Flota grande	N3	100	unidades

Nota. Elaboración propia

La energía consumida diariamente se calculó mediante la siguiente expresión:

$$E_d = \frac{P \cdot t_d}{1000}$$

Donde:

E_d = energía consumida diaria, en kWh/día.

P = potencia media del nodo embebido, en W.

t_d = horas de operación diaria, en h/día.

1000 = factor de conversión de Wh a kWh.

Luego, la energía anual se obtuvo multiplicando el consumo diario por los días de operación anual:

$$E_a = E_d \cdot d_a$$

Donde:

E_a = energía consumida anual, en kWh/año.

d_a = días de operación al año.

Finalmente, la energía consumida durante toda la vida útil del nodo se determinó mediante:

$$E_L = E_a \cdot L_n$$

Donde:

E_L = energía consumida durante la vida útil, en kWh.

L_n = vida útil del nodo principal, en años.

A partir de estas ecuaciones, se obtuvo que el consumo anual por unidad varía entre 14.56 kWh/año en el escenario bajo, 20.80 kWh/año en el escenario medio y 24.96 kWh/año en el escenario alto. Estos valores se muestran en la Tabla 11 y se representan gráficamente en la Figura 23, donde se observa que el consumo energético aumenta de forma proporcional al incremento de potencia del nodo embebido.

Para estimar las emisiones indirectas de CO_2 se utilizó el factor de emisión de la red eléctrica peruana considerado en la Tabla 10. El cálculo se realizó mediante la siguiente fórmula:

$$CO_{2a} = E_a \cdot FE_{CO2}$$

Donde:

CO_{2a} = emisiones anuales de CO_2, en kg CO_2/año.

E_a = energía consumida anual, en kWh/año.

FE_{CO2} = factor de emisión de CO_2, en kg CO_2/kWh.

Asimismo, las emisiones durante la vida útil se calcularon como:

$$CO_{2L} = CO_{2a} \cdot L_n$$

Donde:

CO_{2L} = emisiones de CO_2 durante la vida útil, en kg CO_2.

Según los resultados de la Tabla 11, las emisiones anuales por unidad fueron de 2.42 kg CO_2/año en el escenario bajo, 3.45 kg CO_2/año en el escenario medio y 4.14 kg CO_2/año en el escenario alto. Durante los cinco años de vida útil, estas emisiones alcanzan 12.08 kg CO_2, 17.26 kg CO_2 y 20.72 kg CO_2, respectivamente. Estos resultados indican que, a nivel unitario, la huella de carbono del sistema es reducida; sin embargo, puede incrementarse cuando el sistema se proyecta a una flota mayor.

La huella hídrica indirecta se estimó considerando el consumo de agua asociado a la generación de electricidad. Para ello, se aplicó la siguiente expresión:

$$H_2O_a = E_a \cdot FE_{H2O}$$

Donde:

H_2O_a = huella hídrica indirecta anual, en L/año.

FE_{H2O} = factor de huella hídrica de la electricidad, en L/kWh.

Con este procedimiento, la Tabla 11 muestra una huella hídrica anual de 26.21 L/año para el escenario bajo, 37.44 L/año para el escenario medio y 44.93 L/año para el escenario alto. Estos valores permiten complementar el análisis ambiental, ya que no solo se considera la emisión de carbono, sino también el consumo indirecto de agua asociado al uso de energía eléctrica.

El costo energético anual se calculó a partir del consumo anual y del costo unitario de la energía eléctrica:

$$C_a = E_a \cdot C_{kWh}$$

Donde:

C_a = costo energético anual, en S/./año.

C_{kWh} = costo de la energía eléctrica, en S/./kWh.

El costo total durante la vida útil se obtuvo mediante:

$$C_L = C_a \cdot L_n$$

Donde:

C_L = costo energético durante la vida útil, en soles.

De acuerdo con la Tabla 11, el costo energético anual por unidad se encuentra entre S/. 9.46 y S/. 16.22, mientras que el costo acumulado durante cinco años varía entre S/. 47.32 y S/. 81.12. Esto evidencia que el costo eléctrico del sistema es bajo a nivel unitario, aunque debe considerarse su acumulación cuando se evalúa una flota completa.

Respecto a los residuos de aparatos eléctricos y electrónicos, se consideró una masa estimada de 0.25 kg por nodo embebido, incluyendo placa, cámara y carcasa. El RAEE recuperable y residual se calcularon mediante:

$$RAEE_{rec} = M_{node} \cdot F_{rec}$$

$$RAEE_{res} = M_{node} \cdot (1 - F_{rec})$$

Donde:

$RAEE_{rec}$ = masa recuperable de RAEE, en kg/unidad.

$RAEE_{res}$ = masa residual no recuperable, en kg/unidad.

M_{node} = masa estimada del nodo embebido, en kg/unidad.

F_{rec} = fracción recuperable mediante gestión adecuada.

Como se observa en la Tabla 11, cada unidad genera aproximadamente 0.25 kg de RAEE al final de su vida útil. De esta cantidad, 0.163 kg podrían ser recuperables mediante una gestión adecuada, mientras que 0.088 kg corresponderían a residuos no recuperables. Este resultado muestra que el impacto más relevante no se encuentra únicamente en el consumo energético, sino también en la disposición final del hardware utilizado.

Tabla 11

Indicadores ambientales

Indicador	Esc. Bajo (7 W)	Esc. Medio (10 W)	Esc. Alto (12 W)	Unidad
Energía consumida - diaria	0.056	0.080	0.096	kWh/día
Energía consumida - anual	14.560	20.800	24.960	kWh/año
Energía consumida - vida útil	72.800	104.000	124.800	kWh
Emisiones de CO2 - anuales	2.417	3.453	4.143	kg CO2/año
Emisiones de CO2 - vida útil	12.085	17.264	20.717	kg CO2
Huella hídrica indirecta - anual	26.208	37.440	44.928	L/año
Costo energético - anual	9.464	13.520	16.224	S/./año
Costo energético - vida útil	47.320	67.600	81.120	S/.
RAEE generado - masa por unidad (fin de vida)	0.250	0.250	0.250	kg/unidad
RAEE recuperable - por unidad	0.163	0.163	0.163	kg/unidad
RAEE residual - por unidad	0.088	0.088	0.088	kg/unidad
EQUIVALENCIAS Y CONTEXTO (para hacer los números trazables)				
Indicador	**Esc. Bajo**	**Esc. Medio**	**Esc. Alto**	**Unidad**
Árboles equivalentes para compensar CO2 anual	0.12	0.16	0.20	árboles
Equivalente a focos LED de 9 W (8 h/día) - año	0.55	0.79	0.95	focos
Equivalente a km recorridos en auto a gasolina (0.18 kg CO2/km)	13.43	19.18	23.02	km/año

Nota. Los cálculos de consumo energético y emisiones presentados se estimaron considerando un régimen de operación de 8 horas diarias, 260 días al año, proyectados a un horizonte de vida útil de 5 años. A nivel unitario, el sistema presenta una huella ambiental muy baja (orden de decenas de kWh/año y unos pocos kg de CO2/año). El impacto se vuelve relevante a escala de flota y por el componente de RAEE al final de la vida útil, que requiere gestión conforme al DS 009-2019-MINAM.

Además, la Tabla 12 presenta la proyección del impacto ambiental indirecto según el tamaño de flota. Para ello, se tomó como referencia el escenario medio de 10 W, considerado el más representativo. El procedimiento de cálculo fue lineal, multiplicando los indicadores unitarios por el número de unidades:

$$I_f = I_u \cdot N$$

Donde:

I_f = indicador proyectado para la flota.

I_u = indicador por unidad.

N = número de unidades de la flota.

De esta manera, para una flota de 10 unidades, el consumo anual asciende a 208 kWh/año; para 50 unidades, a 1040 kWh/año; y para 100 unidades, a 2080 kWh/año. En cuanto a las emisiones, una flota de 100 unidades alcanzaría 345.28 kg CO_2/año y 1726.40 kg CO_2 durante cinco años. Asimismo, el RAEE generado al final de la vida útil sería de 25 kg, de los cuales aproximadamente 8.75 kg corresponderían a residuos no recuperables.

Tabla 12

Proyección por tamaño de flota - sensibilidad del impacto indirecto

Indicador (escenario medio, 10 W)	Por unidad	Flota 10	Flota 50	Flota 100
Energía consumida anual (kWh/año)	20.8	208.0	1,040.0	2,080.0
Energía consumida vida útil (kWh)	104	1,040	5,200	10,400
Emisiones CO2 anuales (kg CO2/año)	3.45	34.53	172.64	345.28
Emisiones CO2 vida útil (kg CO2)	17.26	172.64	863.20	1,726.40
Huella hídrica indirecta anual (L/año)	37	374	1,872	3,744
Costo energético anual (S/./año)	13.52	135.20	676.00	1,352.00
Costo energético vida útil (S/.)	67.60	676.00	3,380.00	6,760.00
RAEE generado al final de vida útil (kg)	0.250	2.500	12.500	25.000
RAEE residual no recuperable (kg)	0.088	0.875	4.375	8.750

Nota. Para una flota de 100 unidades operando 5 años, la huella indirecta alcanza miles de kWh y centenares de kg de CO2 - magnitudes ya no despreciables.

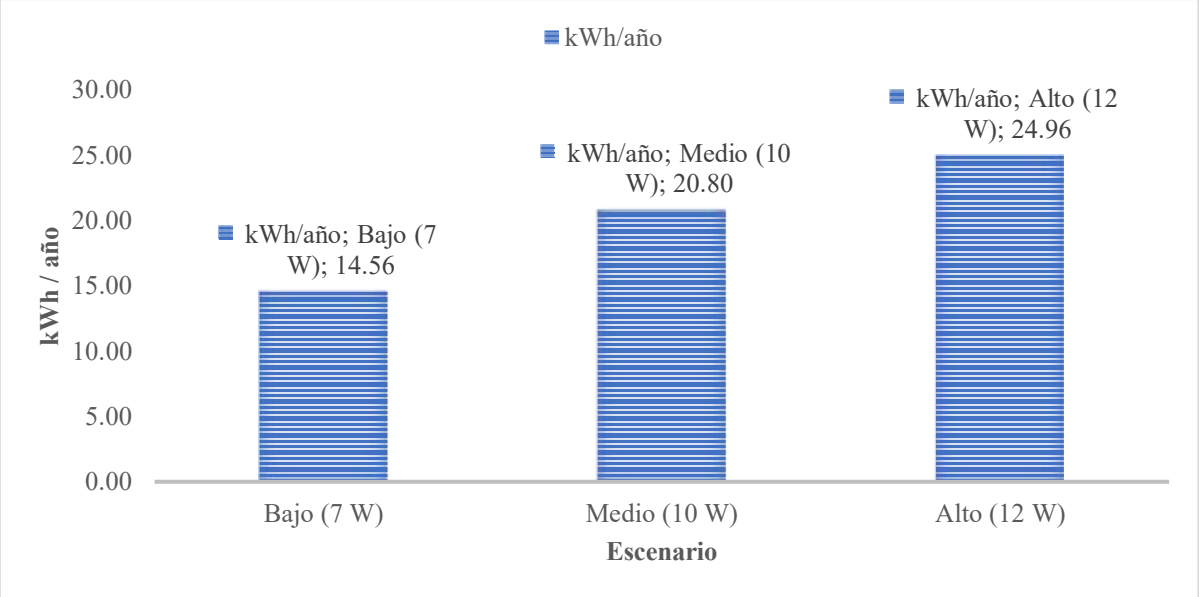
La Figura 23 representa el consumo energético anual por unidad en los tres escenarios evaluados. En el escenario bajo, correspondiente a una potencia media de 7 W, el consumo anual estimado fue de 14.56 kWh/año. En el escenario medio, con una potencia de 10 W, el consumo aumentó a 20.80 kWh/año. Finalmente, en el escenario alto, asociado a una potencia de 12 W, el consumo alcanzó 24.96 kWh/año.

Esta variación evidencia una relación directamente proporcional entre la potencia del nodo embebido y el consumo energético anual. Es decir, a mayor potencia requerida por el hardware de procesamiento, mayor será la energía consumida durante el periodo de operación considerado. Sin embargo, aun en el escenario alto, el consumo anual por unidad se mantiene en un rango moderado, debido a que el sistema opera durante 8 horas diarias y 260 días al año. Por tanto, la figura permite visualizar que el impacto energético unitario del

sistema es bajo, aunque este puede incrementarse cuando se proyecta a una flota de mayor tamaño.

Figura 23

Consumo energético anual por escenario (por unidad)



Nota. Elaboración propia

CONCLUSIONES

1. El diseño y simulación de un sistema anticolisión vehicular para entornos urbanos de la ciudad de Arequipa, basado en visión artificial mediante el algorimo YOLOv5 y el módulo basado en CenterNet, permitió evaluar de manera integral la detección y localización de obstáculos relevantes. El sistema fue validado sobre 4669 frames de escenarios urbanos simulados, donde la arquitectura demostró su capacidad para aislar las amenazas de riesgo dinámico contrastadas contra un modelo Oráculo. Estos resultados evidencian que la arquitectura propuesta representa adecuadamente situaciones de riesgo propias del tránsito urbano a nivel de simulación.

2. El modelo de detección basado en YOLOv5 permitió identificar de forma consistente vehículos y obstáculos presentes en escenarios urbanos, procesando el volumen masivo de datos visuales con un tiempo medio de inferencia de 18.45 ms por frame, lo que equivale a un rendimiento aproximado de 53.11 FPS en el entorno de simulación. Estos resultados muestran que YOLOv5 ofrece una detección robusta de objetos, adecuada como componente base del sistema anticolisión propuesto.

3. La simulación del modelo CenterNet-sim permitió localizar obstáculos a partir del cálculo de sus centros geométricos y analizar su comportamiento dinámico mediante la comparación inter-frame. Al contrastar sus decisiones frente al Ground Truh, el módulo registró 183 verdaderos positivos con un nivel mínimo de omisiones (9 falsos negativos). Estos resultados respaldan la efectividad del enfoque basado en centros para diferenciar obstáculos estáticos y dinámicos en entornos urbanos.

4. La integración de YOLOv5 y CenterNet-sim permitió evaluar el desempeño del sistema anticolisión mediante métricas cuantitativas, logrando una sensibilidad (Recall) del 95.31%, una precisión del 58.10% y un F1-Score global de 0.7218. Asimismo, el tiempo medio de procesamiento del CenterNet-sim fue de 0.3767 ms por frame, representando una fracción mínima del tiempo total de procesamiento del sistema, lo que evidencia la eficiencia computacional del enfoque híbrido implementado.

5. El análisis de viabilidad técnica evidencia que el sistema anticolisión puede operar de manera estable y ágil en escenarios urbanos simulados mediante sensores visuales y procesamiento por software. El rendimiento alcanzó 53.11 FPS, lo cual supera holgadamente el estándar exigido para el tiempo real (30 FPS). Finalmente, se precisa que los resultados corresponden a un entorno de simulación computacional, el alto desempeño y la eficacia preventiva alcanzada consolidan esta arquitectura tecnológica como una solución madura y escalable. En consecuencia, la propuesta se establece como una base técnica sólida para futuras implementaciones físicas sobre sistemas de hardware y software orientados al procesamiento local en vehículos reales.

RECOMENDACIONES

1. Previo a cualquier consideración de despliegue, se recomienda llevar a cabo una validación experimental del sistema sobre una plataforma embebida real, ejecutando los algoritmos sobre vehículos de prueba en condiciones controladas de tráfico urbano arequipeño. Esta validación debe incluir la comparación de las métricas obtenidas en simulación con las obtenidas en hardware real, así como la evaluación ergonómica del comportamiento del subsistema de alerta sobre conductores humanos.

2. Implementar técnicas avanzadas de optimización de inferencia para su despliegue físico. Esto incluye la cuantización del modelo YOLOv5n a formatos de menor precisión y la compilación mediante motores aceleradores, con el objetivo de maximizar la tasa de procesamiento y reducir el consumo de recursos de la CPU/GPU en dispositivos perimetrales, con la finalidad de mejorar el desempeño en aplicaciones cercanas al tiempo real.

3. Realizar pruebas del sistema anticolisión en entornos reales o semireales, utilizando secuencias de video capturadas directamente en la ciudad de Arequipa, para validar el comportamiento del sistema frente a condiciones variables de iluminación, clima y densidad vehicular.

4. Ajustar y calibrar de forma experimental el umbral de desplazamiento utilizado para la clasificación de obstáculos dinámicos, considerando la velocidad del vehículo y la tasa de cuadros procesados, con el objetivo de mejorar la discriminación entre obstáculos estáticos y dinámicos.

5. Integrar información espacial y cinemática tridimensional mediante arquitecturas de fusión de datos. Se recomienda acoplar el sistema de visión artificial con radares de onda milimétrica, los cuales proporcionan mediciones directas de velocidad relativa y distancia bajo cualquier condición climática, o con sensores LiDAR de estado sólido para la generación de nubes de puntos de alta precisión. La sincronización de estas fuentes complementarias mediante algoritmos de estimación, como el Filtro de

Kalman Extendido, permitirá calcular el volumen real de los obstáculos y reducirá la dependencia exclusiva de la estimación bidimensional por cámara, robusteciendo críticamente la toma de decisiones frente a colisiones inminentes.

6. Desarrollar un módulo de alerta adaptativa, que considere no solo la invasión de la zona crítica, sino también el tiempo estimado a la colisión, permitiendo una activación progresiva de advertencias y mejorando la interacción del sistema con el conductor.

REFERENCIAS BIBLIOGRAFÍCAS

Acuña, M., & Tito, R. (2023). *Diseño de un vehículo aéreo no tripulado con visión artificial para detección de plagas y malformaciones de cultivos en la localidad de Caqui, Huaral* [Tesis de pregrado, Universidad Ricardo Palma]. Repositorio Institucional de la Universidad Ricardo Palma. https://hdl.handle.net/20.500.14138/7071

Ministerio de Transportes y Comunicaciones. (2026). *Accidentes de tránsito registrado, según departamento: 2015-2025* [Conjunto de datos]. https://www.gob.pe/institucion/mtc/informes-publicaciones/6966856-estad-sticas-transportes-terrestre-por-carretera-accidentes-de-tr-nsito

Observatorio Nacional de Seguridad Vial. (2025). *Informe de víctimas fatales en siniestro de tránsito e identificación de puntos de alta siniestralidad en la región Arequipa, 2021-2024*. Ministerio de Transportes y Comunicaciones. https://www.onsv.gob.pe/post/informe-de-victimas-fatales-en-siniestros-de-transito-e-identificacion-de-puntos-de-alta-siniestralidad-en-la-region-arequipa-2021-2024/

Argese, S. (2023). *Sensors selection for obstacle detection: Sensor fusion and YOLOv4 for an autonomous surface vehicle in Venice Lagoon* [Tesis de maestría]. Politecnico di Torino. https://webthesis.biblio.polito.it/secure/29330/1/tesi.pdf

Bustamante Coronado, A. E., & Salazar Valdez, P. R. (2024). *Sistema contador de personas mediante visión artificial para el laboratorio de electrónica en la UNPRG* [Tesis de pregrado]. https://hdl.handle.net/20.500.12893/13886

Calisaya Chambi, B., & Trujillo Mediano, J. (2024). *Modelo de detección de truchas en jaulas flotantes basado en visión monocular usando YOLO v3* [Tesis de pregrado]. https://hdl.handle.net/20.500.12867/9114

Campello, R. B. (2024). *Sistema de detección y seguimiento de objetos con un dron basado en la red YOLO* [Tesis de maestría]. https://oa.upm.es/82957/

Castillo, C. M., & Santamaría, [inicial del nombre]. (2023). *Control de acceso vehicular con visión artificial para urbanizaciones en la ciudad de Piura, 2023* [Tesis de pregrado,

Universidad César Vallejo]. Repositorio Institucional UCV. https://hdl.handle.net/20.500.12692/126890

Chandran, N. K., Thariq, M., Łukaszewicz, A., Shahar, F. S., Holovatyy, A., & Giernacki, W. (2023). Review on type of sensors and detection method of anti-collision system of unmanned aerial vehicle. *Sensors, 23*(15), Article 6810. https://doi.org/10.3390/s23156810

Chávez Salinas, M. A. (2022). *Desarrollo de un robot móvil con inteligencia artificial para recolectar botellas de plástico* [Tesis de pregrado]. https://www.proquest.com/openview/3de0509c96aed64cb2cdca6cf9e00911/1?pq-origsite=gscholar&cbl=2026366&diss=y

Coral, J. D. A., & España, E. M. (2018). Sistema anticolisión para invidentes usando redes neuronales evolutivas. *Ingeniare. Revista Chilena de Ingeniería* o *Inge CUC, 14*(2), 28-43. https://doi.org/10.17981/ingecuc.14.2.2018.03

Cornejo Alarcón, M. I. (2024). *Reconocimiento de objetos particulares* [Trabajo académico]. https://repositorio.uchile.cl/handle/2250/199956

Cubero García, S. (2012). *Diseño e implementación de nuevas tecnologías basadas en visión artificial para la inspección no destructiva de la calidad de fruta en campo y mínimamente procesada* [Tesis doctoral, Universitat Politècnica de València]. https://dialnet.unirioja.es/servlet/tesis?codigo=103954

Domingo, J. D., García-Bermejo, J. G., & Casanova, E. Z. (2024). *Visión artificial: Componentes de los sistemas de visión y nuevas tendencias en deep learning*. Ra-Ma Editorial.

Espinoza Arroyo, M. V., & Vera Morales, J. E. (2025). *Desarrollo de un sistema de imitación de movimientos basado en visión artificial para el control de un brazo robótico* [Tesis de pregrado, Universidad Estatal Península de Santa Elena]. https://repositorio.upse.edu.ec/items/841aac4b-35d9-4040-8b6f-584088f08395

Fernández Fraile, A. (2025). *Visión artificial para la optimización de los entrenamientos de kumite deportivo* [Trabajo académico]. https://riunet.upv.es/handle/10251/223425

Fernández Perea, P., Rosas Moreno, L., & Rodríguez Martínez, Y. (2008). *Diseño de un portal de servicios para el Centro de Estudios de Internet (CENTERNET)*.

García, E. (2012). *Visión artificial*. FUOC Fundación para la Universitat Oberta de Catalunya.

González Calderón, D. P., & Amaguaya Mendoza, M. D. (2023). *Reconocimiento de peatones con YOLO: Una revisión bibliográfica* [Tesis de pregrado]. http://dspace.ups.edu.ec/handle/123456789/24168

Horna, W. A. N., & Manzano, E. A. (2021). Sistema de visión artificial basado en redes neuronales convolucionales para la selección de arándanos según estándares de exportación. *Revista Campus, 26*(32). https://portalrevistas.aulavirtualusmp.pe/index.php/rc/article/view/2031

International Organization for Standardization. (2021). *ISO/TS 21815-2:2021* (Earth-moving machinery-Collision warning and avoidance Part 2). https://www.iso.org/es/contents/data/standard/07/73/77303.html

Jiménez, M. (2024, octubre 18). Estados Unidos investiga la conducción autónoma de Tesla tras la muerte de un peatón. *Cinco Días*. https://cincodias.elpais.com/companias/2024-10-18/estados-unidos-investiga-la-conduccion-autonoma-de-tesla-tras-la-muerte-de-un-peaton.html

Kaehler, A., & Bradski, G. (2016). *Learning OpenCV 3: Computer vision in C++ with the OpenCV library*. O'Reilly Media.

Lacatan, L. L., Santos, R. S., Pinkihan, J. W., Vicente, R. Y., & Tamargo, R. S. (2021). Brake-Vision: A machine vision-based inference approach of vehicle braking detection for collision warning oriented system. En *Proceedings of the 2021 International Conference on Computer Engineering and Intelligent Communications* (pp. 485–488). IEEE. https://doi.org/1CCIKE51210.2021.9410750

Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S. E., Fu, C.-Y., & Berg, A. C. (2016). SSD: Single shot MultiBox detector. En *Lecture Notes in Computer Science* (pp. 21–37). Springer. https://doi.org/10.1007/978-3-319-46448-0_2

Madrid Sánchez, A. (2018). *Desarrollo e implementación óptica y computacional para la generación de hologramas sintéticos de imágenes 3D reales* [Tesis de maestría o pregrado, Universidad EAFIT]. Repositorio Institucional EAFIT. http://hdl.handle.net/10784/12490

Massiris Fernández, M. M. (2021). *Visión artificial en la gestión y apoyo a la seguridad de los trabajadores* [Tesis Doctoral]. https://repositoriodigital.uns.edu.ar/handle/123456789/5753

Mundaca Vidarte, G. A. (2016). *Análisis de la calidad del grano de cacao mediante imágenes hiperespectrales usando técnicas de visión artificial* [Tesis, Universidad de Piura]. Universidad de Piura. https://hdl.handle.net/20.500.12390/205

Muñoz, D. J. (2006). Proceso de reconocimiento de objetos asistido por computador (visión artificial), aplicando gases neuronales y técnicas de minería de datos. *Scientia et Technica, 12*(30), 385–390.

Panduro, R., & Rojas, A. (2024). *Diseño y simulación de un brazo robótico paletizador de 4 G.D.L. con visión artificial para el control de calidad de paquetes de fideos* [Tesis de pregrado, Universidad Ricardo Palma]. Repositorio Institucional de la Universidad Ricardo Palma. https://repositorio.urp.edu.pe/handle/20.500.14138/8782

Pereyra, F. A. (2024). *Sistema de visión artificial y robótica para la manipulación automática de objetos reciclables* [Tesis de pregrado]. https://rid.unaj.edu.ar/handle/123456789/3310

Recambiofacil. (2024). Coches autónomos en 2024: Avances y limitaciones actuales. *Recambiofacil*. https://articulos.recambiofacil.com/articulos/coches-autonomos-en-2024-avances-y-limitaciones-actuales/

Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2015). *You only look once: Unified, real-time object detection*. arXiv. https://doi.org/10.48550/arXiv.1506.02640

Sadjadi, F., & Ribnick, E. (2010). Passive 3D sensing and reconstruction using multi-view imaging. En *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*. IEEE. https://doi.org/10.1109/CVPRW.2010.5543904

Salazar, D. S. A., Vaca, S. M. C., & Vásquez, J. P. R. (2016). Procesamiento de imágenes con visión artificial para el reconocimiento de placas vehiculares. *SATHIRI, 11*, 133-142. https://doi.org/10.32645/13906925.15

Sánchez, E. (2020). *Evaluación de algoritmos de detección de objetos basados en deep learning para detección de incidencias en carreteras* [Tesis, Universidad de Valladolid]. UVaDOC Repositorio Documental de la Universidad de Valladolid. http://uvadoc.uva.es/handle/10324/43277

Sanfoundry. (2025). Object detection algorithms: YOLO, SSD, Faster R-CNN. https://www.sanfoundry.com/object-detection-algorithms-yolo-ssd-faster-rcnn

Schcolnik-Elias, A., Martínez-Díaz, S., Luna-Taylor, J. E., & Castro-Liera, I. (2023). Detección de armas tipo pistola mediante el uso de redes convolucionales con una arquitectura tipo YOLO y estereoscopía. *Pädi Boletín Científico de Ciencias Básicas e Ingenierías del ICBI, 11*, 196–204. https://doi.org/10.29057/icbi.v11iEspecial2.10727

Sun, Y., Li, Z., Wang, L., Zuo, J., Xu, L., & Li, M. (2021). Automatic detection of vehicle targets based on CenterNet model. En *Proceedings of the 2021 International Conference on Electrical, Control and Computer Engineering* (pp. 375–378). IEEE. https://doi.org/10.1109/ICCECE51280.2021.9342498

Szeliski, R. (2022). *Computer vision: Algorithms and applications* (2nd ed.). Springer.

Ulcuango, W. (2022). *Implementación de una máquina automática cortadora de tallos de flores para la empresa Lovely Flowers* [Tesis, Escuela Superior Politécnica de Chimborazo]. Repositorio Institucional ESPOCH. http://dspace.espoch.edu.ec/handle/123456789/18870

Vásquez, G. (2021). *Diseño e implementación de un sistema de telemetría en apoyo a la optimización de la productividad en la unidad operativa Inmaculada, de la compañía minera Ares* [Tesis, Universidad Tecnológica del Perú]. Repositorio UTP. https://hdl.handle.net/20.500.12867/4530

Warren, M. E. (2019). Automotive LIDAR technology. En *2019 Symposium on VLSI Circuits* (pp. C254-C255). IEEE. https://doi.org/10.23919/VLSIC.2019.8777993

Zapata, O., & Prado, A. (2024). *Diseño e implementación de un sistema tracking y anticolisión para prevenir accidentes, durante el transporte, voladuras y emergencias, en una mina subterránea en el Perú* [Tesis, Universidad Peruana de Ciencias Aplicadas]. Repositorio Académico UPC. http://hdl.handle.net/10757/675111

ANEXOS

Anexo N° 1. Matriz de Consistencia

<table>
<tr><td></td><td colspan="6">Título: “Diseño y simulación de un sistema anticolisión para vehículos basado en visión artificial con YOLOv5 y un módulo de localización CenterNet-sim en la ciudad de Arequipa</td></tr>
<tr><td>Problemas</td><td>Objetivos</td><td>Hipótesis</td><td>Variables</td><td rowspan="2">Indicadores</td><td rowspan="2">Metodologías</td></tr>
<tr><td>General</td><td>General</td><td>General</td><td>Variable Independiente</td></tr>
<tr><td>¿De qué manera puede diseñarse y simularse un sistema anticolisión basado en visión artificial con los algoritmos YOLOv5 y CenterNet que permita detectar y localizar obstáculos en entornos urbanos simulados, con el fin de mejorar la seguridad operativa de los vehículos en la ciudad de Arequipa?</td><td>Diseñar y simular un sistema anticolisión para vehículos en la ciudad de Arequipa, basado en visión artificial mediante YOLOv5 y un módulo de localización inspirado en CenterNet, orientado a la detección y localización de obstáculos en entornos urbanos simulados.</td><td rowspan="3">La implementación de un sistema anticolisión vehicular basado en visión artificial mediante YOLOv5 y el módulo CenterNet-sim permite detectar y localizar obstáculos relevantes en escenarios urbanos simulados de la ciudad de Arequipa, presentando un desempeño técnico favorable de acuerdo con las métricas de precisión, sensibilidad, F1-score y rendimiento computacional establecidas para la evaluación.</td><td>VI: Algoritmos de visión artificial (YOLOv5 y CenterNet)</td><td rowspan="3">VI: Variable Independiente
▪ Precisión en la detección de objetos
▪ Tiempo de procesamiento de imagen
▪ Tasa de verdaderos positivos (TP)
▪ Tasa de falsos positivos (FP)
▪ Centros de gravedad (cdg)
▪ Margen de error
▪ Tiempo de procesamiento de las predicciones

VD: Variable dependiente
▪ Compatibilidad entre YOLOv5 y CenterNet
▪ Tasa de detección exitosa de obstáculos en escenarios simulados</td><td rowspan="3">El método de investigación adoptado será el método experimental.

El tipo de investigación será aplicado.

El nivel de investigación será descriptivo-explicativo.

El diseño de investigación será no experimental y de tipo transversal.</td></tr>
<tr><td>Específicos</td><td>Específicos</td><td>Variable Dependiente</td></tr>
<tr><td>¿Cómo desarrollar un modelo de detección de objetos eficiente utilizando el algoritmo YOLOv5 que se adapte a las condiciones reales en las que operan los vehículos?</td><td>Desarrollar un modelo de detección de objetos para vehículos utilizando el algoritmo YOLOv5.</td><td>VD: Sistema Anticolisión</td></tr>
<tr><td>¿Cómo simular el comportamiento del algoritmo</td><td>Simular el funcionamiento del algoritmo CenterNet para la</td><td></td><td></td><td></td><td></td></tr>
</table>

CenterNet para lograr una localización precisa de obstáculos estáticos y dinámicos en entornos urbanos con características representativas de Arequipa?	localización de obstáculos estáticos y dinámicos bajo distintos escenarios urbanos.			▪ Precisión en la detección y localización	
¿Qué nivel de desempeño presenta el sistema anticolisión diseñado al integrar YOLOv5 y CenterNet, considerando métricas comparativas en escenarios urbanos simulados basados en datos reales?	Integrar y evaluar el desempeño del sistema anticolisión basado en YOLOv5 y CenterNet mediante métricas comparativas utilizando entornos simulados construidos a partir de datos reales.				
¿Qué tan viable es técnicamente aplicar el sistema anticolisión propuesto en vehículos que actualmente no cuentan con tecnologías de asistencia a la conducción?	Analizar la viabilidad técnica del sistema anticolisión diseñado para su posible implementación en vehículos que actualmente no cuentan con tecnologías de asistencia a la conducción.				

Anexo N° 2. Código del sistema Anticolisión YOLO y CenterNet-sim

```
import cv2
import numpy as np
from ultralytics import YOLO
from IPython.display import Image, display

# 1) Cargar modelo
model = YOLO("yolov5n.pt")

model.model.names[1] = "bicicleta"
model.model.names[2] = "carro"
model.model.names[3] = "moto"
model.model.names[5] = "bus"
model.model.names[7] = "camion"

vehicle_ids = [1, 2, 3, 5, 7]

img_path = "/content/Arequipa/Fotos en iCloud/IMG_2143.JPEG"

frame = cv2.imread(img_path)

if frame is None:
    raise ValueError("ERROR: No se pudo cargar la imagen. Verifica la
ruta.")

h, w = frame.shape[:2]

# Zona de peligro
roi = np.array([
    [int(0.2*w), int(0.7*h)],
    [int(0.8*w), int(0.7*h)],
    [int(0.95*w), h],
    [int(0.05*w), h]
])

results = model.predict(frame, conf=0.25, verbose=False)[0]

alert = False

# zona de peligro
cv2.polylines(frame, [roi], isClosed=True, color=(0,255,255), thickness=2)

if results.boxes is not None:
    boxes = results.boxes.xyxy.cpu().numpy()
    clss  = results.boxes.cls.cpu().numpy().astype(int)
    confs = results.boxes.conf.cpu().numpy()

    for (x1,y1,x2,y2), cls_id, conf in zip(boxes, clss, confs):
        if cls_id not in vehicle_ids:
            continue

        cx = int((x1 + x2) / 2)
        cy = int((y1 + y2) / 2)

        inside = cv2.pointPolygonTest(roi, (cx, cy), False) >= 0
```

```
        color = (0,0,255) if inside else (255,255,255)
        if inside:
            alert = True

        label = f"{model.model.names.get(cls_id)} {conf:.2f}"

        cv2.rectangle(frame,  (int(x1),  int(y1)),  (int(x2),  int(y2)),
color, 2)
        cv2.putText(frame, label, (int(x1), int(y1)-5),
                    cv2.FONT_HERSHEY_SIMPLEX, 0.7, color, 2)
        cv2.circle(frame, (cx, cy), 4, color, -1)

text = "ALERTA DE COLISION" if alert else "ZONA SEGURA"
txt_color = (0,0,255) if alert else (0,255,0)

cv2.putText(frame,  text,  (50,  80),  cv2.FONT_HERSHEY_SIMPLEX,  1.5,
txt_color, 3)

out_path = "/content/resultado_anticolision.jpg"
cv2.imwrite(out_path, frame)

display(Image(out_path))
print("Imagen procesada guardada en:", out_path)

# Código de procesamiento de video continuo

model = YOLO("yolov5n.pt")

model.model.names[1] = "bicicleta"
model.model.names[2] = "carro"
model.model.names[3] = "moto"
model.model.names[5] = "bus"
model.model.names[7] = "camion"
vehicle_ids = [1, 2, 3, 5, 7]

# Video de entrada
video_in = "/content/Arequipa/Fotos en iCloud/IMG_2319.MP4"
video_out = "/content/centernet_sim_anticolision.mp4"

cap = cv2.VideoCapture(video_in)
fps = cap.get(cv2.CAP_PROP_FPS)
w = int(cap.get(cv2.CAP_PROP_FRAME_WIDTH))
h = int(cap.get(cv2.CAP_PROP_FRAME_HEIGHT))

scale = 0.7
nw, nh = int(w*scale), int(h*scale)

fourcc = cv2.VideoWriter_fourcc(*"mp4v")
out = cv2.VideoWriter(video_out, fourcc, fps if fps>0 else 25, (nw, nh))

# Zona de peligro
roi = np.array([
    [int(0.2*nw), int(0.7*nh)],
    [int(0.8*nw), int(0.7*nh)],
    [int(0.95*nw), nh],
    [int(0.05*nw), nh]
])
```

```
prev_centers = []
frame_id = 0

while True:
    ret, frame = cap.read()
    if not ret:
        break
    frame_id += 1

    frame = cv2.resize(frame, (nw, nh))

    res = model.predict(frame, conf=0.25, verbose=False)[0]

    centers_curr = []
    alert = False

    cv2.polylines(frame, [roi], True, (0,255,255), 2)

    if res.boxes is not None:
        boxes = res.boxes.xyxy.cpu().numpy()
        clss  = res.boxes.cls.cpu().numpy().astype(int)

        for (x1,y1,x2,y2), cls_id in zip(boxes, clss):
            if cls_id not in vehicle_ids:
                continue

            cx = int((x1 + x2) / 2)
            cy = int((y1 + y2) / 2)
            centers_curr.append((cx, cy))


            motion = "estatico"
            min_dist = 9999
            for (px, py) in prev_centers:
                d = np.hypot(cx - px, cy - py)
                if d < min_dist:
                    min_dist = d
            if min_dist > 4:
                motion = "dinamico"


            inside = cv2.pointPolygonTest(roi, (cx, cy), False) >= 0

            if inside and motion == "dinamico":
                alert = True

            color  =  (0,0,255)  if  inside  and  motion=="dinamico"  else
(255,255,255)


            cv2.circle(frame, (cx, cy), 6, color, -1)
            cv2.putText(frame, motion, (cx+5, cy-5),
                        cv2.FONT_HERSHEY_SIMPLEX, 0.5, color, 1)

    prev_centers = centers_curr
```

```
    text = "ALERTA COLISION (CenterNet sim)" if alert else "ZONA SEGURA
(CenterNet sim)"
    txt_color = (0,0,255) if alert else (0,255,0)
    cv2.putText(frame, text, (40, 60),
                cv2.FONT_HERSHEY_SIMPLEX, 0.8, txt_color, 2)

    out.write(frame)

cap.release()
out.release()

display(Video(video_out, embed=True))
print("Video guardado como:", video_out)

# Código de Código de evaluación de métricas y contrastación contra modelo
Oráculo

import cv2
import numpy as np
import time
import pandas as pd
from ultralytics import YOLO
from tabulate import tabulate
from tqdm import tqdm

video_in = "Arequipa/Fotos en iCloud/IMG_5893.MP4"
csv_oraculo = "ground_truth_oraculo.csv"
output_excel = "resultados_eval_oraculo_vs_edge_optimizado.xlsx"

df_oraculo = pd.read_csv(csv_oraculo)

scale = 1.0  # Mantenemos la escala alta para no perder sensibilidad
conf_th = 0.15

model = YOLO("yolov5n.pt")
vehicle_ids = [0, 2, 3, 5, 7]

cap = cv2.VideoCapture(video_in)
w = int(cap.get(cv2.CAP_PROP_FRAME_WIDTH))
h = int(cap.get(cv2.CAP_PROP_FRAME_HEIGHT))
total_frames = int(cap.get(cv2.CAP_PROP_FRAME_COUNT))
nw, nh = int(w * scale), int(h * scale)

roi = np.array([
    [int(0.2 * nw), int(0.7 * nh)],
    [int(0.8 * nw), int(0.7 * nh)],
    [int(0.95 * nw), nh],
    [int(0.05 * nw), nh]
])

prev_centers = []
```

```
frame_records = []

TP = 0
FP = 0
FN = 0
TN = 0

# Variables para tus métricas originales
yolo_times = []
center_times = []
yolo_total_dets = 0

for frame_idx in tqdm(range(total_frames)):
    ret, frame = cap.read()
    if not ret:
        break

    frame = cv2.resize(frame, (nw, nh))

    # Temporizador 1: Inferencia YOLO
    t0 = time.time()
    res = model.predict(frame, conf=conf_th, verbose=False)[0]
    t1 = time.time()
    yolo_times.append((t1 - t0) * 1000.0)

    yolo_centers = []
    if res.boxes is not None:
        boxes = res.boxes.xyxy.cpu().numpy()
        clss = res.boxes.cls.cpu().numpy().astype(int)

        for (x1, y1, x2, y2), cls_id in zip(boxes, clss):
            if cls_id not in vehicle_ids:
                continue
            cx = int((x1 + x2) / 2)
            cy = int((y1 + y2) / 2)
            yolo_centers.append((cx, cy))

    yolo_total_dets += len(yolo_centers)

    # Temporizador 2: Lógica CenterNet-sim
    t2 = time.time()
    centers_curr = []
    alerta_edge = 0

    for (cx, cy) in yolo_centers:
        centers_curr.append((cx, cy))
        if prev_centers:
            dists = [np.hypot(cx - px, cy - py) for (px, py) in
prev_centers]
```

```
                min_dist = min(dists)
            else:
                min_dist = 9999

            if min_dist > 4:
                inside = cv2.pointPolygonTest(roi, (cx, cy), False) >= 0
                if inside:
                    alerta_edge = 1

        t3 = time.time()
        center_times.append((t3 - t2) * 1000.0)
        prev_centers = centers_curr

        try:
            alerta_real = df_oraculo.iloc[frame_idx]["Alerta_Real"]
        except IndexError:
            break

        if alerta_edge == 1 and alerta_real == 1:
            TP += 1
        elif alerta_edge == 1 and alerta_real == 0:
            FP += 1
        elif alerta_edge == 0 and alerta_real == 1:
            FN += 1
        elif alerta_edge == 0 and alerta_real == 0:
            TN += 1

        frame_records.append({
            "Frame": frame_idx + 1,
            "Alerta_Oraculo": alerta_real,
            "Alerta_Edge": alerta_edge,
            "YOLO_time_ms": (t1 - t0) * 1000.0,
            "CNT_time_ms": (t3 - t2) * 1000.0
        })

cap.release()

total_evaluados = TP + FP + FN + TN
precision = TP / (TP + FP) if (TP + FP) > 0 else 0.0
recall = TP / (TP + FN) if (TP + FN) > 0 else 0.0
f1_score = 2 * (precision * recall) / (precision + recall) if (precision
+ recall) > 0 else 0.0

# Cálculos de las métricas de rendimiento y compatibilidad
mean_yolo_t = np.mean(yolo_times) if yolo_times else 0.0
mean_cnt_t  = np.mean(center_times) if center_times else 0.0
fps_total = 1000.0 / (mean_yolo_t + mean_cnt_t) if (mean_yolo_t +
mean_cnt_t) > 0 else 0.0
compatibilidad = TP / yolo_total_dets if yolo_total_dets > 0 else 0.0
```

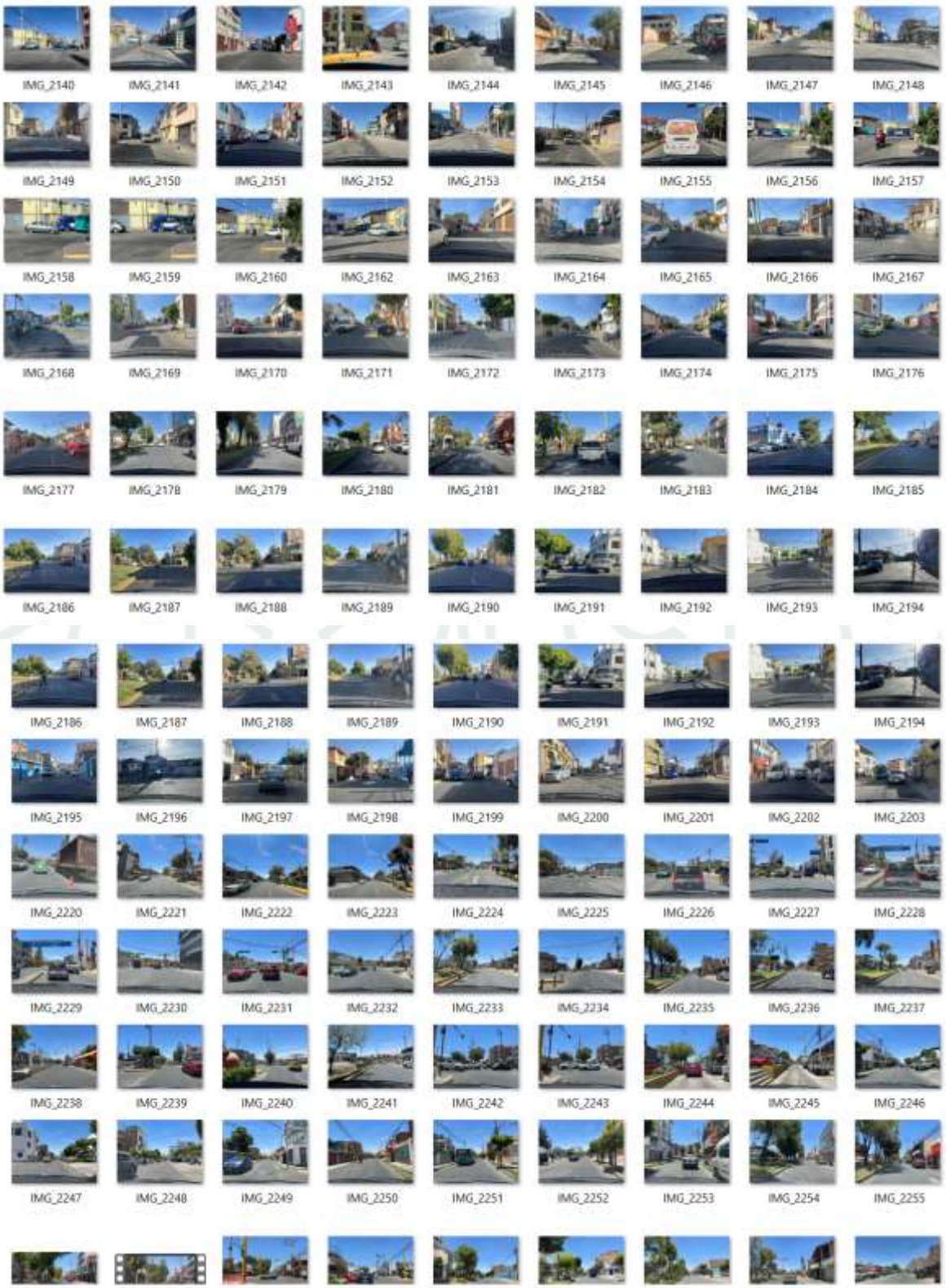
```
global_stats = pd.DataFrame([
    ["Total Frames Evaluados", total_evaluados],
    ["Verdaderas Alertas (TP)", TP],
    ["Falsas Alertas (FP)", FP],
    ["Alertas Omitidas (FN)", FN],
    ["Silencio Correcto (TN)", TN],
    ["Precisión (Confiabilidad de Alerta)", precision],
    ["Sensibilidad / Recall", recall],
    ["F1-Score", f1_score],
    ["Tiempo medio YOLO por frame (ms)", mean_yolo_t],
    ["Tiempo medio CenterNet-sim por frame (ms)", mean_cnt_t],
    ["FPS aproximados del sistema", fps_total],
    ["Compatibilidad YOLOv5 vs CenterNet-sim", compatibilidad],
], columns=["Métrica", "Valor"])

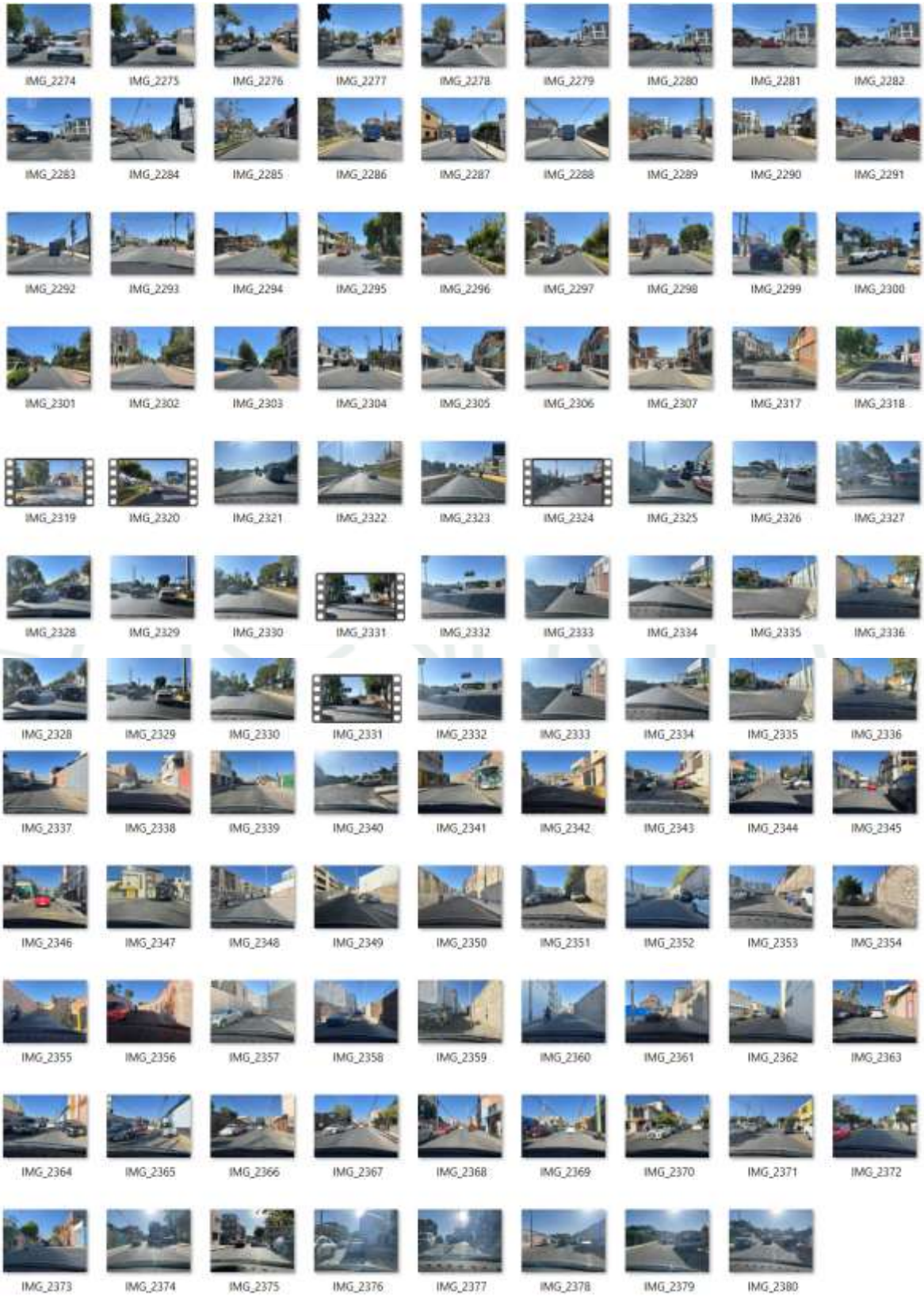
print("\n================ MATRIZ DE CONFUSIÓN Y RENDIMIENTO
================\n")
print(tabulate(global_stats, headers="keys", tablefmt="grid",
showindex=False))

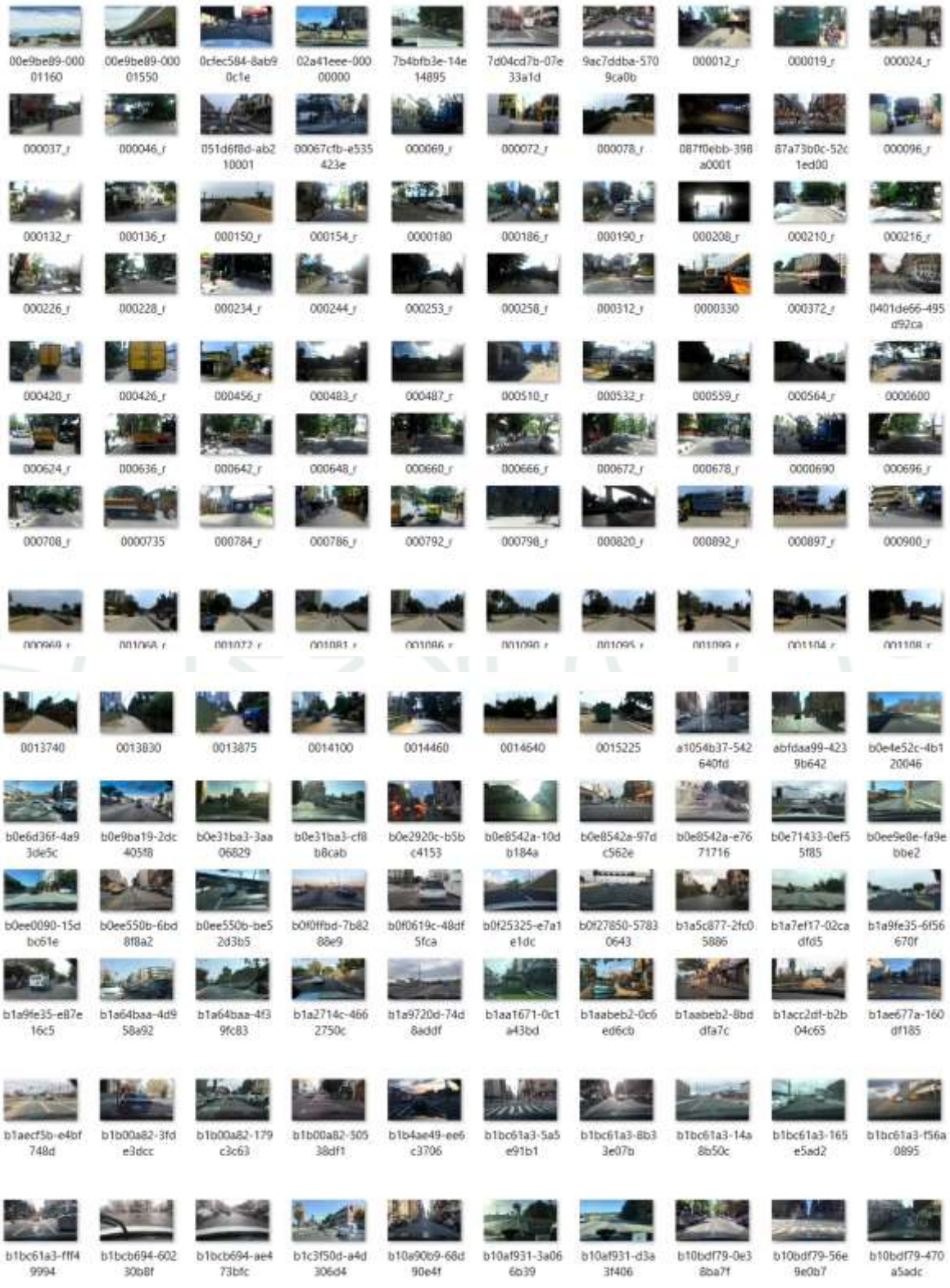
with pd.ExcelWriter(output_excel, engine="openpyxl") as writer:
    global_stats.to_excel(writer, sheet_name="Resumen", index=False)
    pd.DataFrame(frame_records).to_excel(writer,
sheet_name="Desglose_Frames", index=False)

print("\nReporte Excel generado:", output_excel)
```

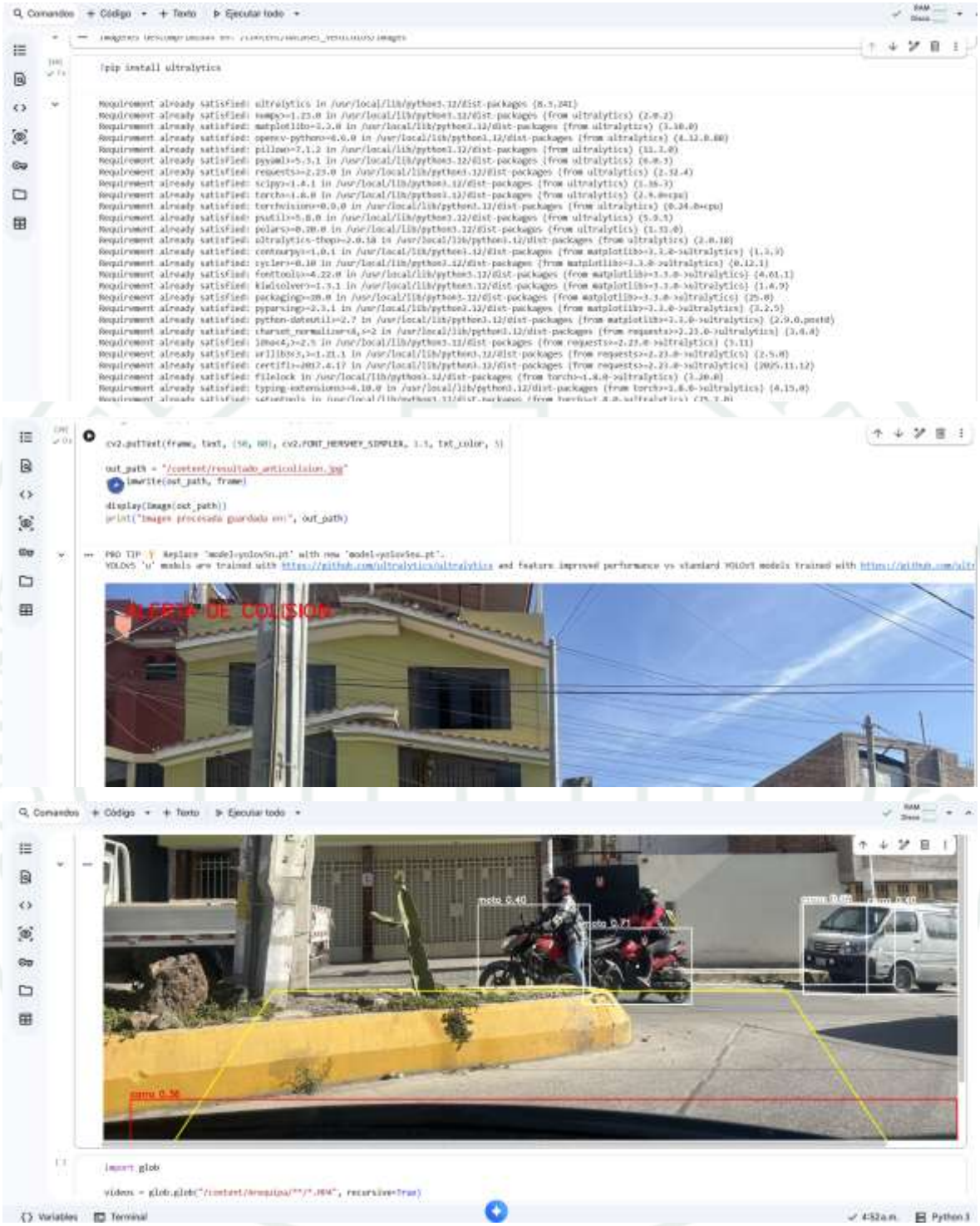
Anexo N° 3. Base de datos

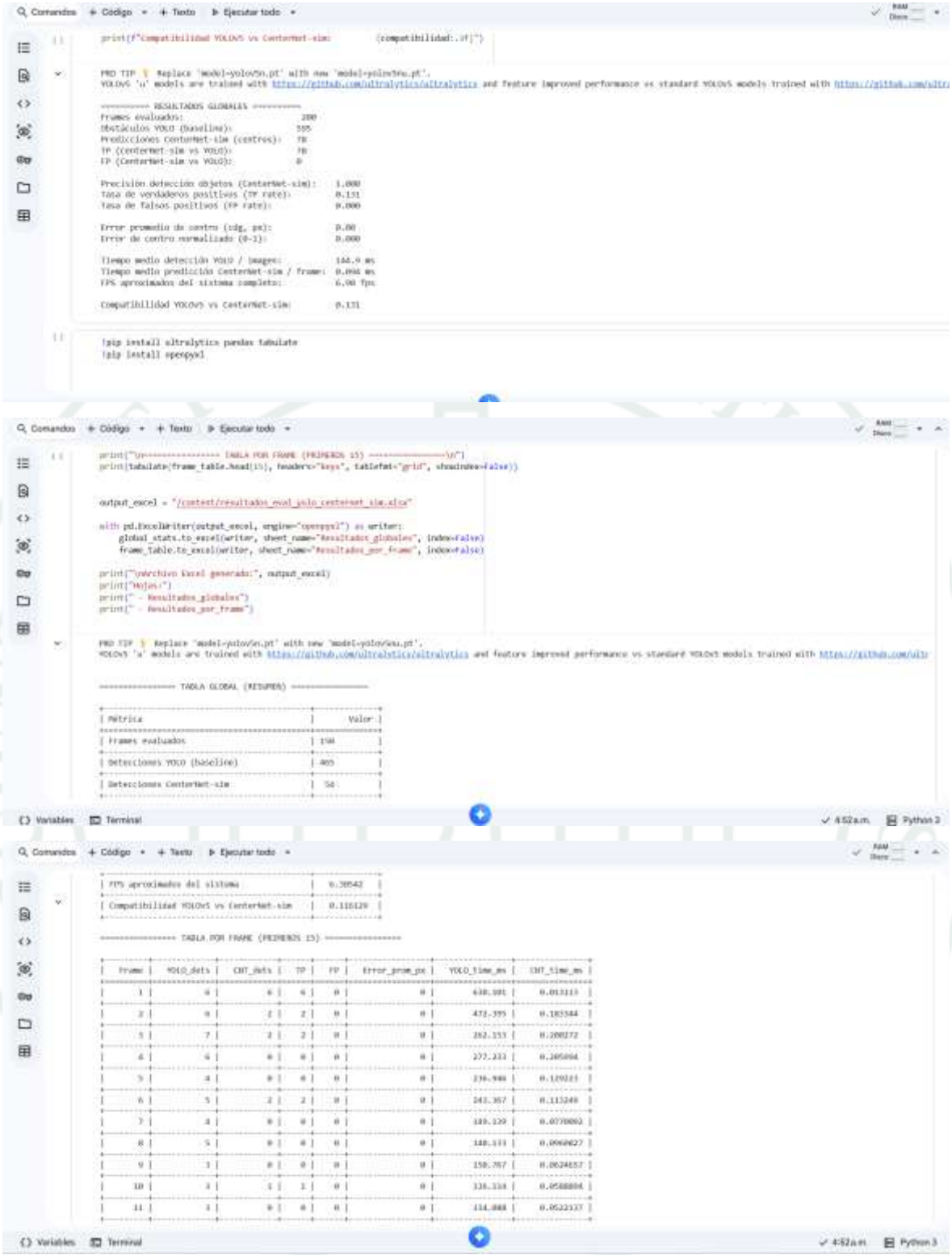



IMG_2274 IMG_2275 IMG_2276 IMG_2277 IMG_2278 IMG_2279 IMG_2280 IMG_2281 IMG_2282
IMG_2283 IMG_2284 IMG_2285 IMG_2286 IMG_2287 IMG_2288 IMG_2289 IMG_2290 IMG_2291
IMG_2292 IMG_2293 IMG_2294 IMG_2295 IMG_2296 IMG_2297 IMG_2298 IMG_2299 IMG_2300
IMG_2301 IMG_2302 IMG_2303 IMG_2304 IMG_2305 IMG_2306 IMG_2307 IMG_2317 IMG_2318
IMG_2319 IMG_2320 IMG_2321 IMG_2322 IMG_2323 IMG_2324 IMG_2325 IMG_2326 IMG_2327
IMG_2328 IMG_2329 IMG_2330 IMG_2331 IMG_2332 IMG_2333 IMG_2334 IMG_2335 IMG_2336
IMG_2328 IMG_2329 IMG_2330 IMG_2331 IMG_2332 IMG_2333 IMG_2334 IMG_2335 IMG_2336
IMG_2337 IMG_2338 IMG_2339 IMG_2340 IMG_2341 IMG_2342 IMG_2343 IMG_2344 IMG_2345
IMG_2346 IMG_2347 IMG_2348 IMG_2349 IMG_2350 IMG_2351 IMG_2352 IMG_2353 IMG_2354
IMG_2355 IMG_2356 IMG_2357 IMG_2358 IMG_2359 IMG_2360 IMG_2361 IMG_2362 IMG_2363
IMG_2364 IMG_2365 IMG_2366 IMG_2367 IMG_2368 IMG_2369 IMG_2370 IMG_2371 IMG_2372
IMG_2373 IMG_2374 IMG_2375 IMG_2376 IMG_2377 IMG_2378 IMG_2379 IMG_2380


00e9be89-000 01160
00e9be89-000 01550
0cfec584-8ab9 0c1e
02a41eee-000 00000
7b4bfb3e-14e 14895
7d04cd7b-07e 33a1d
9ac7ddba-570 9ca0b
000012_r
000019_r
000024_r
000037_r
000046_r
051d6f8d-ab2 10001
00067cfb-e535 423e
000069_r
000072_r
000078_r
087f0ebb-398 a0001
87a73b0c-52c 1ed00
000096_r
000132_r
000136_r
000150_r
000154_r
0000180
000186_r
000190_r
000208_r
000210_r
000216_r
000226_r
000228_r
000234_r
000244_r
000253_r
000258_r
000312_r
0000330
000372_r
0401de66-495 d92ca
000420_r
000426_r
000456_r
000483_r
000487_r
000510_r
000532_r
000559_r
000564_r
0000600
000624_r
000636_r
000642_r
000648_r
000660_r
000666_r
000672_r
000678_r
0000690
000696_r
000708_r
0000735
000784_r
000786_r
000792_r
000798_r
000820_r
000892_r
000897_r
000900_r
0013740
0013830
0013875
0014100
0014460
0014640
0015225
a1054b37-542 640fd
abfdaa99-423 9b642
b0e4e52c-4b1 20046
b0e6d36f-4a9 3de5c
b0e9ba19-2dc 405f8
b0e31ba3-3aa 06829
b0e31ba3-cf8 b8cab
b0e2920c-b5b c4153
b0e8542a-10d b184a
b0e8542a-97d c562e
b0e8542a-e76 71716
b0e71433-0ef5 5f85
b0ee9e8e-fa9e bbe2
b0eee0090-15d bc61e
b0eee550b-6bd 8f8a2
b0eee550b-be5 2d3b5
b0f0ffbd-7b82 88e9
b0f0619c-48df 5fca
b0f25325-e7a1 e1dc
b0f27850-5783 0643
b1a5c877-2fc0 5886
b1a7ef17-02ca dfd5
b1a9fe35-6f56 670f
b1a9fe35-e87e 16c5
b1a64baa-4d9 58a92
b1a64baa-4f3 9fc83
b1a2714c-466 2750c
b1a9720d-74d 8addf
b1aa1671-0c1 a43bd
b1aabeb2-0c6 ed6cb
b1aabeb2-8bd dfa7c
b1acc2df-b2b 04c65
b1ae677a-160 df185
b1aecf5b-e4bf 748d
b1b00a82-3fd e3dcc
b1b00a82-179 c3c63
b1b00a82-505 38df1
b1b4ae49-ee6 c3706
b1bc61a3-5a5 e91b1
b1bc61a3-8b3 3e07b
b1bc61a3-14a 8b50c
b1bc61a3-165 e5ad2
b1bc61a3-f56a 0895
b1bc61a3-fff4 9994
b1bcb694-602 30b8f
b1bcb694-ae4 73bfc
b1c3f50d-a4d 306d4
b10a90b9-68d 90e4f
b10af931-3a06 6b39
b10af931-d3a 3f406
b10bdf79-0e3 8ba7f
b10bdf79-56e 9e0b7
b10bdf79-470 a5adc

Anexo N° 4. Evidencia de las simulaciones




Comandos
+ Código
+ Texto
Ejecutar todo
print(f"Compatibilidad YOLOv5 vs CenterNet-sim: {compatibilidad:.3f}")
PRO TIP 💡 Replace 'model-yolov5n.pt' with new 'model-yolov5nu.pt'.
========== RESULTADOS GLOBALES ==========
Frames evaluados: 200
Obstáculos YOLO (baseline): 555
Predicciones CenterNet-sim (centros): 78
TP (CenterNet-sim vs YOLO): 78
FP (CenterNet-sim vs YOLO): 0
Precisión detección objetos (CenterNet-sim): 1.000
Tasa de verdaderos positivos (TP rate): 0.131
Tasa de falsos positivos (FP rate): 0.000
Error promedio de centro (cdg, px): 0.00
Error de centro normalizado (0-1): 0.000
Tiempo medio detección YOLO / imagen: 144.9 ms
Tiempo medio predicción CenterNet-sim / frame: 0.096 ms
FPS aproximados del sistema completo: 6.90 fps
Compatibilidad YOLOv5 vs CenterNet-sim: 0.131
!pip install ultralytics pandas tabulate
!pip install openpyxl
Variables
Terminal
Python 3
print("\n========== TABLA POR FRAME (PRIMEROS 15) ==========\n")
print(tabulate(frame_table.head(15), headers="keys", tablefmt="grid", showindex=False))
output_excel = "/content/resultados_eval_yolo_centernet_sim.xlsx"
with pd.ExcelWriter(output_excel, engine="openpyxl") as writer:
global_stats.to_excel(writer, sheet_name="Resultados_globales", index=False)
frame_table.to_excel(writer, sheet_name="Resultados_por_frame", index=False)
print("\nArchivo Excel generado:", output_excel)
print("Hojas:")
print(" - Resultados_globales")
print(" - Resultados_por_frame")
PRO TIP 💡 Replace 'model-yolov5n.pt' with new 'model-yolov5nu.pt'.
========== TABLA GLOBAL (RESUMEN) ==========
Métrica	Valor
Frames evaluados	198
Detecciones YOLO (baseline)	465
Detecciones CenterNet-sim	56
4:52 a.m.
FPS aproximados del sistema	6.30542
Compatibilidad YOLOv5 vs CenterNet-sim	0.116129
========== TABLA POR FRAME (PRIMEROS 15) ==========
Frame	YOLO_dets	CNT_dets	TP	FP	Error_prom_px	YOLO_time_ms	CNT_time_ms
1	0	6	6	0	0	630.591	0.013113
2	0	2	2	0	0	472.395	0.185344
3	7	2	2	0	0	262.153	0.200272
4	6	0	0	0	0	277.233	0.105004
5	4	0	0	0	0	236.948	0.129123
6	5	2	2	0	0	243.367	0.113249
7	3	0	0	0	0	189.139	0.0778002
8	5	0	0	0	0	148.133	0.0960027
9	3	0	0	0	0	150.767	0.0624657
10	3	1	1	0	0	138.138	0.0588804
11	3	0	0	0	0	114.888	0.0522137