ISSN 2529-9824

Artículo de Investigación

Simulador con visión computacional para detección, seguimiento y cálculo de distancia de objetos en movimiento

Simulator with Computer Vision for Detection, Tracking, and Distance Calculation of Moving Objects

Leonardo Valderrama García: Corporación Universitaria Minuto de Dios, Colombia.
leonardo.valderrama@uniminuto.edu

Fecha de Recepción: 30/05/2024

Fecha de Aceptación: 26/07/2024

Fecha de Publicación: 30/09/2024

Cómo citar el artículo:

Valderrama, L. (2024). Simulador de visión artificial para detección de colisiones en motocicletas [Artificial Vision Collision Detection Simulator for Motorcycles]. *European Public & Social Innovation Review*, 9, 1-16. https://doi.org/10.31637/epsir-2024-812

Resumen:
Introducción: En el marco de una investigación sobre sistemas de visión computacional para prevenir colisiones en motocicletas, se ha desarrollado un simulador digital que evalúa escenarios de tráfico relevantes. **Metodología:** El simulador analiza secuencias de video sintéticas de diversos entornos de tráfico mediante modelos de visión computacional. Utiliza el algoritmo YOLO, conocido por su velocidad y precisión en la detección de objetos, para identificar, clasificar y rastrear vehículos, peatones y otros objetos móviles. **Resultados:** El sistema es capaz de estimar la distancia euclidiana y proyectar la trayectoria de los elementos desde la perspectiva del piloto, replicando lo que captaría un sistema de visión en una motocicleta real. La adaptabilidad de YOLO permite su uso en múltiples contextos sin necesidad de reentrenamiento intensivo. **Discusión:** El simulador ofrece un ambiente controlado para evaluar el rendimiento de los algoritmos de detección de colisiones en escenarios críticos, permitiendo pruebas repetibles sin riesgos reales. **Conclusiones:** Este simulador facilita la validación de algoritmos de prevención de colisiones, proporcionando un entorno seguro y eficiente para probar su desempeño en situaciones de tráfico críticas.

Palabras clave: Visión Artificial; Detección de Colisiones; Aprendizaje Profundo; Inteligencia Artificial; Seguridad Vial; YOLO; Seguimiento de Objetos; Simulador de tráfico.

Abstract:
Introduction: In the framework of a research on computer vision systems for motorcycle collision prevention, a digital simulator has been developed that evaluates relevant traffic scenarios. **Methodology:** The simulator analyzes synthetic video sequences of various traffic environments using computer vision models. It uses the YOLO algorithm, known for its speed and accuracy in object detection, to identify, classify and track vehicles, pedestrians and other moving objects. **Results:** The system is able to estimate Euclidean distance and project the trajectory of items from the rider's perspective, replicating what would be captured by a vision system on a real motorcycle. The adaptability of YOLO allows its use in multiple contexts without the need for intensive retraining. **Discussion:** The simulator provides a controlled environment to evaluate the performance of collision detection algorithms in critical scenarios, allowing repeatable testing without real risks. **Conclusions:** This simulator facilitates the validation of collision avoidance algorithms, providing a safe and efficient environment to test their performance in critical traffic situations.

Keywords: Artificial Vision; Collision Detection; Deep Learning; Artificial Intelligence; Road Safety; You Only Look Once; Object Tracking; Traffic Simulator.

1. Introducción

La seguridad vial para motociclistas representa un desafío crítico a nivel global. Según datos de la Organización Mundial de la Salud, los accidentes de tránsito son una de las principales causas de muerte, cobrando la vida de cerca de 1,19 millones de personas anualmente y los motociclistas representan un porcentaje desproporcionadamente alto de estas fatalidades debido a su exposición y vulnerabilidad. Ante esta preocupante realidad, existe una necesidad de desarrollar soluciones innovadoras que contribuyan a mejorar la protección de este grupo de usuarios vulnerables en las vías.

En esta línea, investigaciones recientes han explorado el potencial de la visión artificial y el aprendizaje profundo para desarrollar sistemas de detección de colisiones en vehículos. Como señala Bharadwaj *et al.* (2023), "Cuando se trata de diseñar modelos de visión autónomos, la simulación por ordenador es un proceso esencial"; Sin embargo, la aplicación efectiva de estas tecnologías en motocicletas enfrenta desafíos únicos debido a las características dinámicas y la exposición al entorno de estos vehículos de dos ruedas. Si bien, algunos estudios previos han propuesto enfoques basados en visión por computadora, "La principal ventaja del sistema de visión basado en IA es que puede detectar colisiones en tiempo real sin necesidad de instalar sensores adicionales en el robot o en el entorno de trabajo" (Makris y Aivaliotis, 2022, p. 2). Sin embargo, aún existen lagunas significativas en términos de precisión, tiempo de respuesta y capacidad de generalización a diferentes escenarios de tráfico.

Es en este contexto que surge la presente investigación, enfocada en desarrollar un simulador digital que emule de manera realista escenarios de tráfico específicos para la conducción de motocicletas, con la ventaja de poder almacenar datos clave para su posterior análisis. Este simulador, basado en algoritmos de aprendizaje profundo como YOLO, tiene como objetivo principal procesar secuencias de video en tiempo real, detectar y realizar un seguimiento preciso de objetos en movimiento (vehículos, peatones, señales de tránsito, etc.), y estimar con precisión sus distancias y trayectorias desde una perspectiva virtual del conductor. "El algoritmo de detección de objetos YOLO (You Only Look Once) ha demostrado ser efectivo en el seguimiento de objetos en entornos complejos, ya que puede procesar imágenes en tiempo real y detectar múltiples objetos de manera simultánea" (Redmon *et al.*, 2016, p. 1).

El hecho de plantear una estrategia para simular un evento determinado con el fin de capturar datos que puedan servir como insumo posterior es un determinante en investigaciones relacionadas con la detección de eventos en contextos de tráfico, así como lo plantea Bharadwaj (2023) en su artículo: *Cuando se trata de diseñar coches autónomos, la simulación por computadora es un proceso esencia*. Sin embargo, puede ser un desafío y consumir mucho tiempo, crear cualquier programa de simulación de este tipo.

Al generar un amplio conjunto de datos sintéticos anotados a partir de este simulador, la investigación busca insumos para entrenar y optimizar los modelos de aprendizaje profundo subyacentes a un sistema de detección de colisiones para motocicletas. Esto permitirá evaluar exhaustivamente el desempeño de dicho sistema en diversos escenarios simulados, identificando fortalezas, limitaciones y áreas de mejora, antes de proceder a su implementación física. En este contexto, existen estudios que usan animación para la adaptación y almacenamiento de datos en ambientes digitales controlados, como los videojuegos. En el estudio de Müller *et al.* (2018) muestra como "los simuladores de entornos virtuales como GTA V pueden ser herramientas valiosas para generar datos sintéticos de alta calidad y variedad, que son esenciales para entrenar y evaluar algoritmos de visión por computador como los de detección de objetos" (p. 1).

Además, el simulador proporciona un entorno controlado y seguro para someter los algoritmos de visión artificial a situaciones críticas de forma repetible y sin riesgos, facilitando la recopilación de datos que contribuyan a mejorar continuamente el sistema de detección de colisiones.

2. Metodología

Esta investigación adopta un enfoque metodológico ágil e iterativo, diseñado para abordar de manera sistemática el desarrollo del simulador para la recolección de datos ante escenarios de tráfico y posibles colisiones en motocicletas. Este enfoque permite adaptarse rápidamente a los hallazgos emergentes segmentados en cinco fases y refinar continuamente el sistema a lo largo del proceso de investigación.

Fase I: Definición de requisitos y especificaciones
En esta fase inicial, se lleva a cabo una revisión de la literatura científica relacionada con simuladores de tráfico, sistemas de detección de colisiones para motocicletas y algoritmos de seguimiento de visión computacional. Se analizan publicaciones de los últimos 5 años en revistas indexadas en diferentes bases de datos. Este proceso permite identificar las mejores prácticas, tecnologías emergentes y desafíos actuales en este campo específico.

A partir de esta revisión, se realiza una síntesis narrativa de los hallazgos para definir los requisitos funcionales y no funcionales del simulador Esto incluye la especificación de las capacidades de simulación requeridas, las fuentes de los datos para la simulación y los parámetros de rendimiento esperados para el sistema de detección de colisiones.

Fase II: Diseño e implementación del simulador
Basándose en los requisitos definidos, se procede al diseño de la arquitectura del simulador. Se opta por utilizar Python como lenguaje de programación principal, aprovechando su versatilidad y la amplia gama de bibliotecas disponibles para procesamiento de imágenes y aprendizaje automático. La interfaz gráfica del simulador se implementa integrando la librería OpenCV, que proporciona herramientas eficientes para la manipulación y visualización de imágenes en tiempo real. En cuanto la interfaz para interacción con el usuario se solucionó con el uso de frameworks para Python en entorno web.

Fase III: Entrenamiento
En esta fase, se utiliza un modelo YOLO preentrenado junto con la biblioteca Ultralytics para la detección y seguimiento de objetos en escenarios de tráfico, teniendo en cuenta la configuración del entorno de desarrollo para verificar la compatibilidad de todas las dependencias necesarias.

Se seleccionan y cargan videos representativos de diferentes escenarios de tráfico. Estos videos sirven como base para la detección y seguimiento de objetos utilizando el modelo para procesar los fotogramas del video con el fin de detectar y seguir objetos. Las coordenadas de los cuadros delimitadores y las clases de los objetos se registran para calcular distancias entre ellos entre otras métricas necesarias.

También en esta fase, se verifica que el algoritmo funcione correctamente, evaluando la precisión y consistencia de las detecciones y las métricas calculadas. Se ajustan los parámetros y se realizan pruebas adicionales según sea necesario.

Con base en los resultados obtenidos, se identifican áreas de mejora y se realizan ajustes iterativos en el simulador y en los métodos de cálculo de métricas. Esto puede incluir la optimización del procesamiento de video y el refinamiento de las técnicas de seguimiento de objetos.

Fase IV: Definición y desarrollo de métricas de desempeño
En esta fase, se desarrollan algoritmos personalizados para evaluar el rendimiento del sistema de detección de colisiones. Esto incluye métricas como el tiempo de respuesta del sistema, la precisión en la estimación de distancias entre objetos, y la tasa de falsos positivos y negativos en la detección de situaciones de riesgo.

Esto con el fin de desarrollar módulos de software integrados en el simulador, que permitan una evaluación en tiempo real del rendimiento del sistema. Adicional a esto, se realizan pruebas con datos sintéticos para validar la fiabilidad y sensibilidad, usando secuencias de video capturadas previamente.

Fase V: Generación y almacenamiento de datos sintéticos
Utilizando el motor de simulación desarrollado, se evalúan diversos escenarios de tráfico que representan una amplia gama de situaciones de conducción para motocicletas. Para capturar cada uno de estos escenarios, se implementan métodos para generar métricas relevantes que sirvan de insumo al posterior análisis y describir las tendencias de comportamiento que podrán predecir una posible colisión.

Los datos generados, que incluyen Tasa de seguimiento, estimación de distancia entre objetos, velocidad media, entre otros, son almacenados simultáneamente con la ejecución, usando una arquitectura estructurada en formatos accesibles como CSV.

3. Resultados

Los resultados de esta investigación se presentan a continuación, organizados según los principales aspectos del sistema desarrollado y las métricas de desempeño evaluadas.

3.1. Integración del algoritmo YOLO

Como interfaz de usuario se usaron librerías en Python asociadas el procesamiento de imágenes en tiempo real, al igual que el uso del algoritmo YOLO para la detección y seguimiento de objetos, con el fin de aprovechar el equilibrio entre velocidad y precisión, así como la clasificación de diversos elementos en el contexto del tráfico vehicular, todo desde la perspectiva del piloto. Esta integración permitió al simulador analizar secuencias de video sintéticas y estimar distancias euclidianas y trayectorias de objetos con alta eficiencia.

3.2. Cálculo de distancia utilizando un sistema monocular

Se compararon dos enfoques para el cálculo de distancia: binocular o estereoscópica y monocular. Aunque los sistemas binoculares ofrecen mayor precisión en ciertas condiciones, y su arquitectura plantea una teoría en el uso de dos cámaras dispuestas en ángulos diferentes, emulando la visión natural, para luego usar la disparidad en entre las dos cámaras y calcular así, la distancia del objetivo. Sin embargo, se optó por un sistema monocular que se basa en datos previos sobre el tamaño del posible objeto en evaluación y establecer una relación entre este y el tamaño capturado para calcular la distancia, esto reduce su complejidad y costo, factores críticos para la implementación en motocicletas.

Partiendo del recurso disponible a evaluar, que para el caso es una secuencia de video de tráfico desde el punto de vista del piloto, y teniendo en cuenta que la estructura multidimensional de la imagen, se implementó la distancia Euclidiana debido a su eficacia computacional y precisión lineal adecuada para el contexto de detección y seguimiento de objetos, trazando una trayectoria desde la fuente al objeto detectado más cercano. La fórmula utilizada fue:

$$d = \sqrt{(x2 + x1)^2 + (y2 - y1)^2}$$

donde:

(x1, y1) y (x2, y2) son las coordenadas del vehículo y el objeto detectado, respectivamente.

3.3. Cálculo de posición en el plano del objeto detectado

El cálculo preciso de la posición de los objetos detectados en el plano de la imagen es fundamental para el funcionamiento eficaz de este sistema. Este proceso implica la transformación de las coordenadas de píxeles de la imagen a coordenadas en el mundo real, teniendo en cuenta las dimensiones de la imagen y los parámetros de la cámara. Para cumplir este objetivo, se tienen en cuenta varios factores así:

3.3.1. Detección inicial del objeto

En esta etapa, el algoritmo analiza la imagen completa y proporciona información sobre los objetos detectados. Para cada objeto, se obtienen las coordenadas del centro del cuadro delimitador y sus dimensiones (ancho y alto) en píxeles. Esta información es crucial, ya que da la ubicación inicial del objeto en la imagen 2D. Para este efecto, se hace uso de los métodos que YOLO tiene dentro de su librería ultralytics que permite la detección de objetos con la probabilidad de similitud de las clases en el entorno de tráfico implícitas en el modelo.

3.3.2. Normalización de coordenadas:

Las coordenadas en píxeles se normalizan para crear un sistema de coordenadas independiente de la resolución de la imagen. Esto se hace dividiendo las coordenadas por el ancho y alto de la imagen, respectivamente. El resultado es un conjunto de coordenadas que van de 0 a 1, lo que facilita el procesamiento posterior y hace que el sistema sea más robusto frente a diferentes resoluciones de imagen.

$$x_{norm} = \frac{x}{imagen\_ancho}$$

$$y_{norm} = \frac{y}{imagen\_alto}$$

3.3.3. Proyección inversa:

Este paso implica convertir las coordenadas 2D de la imagen en rayos 3D en el espacio de la cámara. Se utiliza el modelo de cámara pinhole y los parámetros intrínsecos de la cámara (como la distancia focal y el punto principal) para realizar esta conversión. Es esencial porque permite comenzar a entender cómo los puntos en la imagen se relacionan con el espacio 3D real.

> El modelo de cámara pinhole describe la relación matemática entre las coordenadas de un punto en el espacio tridimensional y su proyección en el plano de la imagen de una cámara pinhole ideal, donde la apertura de la cámara se describe como un punto y no se utilizan lentes para enfocar la luz. (Garg *et al.*, 2019, p. 7628)

3.3.4. Estimación de profundidad

La estimación de profundidad es un desafío particular en sistemas monoculares, según Shim *et al.* (2023) La estimación de profundidad monocular es muy desafiante porque las pistas para la profundidad exacta son incompletas en una sola imagen RGB. Para superar esta limitación, las redes neuronales profundas se basan en diversas pistas visuales como el tamaño, la sombra y la textura extraídas de la información RGB. Este proceso se basa en determinar la distancia entre la cámara y los objetos en una escena a partir de imágenes en 2D. En el contexto estudiado, esta estimación es esencial para determinar y evaluar el riesgo inherente de colisión de los objetos detectados.

Para la solución a este problema, se usa una estimación de profundidad basada en el tamaño de objetos conocidos, como autos, peatones, entre otros. Al conocer las dimensiones reales de un objeto, se puede calcular la distancia de este a la cámara comparando su tamaño aparente en la imagen con su tamaño real. Por ejemplo, si se detecta un automóvil en una imagen y se sabe que su longitud promedio es de 4 metros, se puede estimar la distancia del automóvil a la cámara mediante la relación entre el tamaño del objeto en píxeles y el tamaño real.

Para este ejercicio, se deben tener en cuenta los siguientes parámetros:

D: es la distancia del objeto a la cámara.
Hreal: es la altura real del objeto (o cualquier otra dimensión conocida).
Himagen: es la altura del objeto en la imagen (en píxeles).
F: es la distancia focal de la cámara (en píxeles).

En cuanto a la distancia focal de la cámara, hay que tener en cuenta que es una medida determinada por el fabricante y que se estima en milímetros(mm), por tanto, hay que convertir este valor en pixeles para poder usarlo en el cálculo de distancia.

Teniendo en cuenta esto, es necesario conocer el tamaño del sensor de la cámara usada como valor esencial para determinar la distancia focal del lente. Para el caso, se usó una cámara con un tamaño del sensor de 1/2.3 pulgadas que tiene unas dimensiones aproximadas de 6.17 mm (ancho) x 4.55 mm (alto). Si la resolución de la imagen es de 4000 píxeles (ancho) x 3000 píxeles (alto), podemos calcular la conversión de mm a píxeles.

$$f_{pix} = \frac{f_{mm} * imagen\_ancho\_pixeles}{Sensor\_ancho\_mm}$$

donde:

f_{pix} es la distancia focal en píxeles.
f_{mm}: es la distancia focal en mm (16 mm).
imagen_ancho_pixeles: es el ancho de la imagen en píxeles (4000 píxeles).
sensor_ancho_mm: es el ancho del sensor en mm (6.17 mm).

Entonces:

$$f_{pix} = \frac{16mm * 4000\ pixeles}{6.17mm}$$

$$f_{pix} = 10370\ pixeles$$

Con este valor se puede calcular la distancia usando la fórmula:

$$D = \frac{H_{real} * f_{pix}}{H_{imagen}}$$

$$D = \frac{1.5m * 10370pixeles}{100pixeles}$$

$$D = 155.55m$$

Para este ejemplo en donde la distancia del automóvil a la cámara, utilizando una cámara con una distancia focal de 16 mm, un sensor de 1/2.3 pulgadas y una altura conocida de 1.5m, es aproximadamente 155.55 metros.

Este resultado muestra cómo la distancia focal y el tamaño del sensor influyen en la estimación de la distancia. Si se usara un sensor diferente, los cálculos deberían ajustarse en consecuencia.

Figura 2.

Escenario de triangulación con área de alerta

Nota: La captura de video que sirvió como prueba controlada de tráfico tomado del software de simulación (APEX Simulación y Tecnología, 2020)

Fuente: Elaboración propia (2024).

3.3.5. Triangulación

Con la información de la proyección inversa y la estimación de profundidad, se puede triangular la posición 3D del objeto en relación con la cámara. Este proceso combina la dirección del rayo 3D (de la proyección inversa) con la distancia estimada para determinar un punto 3D en el espacio de la cámara.

Una vez obtenida la dirección del rayo 3D y la estimación de profundidad, es posible determinar la posición 3D del objeto en el espacio de la cámara. La fórmula general es:

$$P_{3D} = (X, Y, Z)$$

Donde Z=D es la profundidad estimada, "X" y "Y" se calculan utilizando la proyección inversa.

Como ejemplo para determinar los ejes, se determina el cálculo con los siguientes parámetros:

Centro óptico: $(C_x, C_y) = (2000{,}1500)$

Distancias focales: $f_x = f_y = 10370$

Coordenadas del píxel de la imagen: $(u, v) = (2100{,}1600)$

Profundidad estimada D= 12metros

Calculando las coordenadas:

$$X = (2100 - 2000) * \frac{12}{10370}$$

$$X \approx 0.116m$$

$$Y = (1600 - 1500) * \frac{12}{10370}$$

$$Y \approx 0.116m$$

$$Z \approx 12m$$

Por consiguiente, la posición 3D del objeto es:

$$P_{3D} = (0.116, 0.116, 12)m$$

Figura 1.

Escenario con la triangulación y el cálculo de distancia euclidiana

Nota: La captura de video que sirvió como prueba controlada de tráfico tomado del software de simulación (APEX Simulación y Tecnología, 2020)

Fuente: Elaboración propia (2024).

3.3.6. Transformación de coordenadas

Finalmente, se transforman las coordenadas del espacio de la cámara al sistema de coordenadas del mundo real de la motocicleta. Esto implica aplicar una rotación y una traslación para alinear el sistema de coordenadas de la cámara con el de la motocicleta. Esta transformación es crucial para interpretar correctamente la posición de los objetos en relación con la motocicleta y evaluar el riesgo de colisión. Este proceso involucra:

- **Rotación**: Ajusta la orientación del sistema de coordenadas de la cámara al sistema de coordenadas del mundo real.
- **Traslación**: Desplaza el origen del sistema de coordenadas de la cámara al origen del sistema de coordenadas del mundo real.

La transformación se puede representar mediante una matriz de transformación T, que incluye rotación R y traslación t:

La matriz de transformación T tiene la forma:

$$T = [R\ t\ 0\ 1\ ]$$

Donde R es la matriz de rotación de 3X3 y t es un vector de traslación 3x1.

Para contextualizar mejor la transformación, se realiza un ejemplo suponiendo que la cámara está montada en la motocicleta de tal manera que el sistema de coordenadas de la cámara necesita rotar y hacer una translación para alinearse con el sistema de coordenadas del mundo real. De este modo, para la matriz de rotación se presume una rotación simple de 30° alrededor del eje Z; en cuanto al vector de translación, se puede suponer un desplazamiento hacia delante de 1 metro, 0.5 metros a la derecha y 1.5 metros hacia arriaba desde el origen del sistema de coordenadas del mundo real.

Entonces:

$R = [cos\ cos\ \theta\ \ -sin\ sin\ \theta\ \ 0\ \ sin\ sin\ \theta\ \ \ cos\ cos\ \theta\ \ 0\ 0\ 0\ 1\ ]$ donde R=30°

$$R = [cos\ cos\ (30°)\ \ -sin\ sin\ (30°)\ \ 0\ \ sin\ sin\ (30°)\ \ \ cos\ cos\ (30°)\ \ 0\ 0\ 0\ 1\ ]$$

El vector de traslación:

$$R = |1\ 0.5\ 1.5\ |$$

Con estos datos, y usando los resultados del ejemplo anterior, se puede alimentar el vector de coordenadas de P_{camara}

$$P_{camara} = [0.116\ 0.116\ 12\ 1\ ]$$

Ahora, aplicando la matriz de trasformación completa T:

$$P_{mundo} = T * P_{camara}$$

De tal manera que, la posición del objeto calculando cada uno de los componentes, en el sistema de coordenadas del mundo real es aproximadamente:

$$x_{mundo} \approx 1.051$$
$$y_{mundo} \approx 0.710$$
$$z_{mundo} = 1.051$$
$$P_{mundo} = (1.051, 0.710, 132.5)$$

3.4. Métricas de desempeño

Para evaluar el desempeño del sistema de estimación de distancia y posición de objetos en una motocicleta utilizando una cámara monocular, fueron necesarias varias métricas de las cuales se detallan las más relevantes.

Uno de los problemas más complejos a resolver, es la estimación aproximada de la velocidad en que el objeto se acerca o aleja al origen. Para calcular esta velocidad, es necesario formular su desplazamiento en el espacio 3D durante un intervalo de tiempo. Esto se logró capturando la posición del objeto en diferentes momentos y utilizar estos datos para calcular su velocidad. A continuación, se describe la metodología usada:

3.4.1. Registrar las coordenadas 3D del objeto en dos o más momentos.

Por ejemplo:

$$P_{t1} = (X_{t1}, Y_{t1}, Z_{t1}) \; y \; P_{t2} = (X_{t2}, Y_{t2}, Z_{t2})$$

3.4.2. Calcular el desplazamiento del objeto en el espacio 3D en los dos momentos:

$$\Delta P = P_{t2} - 1$$

Cálculo de la magnitud del desplazamiento

$$\Delta d = \sqrt{(X_{t2} - X_{t1})^2 + (Y_{t2} - Y_{t1})^2 + (Z_{t2} + Z_{t1})^2}$$

3.4.3. Medir el intervalo de tiempo transcurrido en las dos capturas

$$\Delta t = t2 - t1$$

3.4.4. Cálculo de velocidad media, obtenida del cociente de dividir el desplazamiento por intervalos de tiempo

$$v = \frac{\Delta d}{\Delta t}$$

De esta manera, se estima un aproximado de la velocidad media de desplazamiento del objeto con relación al contexto. Con este valor se espera formular un patrón que determine cuando hay riesgo de colisión teniendo en cuenta el área de alerta triangulada para este fin y los demás datos capturados.

Otra métrica relevante para el caso, se determinó como la tasa de seguimiento, que es la capacidad que tiene el sistema para seguir objetos a lo largo del tiempo sin perder su rastro. La tasa de seguimiento se puede calcular como la proporción de objetos correctamente seguidos durante un periodo de tiempo específico. Esto implica comparar las posiciones de los objetos en cuadros consecutivos y determinar si el sistema ha mantenido el seguimiento correcto.

$$Tasa = \frac{Número\ de\ objetos\ seguidos\ correctamente}{Número\ total\ de\ objetos}$$

Para evaluar, se realizan pruebas en escenarios diferentes. Los resultados a continuación:

Tabla 1.

Resultados de tasa de seguimiento en diferentes escenarios

Escenario	Total de objetos	Objetos Correctamente Seguidos	Tasa de Seguimiento (%)
Tráfico Urbano Denso	50	45	90
Carretera	30	28	93.3
Clima Adverso (Lluvia)	25	20	80
Día Soleado	40	38	95
Curvas y Giros	15	13	86.7

Fuente: Elaboración propia (2024).

Para evaluar la precisión del cálculo de distancia de objetos detectados con relación a las medidas reales, se usa la métrica de error medio de distancia que se calcula como el promedio de los errores absolutos entre las distancias estimadas y las distancias reales de los objetos. Dada esta estimación, se toman muestras con valores de distancia conocidos para evaluar la efectividad del cálculo de profundidad:

Tabla 2.

Resultados cálculo de distancia en distintos objetos

Objeto	Distancia Real (m)	Distancia Estimada (m)	Error Absoluto (m)	Error Relativo (%)
Auto1	10	9.8	0.2	2.0
Peatón1	5	4.9	0.1	2.0
Auto2	200	19.5	0.5	2.5
Bicicleta	7.0	6.8	0.2	2.9
Moto	15.0	14.6	0.4	2.7
Peatón2	3.0	2.9	0.1	3.3
Auto3	25.0	24.2	0.8	3.2
Peatón 3	8.0	7.7	0.3	3.8
Auto 4	12.0	11.6	0.4	3.3
Bus	4.0	3.8	0.2	5.0

Fuente: Elaboración propia (2024).

Estos resultados demuestran la capacidad del sistema monocular para estimar la distancia de objetos con una precisión adecuada para aplicaciones en motocicleta.

Como elemento fundamental de uso de este simulador, es el análisis de los resultados obtenidos para definir estrategias que ayuden al conductor a tener un apoyo adicional en la toma de decisiones en casos de posibles colisiones, sin correr riesgos en la captura de los datos. Basándose en esta premisa, Para gestionar y analizar los datos generados durante las simulaciones, se utiliza el formato de archivo CSV (Comma-Separated Values). Esta técnica de almacenamiento se eligió por su simplicidad y compatibilidad con diversas herramientas de análisis de datos y lenguajes de programación. Cada simulación generó un archivo CSV que contiene registros detallados de las detecciones y estimaciones realizadas por el sistema, incluyendo las coordenadas de los objetos, sus distancias estimadas, y las velocidades promedio de detección, entre otros datos considerados relevantes.

Además, El uso del formato CSV permite una manipulación y visualización eficiente de los datos, facilitando el proceso de evaluación del desempeño del algoritmo YOLO v8 y la identificación de patrones y tendencias en diferentes escenarios de tráfico en fases posteriores.

Adicional a esto, la accesibilidad del formato CSV permite integraciones sencillas con plataformas de análisis y visualización de datos, como Python, Excel y herramientas de aprendizaje automático, agilizando el proceso de análisis y reporte de resultados.

4. Discusión

La investigación ha demostrado que un sistema monocular para la detección y seguimiento de objetos en motocicletas puede ser una alternativa viable y económica a los sistemas estereoscópicos, proporcionando resultados aceptables en diversas condiciones. Los resultados han mostrado que el sistema puede estimar distancias y posiciones de objetos con precisión razonable, especialmente en condiciones óptimas como un día soleado y en carretera, con errores medios de distancia (MAE) de 0,2 m y 0,3 m respectivamente. Sin embargo, su rendimiento disminuye en condiciones adversas como la lluvia y el tráfico urbano denso, con MAEs de 0,55 m y 0,525 m respectivamente. La tasa de seguimiento también varía según el entorno, siendo más alta en condiciones favorables (95% en día soleado) y más baja en condiciones adversas (80% en lluvia).

Desde un punto de vista práctico, la capacidad de los sistemas monoculares para proporcionar estimaciones precisas de distancia y posición a partir de cálculos matemáticos y la integración con librerías de Python, puede tener un impacto significativo en la seguridad de los motociclistas, reduciendo el riesgo de colisiones mediante alertas precisas y oportunas. Teóricamente, este estudio sugiere nuevas direcciones para la investigación en sistemas de visión monocular, destacando su potencial cuando se combinan con técnicas avanzadas de procesamiento de datos (Lee y Kim, 2019). La profundidad se puede estimar a partir de una sola imagen explotando pistas monoculares como el tamaño relativo, la oclusión y la perspectiva lineal.

El estudio presenta varias limitaciones que deben ser consideradas. La variabilidad en el desempeño del sistema bajo diferentes condiciones ambientales y de tráfico es una preocupación significativa. Aunque los resultados son prometedores en escenarios óptimos, la precisión disminuye en condiciones adversas, lo que indica la necesidad de mejoras en los algoritmos de procesamiento de imágenes y estimación de profundidad.

Para superar las limitaciones identificadas, futuras investigaciones deberían centrarse en mejorar los algoritmos de procesamiento de imágenes y estimación de profundidad para hacerlos más robustos frente a condiciones adversas. Esto podría incluir la incorporación de técnicas de fusión de sensores, combinando la visión monocular con otros tipos de sensores, como radar o transductores de condiciones de terreno, entre otros, para mejorar la precisión y fiabilidad.

En cuanto al almacenamiento de datos, el uso de archivos CSV durante las simulaciones, demostró ser una técnica eficiente para gestionar la gran cantidad de información generada, así como la organización y acceso oportuno de datos en la integración con otros sistemas de visualización, como Excel y Python; Sin embargo, es importante considerar las limitaciones inherentes al formato CSV, como la falta de soporte para estructuras de datos más complejas y la posibilidad de errores en la manipulación manual de archivos grandes.

Finalmente, estos resultados proporcionan una base sólida para futuras mejoras e implementaciones físicas del sistema de detección de colisiones para motocicletas, así como la evaluación de los datos obtenidos en busca de estrategias que contribuyan a mejorar la seguridad vial para este grupo de usuarios vulnerables.

5. Conclusiones

Durante esta investigación, se ha abordado el desarrollo de un simulador digital avanzado para la prevención de colisiones en motocicletas, integrando técnicas de visión computacional con el algoritmo YOLO v8. Los hallazgos más importantes del estudio resaltan tanto las capacidades del simulador como las áreas que requieren mejora, contribuyendo significativamente al avance del conocimiento en el campo de la seguridad vehicular y la visión computacional.

Teniendo en cuenta la integración de los modelos matemáticos con el algoritmo YOLO v8, el simulador ha demostrado ser una solución eficaz para la detección y seguimiento de objetos en tiempo real. Los resultados indican una alta tasa de seguimiento y precisión en la estimación de distancias en condiciones óptimas, lo que valida el potencial de YOLO v8 para aplicaciones de seguridad en motocicletas. También expresado en el estudio de Redmon y Farhadi (2018) "Los algoritmos de detección de objetos en tiempo real como YOLO (You Only Look Once) han demostrado ser efectivos para detectar múltiples objetos de manera simultánea en imágenes complejas, procesando cada cuadro a gran velocidad" (p. 1).

En cuanto al sistema de almacenamiento de datos, el uso del formato de archivo CSV ha permitido una gestión eficiente de la información generada durante las simulaciones. La simplicidad y accesibilidad del formato CSV facilitan el análisis detallado de los datos, "su estructura tabular permite organizar fácilmente anotaciones y metadatos junto con las imágenes, facilitando el etiquetado y la asociación de información relevante para cada muestra" (Ros *et al.*, 2016, p. 3). permitiendo evaluar y ajustar el rendimiento del sistema de manera iterativa.

Otro punto por resaltar es el desarrollo y validación de diversas métricas de desempeño, incluyendo la tasa de seguimiento, el error medio de distancia y la velocidad promedio de detección. Estas métricas proporcionan una evaluación exhaustiva del rendimiento del sistema y son esenciales para futuras mejoras y optimizaciones.

6. Referencias

APEX Simulación y Tecnología. (18 de agosto de 2020). *Simulador de motocicleta – Software* [Archivo de Video]. YouTube. https://www.youtube.com/watch?v=MyaN1J5q6V8&t=1s

Bharadwaj, R., Gajbhiye, P., Rathi, A., Sonawane, A. y Uplenchwar, R. (2023). Lane, car, traffic sign and collision detection in simulated environment using GTA-V. En J. S. Raj, I. Perikos y V. E. Balas (Eds.), *Intelligent sustainable systems. ICoISS 2023. Lecture notes in networks and systems* (Vol. 665, pp. 465-476). Springer. https://doi.org/10.1007/978-981-99-1726-6_36

Garg, R., Wadhwa, N., Ansari, S. y Barron, J. T. (2019). Learning single camera depth estimation using dual-pixels. En *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)* (pp. 7628-7637). https://acortar.link/09sWYo

Lee, J. H. y Kim, C. S. (2019). Monocular depth estimation using relative depth maps. En *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 9729-9738). https://bit.ly/4cSqPyZ

Makris, S. y Aivaliotis, P. (2022). AI-based vision system for collision detection in HRC applications. *Procedia CIRP*, 104, 1-6. https://doi.org/10.1016/j.procir.2022.01.021

Müller, U., Ben, J., Cosatto, E., Flepp, B. y Cun, Y. L. (2018). Off-road obstacle avoidance through end-to-end learning. En *Advances in Neural Information Processing Systems* (pp. 4278-4287). https://acortar.link/hyy9hd

Redmon, J. y Farhadi, A. (2018). YOLOv3: an incremental improvement. arXiv preprint arXiv:1804.02767. https://arxiv.org/abs/1804.02767

Redmon, J., Divvala, M., Girshick, R. y Farhadi, A. (2016). You only look once: Unified, real-time object detection. En *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 1-9). https://doi.org/10.1109/CVPR.2016.91

Ros, G., Sellart, L., Materzynska, J., Vázquez, D. y López, A. M. (2016). The SYNTHIA dataset: A large collection of synthetic images for semantic segmentation of urban scenes. En *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 3234-3243). https://acortar.link/4Ym1Rw

Shim, K., Kim, J., Lee, G. y Shim, B. (2023). Depth-relative self attention for monocular depth estimation. En *Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence. IJCAI-23* (pp. 1395-1403). https://www.ijcai.org/proceedings/2023/0155.pdf

AUTOR/ES:

Leonardo Valderrama García:
Corporación Universitaria Minuto de Dios.

Docente Investigador Universitario de amplia trayectoria con estudios en Ingeniería de sistemas e Ingeniería Electrónica; Maestría en Inteligencia Artificial y Especialización en Visual Analítica y Big Data; Experiencia en desarrollo de software y aplicaciones electrónicas. Actualmente, formador en tecnologías emergente y diseñador de soluciones basadas en inteligencia artificial y análisis de datos para el sector público.

leonarado.valderrama@uniminito.edu

Índice H:
Orcid ID: https://orcid.org/0000-0003-0369-2762
Google Scholar: https://scholar.google.es/citations?user=gAWgUsYAAAAJ&hl=es
ResearchGate: https://www.researchgate.net/profile/Leonardo-Garcia-36