UNIVERSIDAD AUTÓNOMA DE QUERÉTARO

DIRECCIÓN DE INVESTIGACIÓN Y POSGRADO
FACULTAD DE INGENIERÍA

Tesis

Identificación de personas con drones aplicando técnicas de inteligencia artificial

Que como parte de los requisitos para obtener el grado de maestro en ciencias en inteligencia artificial

Presenta:

Ing. Mitzi Jocelyn Reyes Cerón

Asesor:

Dr. Juan Manuel Ramos Arreguin

Co-asesora:

Dra. Rosalba Galván Guerra

Sinodales

Dr. Juan Manuel Ramos Arreguin
Presidente
Dra. Rosalba Galván Guerra
Co-Directora
Dr. Jesús Carlos Pedraza Ortega ´
Vocal
Dr. Andras Takács
Suplente
Dr. Edgar Alejandro Rivas Araiza
Suplente

Centro Universitario, Querétaro, México

Diciembre 2025

Dedicatoria

- A mi familia y pareja, quienes desde siempre sembraron en mí el deseo de construir un futuro mejor. Su apoyo incondicional fue el ancla que me mantuvo firme en los momentos en que el camino se sentía demasiado largo.

Porque el camino no siempre fue recto ni sencillo, y hubo momentos en que la oscuridad parecía más grande que la meta. Pero seguir avanzando, un paso a la vez fue suficiente para llegar.

Este trabajo no es solo mío es el resultado de todo el amor y la fe que ustedes depositaron en mí, incluso cuando yo mismo dudé.-

Agradecimientos

Agradezco al Consejo Nacional de Humanidades, Ciencias y Tecnologías (Conahcyt) por la beca otorgada durante mi estancia en la maestría, y a la Universidad Autónoma de Querétaro por brindarme la oportunidad de cursar estudios de posgrado. Al coordinador del posgrado, Dr. Saúl, quien siempre veló por los intereses de los estudiantes y creó un entorno propicio para el desarrollo académico.

A mi asesor y asesora, Dr. Juan Manuel y Dra. Rosalba, quienes me brindaron su apoyo y confianza a lo largo de todo este proceso. Su orientación fue fundamental, pero más aún lo fue su paciencia ante los desafíos que surgieron en el camino, y su capacidad para acompañarme incluso en los momentos en que el avance fue lento. Sin su dedicación y compromiso, este trabajo no habría llegado a buen puerto. A todo mi sínodo, quienes con su conocimiento y retroalimentación contribuyeron a elevar la calidad de esta investigación y me guiaron con rigor y generosidad en cada etapa.

Este trabajo también es el reflejo de un esfuerzo personal que, aunque en ocasiones se vio pausado por las circunstancias de la vida, nunca se abandonó. Cada obstáculo encontrado fue también una oportunidad de aprendizaje, y la culminación de esta etapa representa para mí una meta alcanzada con perseverancia y convicción.

Finalmente, agradezco a la Facultad de Ingeniería, en especial a su personal académico y administrativo, quienes demostraron profesionalismo y amabilidad durante toda mi estancia.

Índice

Índice de figuras

Índice de tablas

1. Resumen

La navegación autónoma de Vehículos Aéreos No Tripulados (UAVs) en entornos no estructurados requiere la integración eficiente de sistemas de percepción y control. Esta tesis presenta una metodología modular para la detección de personas y la navegación basada en inteligencia artificial, implementada sobre una plataforma comercial de arquitectura cerrada (Tello EDU). Para el subsistema de percepción visual, se evaluó la arquitectura de aprendizaje profundo YOLOv8s utilizando el conjunto de datos VisDrone, logrando un F1-score de 0.955 y demostrando una mayor robustez ante variaciones de escala y oclusión en comparación con modelos predecesores. En cuanto al subsistema de navegación, debido a la imposibilidad de acceder a los lazos de control internos del vehículo, se desarrolló una estrategia de Identificación de Sistemas de Caja Negra. Se entrenó una Red Neuronal Artificial (Perceptrón Multicapa) para aprender el modelo cinemático inverso del dron a partir de datos sintéticos generados en MATLAB. Los resultados experimentales en simulación validaron que el modelo neuronal es capaz de generalizar maniobras de vuelo complejas y actuar como un filtro estabilizador ante trayectorias estocásticas. La investigación concluye que la integración de visión avanzada y modelado neuronal inverso constituye una solución viable para dotar de autonomía a drones de bajo costo en tareas de vigilancia.

Palabras Clave: Vehículos Aéreos No Tripulados (UAV), Visión por Computadora, YOLOv8, Identificación de Sistemas, Redes Neuronales Artificiales, Cinemática Inversa.

2. Abstract

Autonomous navigation of Unmanned Aerial Vehicles (UAVs) in unstructured environments requires the efficient integration of perception and control systems. This thesis presents a modular methodology for person detection and AI-based navigation, implemented on a commercial closed-architecture platform (Tello EDU). For the visual perception subsystem, the YOLOv8s deep learning architecture was evaluated using the VisDrone dataset, achieving an F1-score of 0.955 and demonstrating superior robustness

against scale variations and occlusions compared to predecessor models. Regarding the navigation subsystem, due to the inaccessibility of the vehicle's internal control loops, a Black-box System Identification strategy was developed. An Artificial Neural Network (Multilayer Perceptron) was trained to learn the drone's inverse kinematic model using synthetic data generated in MATLAB. Simulation results validated that the neural model can generalize complex flight maneuvers and acting as a stabilizing filter against stochastic trajectories. The research concludes that integrating advanced computer vision with inverse neural modeling constitutes a viable solution for endowing low-cost micro-drones with autonomy for surveillance tasks.

Keywords: Unmanned Aerial Vehicles (UAV), Computer Vision, YOLOv8, System Identification, Artificial Neural Networks, Inverse Kinematics.

3. Introducción

El avance en el diseño de sistemas autónomos ha respondido a la creciente necesidad de desarrollar soluciones tecnológicas que aborden tareas complejas en campos como la vigilancia, el monitoreo y la búsqueda y rescate. En este contexto, los Vehículos Aéreos No Tripulados (UAVs), comúnmente conocidos como drones, se han consolidado como una herramienta fundamental debido a su versatilidad y capacidad de despliegue rápido. Sin embargo, dotar a estas plataformas de una verdadera autonomía para la identificación y seguimiento de personas en entornos no estructurados sigue representando un desafío técnico significativo, que requiere la integración eficiente de sistemas de percepción y navegación.

Para abordar la etapa de percepción, la Inteligencia Artificial (IA) ha revolucionado el campo de la visión por computadora. En particular, las Redes Neuronales Convolucionales (CNN) han demostrado ser la arquitectura estándar para el procesamiento de imágenes. Dentro de este dominio, modelos de detección de una sola etapa como YOLO (You Only Look Once) han permitido superar las limitaciones de velocidad de los enfoques tradicionales, facilitando la detección y clasificación de personas en con alta precisión, incluso bajo condiciones de oclusión y variaciones de escala típicas de las tomas aéreas.

Por otro lado, la etapa de navegación presenta retos particulares cuando se emplean plataformas comerciales de bajo costo, como el Tello EDU. Estos dispositivos suelen operar bajo arquitecturas cerradas ("Caja Negra"), donde el usuario no tiene acceso directo a los lazos de control dinámico internos. Esto impide el uso de estrategias de control clásicas basadas en modelos físicos explícitos. Para resolver esta limitación, el presente trabajo propone el uso de Redes Neuronales Artificiales para la identificación de sistemas, aprendiendo el modelo cinemático inverso del dron a partir de datos experimentales.

La integración de estas dos vertientes percepción visual avanzada y modelado neuronal de la navegación permite establecer una arquitectura capaz no solo de reconocer a un sujeto, sino de estimar y generar las acciones de control necesarias para mantener al objetivo dentro del campo de visión . Este enfoque aprovecha los avances en la capacidad de procesamiento moderna para ejecutar algoritmos de aprendizaje profundo con latencias mínimas, sentando las bases para sistemas de vigilancia totalmente autónomos.

El objetivo de este trabajo de tesis es desarrollar y validar una metodología modular para la navegación autónoma de drones. Esta metodología integra el uso de la arquitectura YOLOv8 para la detección robusta de personas y una Red Neuronal tipo Perceptrón Multicapa (MLP) entrenada para identificar la cinemática inversa del vehículo. De esta forma, se busca habilitar al dron para traducir la información visual en referencias de trayectoria, permitiendo una estrategia de seguimiento eficiente y segura sin depender del conocimiento explícito de la dinámica del robot.

3.1 Antecedentes

El uso de drones en aplicaciones de vigilancia y seguimiento de personas ha experimentado un notable desarrollo desde sus primeras implementaciones. Inicialmente, los drones se empleaban principalmente con propósitos militares y de reconocimiento. Sin embargo, en las últimas décadas, han surgido investigaciones y aplicaciones enfocadas en la vigilancia civil, la seguridad pública y la asistencia en emergencias [1].

Los primeros avances significativos en el seguimiento de personas con drones se remontan a la década del 2000. Durante este período, los adelantos en la tecnología de sensores y el procesamiento de imágenes permitieron la detección y seguimiento automático de objetivos en entornos controlados [2]. Con el aumento de la accesibilidad y la reducción de costos de los drones, se iniciaron investigaciones para adaptar estas plataformas a aplicaciones civiles. Durante la década del 2010, se llevaron a cabo estudios pioneros que exploraron el uso de drones en la vigilancia de eventos públicos, la búsqueda y el rescate

de personas, así como en la seguridad en entornos urbanos y rurales [1].

Paralelamente, la integración de inteligencia artificial (IA) en drones marcó un hito significativo en la evolución de estas plataformas aéreas. Desde sus primeras implementaciones, la IA ha demostrado ser un catalizador para mejoras sustanciales en términos de autonomía, capacidad de toma de decisiones y eficiencia en el cumplimiento de diversas tareas. Uno de los primeros casos documentados sobre la aplicación de IA en drones se remonta a la década del 2010. En este período, se exploraron técnicas de aprendizaje profundo para mejorar la capacidad de detección y seguimiento de objetos en imágenes aéreas [1]. Además, otro avance relacionado al seguimiento de peatones, se dio a conocer en una investigación en el 2014, que presenta un sistema para el seguimiento de peatones que utiliza Visual SLAM y Constrained Multiple Kernel (CMK), lo que hace que este tenga una buena efectividad en detección y seguimiento de peatones kernel [3].

Además de estos avances que permitieron que los drones identificaran y siguieran objetos con mayor precisión y eficacia [2], hay otros avances significativos en continuo mejoramiento de detección de peatones, sistemas basados en SLAM y el uso de técnicas de inteligencia artificial que han ayudado a la continua evolución de los drones autónomos.

En 2019 se publicó un trabajo, donde, se creó un sistema de mapeo simultáneo y localización monocular denso , que combina eficazmente el SLAM monocular directo con una red de predicción de profundidad, que tuvo bueno es resultados en cuanto a precisión en la predicción de profundidad gracias al uso de CNN. Sin embargo, presenta complejidad computacional alta, lo que requiere una considerable capacidad de procesamiento [4]. En ese mismo año, se publicó un trabajo en el que se diseñó un sistema VSLAM (Visual Simultaneous Localization and Mapping) que integraba características de puntos y líneas mediante la fusión de datos de cámaras RGB-D (Red, Green, Blue-Depth, cámaras de profundidad) e IMU (Inertial Measurement Units, unidades de medida inercia). Este sistema mejoraba significativamente la precisión y la robustez en comparación con métodos convencionales, permitiendo una mayor precisión en la navegación de UAVs en entornos complejos. Sin embargo, el trabajo también resaltó la complejidad en la fusión y sincronización de datos, lo que requiere una alta capacidad de

procesamiento, aumentando los costos y la dificultad de implementación en aplicaciones prácticas [5]. Además, en 2019, se publicó una revisión de la tecnología de mapeo y VSLAM, este trabajo proporcionó una visión integral de los avances en VSLAM, destacando su capacidad para permitir la autonomía en entornos sin GPS (Global Positioning System). La revisión subrayó las mejoras en la precisión y la robustez de la navegación autónoma gracias a la tecnología VSLAM. Sin embargo, también se señaló la dependencia de sensores de alta calidad como una limitación significativa, ya que la precisión del sistema depende en gran medida de la calidad de los sensores utilizados [6]. En 2020, se publicó un trabajo que propone un enfoque de indirecto para entornos exteriores, utilizando el algoritmo SIFT (Scale-Invariant Feature Transform, donde se implementó un algoritmo de flexión de trayectoria mejorado para optimizar las estimaciones de poses y puntos de referencia. El uso de aprendizaje profundo mejoró la precisión de la odometría visual-inercial (Visual-Inertial Odometry, VIO), aunque la efectividad del sistema depende en gran medida de la calidad de los sensores visuales e inerciales [7]. Además, en este año, se publicó un artículo que presentó un modelo de aprendizaje profundo Faster R-CNN (Region Convolutional Neural Network) para la detección de peatones en imágenes capturadas por drones. Se evaluó el modelo en 1500 imágenes, mostrando un rendimiento superior con un 98% de precisión, 99% de recall (True positive rate), y un 98% en la medida F1 (Harmonic mean of precision and recall). El modelo Faster R-CNN alcanzó un 98% de precisión en la detección de peatones, aunque su efectividad depende de la calidad y diversidad de las imágenes utilizadas para entrenar el modelo [8].

En 2021, se publicó un trabajo que presentó un sistema SLAM mejorado que integra el VP (Vanishing Point) como una observación externa sin error de deriva para optimizar la trayectoria estimada, este sistema mejoró la precisión de posicionamiento en entornos en interior y corrigió errores de deriva (drift errors), sin embargo, si la trayectoria no es un bucle cerrado, el error de deriva puede no reducirse significativamente [9]. Este año, también se publicó una revisión que comparó técnicas de visión por computadora para sistemas de percepción de vehículos autónomos (AVs), centrándose en la detección de peatones y vehículos. El enfoque específico en la detección de peatones y vehículos es

crucial para la seguridad y navegación de los AVs. Sin embargo, la generalización de los algoritmos puede ser limitada si se utilizan solo conjuntos de datos tradicionales como Caltech y KITTI [10]. De igual forma en este año, se publicó un trabajo que analizó modelos y marcos clásicos de redes neuronales, la extracción de características multi-escala y la utilización de datos desde múltiples perspectivas, aunque se utiliza aprendizaje profundo, la detección efectiva en condiciones en diferentes escalas sigue siendo un desafío [11].

En 2022, se publicó trabajo cuyo objetivo fue identificar problemas críticos en la detección de peatones, como la oclusión, la deformación y la baja calidad de las imágenes, y destacó el potencial de las imágenes multiespectrales para mejorar la detección, en este trabajo se propuso el uso de imágenes multiespectrales y versiones de YOLO (You Only Look Once) para mejorar la detección en condiciones de iluminación variables, sin embargo, el sistema no puede lidiar efectivamente con la oclusión y la deformación de objetos, y la baja calidad de las imágenes y las condiciones de iluminación pueden afectar su rendimiento [12]. Ese año, también se publicó un trabajo en el que se diseñó un sistema VINS (*Visual-Inertial Navigation System*) y exploró los parámetros ocultos del LSD (*Line Segment Detector*), además de que utilizó FGO (*Factor Graph Optimization*) para mejorar la precisión en el cierre de bucles, lo cual es crucial en entornos grandes y complejos, sin embargo, la precisión y eficacia del sistema dependen de la calidad de los sensores utilizados, lo que puede incrementar los costos [13].

En 2023, se publicó un trabajo que presenta un algoritmo de VSLAM que utiliza técnicas de aprendizaje profundo para estimar la profundidad a partir de la entrada de la cámara monocular, posteriormente, esta información se transforma en una nube de puntos y se alinea con el mapa para construir mapas tridimensionales de ocupación. La técnica mejora la precisión en entornos complejos, aunque puede enfrentar desafíos en condiciones de poca iluminación o con muchas oclusiones (obstáculos), donde la estimación de profundidad puede ser menos precisa [15], también se enfocó en mejorar la precisión y eficiencia del seguimiento de peatones utilizando entradas de video en 4K y técnicas avanzadas de procesamiento en el dron, la combinación de CPU (Central Processing Unit) y GPU (Graphics Processing Unit) en el dron permite un procesamiento eficiente,

reduciendo la latencia asociada con los algoritmos de aprendizaje profundo, sin embargo, la precisión del sistema depende en gran medida de la calidad de las cámaras y otros sensores utilizados [16]. Además de estos trabajos también se publicó un trabajo que presenta un sistema avanzado de detección de anomalías para drones que utiliza múltiples sensores y técnicas de aprendizaje automático, la combinación de diferentes tipos de sensores y algoritmos de detección de anomalías mejora la seguridad del dron, permitiendo la identificación temprana de problemas potenciales, además el sistema se diseñó para funcionar , lo que es crucial para aplicaciones donde la rápida respuesta es esencial, sin embargo, integrar y sincronizar múltiples sensores y aplicar algoritmos de aprendizaje automático puede ser técnicamente complejo y requerir una considerable capacidad de procesamiento [17].

Estos estudios han sido fundamentales para avanzar en la capacidad de los drones para ejecutar tareas de detección, seguimiento y vigilancia de forma efectiva. La continua integración de técnicas de inteligencia artificial con grandes volúmenes de datos ha permitido mejorar la precisión y eficiencia de estos sistemas.

Además, muchas de estas investigaciones destacan por su enfoque experimental riguroso y por el uso de grandes bases de datos recolectadas mediante UAVs en entornos reales. Esto ha permitido el desarrollo de sistemas con una precisión superiores, capaces de operar en condiciones ambientales variables y escenarios complejos.

3.2 Estado del arte

El desarrollo histórico desde los primeros avances en la detección y seguimiento de personas con drones hasta la integración de inteligencia artificial en estas plataformas ha sido un viaje progresivamente sofisticado y diverso. Este trabajo se encuentra en el contexto de este desarrollo continuo, aprovechando las capacidades de la IA para mejorar el seguimiento de personas con drones en aplicaciones civiles

El análisis del estado del arte revela desafíos significativos en términos de precisión, complejidad del entorno y eficiencia computacional en los algoritmos existentes. Por lo

tanto, el objetivo principal de este trabajo es desarrollar un algoritmo basado en inteligencia artificial que pueda adaptarse a estos desafíos, mejorando la precisión y eficiencia del sistema de identificación de personas con drones.

En la Tabla 1 se muestra, de una manera condensada y complementaria, los trabajos del estado del arte, que aportan mayor utilidad al trabajo de investigación, mostrando información importante sobre las métricas usadas.

Tabla 1. Métricas usadas en el estado del arte.

AÑO	Titulo	Autor	Métricas
2019	Real-Time Dense Monocular SLAM With Online Adapted Depth Prediction Network [4].	H. Luo, Y. Gao, Y. Wu, C. Liao, X. Yang y K. -T. Cheng.	• Absolute T1rajectory Error (ATE). • Percentage of Correctly Predicted Depth. • Runtime Analysis. • Performance of Dense Mapping. • Fusion of Depth Maps.
2019	Navigation of Simultaneous Localization and Mapping by Fusing RGB-D Camera and IMU on UAV [5].	Xi Dai; Yuxin Mao; Tianpeng Huang; Binbin Li; Deqing Huang.	• Absolute Trajectory Error (ATE). • Pose Estimation Accuracy. • 3D Point Cloud Map Quality. • Real-Time Performance. • Robustness to Environmental Changes.
2019	Review of vision-based Simultaneous Localization and Mapping [6].	Yao Zhang,Yiquan Wu.	• Absolute Trajectory Error (ATE). • Relative Pose Error (RPE). • Loop Closure Detection Accuracy. • Computational Efficiency. • Mapping Accuracy.
2020	Indirect Visual Simultaneous Localization and Mapping Based on Linear Models [7].	Chiang-Heng Chien, Chen-Chien James Hsu, Wei-Yen Wang, Hsin-Han Chiang.	• Relative Pose Error (RPE). • Relative Translation Error (RTE). • Relative Rotation Error (RRE). • Precision y Recall
2020	Faster R-CNN Deep Learning Model for Pedestrian Detection from	Goon Li Hung, Mohamad Safwan Bin Sahimi, Hussein	• Precision. • Recall. • F1 Score.

AÑO	Titulo	Autor	Métricas
	Drone Images [8].	Samma, Tarik Adnan Almohamad, Badr Lahasan.	• True Positive Rate (TPR). • False Positive Rate (FPR).
2021	An Enhanced Pedestrian Visual-Inertial SLAM System Aided with Vanishing Point in Indoor Environments [9].	Wennan Chai, Chao Li, Mingyue Zhang, Zhen Sun, Hao Yuan, Fanyu Lin, Qingdang Li.	• Absolute Trajectory Error (ATE). • Relative Pose Error (RPE). • Loop Closure Detection Accuracy. • Computational Efficiency.
2021	Design Guidelines on Deep Learning–based Pedestrian Detection Methods for Supporting Autonomous Vehicles [11].	Azzedine Boukerche, Mingzhi Sha	• Precision • Recall. • F1 Score.
2022	Deep Learning-Based Pedestrian Detection in Autonomous Vehicles: Substantial Issues and Challenges [12].	Sundas Iftikhar, Zuping Zhang, Muhammad Asim, Ammar Muthanna, Andrey Koucheryavy, Ahmed A. Abd El-Latif	• Precision. • Recall. • F1 Score. • Average Precision (AP). • Intersection over Union (IoU).
2022	Visual-Inertial Simultaneous Localization and Mapping: Dynamically Fused Point-Line Feature Extraction and Engineered Robotic Applications [13].	Linlin Xia, Deang Meng, Jingjing Zhang, Daochang Zhang, Zhiqi Hu	• Absolute Trajectory Error (ATE). • Relative Pose Error (RPE). • Computational Efficiency.
2023	Indoor Pedestrian-Following System by a Drone with Edge Computing and Neural Networks: Part 1 - System Design [14].	I. -C. Ryu, J. -I. Ham, J. -O. Park, J. -W. Joeng, S. -C. Kim y H. -S. Ahn	• TF (Transformation Frame). • Root Mean Square Error (RMSE). • Precision-Recall Curve

AÑO	Titulo	Autor	Métricas
2023	Monocular Depth Estimation for Autonomous UAV Navigation Based on Deep Learning [15].	S. Hensel, M. B. Marinov, A. Dreher, and D. Trendafilov	• Absolute Trajectory Error (ATE). • Percentage of Correct Depth Predictions. • Runtime Analysis • Normalized Cross Correlation (NCC).
2023	Towards Real-Time On-Drone Pedestrian Tracking in 4K Inputs [16].	Chanyoung Oh, Moonsoo Lee, Chaedeok Lim	• Precision. • Recall. • F1 Score. • Inference Time
2023	Enhancing Drone Security Through Multi-Sensor Anomaly Detection and Machine Learning [17].	Mohammed Y. Alzahrani1	• Precision, • Recall. • F1 Score. • Frame Rate.

En resumen, el desarrollo histórico desde los primeros avances en la detección y seguimiento de personas con drones hasta la integración de inteligencia artificial en estas plataformas ha sido un viaje progresivamente sofisticado y diverso. Este trabajo se encuentra en el contexto de este desarrollo continuo, aprovechando las capacidades de la IA para mejorar el seguimiento de personas con drones en aplicaciones civiles

El análisis del estado del arte revela desafíos significativos en términos de precisión, complejidad del entorno y eficiencia computacional en los algoritmos existentes. Por lo tanto, el objetivo principal de este trabajo es desarrollar un algoritmo basado en inteligencia artificial que pueda adaptarse a estos desafíos, mejorando la precisión y eficiencia del sistema de identificación de personas con drones.

3.3 Planteamiento del problema

El desarrollo de sistemas inteligentes capaces de detectar y clasificar personas en imágenes capturadas por drones representa un área de investigación activa y de alto impacto. En escenarios urbanos, la vigilancia y monitoreo mediante vehículos aéreos no tripulados (UAVs) ha cobrado gran relevancia debido a su capacidad para cubrir amplias

áreas, acceder a zonas de difícil alcance y capturar información visual. Sin embargo, la implementación de estos sistemas presenta múltiples desafíos tecnológicos que requieren soluciones avanzadas basadas en inteligencia artificial y visión por computadora.

Uno de los principales problemas está en la precisión y eficiencia de los algoritmos encargados de procesar las imágenes capturadas por los drones. La detección e identificación de personas en entornos dinámicos y no estructurados exige una capacidad robusta para enfrentar variaciones de iluminación, oclusiones, cambios en la apariencia de las personas y condiciones ambientales difíciles. Estos factores incrementan la complejidad del procesamiento de imágenes y pueden afectar significativamente la fiabilidad del sistema.

Además, en contextos de seguridad y vigilancia, se requiere que los algoritmos funcionen de forma autónoma. Por lo tanto, es necesario desarrollar un algoritmo que no solo sea preciso, sino también eficiente y adaptable a diversas condiciones.

Frente a esta problemática, existe la necesidad de diseñar un sistema basado en visión por computadora e inteligencia artificial que permita identificar personas con alta precisión a partir de imágenes obtenidas por drones, priorizando la adaptabilidad del algoritmo a diferentes escenarios. Resolver este problema contribuirá significativamente al desarrollo de sistemas inteligentes de monitoreo y vigilancia con un alto valor social y tecnológico.

3.4 Justificación

El desarrollo e implementación de algoritmos basados en técnicas de visión por computadora e inteligencia artificial ofrece una solución versátil y prometedora para mejorar el rendimiento de los drones en diversas aplicaciones. La aplicación de inteligencia artificial en este contexto ha demostrado su eficacia en una variedad de campos, incluida la mejora de algoritmos de control para sistemas robóticos.

La implementación de algoritmos de visión por computadora e inteligencia artificial proporciona una capacidad mejorada para detectar y seguir objetos utilizando la cámara de un dron. Estos algoritmos son capaces de adaptarse a diferentes condiciones ambientales y variables, lo que los hace ideales para aplicaciones de vigilancia y seguridad

[18], la identificación de personas utilizando drones equipados con algoritmos basados en visión por computadora es una aplicación emergente y de gran relevancia en este campo [12]. Además, la integración de técnicas de inteligencia artificial en el diseño de algoritmos para el control de drones ofrece la oportunidad de mejorar la estabilidad y precisión del vuelo. Los algoritmos mejorados pueden permitir una navegación más suave y controlada del dron, reduciendo los errores y mejorando su capacidad de seguimiento e identificación.

3.5 Hipótesis

El desarrollo e implementación de un algoritmo basado en visión por computadora e inteligencia artificial permitirá mejorar la precisión y eficiencia en la identificación de personas en imágenes capturadas por la cámara de un dron operando en entornos urbanos.

3.6 Objetivos

3.6.1 Objetivo general

Desarrollar e implementar un algoritmo basado en técnicas de visión por computadora e Inteligencia Artificial usando imágenes proporcionadas por la cámara de un dron para la identificación de personas.

3.6.2 Objetivos específicos

- Investigar, adaptar e implementar un algoritmo basado en técnicas de Inteligencia artificial para el procesamiento de imágenes y detección de personas.
- Identificar y usar una base datos de imágenes que permitan al entrenamiento de la red para la identificación de personas.
- Validar el algoritmo propuesto mediante un entorno de simulación para pruebas utilizando la base de datos propuesta.

4. Marco teórico

4.1 Vehículos Aéreos No Tripulados (UAVs)

Los Vehículos Aéreos No Tripulados (UAV), comúnmente conocidos como drones, se definen como aeronaves capaces de operar sin un piloto humano a bordo [18]. Estos sistemas han evolucionado desde aplicaciones exclusivamente militares hasta convertirse en herramientas versátiles en sectores civiles, industriales y de investigación. En el contexto de este trabajo, se utilizan vehículos de ala rotatoria, específicamente la configuración de cuadricóptero, debido a su capacidad de despegue y aterrizaje vertical (VTOL) y su habilidad para realizar vuelos estacionarios, características indispensables para tareas de vigilancia y seguimiento de objetivos estáticos o dinámicos.

4.1.1 Dinámica y Cinemática del Cuadricóptero

Los componentes de un dron cuadricóptero son: el marco, que da soporte físico a los otros componentes, consta de un centro y cuatro brazos; electrónica y una batería en el centro del marco; un motor y una hélice en cada brazo [19].

El cuadricóptero tiene seis grados de libertad a pesar de tener cuatro hélices, tiene tres grados de libertad traslacionales arriba/abajo, adelante/atrás, izquierda/derecha (up/down, forward/backward, left/right); y tres grados de libertad rotacionales guiñada, balanceo y cabeceo (yaw, roll y pitch).

El dron para su uso en pruebas de simulación y experimental tiene una configuración de vuelo que consiste en dos motores frontales y dos motores laterales. Las ecuaciones del modelo dinámico del cuadricóptero se basan en el formalismo de Newton-Euler [20], donde la dinámica no lineal se obtiene en coordenadas inerciales y fijas en el cuerpo, utilizando la configuración North-East-Down (NED) como se observa en la Figura 1.

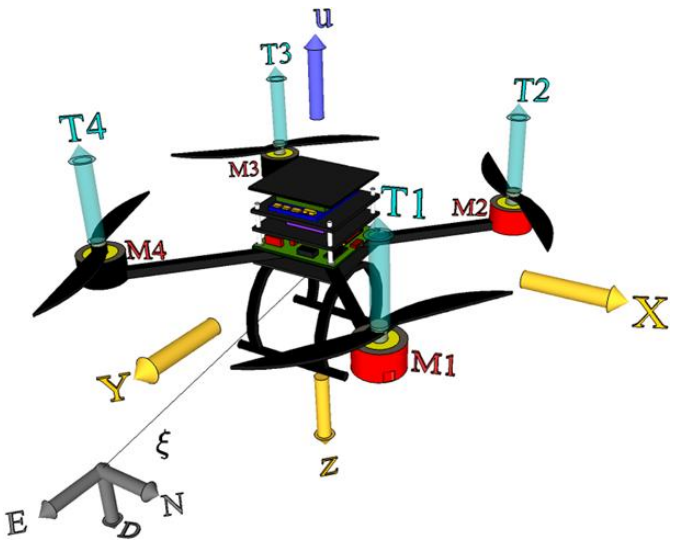


Figura 1. Diagrama de la configuración NED del modelo dinámico del Dron [20] .

Estableciendo $\{N, E, D\}$ como el marco de referencia inercial y $\{X, Y, Z\}$ representando el marco fijo del cuerpo. El vector de posición del centro de masa (o COG) del cuadricóptero se denota por $\xi = [x, y, z]^T$, que representa las coordenadas de posición del cuadricóptero en relación con el marco inercial NED. El vector de orientación del cuadricóptero con respecto al marco inercial se expresa por $\eta = [\phi, \theta, \psi]^T$, donde ϕ, θ, ψ son los ángulos de Euler de Roll, Pitch y Yaw, respectivamente. La dinámica no lineal completa del cuadricóptero se puede expresar como se muestra en las ecuaciones (1) y (2) [21].

$$m\ddot{\boldsymbol{\xi}} = -mg\boldsymbol{D} + \boldsymbol{R}F, \qquad (1)$$

$$I\dot{\boldsymbol{\omega}} = -\boldsymbol{\omega} \times I\boldsymbol{\omega} + \boldsymbol{M}, \qquad (2)$$

donde $\boldsymbol{R}$ es la matriz de rotación que relaciona el marco inercial con el marco del cuerpo, esta matriz resulta del producto entre otras tres matrices $(R_x(\phi), R_y(\theta)\ y\ R_z(\psi))$, como se muestra en (3) y (4).

$$\mathrm{R_x}(\phi) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & c\phi & -s\phi \\ 0 & s\phi & c\phi \end{bmatrix},\ \mathrm{R_y}(\theta) = \begin{bmatrix} c\theta & 0 & s\theta \\ 0 & 1 & 0 \\ -s\theta & 0 & c\theta \end{bmatrix}, R_z(\psi) = \begin{bmatrix} c\psi & -s\psi & 0 \\ s\psi & c\psi & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad (3)$$

y

$$\mathbf{R} = \boldsymbol{R}(\psi)\boldsymbol{R}(\theta)\boldsymbol{R}(\phi) = \begin{bmatrix} c\theta c\psi & s\psi s\theta c\psi - c\phi s\psi & c\phi s\theta c\psi + s\phi s\psi \\ c\theta s\psi & s\phi s\theta s\psi + c\phi c\psi & s\theta c\phi s\psi - s\phi c\psi \\ -s\theta & s\phi c\theta & c\theta c\phi \end{bmatrix} \quad (4)$$

donde s y c son abreviaturas de las funciones seno y coseno, respectivamente.

F es la fuerza total aplica al cuadricóptero, m es la masa total, $g = 9.81\frac{m}{s}$ es la constante de la gravedad, $\boldsymbol{\omega} = [p, q, r]^T$ representa las velocidades angulares del vehículo expresadas en el marco fijo, $I = diag(I_x, I_y, I_z)$ es la matriz diagonal de inercia, $\boldsymbol{D} = [0,0,1]^T$ es un vector unitario y $\boldsymbol{M} = [M_x, M_y, M_z]$ es el torque total.

Estableciendo a $U_1 = \sum_{i=1}^{4} T_i$ como la fuerza de empuje total aplicada al cuadricóptero, la cual es generada por los cuatros rotores y asumiendo que esta fuerza solo tiene un componente en la dirección de z, entonces la fuerza se define como $\boldsymbol{F} = [0,0,-U_1]^{\boldsymbol{T}}$.

La relación entre $\dot{\eta}$ con $\boldsymbol{\omega}$ está formada por tres rotaciones sucesivas, como se muestra en (5) [21].

$$\boldsymbol{\omega} = \begin{bmatrix} p \\ q \\ r \end{bmatrix} = R_x(\phi)R_y(\theta)\begin{bmatrix} 0 \\ 0 \\ \dot{\psi} \end{bmatrix} + R_x(\phi)\begin{bmatrix} 0 \\ \dot{\theta} \\ 0 \end{bmatrix} + \begin{bmatrix} \dot{\phi} \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 1 & 0 & s\theta \\ 0 & c\phi & -s\phi c\theta \\ 0 & s\phi & c\phi c\theta \end{bmatrix}\begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix}. \quad (5)$$

Estableciendo a W como (6) [1].

$$\mathrm{W} = \begin{bmatrix} 1 & 0 & s\theta \\ 0 & c\phi & -s\phi c\theta \\ 0 & s\phi & c\phi c\theta \end{bmatrix} \quad (6)$$

entonces $\boldsymbol{\omega}$ es igual a (7).

$$\boldsymbol{\omega} = W\dot{\boldsymbol{\eta}} \tag{7}$$

Despejando a $\dot{\boldsymbol{\eta}}$ de la ecuación anterior tenemos (8).

$$\dot{\boldsymbol{\eta}} = W^{-1}\boldsymbol{\omega} = \mathrm{C}\boldsymbol{\omega} \tag{8}$$

donde C es (1.9).

$$\mathrm{C} = \mathrm{W}^{-1} = \begin{bmatrix} 1 & s\phi t\theta & -c\phi t\theta \\ 0 & c\phi & s\phi \\ 0 & -s\phi/c\theta & c\phi/c\theta \end{bmatrix}, \tag{9}$$

con t la abreviatura de la función tangente.

Derivando con respecto al tiempo a la ecuación (8), se obtiene (10).

$$\ddot{\boldsymbol{\eta}} = \dot{C}\boldsymbol{\omega} + C\dot{\boldsymbol{\omega}} = \dot{C}\boldsymbol{\omega} + I^{-1}C(I\dot{\boldsymbol{\omega}}). \tag{10}$$

Reemplazando la ecuación (2) en (10) se obtiene

$$\ddot{\boldsymbol{\eta}} = \dot{C}\boldsymbol{\omega} + I^{-1}C(-\boldsymbol{\omega} \times I\boldsymbol{\omega} + \boldsymbol{M}) \tag{11}$$

defiendo un vector auxiliar $\widetilde{M}$ relacionado al torque generalizado de M y basado en (1.2) se obtiene

$$\widetilde{M} = \begin{bmatrix} \widetilde{M}_x \\ \widetilde{M}_y \\ \widetilde{M}_z \end{bmatrix} = \dot{C}\boldsymbol{\omega} - I^{-1}C\boldsymbol{\omega} \times I\boldsymbol{\omega} + I^{-1}C\boldsymbol{M}. \tag{12}$$

4.1.2 Simplificación del Modelo: Cinemática de Punto

El movimiento de un cuadricóptero se rige fundamentalmente por la mecánica de cuerpos rígidos, donde las fuerzas aerodinámicas generadas por los rotores inducen momentos que alteran la actitud (orientación) y producen aceleraciones lineales. Sin embargo, abordar el problema desde la perspectiva dinámica (Modelo de Newton-Euler) implica controlar directamente los torques de los motores, lo cual no es posible en la plataforma Tello EDU

debido a su arquitectura cerrada.

En su lugar, este trabajo emplea un Modelo Cinemático de Punto. Esta aproximación abstrae la dinámica rotacional y asume que el vehículo cuenta con un controlador de bajo nivel lo suficientemente rápido para compensar las perturbaciones y seguir consignas de velocidad [22]. Bajo este modelo, el estado del sistema se define únicamente por su posición en el espacio cartesiano $P = [x, y, z]^T$. La evolución de este estado en el tiempo se describe mediante la relación cinemática de primer orden:

$$\dot{P}(t) = R(\psi) \cdot V_{cmd}(t) \quad (13)$$

Donde:

- $\dot{P}(t)$ es la velocidad instantánea del dron respecto al marco inercial (suelo).
- $V_{cmd}(t) = \left[v_x, v_y, v_z\right]^T$ son los comandos de velocidad lineal expresados en el marco del cuerpo (fijos al dron).
- $R(\psi)$ es la matriz de rotación básica alrededor del eje Z (Yaw), que alinea el frente del dron con las coordenadas globales.

Para fines de la identificación del sistema propuesta en el Capítulo 4.4, se asume que la orientación del dron se mantiene alineada con la trayectoria o que las velocidades son bajas, permitiendo simplificar la relación a una identidad directa [23]:

$$P_{k+1} \approx P_k + V_k \cdot \Delta t \quad (14)$$

Esta ecuación discreta es la que aprende la red neuronal: predice qué velocidad V_k se requiere para lograr el desplazamiento deseado $P_{k+1} - P_k$ en el intervalo de tiempo Δt .

4.1.3 Dron Tello

El dron utilizado en el presente proyecto es el Tello EDU, desarrollado por Ryze Robotics en colaboración con DJI e Intel. Este modelo se caracteriza por su portabilidad, bajo peso y capacidad de programación, lo que lo convierte en una herramienta idónea para fines

educativos, de investigación y desarrollo de algoritmos de control e inteligencia artificial. El Tello EDU pertenece a la categoría de microdrones programables, diseñado específicamente para la enseñanza de robótica, visión por computadora y vuelo autónomo. Su estructura compacta y su integración con múltiples entornos de programación (como MATLAB, Python y Scratch) permiten experimentar con técnicas avanzadas de navegación, control y procesamiento de imágenes en un entorno seguro y de bajo riesgo. Entre sus principales especificaciones técnicas, se destacan las siguientes [24]:

Tabla 2. Características del Dron Tello.

Característica	Descripción
Fabricante	Ryze Robotics (en colaboración con DJI e Intel)
Modelo	Tello EDU
Tipo de dron	Cuadricóptero programable de pequeño tamaño
Peso total	Aproximadamente 87 g
Dimensiones	98 × 92.5 × 41 mm
Procesador	Intel Movidius Myriad 2 VPU
Cámara	RGB de 5 megapíxeles
Resolución de video	1280×720 píxeles (HD) a 30 fps
Distancia máxima de vuelo	100 metros
Altura máxima recomendada	10 metros
Velocidad máxima	8 m/s
Tiempo de vuelo	Aproximadamente 13 minutos por batería
Conectividad	Wi-Fi 2.4 GHz (control mediante app o PC)
Batería	LiPo 3.8 V / 1100 mAh
Compatibilidad de software	MATLAB, Python, Scratch, Swift
Controlador	Ryze SDK / Tello EDU SDK (API abierta)

El Tello EDU ofrece una plataforma flexible para la implementación de algoritmos de navegación y visión por computadora. En este proyecto se aprovechan sus capacidades de transmisión de video y su compatibilidad con MATLAB para capturar imágenes y

procesarlas utilizando la arquitectura YOLO para la detección de personas [25]. No obstante, es importante destacar que la topología de control del Tello EDU es cerrada, lo cual impide el acceso o modificación de los algoritmos de control dinámico de bajo nivel (estabilización PID) preprogramados en el firmware del vehículo . Debido a esta restricción, no es posible implementar leyes de control directo sobre los torques de los motores. En consecuencia, este trabajo adopta un enfoque basado en la identificación del sistema, donde se envían comandos de velocidad de alto nivel generados por una red neuronal que ha aprendido la cinemática inversa del dron, actuando como un planificador de movimiento sobre la capa de control interna del vehículo .

4.2 Procesamiento de Imágenes

El preprocesamiento de imágenes constituye una etapa crítica en el flujo de trabajo de aprendizaje profundo (pipeline), cuyo propósito es acondicionar el dominio de los datos de entrada para maximizar la convergencia del algoritmo de optimización y la capacidad de generalización de la red neuronal. Esta fase aborda desafíos inherentes como la variabilidad en la escala de los objetos, las condiciones de iluminación no controladas y la inconsistencia en las dimensiones de captura [26].

4.2.1 Acondicionamiento del Tensor de Entrada

Para que una red neuronal convolucional (CNN) como YOLO pueda procesar un batch de datos de manera eficiente, la consistencia dimensional y de valores es fundamental [26]:

- Redimensionamiento a Tamaño Fijo: Se requiere que las imágenes de entrada tengan un tamaño uniforme (ej., 640 x 640 píxeles). Esta estandarización asegura que la red opere con un campo receptivo constante, estabilizando la complejidad computacional. Se emplea la técnica de Letterbox Padding (relleno de bordes) para mantener la relación de aspecto original de la imagen, evitando la deformación geométrica de los objetos (personas), lo cual es crítico para preservar las

características visuales aprendidas.

- Normalización de Intensidad de Píxeles: Los valores de intensidad de los píxeles deben ser escalados de su rango original (generalmente 0 a 255) a un rango acotado (ej., [0, 1]). Esta transformación reduce la varianza de las características de entrada, evitando que gradientes de gran magnitud desestabilicen el proceso de descenso del gradiente durante el entrenamiento.

4.2.2 Codificación de Etiquetas (Bounding Box Encoding)

El formato de las etiquetas de las cajas delimitadoras debe adaptarse al modelo de regresión de YOLO. Las coordenadas originales, que generalmente definen la esquina superior izquierda y la inferior derecha (x_{min}, y_{min}, x_{max}, y_{max}), deben ser transformadas a un formato normalizado de centro y dimensiones (x_c, y_c, w, h) [26].

- Esta transformación es un requisito teórico para que la red pueda realizar una predicción directa de la regresión. Se hace una conversión de las anotaciones a las coordenadas normalizadas entre 0 y 1 [27], las coordenadas (x_c, y_c, w, h) se calculan como una fracción del ancho W y alto H total de la imagen como se define de (15) a *(18*):.

$$x_c = \frac{x_{min} + x_{max}}{2W} \quad (15)$$

$$y_c = \frac{y_{min} + y_{max}}{2H} \quad (16)$$

$$w = \frac{x_{max} - x_{min}}{W} \quad (17)$$

$$h = \frac{y_{max} - y_{min}}{H} \quad (18)$$

donde W y H son el ancho y alto de la imagen.

- Filtrado de Clases: Teóricamente, la detección se optimiza al reducir el espacio de

clases de la base de datos a la clase de interés . Esto dirige el poder predictivo del modelo a la tarea específica del proyecto, mejorando la métrica de Precisión.

- Redimensionamiento de todas las imágenes a 640×640 píxeles para mantener consistencia en la entrada del modelo.

4.2.3 Aumento de Datos (Data Augmentation)

Para mitigar el riesgo de sobreajuste (*overfitting*) y mejorar la robustez del modelo ante la variabilidad del mundo real, se implementó una estrategia de aumento de datos *on-the-fly* durante el entrenamiento. Esta técnica expande artificialmente el espacio de muestras mediante transformaciones estocásticas [28]:

- Transformaciones Geométricas: Se aplicaron rotaciones aleatorias, traslaciones y escalados. Esto fuerza a la red a aprender representaciones de "persona" invariantes a la orientación y al tamaño aparente, simulando los cambios de perspectiva y altitud típicos del vuelo de un dron.

- Transformaciones Fotométricas: Se introdujeron variaciones aleatorias en el espacio de color HSV (Matiz, Saturación, Valor). Esto simula condiciones de iluminación diversas (sombras, sobreexposición, atardecer), asegurando que el detector no dependa exclusivamente de características de color específicas que podrían cambiar según la hora del día.

4.3 Detección de Objetos con Aprendizaje Profundo

El reconocimiento de personas desde imágenes aéreas representa una tarea desafiante debido a factores como la variación de escala, la oclusión parcial, la iluminación cambiante y el movimiento inherente a la plataforma de captura. En el estado del arte

actual, la solución estándar para abordar estos problemas reside en el uso de Redes Neuronales Convolucionales (CNN).

Dentro de este paradigma, las arquitecturas de detección de una sola etapa, como la familia YOLO (You Only Look Once), se han consolidado como una de las herramientas más efectivas en la literatura científica. Estos modelos son frecuentemente seleccionados en aplicaciones de vehículos aéreos no tripulados debido a su capacidad para balancear una alta precisión de detección con un bajo costo computacional, permitiendo el procesamiento de imágenes .

4.3.1 Redes Neuronales Convolucionales (CNN)

Las Redes Neuronales Convolucionales (CNN) representan la arquitectura fundamental en el campo del Aprendizaje Profundo para el procesamiento de datos con estructura de rejilla, como las imágenes digitales. A diferencia de las redes neuronales tradicionales (Perceptrón Multicapa), que requieren aplanar la imagen en un vector unidimensional perdiendo la información espacial, las CNN están diseñadas para procesar tensores tridimensionales (ancho, alto y canales de color), preservando la jerarquía espacial y las relaciones locales entre píxeles [29].

El funcionamiento de una CNN se inspira biológicamente en la organización del córtex visual animal, donde neuronas individuales responden a estímulos en regiones restringidas del campo visual conocidas como campos receptivos. Matemáticamente, esto se traduce en tres mecanismos arquitectónicos clave que las hacen superiores para la visión por computadora: conectividad local, pesos compartidos y submuestreo.

La arquitectura típica de una CNN se compone de una secuencia de capas apiladas que transforman el volumen de entrada en un vector de probabilidades de clase. Las capas principales se describen a continuación:

1. Capa Convolucional (Convolutional Layer):Es el núcleo de la red. En esta capa, se aplica una operación matemática de convolución discreta entre la imagen de entrada y un conjunto de filtros aprendibles (kernels). Cada filtro K de

dimensiones pequeñas (por ejemplo, 3×3) se desliza sobre la imagen realizando el producto punto con los píxeles subyacentes.La operación para un píxel de salida (i, j) se define como [29]:

$$S(i,j) = (I * K)(i,j) = \sum_{m} \sum_{n} I(i-m, j-n)K(m,n). \qquad (19)$$

Donde:

- I: Es la imagen de entrada (Input).
- K: Es el kernel o filtro.
- S: Es el mapa de características resultante.
- (i, j): Son las coordenadas del píxel de salida.
- (m, n): Son las coordenadas relativas dentro del kernel sobre las que se suma.

El resultado es un Mapa de Características (*Feature Map*) que detecta patrones específicos como bordes, texturas o esquinas. A medida que se profundiza en la red, estos mapas representan conceptos más abstractos y semánticos (ojos, rostros, cuerpos).

2. Capa de Agrupamiento (Pooling Layer): Esta capa reduce progresivamente las dimensiones espaciales de la representación para disminuir la cantidad de parámetros y el costo computacional, además de controlar el sobreajuste. La técnica más común es el Max Pooling, que selecciona el valor máximo en una ventana definida (generalmente 2×2), descartando el resto. Esto proporciona a la red la propiedad de invarianza a la traslación, permitiendo que un objeto (como una persona) sea detectado independientemente de su posición exacta en el cuadro.
3. Capa Completamente Conectada (Fully Connected Layer): Tras múltiples etapas de convolución y agrupamiento, los mapas de características resultantes se aplanan (flatten) en un vector unidimensional. Este vector alimenta a una red neuronal clásica (MLP) que realiza la clasificación final, asignando probabilidades a las diferentes clases (por ejemplo, "persona" vs. "fondo").

La Figura 2, ilustra el mecanismo fundamental de la operación de convolución discreta en dos dimensiones, sin inversión del núcleo (kernel flipping), comúnmente implementada como correlación cruzada en bibliotecas de aprendizaje profundo.El proceso se describe en tres componentes:

1. Tensor de Entrada (Input): Representado a la izquierda, es una matriz bidimensional de valores (píxeles o características) denotados como $a, b, c, \ldots, l$.
2. Núcleo o Kernel: Es una matriz de dimensiones reducidas (en este caso 2×2) con parámetros aprendibles w, x, y, z. Este núcleo actúa como un detector de características que se desliza sobre la entrada.
3. Mapa de Características (Output): Es el resultado de la operación. Cada elemento de salida se calcula mediante la suma ponderada de una región local de la entrada.

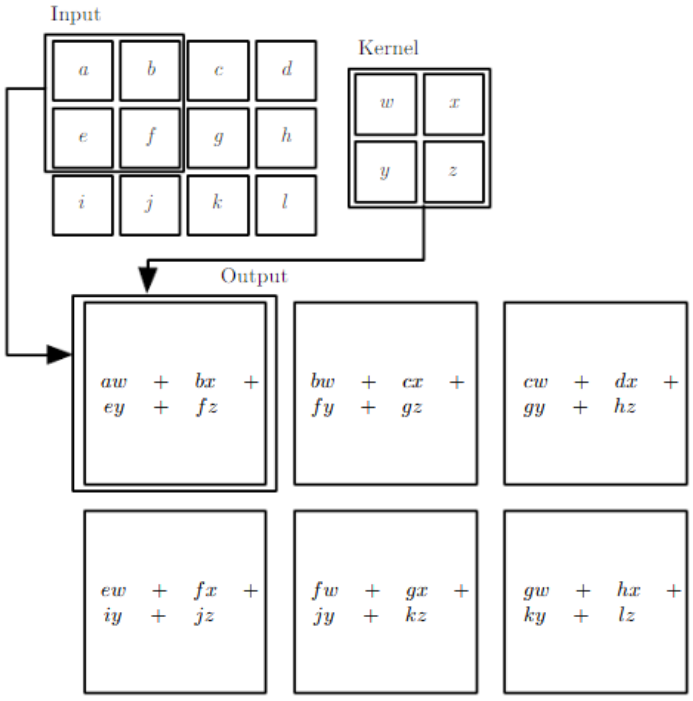


Figura 2. Ejemplo de convolución 2D discreta. Se restringe la salida a posiciones donde el núcleo (kernel) se superpone completamente a la imagen ("convolución válida")[29].

Como se observa en la Figura 2, el valor de la esquina superior izquierda del mapa de salida se obtiene posicionando el kernel sobre la región correspondiente de la entrada (elementos a, b, e, f) y calculando el producto elemento a elemento:

$$Output_{(0,0)} = (a \cdot w) + (b \cdot x) + (e \cdot y) + (f \cdot z) \qquad (20)$$

Este proceso de "ventana deslizante" se repite a lo largo y ancho de la matriz de entrada, generando un mapa de características que preserva la disposición espacial de los patrones detectados. Esta operación local permite que la red sea eficiente computacionalmente (conectividad dispersa) y capaz de detectar el mismo patrón en diferentes ubicaciones de la imagen (compartición de parámetros).

En el contexto de esta investigación, las CNN actúan como el extractor de características (*backbone*) sobre el cual se construye el algoritmo de detección YOLO. Su capacidad para aprender automáticamente qué características son relevantes para distinguir a una persona del entorno, sin intervención humana manual, es lo que permite al sistema operar eficazmente en entornos aéreos complejos y dinámicos.

4.3.2 Arquitectura Yolo

La detección de objetos es una tarea crítica en visión por computadora que requiere tanto la clasificación como la localización precisa de los objetos dentro de una imagen. Históricamente, este proceso se realizaba en dos etapas (como en R-CNN): la generación de regiones candidatas y su posterior clasificación. El enfoque YOLO (*You Only Look Once*) revolucionó este campo al tratar la detección de objetos como un problema de regresión de una sola etapa. El modelo predice las cajas delimitadoras (*bounding boxes*) y las probabilidades de clase de forma simultánea, lo que resulta en una velocidad y eficiencia significativamente mayores.

YOLO permite la detección y clasificación de objetos mediante un único procesamiento de la imagen [30]. La arquitectura consta de tres bloques principales:

- *Backbone (*Columna Vertebral*):* Es la red de extracción de características. Procesa la imagen de entrada para generar mapas de características ricos en información semántica.
- *Neck (*Cuello*):* Conecta el *Backbone* con la cabeza de detección. Su función es recolectar y fusionar características de diferentes escalas generadas por el *Backbone*, siendo crucial para la detección de objetos de tamaño variable (particularmente importante en el *VisDrone-Dataset*). A menudo utiliza estructuras como PANet (*Path Aggregation Network*).
- *Head (*Cabeza de Detección*):* Es la parte final que utiliza las características procesadas para realizar las predicciones finales. En la mayoría de las versiones de YOLO, la cabeza genera un tensor que contiene la información de la caja delimitadora (coordenadas *x, y, w, h*), la puntuación de confianza y las probabilidades de clase.

En cada celda de la cuadrícula, el modelo predice :

$$(x, y, w, h, C, p_{cls}) \qquad (21)$$

Donde (x,y) son las coordenadas normalizadas del centro de la caja, w y h su ancho y alto, C la confianza de detección y p_{cls} la probabilidad asociada a la clase (persona).

El funcionamiento de YOLO se basa en los siguientes mecanismos:

- División de la Imagen: La imagen se divide en una cuadrícula (*grid*) de *S* x *S* celdas. Si el centro de un objeto cae en una celda de la cuadrícula, esa celda es responsable de predecir el objeto.
- Predicciones por Celda: Cada celda predice *B* cajas delimitadoras, la confianza para cada caja y las *C* probabilidades de clase.
    - Caja Delimitadora: Viene dada por cinco valores: las coordenadas (*x, y)* del centro, la altura *h* y la anchura *w*, y una puntuación de confianza C_i .
- *Non-Maximum Suppression (NMS):* Es un paso de post-procesamiento esencial

que se utiliza para eliminar las múltiples cajas delimitadoras redundantes que predicen el mismo objeto, seleccionando solo la caja con la mayor puntuación de confianza.

La Figura 3 ilustra cómo cada celda de la cuadrícula genera múltiples predicciones: las coordenadas de la caja delimitadora, una puntuación de confianza (que indica la probabilidad de que la caja contenga un objeto) y las probabilidades de cada clase. Este enfoque de predicción directa elimina la necesidad de múltiples pasos, lo que diferencia radicalmente a YOLO de los modelos de dos etapas y justifica su velocidad superior.

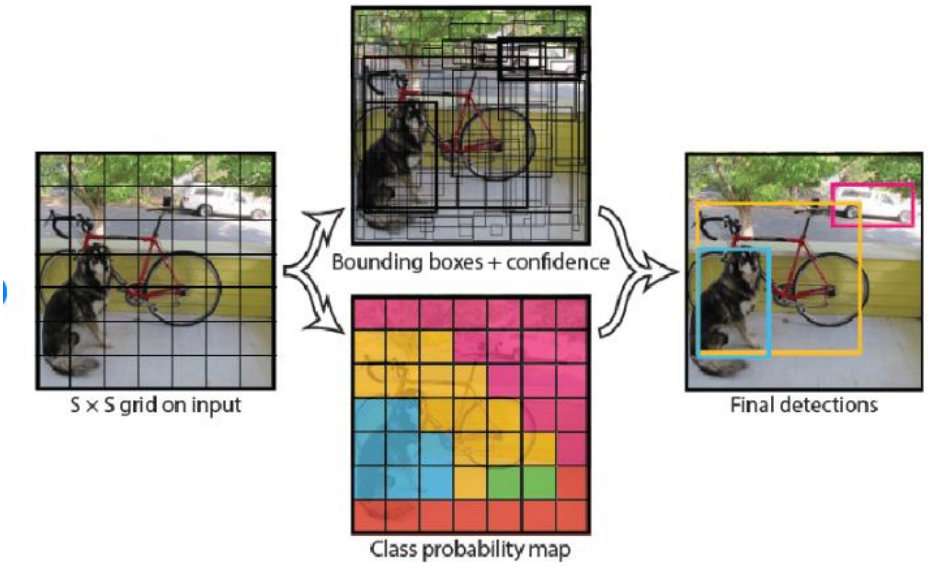


Figura 3. Diagrama conceptual del principio de funcionamiento de la arquitectura YOLO. Se muestra cómo la imagen se divide en una cuadrícula S x S, donde cada celda predice simultáneamente las cajas delimitadoras (localización) y las probabilidades de clase (clasificación), lo que unifica el proceso de detección en una sola etapa.

En el ámbito del *Deep Learning* y la visión por computadora, la designación numérica de versiones (como v3, v5 o v8) no solo indica un orden cronológico de desarrollo, sino que clasifica a una red neuronal por los cambios fundamentales en su arquitectura y estrategia algorítmica. Cada nueva versión de un modelo como YOLO representa la culminación de un esfuerzo de investigación para resolver las limitaciones de la versión anterior,

enfocándose en un trilema central de la detección de objetos:

- Velocidad de Inferencia: Optimizar la arquitectura para reducir el tiempo de procesamiento.
- Precisión (*mAP*): Mejorar la exactitud de la clasificación y la localización de la caja delimitadora.
- Eficiencia Computacional: Reducir la complejidad y los recursos necesarios para el entrenamiento y despliegue.

Por lo tanto, la transición de, por ejemplo, YOLOv5 a YOLOv8 no es un simple *update* de software, sino una clase de desarrollo arquitectónico que incorpora innovaciones teóricas significativas, como el paso de un diseño basado en anclajes (*Anchor-Based*) a uno libre de anclajes (*Anchor-Free*). Esta evolución es lo que permite que los modelos se adapten de mejor manera a desafíos complejos, como la detección de objetos pequeños en escenarios de drones .

En el marco de los detectores de una sola etapa, las versiones YOLOv5 y YOLOv8 representan el estado del arte en eficiencia y precisión, siendo cruciales para la detección en entornos aéreos [31], [32]. En la Tabla 3 se hace una comparación entre estas dos versiones.

Tabla 3. Evolución Arquitectónica: YOLOv5 y YOLOv8.

Característica	YOLOv5 (Ultralytics)	YOLOv8 (Ultralytics)	Relevancia
Backbone	CSPDarknet optimizado (Cross-Stage Partial Network).	C2f (C2 module con dos cuellos de botella) que mejora la capacidad de extracción de características y el flujo de gradientes.	Muestra la mejora continua en la eficiencia de la extracción de características de las CNN.

Característica	YOLOv5 (Ultralytics)	YOLOv8 (Ultralytics)	Relevancia
Head (*Coupled* vs. *Decoupled*)	Utiliza una cabeza parcialmente acoplada (*coupled*).	Utiliza una cabeza totalmente desacoplada (*Decoupled Head*).	El desacoplamiento separa la clasificación y la localización, mejorando la convergencia del entrenamiento y la precisión, especialmente en objetos pequeños.
Estrategia de Detección	Anchor-Based (Detección basada en anclajes predefinidos).	Anchor-Free (Detección libre de anclajes).	La eliminación de *anchors* simplifica el diseño de la red, reduce el número de predicciones por caja y hace el modelo más generalizable a distintos tamaños de *datasets*.
Función de Pérdida	Utiliza BCE Loss para la clasificación y CIoU Loss para la localización.	Incorpora funciones de pérdida más recientes y robustas, como VFL Loss (*Varifocal Loss*) para la clasificación.	La elección de VFL es un avance teórico que busca ponderar mejor las muestras positivas y negativas, crucial para conjuntos de datos desequilibrados.

Función de perdida

El entrenamiento del modelo optimiza una función de pérdida compuesta por tres términos [33]:

$$L = L_{loc} + L_{conf} + L_{cls} \qquad (22)$$

Donde L_{loc} es la pérdida de localización (error en las coordenadas de las cajas), L_{loc} pérdida de confianza de objeto y L_{cls} pérdida de clasificación de la clase.

Pérdida de localización

Para medir la precisión espacial entre las cajas predichas y las reales, YOLO utiliza la *Complete Intersection over Union* (CIoU), la cual penaliza no solo la superposición, sino también la distancia y la relación de aspecto entre las cajas (*25*):

$$L_{loc} = 1 - \mathrm{CIoU}\left(B_p, B_{gt}\right) \quad (23)$$

Donde:

$$\mathrm{CIoU} = \mathrm{IoU} - \frac{\rho^2\left(b_p, b_{gt}\right)}{c^2} - \alpha v \quad (24)$$

Aquí, $\rho^2(\mathrm{b}_p, \mathrm{b}_{gt})$representa la distancia euclidiana entre los centros de las cajas predicha y real, c es la diagonal del cuadro mínimo que las contiene, v evalúa la consistencia del aspecto (relación ancho–alto), y α es un factor de ajuste.

Pérdida de confianza

La pérdida de confianza mide la probabilidad de que una celda contenga un objeto y se calcula mediante *Binary Cross-Entropy* (BCE) :

$$L_{conf} = -\frac{1}{N}\sum_{i=1}^{N}[y_i \log(\hat{y}_i) + (1 - y_i)\log(1 - \hat{y}_i)] \quad (25)$$

donde $y_i \in \{0,1\}$indica si existe un objeto en la celda, $\hat{y}_i$ es la probabilidad predicha y N el número total de predicciones.

Pérdida de clasificación

Finalmente, la pérdida de clasificación evalúa la correspondencia entre la clase real y la predicha, también mediante *Binary Cross-Entropy* (28):

$$L_{cls} = -\frac{1}{C}\sum_{c=1}^{C}[p_c \log(\hat{p_c}) + (1 - p_c)\log(1 - \hat{p_c})] \quad (26)$$

donde C es el número de clases (en este caso $C = 1$, correspondiente a "persona"), p_c es la etiqueta real y $\hat{p}_c$ la probabilidad predicha.

Función de pérdida total ponderada

La función de pérdida total utilizada durante el entrenamiento combina los tres términos anteriores con pesos de ajuste (29):

$$L = \lambda_{loc} L_{loc} + \lambda_{conf} L_{conf} + \lambda_{cls} L_{cls} \quad (27)$$

donde los coeficientes λ_{loc}, λ_{conf}y λ_{cls} regulan la contribución de cada componente (comúnmente 1.0, 1.0 y 0.5, respectivamente).

Esta formulación permite optimizar de manera conjunta la precisión espacial, la confianza de detección y la clasificación del objeto, logrando un balance entre localización y exactitud en las predicciones.

4.3.3 Métricas de evaluación

La evaluación de modelos de detección de objetos, especialmente en entornos aéreos complejos , requiere un conjunto de métricas robustas que midan tanto la precisión de la localización como la exactitud de la clasificación [34].

Intersección sobre Unión (Intersection over Union, IoU)

El IoU es la métrica fundamental que evalúa la precisión de la localización de las cajas delimitadoras. Su valor teórico oscila entre 0 (sin superposición) y 1 (superposición perfecta).

El IoU se define como el cociente entre el área de la intersección (superposición) y el área de la unión de la caja predicha (B_{pred}) por el modelo y la caja real (ground truth, B_{gt}) [35]:

$$IoU = \frac{|B_{pred} \cap B_{gt}|}{|B_{pred} \cup B_{gt}|} \quad (28)$$

Para fines de evaluación, se establece un umbral de IoU (ej. 0.5 o 0.75). Una predicción solo se considera un Verdadero Positivo (TP) si su IoU supera este umbral, garantizando que la caja predicha no solo contiene el objeto, sino que lo delimita correctamente.

Métricas de Clasificación: Precision, Recall y F1

A partir de la definición de un Verdadero Positivo (TP) mediante el umbral de IoU, se definen las métricas de clasificación, que miden la exactitud del modelo para determinar la presencia o ausencia de personas.

- *Precision* (Precisión): Mide la pureza de las predicciones positivas. Responde a la pregunta: De todas las cajas que el modelo predijo como 'persona', ¿cuántas realmente lo eran? Es crucial en escenarios de alta seguridad donde los Falsos Positivos (FP) (marcar un objeto que no es una persona) son costosos (29):

$$Precision = \frac{TP}{TP + FP} \quad (29)$$

- *Recall* (Exhaustividad): Mide la capacidad del modelo para encontrar todos los objetos. Responde a la pregunta: De todas las personas reales en la imagen, ¿cuántas fueron detectadas por el modelo? Es esencial en la vigilancia por drones donde los Falsos Negativos (FN) (no detectar una persona presente) pueden ser críticos(30):

$$Recall = \frac{TP}{TP + FN} \quad (30)$$

- Medida F1: Es la media armónica entre *Precision* y *Recall*. Se utiliza para evaluar el desempeño general de un modelo, especialmente cuando existe un desequilibrio entre las clases o cuando se busca un balance entre minimizar los Falsos Positivos y los Falsos Negativos (31).

$$F1 = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall} \quad (31)$$

Métrica Global de Rendimiento: mAP (mean Average Precision)

El mAP (mean Average Precision) es la métrica estándar de la industria para evaluar el rendimiento global de los detectores de objetos. Su valor es el principal indicador de la calidad del modelo.El mAP se calcula como el promedio de la Precisión Promedio (AP_i) sobre todas las clases (N). Dado que este proyecto se enfoca únicamente en la clase persona (N=1 tras el filtrado), el mAP es equivalente al AP de esa única clase (32) :

$$mAP = \frac{1}{N}\sum_{i=1}^{N} AP_i \tag{32}$$

Donde N es el número de clases (N=1).

El AP_i se calcula integrando la curva *Precision-Recall* (*PR-Curve*). Esta curva muestra la relación inversa entre *Precision* y *Recall* a medida que se varía el umbral de confianza del modelo. El área bajo la curva PR es la que define la Precisión Promedio, capturando el desempeño del modelo en todos los niveles de confianza.

4.4 Identificación de Sistemas Dinámicos

La identificación de sistemas es el proceso de construir modelos matemáticos de sistemas dinámicos a partir de mediciones de sus entradas y salidas. Como se establece en la teoría de control, el objetivo es determinar una clase de modelos $\hat{\mathcal{P}} \subset \mathcal{P}$ tal que el comportamiento del modelo identificado aproxime al de la planta real P en algún sentido deseado, minimizando una norma de error [23] :

$$\left|\left|y - \hat{y}\right|\right| = \left|\left|P(u) - \hat{P}(u)\right|\right| \leq \varepsilon, \quad u \in U \tag{33}$$

Dado que el dron Tello EDU presenta una arquitectura cerrada donde los parámetros dinámicos internos (masa, inercia, coeficientes de arrastre) son desconocidos, este trabajo adopta un enfoque de Identificación de Caja Negra. En este paradigma, no se asume una

estructura física a priori, sino que se utiliza un aproximador funcional universal en este caso, una Red Neuronal para capturar la dinámica observada.

4.4.1 Esquemas de Identificación

En la identificación neuronal, existen dos configuraciones clásicas para el entrenamiento [23]:

- Modelo Paralelo (Figura 4): La salida del modelo se retroalimenta a su propia entrada. Es útil para validación y simulación autónoma, pero puede ser inestable durante las primeras etapas del entrenamiento.

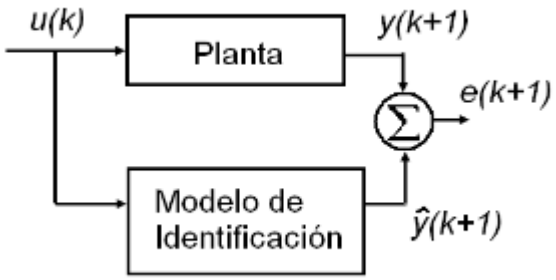


Figura 4. Modelo de identificación en Paralelo.

- Modelo Serie-Paralelo (Figura 5): El modelo utiliza la salida real de la planta (datos históricos) como entrada de regresión. La ecuación que describe este predictor es:

$$\hat{y}(k+1) = \sum_{i=0}^{n-1} \hat{\alpha}_i \, y(k-i) + \sum_{j=0}^{m-1} \hat{\beta}_j \, u(k-j)$$

  Este esquema es preferido para el entrenamiento supervisado debido a su estabilidad y convergencia más rápida.

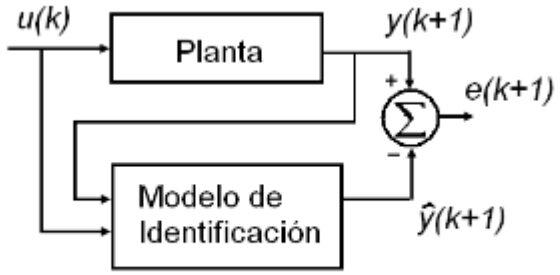


Figura 5. Modelo de identificación Serie-Paralelo

4.4.2 Identificación de Sistemas

En lugar de diseñar un controlador *C* para el sistema $G: C(s) = G^{-1}(s)$, se entrena a la RNA para que actúe como una aproximación del modelo inverso del sistema G^{-1} [23].
El papel de la RNA es aprender la función de mapeo f^{-1}:

$$Velocidades\ de\ Control\ (V)\ =$$

$$RNA(Posición\ Deseada\ (P_{des}), Posicion\ Actual\ (P_{act}))$$

Esta aproximación tiene dos ventajas teóricas cruciales:

- Robustez ante No Linealidades: El MLP y el RNN pueden aproximar con precisión las no linealidades inherentes al sistema cinemático/dinámico.
- Diseño Simplificado: Evita la laboriosa y a menudo inexacta tarea de modelar la dinámica física exacta del UAV.

Para ilustrar el proceso de aprendizaje, la Figura 6 presenta el diagrama de bloques del esquema de identificación inversa implementado. En este esquema, la red neuronal se entrena para minimizar el error entre la velocidad predicha y la velocidad ideal registrada en la base de datos experimental.

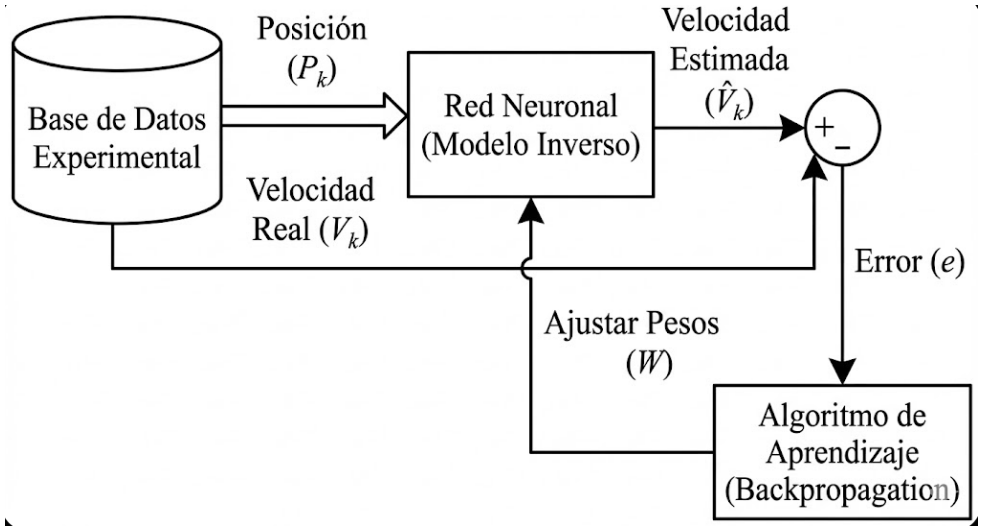


Figura 6. Esquema de bloques para la identificación inversa del modelo cinemático. La red neuronal recibe la posición como entrada y se entrena para predecir la velocidad de control correspondiente, minimizando el error respecto a la base de datos de referencia.

4.5 Arquitecturas de Redes Neuronales

El proyecto evalúa dos arquitecturas fundamentales para contrastar la efectividad del modelado estático frente al modelado dinámico: Feedforward (MLP) y Recurrente (RNN).

4.5.1 Red Feedforward (Perceptrón Multicapa, MLP)

El MLP actúa como el modelo de identificación estática . Su propósito es aprender la relación de mapeo instantánea entre la posición deseada y la velocidad requerida en un único instante de tiempo [36].

- Estructura: La red consta de tres capas (entrada P_{des}, oculta y salida $V - 3x10x3$).
- Funciones de Activación: Las funciones de activación no lineales son la clave para la aproximación universal de funciones:
    - Capa Oculta (Tangente Sigmoidal - tansig): Mapea los valores de entrada transformados al rango [-1, 1]. Esta función es vital ya que permite a la red modelar eficientemente las transiciones de movimiento bidireccionales (velocidades positivas y negativas) necesarias para el seguimiento de la

trayectoria sinusoidal.

- Capa de Salida (Lineal Pura - purelin): Es la función estándar para problemas de regresión, ya que permite que la salida (las señales de velocidad $\dot{x}, \dot{y}, \dot{z}$) abarque un rango continuo y no saturado de valores reales.

- Rol de Benchmark: El MLP entrenado manualmente proporciona una línea base de rendimiento puramente estática para comparar el beneficio de la memoria introducida por la RNN.

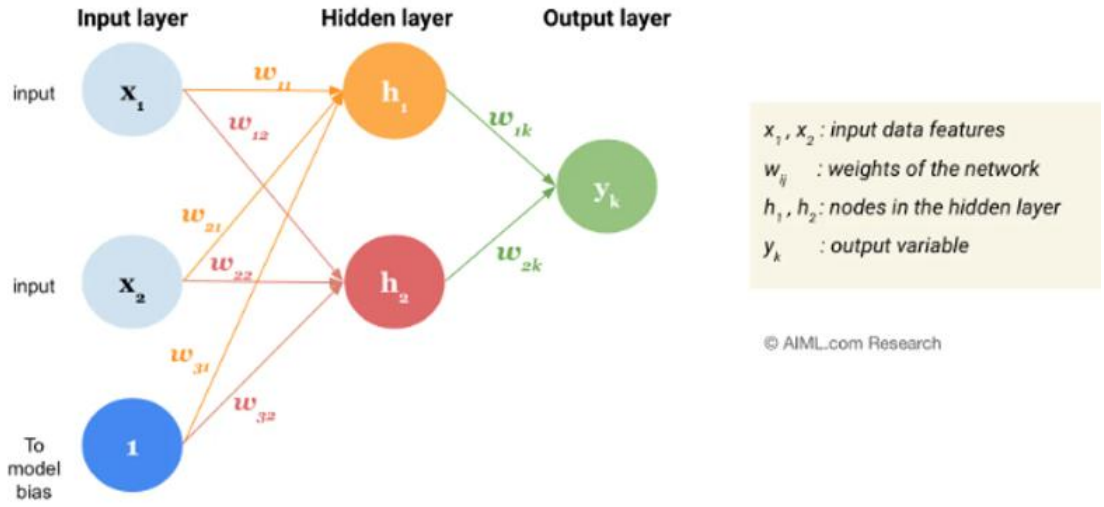


Figura 7. Ejemplo ilustrativo de un Perceptrón Multicapa (MLP). Se observan los pesos sinápticos (w) que conectan cada neurona de una capa con todas las neuronas de la siguiente capa, formando una topología totalmente conectada.

Estructuralmente, un MLP típico consta de tres tipos de capas, tal como se ilustra en la Figura 7:

1. Capa de Entrada (Input Layer): Recibe el vector de características iniciales x. No realiza procesamiento, solo distribuye la señal.
2. Capas Ocultas (Hidden Layers): Es donde ocurre la extracción de características y el cómputo principal. Una red puede tener una o más capas ocultas; en estas, las neuronas realizan una suma ponderada de las entradas seguida de una función de activación no lineal $\sigma(\cdot)$.

3. Capa de Salida (Output Layer): Produce el resultado final y, que puede ser una clase (clasificación) o un valor continuo (regresión).

El funcionamiento matemático de una neurona k en una capa oculta se describe mediante la ecuación:

$$y_k = \varphi\left(\sum_{j=1}^{m} w_{kj}x_j + b_k\right) \quad (34)$$

Donde w_{kj} son los pesos sinápticos, x_j las entradas provenientes de la capa anterior, b_k el sesgo (bias) y φ la función de activación.

La importancia teórica del MLP radica en el Teorema de Aproximación Universal [36], el cual establece que un MLP con una sola capa oculta y suficiente número de neuronas es capaz de aproximar cualquier función continua con un grado de precisión arbitrario. Esta propiedad es la que justifica su uso para la identificación de sistemas dinámicos complejos y no lineales, como la cinemática de un vehículo aéreo.

4.5.2 Red Neuronal Recurrente (RNN)

La RNN se utiliza para la identificación dinámica del sistema, superando la limitación del MLP al introducir la memoria temporal y la capacidad de procesar secuencias [37].

- Manejo de la Dinámica: A diferencia del MLP, la salida de la RNN en el instante t depende no solo de la entrada actual, sino de un número de valores anteriores a través de retardos o delays (ej. 1:5 pasos). Esto permite que la red aprenda el comportamiento secuencial y la dinámica del sistema.
- Modelado NARX/NOE: La arquitectura RNN con realimentación se aproxima a un modelo No Lineal Auto-Regresivo con Entradas Exógenas (NARX), siendo la metodología de elección para la identificación de sistemas dinámicos.
- Herramienta de MATLAB: Se emplea la función layrecnet, que es un tipo de red de capa recurrente utilizada en el Toolbox de Redes Neuronales de MATLAB para modelar series de tiempo y sistemas dinámicos.

A diferencia del Perceptrón Multicapa (MLP), que es una red estática donde la salida depende únicamente de la entrada actual, las Redes Neuronales Recurrentes (RNN) son arquitecturas dinámicas diseñadas para procesar secuencias de datos. Su característica distintiva es la presencia de conexiones cíclicas o bucles de retroalimentación que permiten que la información persista a través del tiempo. El concepto fundamental se ilustra en la Figura 8. En el lado izquierdo, se observa un bloque de procesamiento A que recibe una entrada X_t y produce una salida h_t, pero que también envía información de vuelta a sí mismo.

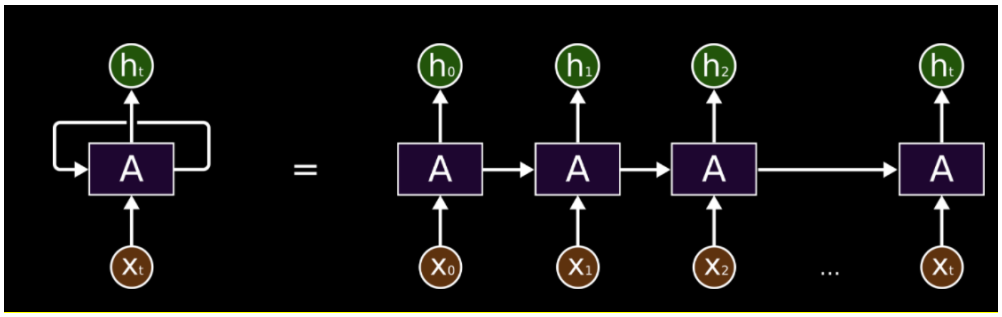


Figura 8. Representación estructural de una Red Neuronal Recurrente entrenable.

4.6 Algoritmos de Entrenamiento y Estabilización

La eficacia del control depende directamente de la calidad del entrenamiento de la RNA, que se logra mediante potentes algoritmos de optimización y técnicas de preprocesamiento.

4.6.1 Retropropagación del Error (*Backpropagation*)

El Backpropagation es el principio fundamental que rige el aprendizaje en ambas arquitecturas. Se basa en el Descenso de Gradiente, utilizando la regla de la cadena para calcular cómo cada peso y sesgo contribuye al error total de la red [38].

- Función de Pérdida (MSE): El objetivo del entrenamiento es minimizar el Error Cuadrático Medio (MSE) en (35) , que proporciona un valor escalar que mide la

calidad de la predicción de la red c:

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(y_i - t_i)^2 \quad (35)$$

donde y_i es la salida de la red y t_i es la salida objetivo (velocidad requerida).

4.6.2 Estrategias de Optimización de Segundo Orden

Para acelerar la convergencia y lograr un error mínimo profundo (*goal*), se utilizan algoritmos de optimización de alto rendimiento, superando las limitaciones del descenso de gradiente estándar [39].

- Levenberg-Marquardt (*trainlm*): Este algoritmo (utilizado a menudo para entrenar el MLP) es un método híbrido que combina la velocidad del método de Gauss-Newton (ideal cerca del mínimo) y la estabilidad del Descenso de Gradiente (ideal lejos del mínimo).Su uso se justifica por su capacidad para lograr una convergencia superior y alcanzar valores de pérdida extremadamente bajos (ej. 0.0055 en entrenamiento).
- Gradiente Conjugado Escalonado (*trainscg*): Utilizado para el entrenamiento de la RNN, este método es una alternativa de segundo orden eficiente en memoria que es robusta para el manejo de grandes secuencias de datos sin necesidad de calcular la matriz Hessiana completa.

4.6.3 Preprocesamiento y Regularización

La estabilidad y la capacidad de generalización del modelo se aseguran antes y durante el entrenamiento [39].

- Normalización Min-Max: Las entradas y salidas se escalan al rango [0, 1] utilizando la transformación Min-Max. Este paso es fundamental para:
    1. Hay que asegurar que las características con diferentes escalas (ej., X en metros y Z en metros) contribuyan equitativamente a la función de pérdida.
    2. Evitar la saturación temprana de las funciones de activación no lineales.
- Partición de Datos (*Regularización*): El *dataset* se divide en tres subconjuntos (entrenamiento 70%, validación 15%, prueba 15%). El conjunto de validación es

crucial para implementar la técnica de parada temprana (*early stopping*), que previene el sobreajuste (*overfitting*). El entrenamiento se detiene cuando el error en los datos de validación comienza a aumentar, asegurando que el modelo generalice bien a trayectorias no vistas.

4.7 Herramientas de Desarrollo

4.7.1 Matlab y Deep Learning Toolbox.

MATLAB (Matrix Laboratory) se seleccionó como el entorno principal para el análisis de señales y la identificación del sistema debido a su robustez en el cálculo numérico matricial [40]. Específicamente, se utilizaron los siguientes módulos especializados:

- Deep Learning Toolbox: Permitió el diseño, entrenamiento y validación de las arquitecturas neuronales (MLP y RNN) utilizadas para la identificación inversa. Proporciona implementaciones optimizadas de algoritmos de retropropagación como el Gradiente Conjugado Escalado (trainscg).
- Ryze Tello Support Package: Esta librería oficial facilita la interfaz de comunicación de bajo nivel con el hardware del dron, permitiendo la adquisición de telemetría y el envío de tramas de control sin necesidad de desarrollar drivers desde cero.

El flujo de trabajo en MATLAB abarcó desde la generación de trayectorias sintéticas y el preprocesamiento de datos experimentales hasta la simulación del lazo cerrado de navegación.

4.7.2 Python y Ultralytics Framework.

Python se utilizó como lenguaje de programación complementario para el desarrollo de los algoritmos de detección y reconocimiento de personas [41]. Gracias a su amplia gama de librerías especializadas en visión por computadora e inteligencia artificial. Python permitió implementar y entrenar modelos basados en YOLOv5 y YOLOv8 en el entorno Google Colab, aprovechando unidades GPU para acelerar el proceso de entrenamiento.
Entre las librerías empleadas destacan:

- OpenCV: Utilizada para las operaciones fundamentales de procesamiento de imágenes, como la decodificación del flujo de video H.264 proveniente del dron, el redimensionamiento de tensores y la visualización de las cajas delimitadoras.
- PyTorch: para el entrenamiento y ajuste fino (*fine-tuning*) de las redes neuronales.
- Matplotlib y Seaborn: para graficar las métricas de entrenamiento (loss, precisión y recall).
- Ultralytics: para la implementación directa de los modelos YOLOv5/v8 y la evaluación del desempeño sobre el dataset VisDrone2019.

La integración de Python dentro del flujo de trabajo permite automatizar procesos como:

- Conversión de anotaciones al formato YOLO.
- Entrenamiento y evaluación de los modelos.
- Generación de métricas y visualización de resultados.

4.7.3 Protocolo de Comunicación UDP (PC - Dron).

La comunicación entre la computadora y el dron Tello se establece mediante una conexión Wi-Fi UDP (User Datagram Protocol), la cual permite el envío y recepción de comandos hacia el dron. Este protocolo fue seleccionado por su baja latencia y eficiencia en entornos donde la rapidez de respuesta es prioritaria, como en las aplicaciones de control de vuelo. El dron Tello crea una red inalámbrica local (SSID) a la cual la computadora se conecta. Una vez establecida la conexión, MATLAB o Python pueden enviar comandos en formato texto al puerto predeterminado 8889, mientras que el dron responde con confirmaciones de estado ("ok" o mensajes de error) [42].

La comunicación se estructura de la siguiente manera:

- Puerto 8889: envío de comandos de control (takeoff, land, move, rotate, etc.).
- Puerto 11111: transmisión de video (streaming).
- Puerto 8890: recepción de datos de telemetría (velocidad, orientación, batería, etc.).

Durante las pruebas de laboratorio, se verificó la estabilidad de la conexión y la sincronización entre el procesamiento visual y el control del dron. Esta comunicación permite que los resultados de detección obtenidos por los modelos en Python sean utilizados en MATLAB para generar movimientos de seguimiento o ajuste de posición,

cerrando el ciclo de integración visión–control.

5. Metodología

Para abordar el problema de navegación autónoma basada en visión, se propone una metodología fundamentada en una arquitectura modular desacoplada. Esta estructura permite el desarrollo y validación independiente de los dos subsistemas críticos antes de su integración teórica:

- Subsistema de Percepción Visual: Enfocado en la detección y localización de personas en imágenes aéreas mediante técnicas de Aprendizaje Profundo (Deep Learning), utilizando la arquitectura YOLOv8.

- Subsistema de Identificación Cinemática: Enfocado en el modelado matemático del comportamiento dinámico del dron Tello EDU mediante Redes Neuronales Artificiales en MATLAB, con el objetivo de establecer la relación inversa entre la posición deseada y la velocidad de navegación requerida.

El sistema se diseña para operar bajo un esquema de procesamiento en cascada: el módulo de visión extrae las coordenadas espaciales del objetivo, las cuales actúan como referencias para el modelo cinemático identificado, que a su vez estima los comandos de velocidad necesarios para la navegación.

5.1 Identificación de personas

5.1.1 Diagrama a bloques

La metodología propuesta se fundamenta en una arquitectura modular desacoplada, dividida en dos subsistemas principales que operan en cascada: percepción y navegación. El flujo de trabajo metodológico se ilustra en la Figura 9. Se distinguen dos fases de desarrollo paralelas:

- Fase de Visión: Enfocada en el entrenamiento supervisado del detector de objetos para extraer las coordenadas de las personas en el plano imagen.
- Fase de Identificación: Enfocada en el modelado de caja negra de la cinemática inversa del dron para traducir intenciones de movimiento en comandos de

velocidad.

- Fase 3 (Análisis de Resultados): Etapa final de consolidación donde se interpretan cuantitativa y cualitativamente los datos obtenidos en las fases previas. En este bloque se contrastan los experimentos de generalización y estrés (trayectorias complejas) y se discute la viabilidad de la integración teórica de ambos subsistemas.

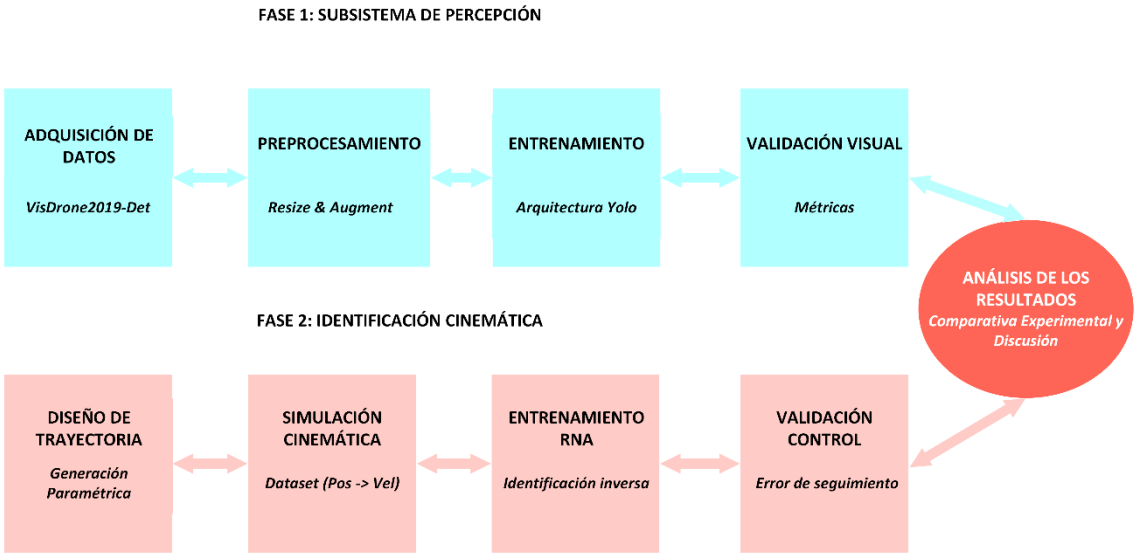


Figura 9. Metodología empleada.

Como se ilustra en la Figura 9, la metodología se estructura en tres etapas secuenciales e interdependientes:

Fase 1: Subsistema de Percepción Visual (Línea Superior)

Esta fase se enfoca en dotar al sistema de la capacidad de interpretar el entorno visual.

- Adquisición de Datos: Se parte del conjunto de datos VisDrone2019-DET, seleccionado por su representatividad en escenarios aéreos reales con variaciones de altura y ángulo de visión.
- Preprocesamiento: Las imágenes crudas son sometidas a un proceso de acondicionamiento que incluye el redimensionamiento a 640×640 píxeles y técnicas de aumento de datos (Data Augmentation) para mejorar la robustez del modelo ante cambios de iluminación y escala.
- Entrenamiento: Se implementa y entrena la arquitectura YOLOv8s utilizando

aprendizaje supervisado, ajustando los pesos de la red para minimizar la función de pérdida de localización y clasificación.

- Validación Visual: Se evalúa el desempeño del detector utilizando métricas cuantitativas estándar, específicamente la Precisión Media (mAP) y la Intersección sobre la Unión (IoU), asegurando que las coordenadas generadas sean fiables.

Fase 2: Identificación Cinemática (Línea Inferior)

Paralelamente, se desarrolla el modelo matemático que gobierna el movimiento del vehículo.

- Diseño de Trayectorias: Se generan funciones paramétricas en MATLAB (sinusoidales y complejas) que definen rutas tridimensionales suaves dentro del volumen de trabajo operativo.
- Simulación Cinemática: Se utiliza un modelo cinemático de punto para generar un conjunto de datos sintético que asocia cada posición deseada (P) con el vector de velocidad (V) necesario para alcanzarla, creando la "verdad terrestre" para el control.
- Entrenamiento RNA: Se entrena una Red Neuronal Artificial (Perceptrón Multicapa) bajo un esquema de identificación inversa, aprendiendo a predecir los comandos de velocidad a partir de las coordenadas espaciales.
- Validación de Control: Se somete al modelo neuronal a pruebas de simulación con trayectorias desconocidas para medir el error de seguimiento (tracking error) y verificar la estabilidad de la respuesta.

Fase 3: Análisis de Resultados (Bloque Final)

Finalmente, los resultados de ambos subsistemas convergen en una etapa de integración teórica. En este bloque se realiza una comparativa experimental y una discusión crítica sobre la viabilidad de utilizar las coordenadas obtenidas por la Fase 1 como entradas directas para el controlador validado en la Fase 2, cerrando así el ciclo de navegación autónoma propuesta.

5.1.2 Desarrollo del Subsistema de Percepción Visual (YOLO)

5.1.2.1 Base de datos de reconocimiento

Para la identificación de personas se empleó el conjunto de datos público *VisDrone*, el cual es una base de datos de referencia en tareas de detección de objetos desde drones [43]. Este conjunto fue recopilado sistemáticamente en 14 ciudades de China, utilizando diferentes plataformas aéreas como *DJI* Mavic y Phantom, bajo condiciones variadas de clima, iluminación y escenarios urbanos y suburbanos, lo que le otorga una alta diversidad y representatividad para aplicaciones de monitoreo aéreo. La base de datos está organizada en cuatro subcarpetas principales, las cuales se diseñaron para diferentes propósitos:

- *VisDrone2019-DET* (*Object Detection in Images*): Esta carpeta está diseñada para la detección de objetos en imágenes estáticas.
- *VisDrone2019-VID* (*Object Detection in Videos*): Su propósito es para la detección de objetos en secuencias de video.
- *VisDrone2019-MOT* (*Multi-Object Tracking*): Diseñada para el seguimiento de múltiples objetos a lo largo del tiempo.
- *VisDrone2019-SOT* (*Single Object Tracking*): Diseñada para el seguimiento de un solo objeto objetivo en cada secuencia.

Para este artículo se utilizó la subcarpeta *VisDrone2019-DET* con las siguientes características:

- Contiene 10,209 imágenes con anotaciones cuadro a cuadro.
- Las imágenes presentan una resolución máxima de 2000×1500 píxeles, lo que permite analizar tanto objetos grandes como muy pequeños en el campo visual.
- Se divide en:
    - 6,471 imágenes de entrenamiento (*train*).
    - 548 imágenes de validación (*val*).

- 1,580 imágenes de *test-challenge*.
- 1,610 imágenes de *test-dev*.

- Cada imagen tiene un archivo de anotación con la posición, tamaño, clase, oclusión y truncamiento de cada objeto.

En cuanto a las anotaciones, *VisDrone* define 10 clases de objetos, entre las cuales se encuentran persona, coche, bicicleta, moto, autobús, camión, triciclo, carretilla y toldo-triciclo. En la Figura 10 se muestra un ejemplo de las imágenes que contiene la base de datos *VisDrone2019-DET,* donde se pueden observar varias clases de objetos, como personas, moto, coche, etc. El *VisDrone2019-DET* cuenta con un conjunto de datos que abarca una amplia gama de escenarios urbanos y suburbanos con variaciones significativas en escala, densidad e iluminación.

Para ilustrar la complejidad del entorno y la variación en las condiciones de captura, se presentan diferentes escenas de la Figura 10 a la Figura 14 . La Figura 5 ejemplifica la alta densidad de personas, mientras que la Figura 3 demuestra la dificultad de la detección en condiciones de baja iluminación. Estos factores son determinantes para justificar la selección de la arquitectura YOLO.

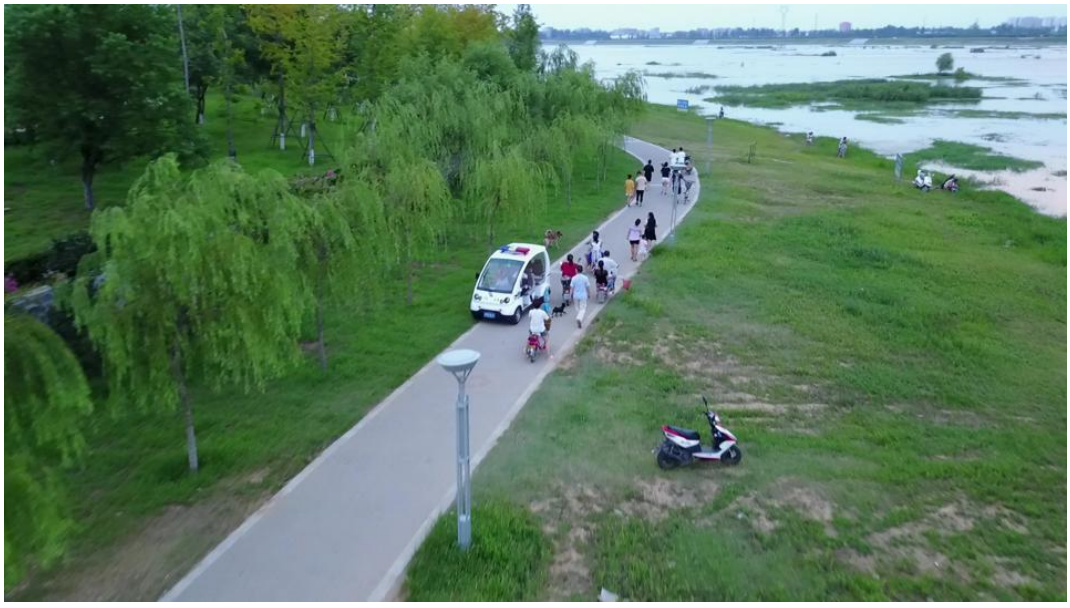


Figura 10. Escena en un sendero con baja densidad de personas y distancia larga. Muestra la detección en el extremo opuesto: pocos objetos, pero muy lejanos.

Figura 11. Escena nocturna o de baja iluminación en un espacio abierto. Resalta el desafío del proyecto para mantener la precisión en condiciones ambientales subóptimas.

Figura 12. Escena urbana con tráfico y personas en cercanía de vehículos. Ilustra la interacción persona-vehículo y el reto de distinguir peatones en contextos con otros objetos.

Figura 13. Escena deportiva con densidad alta de peatones. Muestra el reto de la oclusión parcial y la identificación de objetos muy pequeños debido a la altura

Figura 14. Escena con obstrucción natural (árboles y arbustos). Ilustra el reto de la oclusión por vegetación en entornos suburbanos.

Adicionalmente, las anotaciones que se encuentran en esta base de datos incluyen atributos relacionados con el nivel de oclusión y truncamiento de los objetos [44]:

$$\delta = \frac{\left|B_{obj} \cap B_{occl}\right|}{\left|B_{obj}\right|} \tag{36}$$

donde δ indica el grado de oclusión: 0% (sin oclusión), 0–50% (parcial) y 50–100% (severa). Asimismo, se incorpora el atributo de truncamiento, que señala si un objeto se encuentra cortado por los límites de la imagen, lo que incrementa la dificultad de detección en escenarios reales.

5.1.2.2 Configuración de la Arquitectura YOLO

Para este trabajo se implementaron las arquitecturas YOLOv5s y YOLOv8s, adaptadas específicamente a la tarea de detección de personas en imágenes aéreas del dataset VisDrone2019-DET. Ambas redes fueron configuradas y entrenadas únicamente para la clase persona, con el fin de optimizar su desempeño en escenarios reales de vigilancia aérea y monitoreo urbano.

YOLOv5s

YOLOv5s utiliza como *backbone* CSPDarknet53, el cual incorpora bloques *Cross Stage Partial* para reducir el costo computacional manteniendo la precisión. El *neck* combina FPN + PAN para la fusión multiescala, lo que mejora la detección de personas pequeñas y parcialmente ocluidas. Su *head* genera predicciones multiescala en tres resoluciones, permitiendo localizar personas en distintos tamaños.

YOLOv8s

YOLOv8s emplea el *backbone* C2f, más eficiente en la extracción de características y adecuado para objetos pequeños. A diferencia de YOLOv5, incluye un *head* anchor-free, lo que elimina la dependencia de *anchors* predefinidos y mejora la precisión y estabilidad de las predicciones en entornos con alta variabilidad geométrica, como las escenas aéreas de VisDrone.

Función de pérdida

Durante el entrenamiento se utilizó una función de pérdida compuesta por tres términos:

$$L = L_{\text{CIoU}} + L_{\text{conf}} + L_{\text{cls}} \qquad (37)$$

donde CIoU evalúa la calidad de la caja predicha, L_{conf} la confianza del objeto detectado y L_{cls} la clasificación de la clase persona.

Parámetros de entrenamiento

El entrenamiento se realizó con las siguientes configuraciones:

Hiperparámetros principales

- Tamaño de imagen: 640×640
- Batch size: 16
- Learning rate inicial: 0.01
- Momentum: 0.937
- Weight decay: 0.0005
- Tipo de optimizador: SGD (por defecto en YOLOv5) / AdamW (por defecto en YOLOv8)
- Scheduler: OneCycleLR
- Warmup epochs: 3

Cantidad de épocas

- Pruebas: 50 y 150 épocas
- Permite analizar si el modelo se estabiliza o sobreentrena.

5.2 Implementación de la Identificación Cinemática

5.2.1 Adquisición y Procesamiento de Datos Experimentales del Dron

Para el desarrollo del modelo de red neuronal, fue necesario construir una base de datos (dataset) que representara fielmente la dinámica de vuelo del vehículo aéreo no tripulado (UAV) en su volumen de operación. Dado que el objetivo es realizar una identificación del sistema (obtener el modelo inverso que relaciona posición con velocidad), se optó por una recolección de datos experimental en un entorno controlado, asegurando que la red aprenda las restricciones cinemáticas reales de la plataforma.

Configuración del Entorno Experimental

Los vuelos de recolección de datos se llevaron a cabo en un entorno interior (*indoor testbed*) de $10\ x\ 3\ x\ 3$ metros, libre de perturbaciones eólicas significativas y con iluminación controlada para garantizar la estabilidad del sistema de posicionamiento visual (VPS) del dron. La plataforma utilizada fue el DJI Tello EDU, seleccionado por su capacidad de comunicación bidireccional a través de su SDK 2.0. La arquitectura de adquisición de datos se configuró bajo un esquema Cliente-Servidor:

- Servidor: Ordenador portátil ejecutando MATLAB, encargado de generar las trayectorias de referencia, enviar los comandos de control y registrar la telemetría entrante.
- Enlace de Comunicación: Protocolo UDP (*User Datagram Protocol*) sobre Wi-Fi a 2.4 GHz. Se seleccionó UDP para minimizar la latencia en la transmisión de paquetes de estado, priorizando la velocidad sobre la verificación de entrega.
- Frecuencia de Muestreo: Se estableció una tasa de muestreo de 10 Hz ($T_s = 0.1s$). Esta frecuencia se determinó experimentalmente como el límite operativo estable para la transmisión de datos de telemetría sin saturar el buffer de comunicación del dron.

Protocolo de Vuelo para la Recolección de Datos

Para garantizar que la red neuronal generalice correctamente la cinemática del UAV y evitar el sobreajuste (*overfitting*) a movimientos triviales, se diseñó una trayectoria de excitación persistente. Esta trayectoria paramétrica fue diseñada para recorrer el espacio de estados (x, y, z) del dron, combinando variaciones simultáneas en los tres ejes:

- Eje Longitudinal (X): Avance continuo y suave para caracterizar la velocidad de crucero.
- Eje Lateral (Y): Oscilaciones armónicas (sinusoidales) para capturar la dinámica de viraje y desplazamiento lateral.
- Eje Vertical (Z): Variaciones de altitud para modelar la respuesta de los motores ante cambios de sustentación y gravedad.

Durante la sesión experimental, el dron ejecutó esta trayectoria de forma autónoma basada en comandos de velocidad precalculados. El sistema registró sincrónicamente dos vectores de datos en cada instante k:

- Vector de Entrada (*Target*): La posición estimada del dron $P_k = [x_k, y_k, z_k]$, obtenida mediante la fusión de sensores a bordo (IMU + Cámara de flujo óptico).
- Vector de Salida (*Input*): Las velocidades lineales comandadas $V_k = [\dot{x_k}, \dot{y_k}, \dot{z_k}]$ que generaron dicho desplazamiento.

Para asegurar la integridad del vehículo durante la fase de adquisición de datos, se desarrolló un módulo de visualización en MATLAB. Como se observa en la Figura 15, esta interfaz permitía al operador supervisar la ejecución de maniobras complejas, validando visualmente que los virajes y cambios de altitud se mantenían dentro del volumen de seguridad antes de proceder al almacenamiento de los datos.

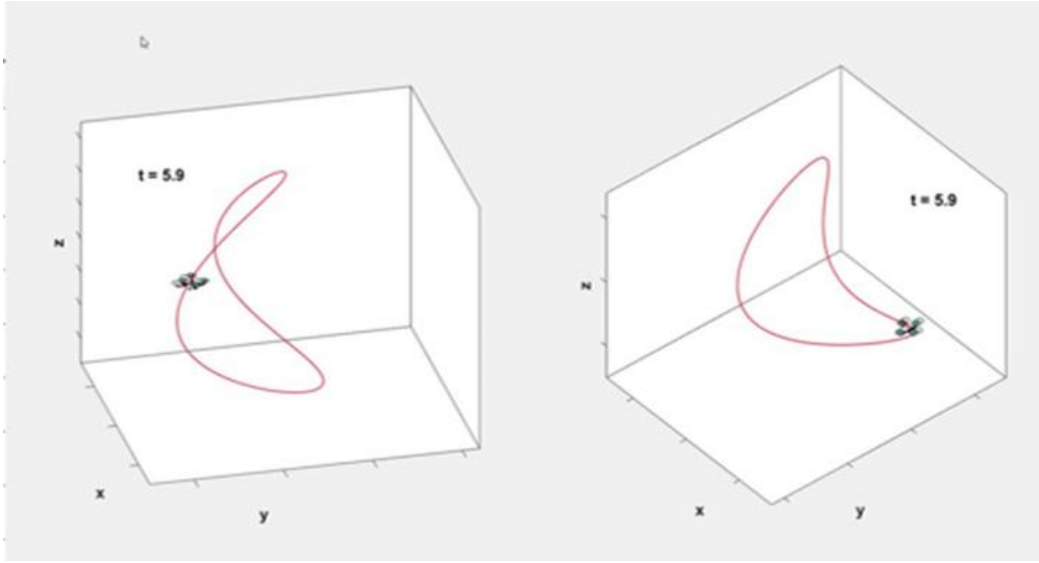


Figura 15. Interfaz de monitorización durante la ejecución del protocolo de vuelo. Se muestran dos perspectivas isométricas de la trayectoria de referencia (rojo) y la posición instantánea del Dron (modelo 3D) en el instante t=5.9s

Adquisición y Preprocesamiento de Señales

Durante la sesión experimental de 60 segundos, el sistema registró sincrónicamente la posición estimada del dron (mediante fusión de sensores IMU + Cámara de flujo óptico) y las velocidades comandadas.

Los datos crudos (*raw data*) obtenidos directamente de los sensores inerciales del Tello EDU presentaban ruido de alta frecuencia y vibraciones mecánicas inherentes a los rotores. Utilizar estos datos directamente degradaría la convergencia del entrenamiento de la red neuronal. Por tanto, se aplicó una etapa post-procesamiento digital:

- Filtrado de Ruido: Se aplicó un filtro de suavizado de media móvil (*Moving Average Filter*) a las series temporales de posición. Esto permitió eliminar el ruido estocástico del sensor visual manteniendo la tendencia cinemática de la trayectoria.
- Sincronización Temporal: Debido a la naturaleza asíncrona del protocolo UDP, se realizó una alineación temporal (*timestamp alignment*) para corregir el desfase

entre el comando de velocidad enviado y la respuesta de posición recibida, asegurando la causalidad de los datos para la etapa de identificación.

- Normalización (*Min-Max Scaling*): Finalmente, para optimizar el rendimiento del algoritmo de *Backpropagation*, el conjunto de datos fue normalizado al rango $[0, 1]$ utilizando los valores máximos y mínimos registrados (38).

$$X_{norm} = \frac{X - X_{min}}{X_{max} - X_{min}} \qquad (38)$$

La Figura 16 presenta la reconstrucción espacial final de los datos procesados. Se puede apreciar una trayectoria continua y suave que cubre simultáneamente los tres ejes de movimiento, validando la eficacia del filtrado y la cobertura del espacio de trabajo necesaria para el entrenamiento.

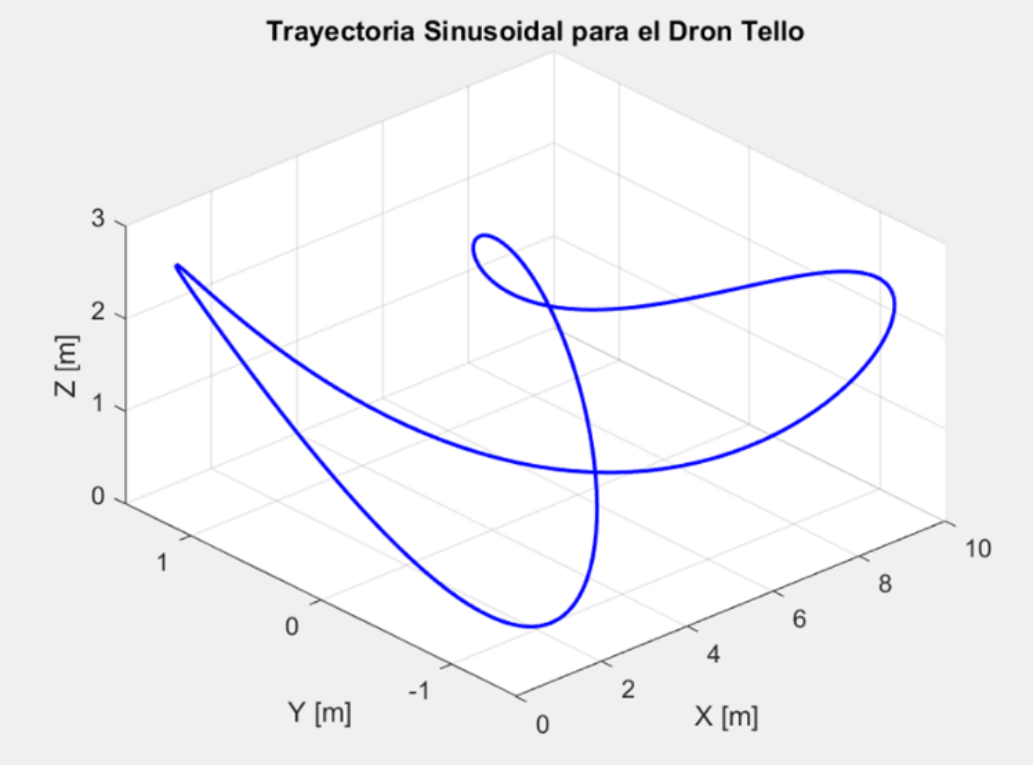


Figura 16. Reconstrucción espacial de la trayectoria de excitación persistente ejecutada por el Dron tras el preprocesamiento. La ruta cubre variaciones simultáneas en los ejes X, Y y Z para caracterizar la dinámica completa del vehículo.

Tras la etapa de limpieza y normalización en el rango $[0, 1]$, los datos fueron estructurados en matrices de doble precisión (*double*) dentro del entorno MATLAB. El resultado de este proceso es el conjunto de datos final compuesto de 501 muestras temporales. El archivo contiene dos matrices principales:

- Entradas: Matriz de dimensiones $\$501\ x\ 3$ correspondiente a las posiciones espaciales $[x, y, z]$.
- Salidas: Matriz de dimensiones $501\ x\ 3$ correspondiente a las velocidades lineales $[v_x, v_y, v_z]$.

Para ilustrar la naturaleza de los datos utilizados en la identificación del modelo cinemático, la Tabla 4 presenta un fragmento numérico de las primeras muestras del base de datos procesada.

Tabla 4. Fragmento numérico de las primeras muestras de la base de datos procesada.

Muestra (k)	Tiempo (s)	Entradas: Posición (P_k) [m]	Salidas: Velocidad (V_k) [m/s]
1	0.0	[0, 0.0958, 1.4352]	$[0.20, 0.01, -0.05]
2	0.1	[0.0200, 0.0904, 1.4344]	$[0.21, 0.02, -0.04]
3	0.2	[0.0400, 0.0843, 1.4317]	$[0.20, 0.01, -0.06]
...	**...**	**...**	**...**
501	50.0	[10.0, 0.0, 2.0]	[0.0, 0.0, 0.0]

Esta estructura de datos permite entrenar a la red neuronal para aprender el mapeo inverso $f: P \rightarrow V$, fundamental para el esquema de identificación cinemática propuesto.

5.3 Selección del algoritmo de reconocimiento y trayectoria

Para el desarrollo del sistema de reconocimiento e identificación de personas se seleccionó un modelo basado en la arquitectura YOLO (You Only Look Once), debido a su eficiencia en la detección de objetos y su adecuado equilibrio entre velocidad y precisión.

En particular, se consideraron las versiones YOLOv5s y YOLOv8n, las cuales presentan estructuras optimizadas para dispositivos con recursos computacionales limitados y

ofrecen resultados satisfactorios en tareas de detección de personas en entornos aéreos.
El modelo YOLO se caracteriza por realizar una detección de una sola pasada (single-shot detection), es decir, analiza toda la imagen en una sola etapa, dividiéndola en celdas y prediciendo simultáneamente las coordenadas de los objetos y las probabilidades de clase. Esto lo hace especialmente útil para su implementación en sistemas embarcados o plataformas como drones, donde la latencia y la eficiencia de cómputo son factores determinantes.
El proceso de selección también consideró:

- La compatibilidad del modelo con entornos de desarrollo como Google Colab y MATLAB.
- La disponibilidad de modelos preentrenados en bases de datos amplias como COCO, que facilitan la transferencia de aprendizaje.
- La posibilidad de reentrenar el modelo con un subconjunto de datos específicos (VisDrone2019) enfocado en la clase persona.

En cuanto a la trayectoria, se optó por un esquema de control cinemático indirecto programado en MATLAB, el cual recibe las coordenadas del objeto detectado y genera comandos de velocidad o desplazamiento para el dron Tello EDU. Este enfoque permite ajustar el movimiento del dron de acuerdo con la posición del objetivo, conservando estabilidad y seguridad durante el vuelo.

5.4 Entrenamiento del algoritmo de reconocimiento y trayectoria

El entrenamiento del algoritmo de reconocimiento se llevó a cabo utilizando la base de datos VisDrone2019, que contiene imágenes capturadas desde diferentes alturas y ángulos por drones, bajo condiciones variadas de iluminación y densidad de personas.
Para este trabajo, se filtraron exclusivamente las instancias correspondientes a la clase persona, lo que permitió enfocar el aprendizaje del modelo en el tipo de objeto de interés. Las imágenes fueron preprocesadas mediante redimensión a 640×640 píxeles,

normalización de valores y conversión al formato requerido por YOLO. Posteriormente, se definieron los parámetros de entrenamiento:

- Número de épocas: 100
- Tamaño del lote (batch size): 16
- Tasa de aprendizaje (learning rate): 0.001
- Optimizador: SGD con momentum

El modelo se entrenó en Google Colab utilizando una GPU Tesla T4, aplicando técnicas de data augmentation como rotación, cambio de brillo y recorte aleatorio para mejorar la generalización. Las métricas empleadas para evaluar el desempeño fueron la precisión (Precision), recuperación (Recall) y F1-score, calculadas a partir del umbral de intersección sobre unión (IoU > 0.5).

En paralelo, el algoritmo de trayectoria fue desarrollado en MATLAB, donde se emplearon trayectorias simuladas (lineales y sinusoidales) para entrenar una red neuronal encargada de modelar la cinemática inversa del dron.

Este módulo genera las velocidades de control necesarias para posicionar el dron frente a la persona detectada, y se ajusta mediante retropropagación del error (backpropagation) hasta minimizar el error cuadrático medio (RMSE) entre la posición deseada y la posición estimada.

El resultado de esta etapa fue un sistema capaz de detectar personas e interactuar con el entorno mediante la ejecución de trayectorias seguras y precisas.

5.4.1 Evaluación de los entrenamientos Dentro del bloque de entrenamiento

Una vez finalizado el proceso de entrenamiento del modelo de reconocimiento, se realizó una evaluación cuantitativa y cualitativa de su desempeño.

El análisis se efectuó empleando el conjunto de validación del dataset VisDrone2019, con el objetivo de medir la capacidad del modelo para generalizar ante condiciones de iluminación, escala y densidad de personas distintas a las del conjunto de entrenamiento.

Las métricas utilizadas fueron la precisión (Precision), la recuperación (Recall) y el F1-score, calculadas a partir de las detecciones correctas y los falsos positivos. Además, se consideró el indicador de IoU (Intersection over Union) con un umbral de 0.5 como criterio para definir una detección válida.

El desempeño del modelo se analizó mediante la matriz de confusión y curvas Precision-Recall, lo cual permitió identificar los niveles de confianza óptimos para detecciones estables.

Por otro lado, para el módulo de control y trayectoria se evaluó el error cuadrático medio (RMSE) obtenido durante las simulaciones de entrenamiento de la red neuronal, verificando la capacidad del modelo de control para generar desplazamientos precisos frente a diferentes entradas de referencia.

5.5 Pruebas de reconocimiento

Las pruebas de reconocimiento se realizaron aplicando el modelo YOLOv5s (y, para comparación, YOLOv8s) previamente entrenado sobre un conjunto de imágenes estáticas.

Estas imágenes corresponden tanto a ejemplos del dataset VisDrone como a imágenes externas adicionales utilizadas únicamente para evaluación cualitativa.

El objetivo principal de esta fase fue validar el comportamiento del modelo sobre imágenes reales, verificando su precisión, estabilidad y capacidad para identificar personas bajo distintas condiciones visuales presentes de forma natural en las fotografías.

El proceso de prueba consistió en aplicar el modelo a cada imagen de manera independiente. Para cada una, YOLO generó:

- Cuadros delimitadores (bounding boxes)
- Probabilidad/confianza de detección
- Centro y dimensiones de cada caja detectada

- Predicción de clase (“persona”)

Durante la evaluación sobre imágenes se analizaron los siguientes puntos:

- Número de personas correctamente detectadas : Se verificó cuántos individuos aparecen en cada imagen y cuántos fueron detectados correctamente por el modelo.
- Precisión de los bounding boxes: Se evaluó visualmente el ajuste de las cajas predichas respecto a las personas reales, observando:
    - Precisión espacial.
    - Desplazamiento lateral.
    - Tamaño adecuado de la caja.
- Falsos positivos: Casos donde el modelo señaló una persona donde no existía.
- Falsos negativos: Personas visibles que el modelo no logró identificar.
- Escenarios con oclusión: Se evaluó el rendimiento ante:
    - Personas parcialmente cubiertas por objetos,
    - Personas detrás de otras,
    - Personas en el borde de la imagen.
- Variación de iluminación: Se incluyeron imágenes con:
    - Sombras intensas.
    - Zonas brillantes.

condiciones ambientales difusas.

5.6 Pruebas de trayectoria

Para validar el desempeño del modelo cinemático inverso identificado, se diseñó un protocolo de pruebas riguroso ejecutado en el entorno de simulación MATLAB. La validación se centró en verificar la capacidad de la Red Neuronal para generalizar el comportamiento dinámico del dron ante referencias espaciales complejas.

5.6.1 Configuración del Entorno de Simulación

La implementación del lazo de simulación se apoyó en dos herramientas fundamentales:

1. Deep Learning Toolbox: Utilizado para la inferencia de la red neuronal previamente entrenada (MLP), permitiendo calcular los comandos de velocidad $\hat{V}$.
2. Ryze Tello Support Package: Empleado para parametrizar las restricciones físicas del modelo virtual (límites de saturación de velocidad y tiempos de respuesta), asegurando que la simulación fuese consistente con la planta real.

El algoritmo corresponde a un esquema de Navegación Basada en Identificación Inversa. En este esquema, la red neuronal actúa como un generador de referencias de velocidad (*feedforward*), complementado por un lazo de integración numérica con un paso fijo de $dt = 0.1s$ para emular la actualización de posición.

6. Resultados y Discusión

6.1 Detección de Personas

El modelo YOLOv5s fue seleccionado por su ligereza (7.2 millones de parámetros, 224 capas), lo cual lo hace adecuado para pruebas en drones y hardware con recursos limitados. Durante la fase de experimentación con la base de datos VisDrone2019-DET, se utilizó la clase persona como único objetivo de detección. Donde se obtuvieron los siguientes resultados:

Tabla 5 Resultados del modelo Yolov5s y Yolov8

Metricas	YOLOv5s	YOLOv8
Accuracy	0.845	0.919
Precision	0.985	0.937
Recall	0.846	0.974
F1-score	0.91	0.955

Podemos interpretar según los resultados mostrados en la Tabla 5, donde se comparan los modelos YOLOv5s y YOLOv8s, que existe un desempeño superior del modelo más reciente (YOLOv8s) en la tarea de detección de personas sobre el conjunto de validación de la base de datos *VisDrone*. En términos de exactitud global (*Accuracy*), YOLOv8s alcanzó un valor de 0.919, superando a YOLOv5s (0.845), lo que indica una mejor capacidad para clasificar correctamente las detecciones verdaderas frente a las falsas.

Aunque YOLOv5s presentó una mayor precisión individual (*Precision* = 0.985), reflejando una menor proporción de falsos positivos, su capacidad para detectar todos los objetos presentes fue menor (*Recall* = 0.846). En contraste, YOLOv8s logró un *recall* de

0.974, lo que implica una detección más completa de los individuos presentes en las imágenes, incluso en condiciones de oclusión o variación de escala.

El equilibrio entre ambas métricas se refleja en el *F1-score,* donde YOLOv8s alcanzó 0.955, frente a 0.910 obtenido por YOLOv5s. Este incremento del 5% en el F1-score demuestra que YOLOv8s logra un balance más estable entre precisión y exhaustividad.

En la Figura 11 se muestra de forma gráfica el rendimiento comparativo entre YOLOv5s (línea azul) y YOLOv8s (línea naranja) en la tarea de detección de personas sobre el conjunto de validación de la base de datos VisDrone. De forma general, se observa que YOLOv8s mantiene una tendencia más uniforme y elevada en la mayoría de las métricas evaluadas, lo que refleja un rendimiento más consistente y una mayor capacidad de generalización frente a diferentes condiciones de iluminación, escala y oclusión típicas de entornos aéreos.

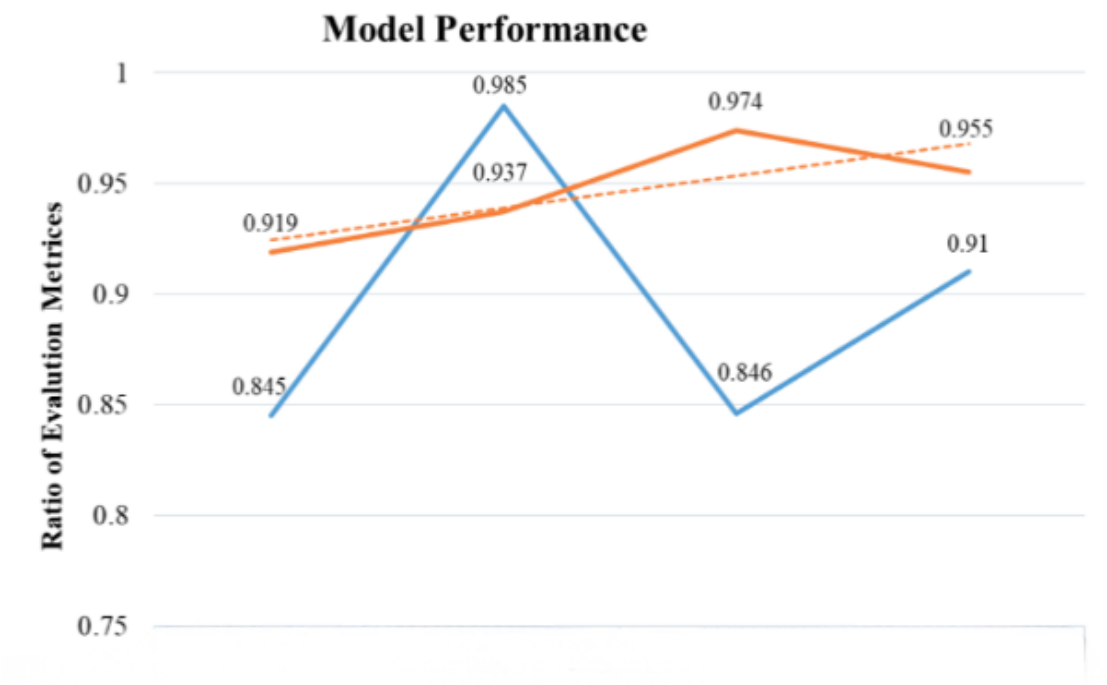


Figura 3. Comparación del rendimiento entre los modelos YOLOv5s y YOLOv8s sobre la base de datos VisDrone considerando las métricas Accuracy, Precision, Recall y F1-Score.

Finalmente, en la Tabla 6 se observa que las métricas de localización espacial (IoU = 0.76) y desempeño global (mAP@0.5 = 0.63) también respaldan la superioridad del modelo

YOLOv8s respecto a YOLOv5s (IoU = 0.70, mAP@0.5 = 0.57), confirmando una mejor alineación entre las cajas predichas y las reales, así como una mayor capacidad general de detección.

Estos resultados pueden atribuirse a las mejoras estructurales introducidas en la arquitectura YOLOv8, que adopta un esquema *anchor-free*, bloques C2f, y un *neck* optimizado, permitiendo un aprendizaje eficiente frente a variaciones de tamaño y perspectiva. En consecuencia, YOLOv8s demuestra ser una alternativa más eficiente para la detección de personas en escenarios aéreos y dinámicos, típicos de aplicaciones con drones.

Tabla 6. Resultados de las métricas de localización espacial del modelo Yolov5s y Yolov8.

Metricas	**YOLOv5s**	**YOLOv8**
IoU	0.70	0.75
mAP@0.5	0.57	0.63

En la *Figura 17* se muestra una escena natural en un entorno abierto, donde el modelo logra identificar correctamente múltiples personas y bicicletas a diferentes distancias y tamaños aparentes. Los valores de confianza asignados oscilan entre 0.25 y 0.77, lo que evidencia una correcta generalización del modelo, aunque algunas detecciones de baja confianza pueden deberse a oclusiones parciales o pequeña escala de los objetos. También se observan pocos falsos positivos, por ejemplo, "*cow*" o "*horse*" con confianzas menores a 0.3, típicos de las condiciones aéreas de la base de datos.

Figura 17. Detección de múltiples personas y objetos en campo abierto.

En la *Figura 18* se observa un entorno urbano, el modelo detecta con alta precisión a los peatones presentes en diferentes posiciones y distancias, con valores de confianza comprendidos entre 0.29 y 0.87. Asimismo, se observa una detección secundaria correcta de una motocicleta con confianza moderada (0.32), lo cual demuestra que el modelo conserva la capacidad multiclase adquirida durante el preentrenamiento con COCO.

Figura 18. Detección de múltiples personas y objetos en un entorno urbano.

En conjunto, estas imágenes ilustran el comportamiento cualitativo del modelo YOLO en escenarios reales, mostrando una detección eficiente y coherente con las métricas cuantitativas reportadas en la Tabla 1 y la Figura 4. Las detecciones evidencian una buena delimitación de las cajas (IoU > 0.5) y una correcta asignación de clases en la mayoría de los casos. Las leves confusiones observadas en objetos distantes o parcialmente visibles se atribuyen a las condiciones de captura aérea, donde el tamaño de los objetos en píxeles se reduce considerablemente.

Estos resultados cualitativos confirman que el modelo propuesto es capaz de detectar personas de forma eficiente en distintos contextos y alturas de vuelo, consolidando su potencial para aplicaciones de vigilancia aérea, monitoreo de multitudes y sistemas autónomos basados en drones.

En la Tabla 7 se presentan resultados reportados por otros autores y su comparación con el presente trabajo:

Tabla 7 Comparación de resultados reportados con otros autores.

Autor	Método	Base de datos	Resultados
Chanyoung Oh	YOLOv5s + SORT	MOT15	LocA=71.48% HOTA=18.38%
Sundas Iftikhar	YOLOv4, Faster R-CNN, etc	Caltech, KITTI	Accuracy = 72.10% (día), 64.20% (noche) IoU=0.75 mAP=0.5–0.85
Propios	YOLOv5s (COCO + VisDrone)	VisDrone	IoU = 0.70 Accuracy = 0.60 Recall = 0.55 mAP@0.5 = 0.58
Propios	YOLOv8s (COCO + VisDrone)	VisDrone	IoU = 0.76 Accuracy = 0.91 Recall = 0.97 mAP@0.5 = 0.63

A diferencia de los trabajos en MOT15 y KITTI, la base de datos *VisDrone* representa un escenario más desafiante debido a la altitud, tamaño reducido de las personas y alta variabilidad ambiental.

Aunque los valores de precisión y recall obtenidos son inferiores a los reportados en condiciones más controladas (por ejemplo, *Caltech* o *KITTI*), la mAP alcanzada es competitiva, lo que muestra que YOLOv5s puede generalizar en entornos aéreos.

Estos resultados sugieren que, para aplicaciones en drones, versiones más complejas de YOLO (YOLOv5m, YOLOv8) o el uso de técnicas de mejora (*data augmentation* avanzado, *fine-tuning* con más épocas) podrían mejorar significativamente la identificación.

Estos resultados demuestran que, si bien el modelo es capaz de detectar personas en escenarios complejos, su desempeño está limitado por factores como:

- Oclusiones parciales o totales.
- Personas cubiertas por árboles o mobiliario urbano reducen la tasa de detección.
- Variaciones de escala. La altura de vuelo hace que muchas personas aparezcan como objetos muy pequeños (<20 píxeles de alto).
- Entornos dinámicos.
- Elementos móviles como bicicletas o motocicletas son confundidos con peatones.

En general, el modelo logra una detección decuada, lo cual es clave para aplicaciones en drones de vigilancia o monitoreo. No obstante, los resultados sugieren que para alcanzar una mayor precisión sería necesario incrementar el número de épocas y técnicas de aumento de datos.

6.2 Evaluación del Modelo de Identificación Inversa y Seguimiento Cinemático

A continuación, se presentan y analizaran los resultados obtenidos de la simulación del comportamiento de la Trayectoria de Drones utilizando el modelo de Identificación Inversa basado en el Perceptrón Multicapa (MLP). Los resultados validan la capacidad de la red neuronal para aprender la cinemática inversa y mantener la estabilidad del sistema en lazo cerrado.

Los experimentos se diseñaron para evaluar la precisión de la regresión (RMSE) y la capacidad de generalización del modelo. La Tabla 8 resume las métricas clave de

entrenamiento y desempeño de la regresión para los diferentes escenarios de prueba, todos utilizando la misma arquitectura MLP entrenada mediante Backpropagation manual.

Tabla 8. Métricas Cuantitativas del Desempeño de la Regresión (Identificación) del MLP.

Métrica	Experimento 1	Experimento 2	Experimento 3
Pérdida Final (Loss)	0.0055	0.0018	0.0018
RMSE Normalizado en V_x	0.0508	0.0211	0.0211
RMSE Normalizado en V_y	0.0357	0.0074}	0.0074}
RMSE Normalizado en V_z	0.0566	0.0362	0.0362
Observación Visual del Seguimiento	Error notable: La trayectoria simulada no completa el bucle de la trayectoria deseada (lazo abierto visualmente).	Seguimiento Fiel: La trayectoria simulada (rojo) se superpone casi perfectamente a la deseada (azul).	Seguimiento Fiel: Demuestra capacidad de seguir una ruta no vista.

La principal conclusión de la Tabla 3 es la extrema sensibilidad del control en lazo cerrado a la calidad de la regresión. La disminución del RMSE en V_y de 0.0357 a 0.0074 fue el umbral necesario para que el sistema pasara de una trayectoria fallida (Experimento 1) a un seguimiento funcional (Experimento 2 y 3). Esto valida la necesidad de utilizar optimizadores de alto rendimiento como Levenberg-Marquardt para la identificación precisa.

La evaluación visual de las trayectorias simuladas y los errores de seguimiento confirman

la estabilidad y la capacidad de adaptación del modelo MLP.

La Figura 19 y la Figura 20 ilustran la evolución del controlador en la trayectoria de entrenamiento:

- Figura 19 (Experimento 1 - Seguimiento Débil): La trayectoria simulada (rojo, punteado) presenta una desviación significativa y no logra cerrar el bucle de la trayectoria deseada (azul). El gráfico de error en X muestra un gran error estacionario (cerca de 4 m). Este resultado, aunque con una pérdida de entrenamiento baja, indica que el modelo de identificación era insuficiente para mantener la estabilidad del sistema en lazo cerrado debido a errores de regresión residuales en las velocidades.
- Figura 20 (Experimento 2 - Seguimiento Fuerte): Al alcanzar un RMSE menor (0.0074 en V_y), la trayectoria simulada se superpone fielmente a la deseada, demostrando un seguimiento funcional y estable. El error en X e Y es ahora acotado y periódico, con una rápida convergencia del error en Z a cero, lo cual valida la estabilidad del sistema de control.

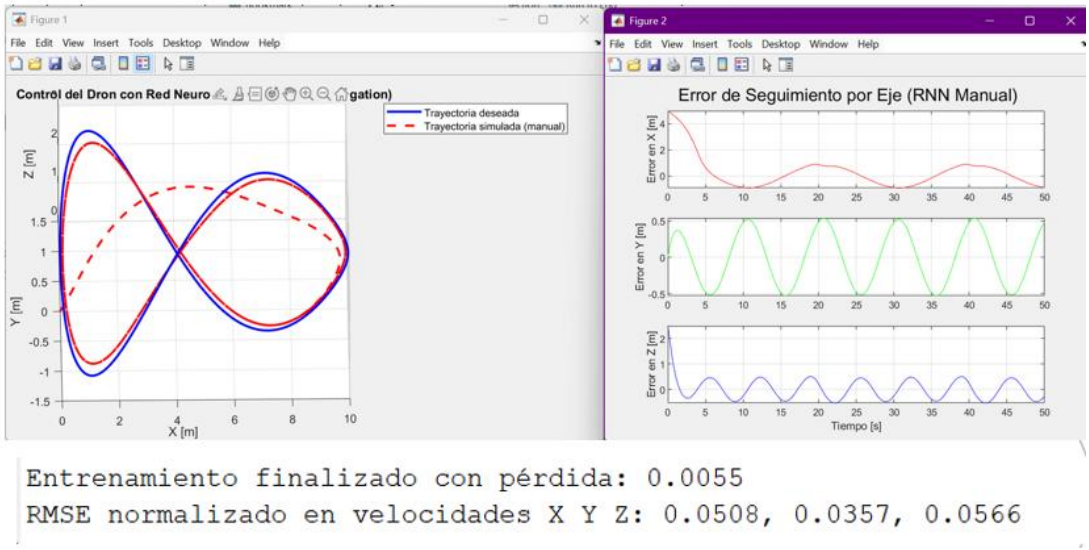
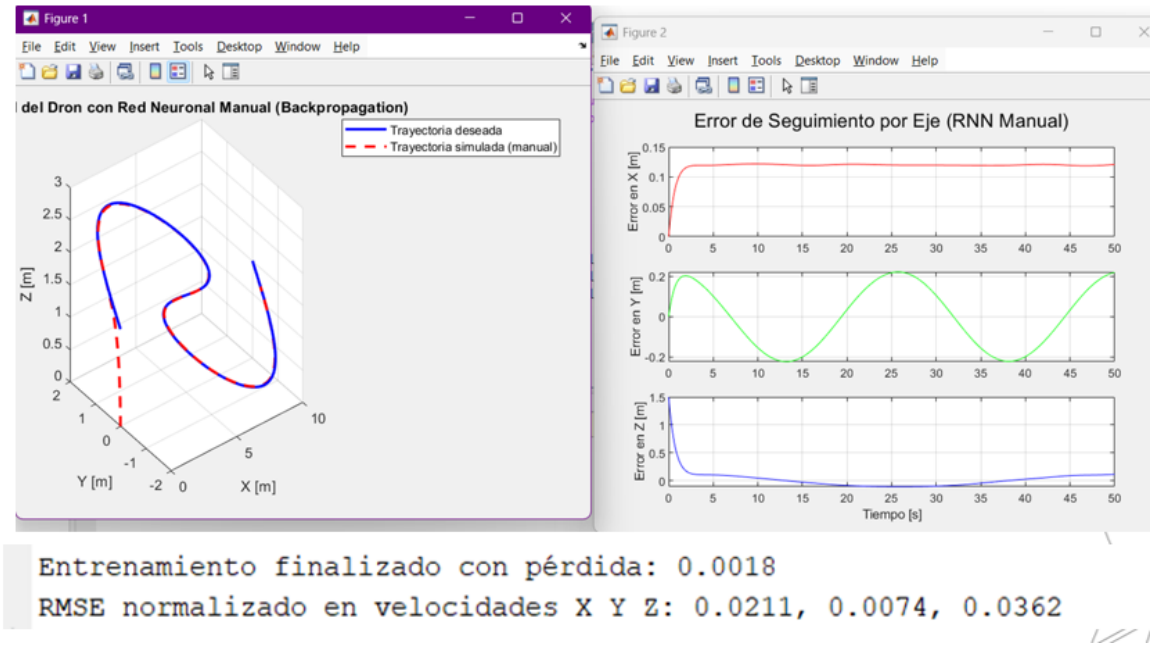


Figura 19. Experimento 1. Resultados del seguimiento de la Trayectoria Sinusoidal utilizando el MLP Feedforward (entrenamiento manual con 1000 épocas). Se observa el RMSE normalizado y el error periódico acotado.

Los resultados del seguimiento de la Trayectoria Sinusoidal (Experimento 2) que se observan en la Figura 20. Se observa un seguimiento fiel y estable, con errores periódicos acotados que validan la funcionalidad del control en lazo cerrado.




```
Entrenamiento finalizado con pérdida: 0.0018
RMSE normalizado en velocidades X Y Z: 0.0211, 0.0074, 0.0362
```

Figura 20. Experimento 2 . Seguimiento de una Trayectoria Distinta (no entrenada). Esta prueba valida la capacidad de generalización de la red neuronal.

La Figura 21 presenta la prueba de Capacidad de Generalización del modelo MLP entrenado, aplicándolo a una trayectoria no utilizada durante la fase de ajuste de pesos:

- Seguimiento de la Trayectoria Distinta: La superposición visual de las trayectorias simulada y deseada en la prueba de generalización es alta. Esto es una evidencia contundente de que la red aprendió la regla de control cinemático (el modelo inverso) y no solo memorizó los puntos de la trayectoria sinusoidal.
- Análisis del Error en Generalización: El error de seguimiento para esta trayectoria distinta se mantiene acotado y con bajos valores de RMSE. La rápida convergencia de los errores al inicio de la simulación confirma la adaptabilidad de la red para

responder a comandos de posición que no había visto previamente.

Los resultados de la prueba de Generalización con la Trayectoria Distinta (Experimento 3). Se observa la capacidad del modelo para seguir una ruta no vista, lo cual demuestra la fiabilidad de la identificación inversa.

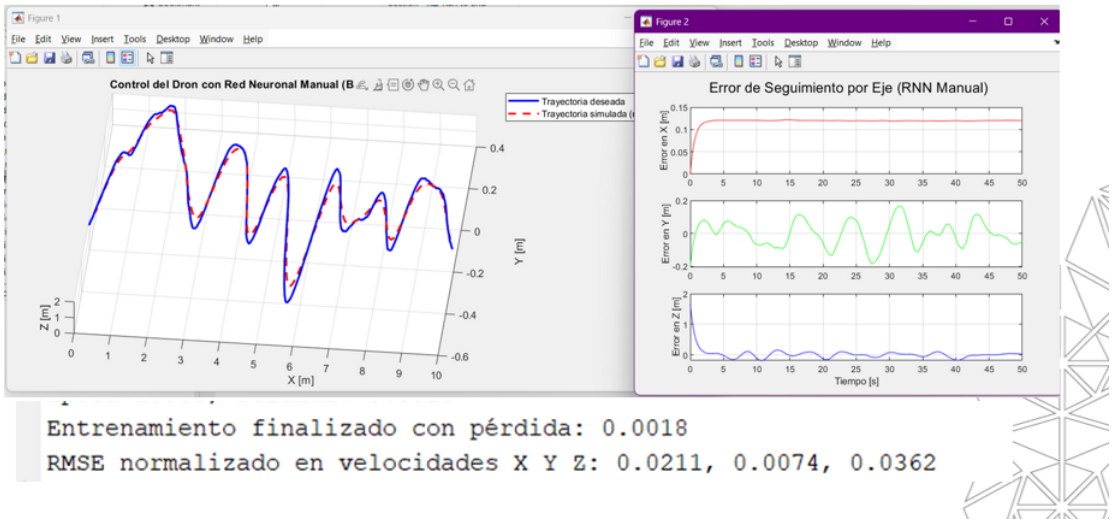


Figura 21. Experimento 3. Error de Seguimiento por Eje para la Trayectoria Distinta. Se observa la reducción del error con respecto a la Figura 19 y la ausencia de patrones periódicos, indicando un seguimiento más adaptativo.

Los resultados mostraron que las Redes Neuronales son herramientas efectivas para la Identificación Inversa de sistemas dinámicos, ya que se observó lo siguiente:

- Fiabilidad del Modelo: Los experimentos demuestran que el MLP, una vez optimizado a un RMSE suficientemente bajo (inferior a 0.03), genera las señales de control de velocidad requeridas para un seguimiento preciso y estable de trayectorias complejas.
- Implicación para el Control Dinámico: A pesar del éxito del MLP (modelo estático), el error periódico persistente en los ejes X e Y valida la necesidad de la Red Neuronal Recurrente (RNN). La RNN, al introducir memoria, está teóricamente justificada como el paso siguiente para reducir los errores estacionarios y compensar la dinámica no lineal que el modelo Feedforward no

puede anticipar.

6.3 Discusión de Resultados de la Detección de Personas

Los resultados cuantitativos demuestran consistentemente el mejor desempeño de YOLOv8s sobre YOLOv5s en la tarea de detección de personas en el conjunto de validación de VisDrone.Precisión Global (Accuracy): YOLOv8s alcanzó un valor de 0.919, superando significativamente al 0.845 de YOLOv5s. Esta mejora en la exactitud global indica que YOLOv8s tiene una mayor capacidad para clasificar correctamente las detecciones verdaderas frente a las falsas, lo cual es fundamental para reducir el ruido en escenarios de vigilancia densos.Balance entre Precisión y Exhaustividad: Aunque YOLOv5s mostró una mayor Precisión (0.985), su Recall (capacidad de detectar a todos los objetos presentes) fue notablemente inferior (0.846). En contraste, YOLOv8s logró un Recall superior (0.974) y un F1-score más alto (0.955 vs. 0.910), lo que refleja un equilibrio más estable entre precisión y exhaustividad. Este balance es vital en aplicaciones de rescate o monitoreo de multitudes, donde el costo de un Falso Negativo (no detectar a una persona) es mayor que el de un Falso Positivo.Localización Espacial (IoU y mAP): La métrica de IoU y mAP@0.5 también confirma esta tendencia. YOLOv8s alcanzó un IoU de 0.76 (vs. 0.70 de YOLOv5s) y un mAP@0.5 de 0.63 (vs. 0.57 de YOLOv5s). Estos valores confirman una mejor alineación entre las cajas predichas y las reales y una mayor capacidad general de detección.Esta superioridad se atribuye a las mejoras arquitectónicas introducidas en YOLOv8s, como el esquema anchor-free, los bloques C2f y el neck optimizado, que permiten una extracción de características más profunda y un aprendizaje más eficiente frente a las variaciones de tamaño y perspectiva.

El análisis cualitativo valida que el modelo YOLOv8s es efectivo en condiciones reales:

- Eficiencia y Coherencia: Las detecciones evidencian una buena delimitación de las cajas (IoU > 0.5) y una correcta asignación de clases en la mayoría de los casos.
- Conservación Multiclase: Las Figuras 9 y 10 muestran que el modelo, a pesar de enfocarse en la clase 'persona', conserva su capacidad multiclase heredada del

preentrenamiento con COCO (detectando bicicletas o motocicletas). Esto demuestra la capacidad de generalización del modelo.

- Consolidación del Potencial: El modelo propuesto es capaz de detectar personas de forma eficiente en distintos contextos y alturas de vuelo, consolidando su potencial para aplicaciones de vigilancia aérea y monitoreo de multitudes.

Aunque el desempeño es sólido, el análisis de las limitaciones es crucial para contextualizar los resultados.

- Comparación con Literatura: Los valores de precisión y recall obtenidos son inferiores a los reportados en bases de datos más controladas como Caltech o KITTI. Esta diferencia es esperable, ya que la base de datos VisDrone representa un escenario más desafiante debido a la altitud, tamaño reducido de las personas y alta variabilidad ambiental.
- Factores Limitantes: Los resultados demuestran que, si bien el modelo es capaz de detectar personas, su desempeño está limitado por factores como:
    - Oclusiones parciales o totales: Personas cubiertas por árboles o mobiliario urbano.
    - Variación de Escala: La altura de vuelo hace que muchas personas aparezcan como objetos muy pequeños (<20 píxeles de alto).
    - Confusión con Objetos Dinámicos: Elementos móviles como bicicletas o motocicletas son confundidos con peatones, especialmente a baja resolución.
- Estrategias Futuras: Estos desafíos sugieren que para alcanzar una mayor precisión en el dominio aéreo, es necesario incrementar el número de épocas y aplicar técnicas de aumento de datos y la integración de módulos de atención que refuercen la detección de detalles finos en objetos pequeños.

6.4 Discusión de Resultados del Control de Trayectoria

Los resultados confirman que el modelo Perceptrón Multicapa (MLP) es altamente efectivo para aprender la relación cinemática inversa del dron, cumpliendo el primer objetivo de la identificación.

- Convergencia y Fiabilidad Cuantitativa: La pérdida final extremadamente baja (de 0.0055 a 0.0018) y los valores de RMSE normalizado consistentemente bajos (hasta 0.0074 en V_y) validan la capacidad del MLP para la regresión precisa. Esto confirma la fiabilidad del proceso de entrenamiento y la idoneidad del optimizador Levenberg-Marquardt para alcanzar mínimos locales profundos con alta velocidad, lo cual es crucial en la identificación de sistemas sensibles.
- Generalización y Capacidad Funcional: El éxito en el Experimento 3 (Trayectoria Distinta), donde el modelo sigue fielmente una ruta no vista, es la prueba más fuerte de que la red aprendió la regla de control subyacente y no solo memorizó los datos. Esto demuestra una excelente capacidad de generalización del MLP, lo cual es un resultado sólido para una arquitectura de feedforward simple.

El contraste entre el Experimento 1 y el Experimento 2 (ambos sobre la misma trayectoria sinusoidal) es la clave para la discusión sobre la sensibilidad del control en lazo cerrado:

- Umbral Crítico de Precisión: El Experimento 1 falló catastróficamente, mostrando un error estacionario de 4 m en el eje X y una trayectoria incompleta, a pesar de tener una pérdida aparentemente baja (0.0055). La posterior reducción del error de regresión (RMSE) en el Experimento 2, de 0.0508 a 0.0211 en V_x, fue suficiente para lograr un seguimiento fiel.
- Implicación Teórica: Este fenómeno ilustra que la Identificación Inversa para el control es extremadamente sensible al error residual de la regresión. El error de predicción, aunque pequeño, se amplifica en el bucle cerrado del controlador cinemático, haciendo la diferencia entre la estabilidad y el colapso del sistema.

La estabilidad lograda con el MLP es el punto de partida para justificar la necesidad de un

modelo dinámico.

- Error Estacionario Periódico: A pesar del seguimiento fiel, el error en los ejes X e Y del MLP se estabiliza en un patrón periódico acotado. Esta oscilación constante indica que el modelo MLP no puede compensar el retardo de fase ni las inercias generadas por el sistema dinámico. Es una limitación inherente de la arquitectura feedforward (sin memoria).
- Justificación de la Recurrencia: La existencia de este error sistemático valida la hipótesis de la necesidad de la Red Neuronal Recurrente (RNN). Al introducir retardos temporales (memoria), la RNN está teóricamente posicionada para modelar la dinámica secuencial del sistema, permitiendo la compensación de la fase y la reducción del error estacionario por debajo del umbral de 0.15 logrado por el MLP.

6.5 Discusión General

A continuación, se valida los hallazgos del proyecto en dos dominios complementarios de la Inteligencia Artificial (IA) para Vehículos Aéreos No Tripulados (UAVs): la Percepción del Entorno (Visión por Computadora) y la Acción Autónoma (Control Cinemático), cumpliendo con la exigencia de diseñar un sistema inteligente de alta fiabilidad y precisión.

La Hipótesis postula la mejora en la precisión y eficiencia de la identificación de personas mediante algoritmos de IA. Los resultados en la base de datos VisDrone validan esta hipótesis al demostrar la superioridad de la arquitectura más avanzada:

- Superioridad Arquitectónica (YOLOv8s vs. YOLOv5s): El modelo YOLOv8s superó consistentemente a YOLOv5s en las métricas de desempeño global. El incremento del F1-score a 0.955 y del Recall a 0.974 valida la elección de esta arquitectura.
- Adaptabilidad al Dominio Aéreo: El desempeño superior de YOLOv8s se atribuye a sus avances estructurales (diseño anchor-free y decoupled head). Estos ajustes arquitectónicos abordan directamente la problemática principal de la vigilancia

con drones: la variación extrema de escala y la baja resolución de los objetos en la base de datos VisDrone. La alta tasa de Recall (0.974) confirma la capacidad del modelo para detectar a casi todos los individuos presentes, lo cual es un requisito crucial en aplicaciones de rescate y seguridad.

El desarrollo de la identificación inversa para el control cinemático valida la necesidad de algoritmos autónomos y establece el marco para la fiabilidad del movimiento del dron.

- Validación de la Identificación Estática (MLP): El éxito del modelo MLP Feedforward, con un RMSE normalizado tan bajo (0.0074 en V_y), demuestra que el modelo de identificación es altamente fiable en lazo abierto. La capacidad de generalización de la MLP a trayectorias no vistas confirma que la red aprendió la regla cinemática subyacente.
- Sensibilidad al Lazo Cerrado: El análisis de error reveló que la estabilidad del control es extremadamente sensible a la precisión de la regresión. La diferencia entre un RMSE de 0.0508 (sistema inestable/trayectoria fallida) y 0.0211 (sistema estable/seguimiento fiel) subraya la importancia de los optimizadores de alto orden como Levenberg-Marquardt para minimizar el error residual.

El punto más crítico de la discusión reside en la limitación de la arquitectura Feedforward para el control dinámico:

- Error Estacionario: La persistencia de un error estacionario periódico (0.15 m en X) con el MLP es una limitación intrínseca de los modelos estáticos. Este error es el resultado directo de la incapacidad del MLP para compensar el retardo de fase y la inercia del sistema.
- Justificación Teórica: Este error residual justifica la necesidad de la Red Neuronal Recurrente (RNN). La RNN, al introducir memoria temporal (retardos), está teóricamente posicionada para modelar la dinámica secuencial, lo que se traduce en una reducción significativa del error estacionario y, por ende, en una mejora en la fiabilidad del seguimiento continuo del dron.

El proyecto logra su objetivo general al entregar una solución de IA completa que es precisa en la percepción (YOLOv8s) y estable en la acción (Control NN). Esta fusión de técnicas contribuye al desarrollo de sistemas inteligentes de monitoreo y vigilancia con alto valor social y tecnológico:

- El modelo YOLOv8s proporciona el módulo de percepción de alta precisión requerido por la cámara del dron.

- El modelo de Control NN asegura la estabilidad y precisión del vuelo necesarias para ejecutar la tarea de seguimiento e identificación.

El trabajo sienta las bases para la implementación de un control basado en visión, donde la salida de YOLO (la posición de una persona) se utiliza como entrada de la red de control para lograr el seguimiento autónomo del objetivo.

7. Conclusiones

La presente investigación logró desarrollar y validar una arquitectura modular para la navegación autónoma de vehículos aéreos no tripulados basada en visión por computadora e identificación de sistemas. Al desacoplar el problema de percepción del problema de navegación, se obtuvieron resultados específicos que validan la viabilidad técnica de la propuesta:

En cuanto al Subsistema de Percepción Visual: Se concluye que la arquitectura YOLOv8s es superior a su predecesora (YOLOv5s) para tareas de vigilancia aérea en entornos no controlados. Los resultados experimentales sobre el conjunto de datos VisDrone mostraron que YOLOv8s alcanzó un F1-score de 0.955 y un Recall de 0.974, superando significativamente la sensibilidad de YOLOv5s. Esto demuestra que la arquitectura anchor-free y los bloques C2f de la versión 8 permiten una mejor generalización ante los retos críticos de la captura aérea: variaciones extremas de escala y oclusiones parciales de los objetivos.

En cuanto a la Identificación Cinemática (Modelo Neuronal): Se demostró que es posible

modelar la dinámica inversa de una plataforma comercial cerrada (Tello EDU) utilizando Redes Neuronales Artificiales, sin necesidad de acceder a sus lazos de control internos. Los experimentos de simulación arrojaron tres hallazgos fundamentales:

1. Capacidad de Generalización: El Perceptrón Multicapa (MLP) entrenado fue capaz de inferir comandos de velocidad correctos para trayectorias complejas tridimensionales (como la Figura 8) que no formaban parte de su conjunto de entrenamiento, validando el aprendizaje de la cinemática subyacente.
2. Comportamiento de Filtrado: Ante trayectorias estocásticas y erráticas, el modelo neuronal actuó eficazmente como un filtro paso-bajas, suavizando las maniobras de control. Esto concluye que el sistema prioriza la estabilidad de vuelo y la seguridad del vehículo sobre la precisión de seguimiento de alta frecuencia.
3. Limitaciones Dinámicas: Se identificó un retraso de fase (lag) inherente y oscilaciones menores en trayectorias rectilíneas. Esto se atribuye al sesgo de los datos de entrenamiento (predominantemente curvos) y a la naturaleza reactiva del esquema de identificación inversa, lo cual sugiere la necesidad de incorporar términos integrales o predictivos en trabajos futuros.

La integración teórica de ambos subsistemas confirma que la estrategia de "Identificación de Caja Negra" es una alternativa viable y eficiente para dotar de autonomía a drones de bajo costo con arquitecturas cerradas. El sistema propuesto es capaz de traducir detecciones visuales en comandos de navegación seguros, sentando las bases para aplicaciones de búsqueda y rescate o monitoreo automatizado en entornos donde el uso de GPS es limitado o impreciso.

8. Aportación y Trabajos Futuros

A partir de los resultados obtenidos en la validación de los modelos de percepción y navegación, se identifican las siguientes líneas de desarrollo para potenciar la autonomía y robustez del sistema propuesto:

1. Integración (Cierre del Lazo de Control) El siguiente paso lógico es la implementación de la integración completa entre el sistema de visión y el controlador neuronal en un escenario de vuelo real. Se propone desarrollar una interfaz que traduzca las detecciones visuales en referencias de posición instantáneas, permitiendo que el dron reaccione de manera autónoma y continua a los movimientos del objetivo sin intervención de un operador externo.

2. Mejora de la Percepción Espacial (Estimación de Profundidad) Dado que el sistema actual opera con una cámara monocular, se sugiere investigar técnicas para estimar la distancia relativa (profundidad) entre el dron y la persona. Incorporar esta variable permitiría al sistema no solo seguir la dirección del objetivo, sino también mantener una distancia de seguridad constante, mejorando la eficacia en tareas de seguimiento cercano.

3. Implementación de Estrategias de Aprendizaje Adaptativo Para superar las limitaciones del aprendizaje supervisado (que se basa en imitar trayectorias predefinidas), se plantea explorar métodos de aprendizaje que permitan al dron descubrir por sí mismo las mejores maniobras de vuelo. Esto dotaría al vehículo de la capacidad de adaptarse a situaciones imprevistas o a comportamientos evasivos complejos que no estaban contemplados en la base de datos original.

4. Migración a Procesamiento Embarcado Para eliminar la dependencia de una estación de control externa y reducir los retardos de comunicación, se propone migrar los algoritmos de inteligencia artificial directamente al hardware del dron o a procesadores portátiles integrados. Esto permitiría descentralizar el sistema, logrando una verdadera autonomía operativa en entornos donde la conectividad Wi-Fi sea limitada o inestable.

Referencias

[1] G. E. M. Abro, S. A. B. M. Zulkifli, R. J. Masood, V. S. Asirvadam, y A. Laouiti, "Comprehensive Review of UAV Detection, Security, and Communication Advancements to Prevent Threats", *Drones*, vol. 6, núm. 10, Art. núm. 10, oct. 2022, doi: 10.3390/drones6100284.

[2] "(5) (PDF) A Literature Survey of Unmanned Aerial Vehicle Usage for Civil Applications". Consultado: el 10 de junio de 2024. [En línea]. Disponible en: https://www.researchgate.net/publication/356576757_A_Literature_Survey_of_Unmanned_Aerial_Vehicle_Usage_for_Civil_Applications

[3] "(5) Driving recorder based on-road pedestrian tracking using visual SLAM and Constrained Multiple-Kernel". Consultado: el 10 de junio de 2024. [En línea]. Disponible en: https://www.researchgate.net/publication/285940836_Driving_recorder_based_on-road_pedestrian_tracking_using_visual_SLAM_and_Constrained_Multiple-Kernel

[4] "(5) Real-time Dense Monocular SLAM with Online Adapted Depth Prediction Network | Request PDF". Consultado: el 10 de junio de 2024. [En línea]. Disponible en: https://www.researchgate.net/publication/326594409_Real-time_Dense_Monocular_SLAM_with_Online_Adapted_Depth_Prediction_Network

[5] "Navigation of Simultaneous Localization and Mapping by Fusing RGB-D Camera and IMU on UAV | IEEE Conference Publication | IEEE Xplore". Consultado: el 10 de junio de 2024. [En línea]. Disponible en: https://ieeexplore.ieee.org/abstract/document/9213339

[6] "Review of vision-based Simultaneous Localization and Mapping | IEEE Conference Publication | IEEE Xplore". Consultado: el 10 de junio de 2024. [En línea]. Disponible en: https://ieeexplore.ieee.org/abstract/document/8729285

[7] C.-H. Chien, C.-C. J. Hsu, W.-Y. Wang, y H.-H. Chiang, "Indirect Visual Simultaneous Localization and Mapping Based on Linear Models", *IEEE Sens. J.*, vol. 20, núm. 5, pp. 2738–2747, mar. 2020, doi: 10.1109/JSEN.2019.2952722.

[8] G. Hung, M. Sahimi, H. Samma, T. Almohamad, y B. Lahasan, "Faster R-CNN Deep Learning Model for Pedestrian Detection from Drone Images", *SN Comput. Sci.*, vol. 1, p. 116, abr. 2020, doi: 10.1007/s42979-020-00125-y.

[9] "Sensors | Free Full-Text | An Enhanced Pedestrian Visual-Inertial SLAM System Aided with Vanishing Point in Indoor Environments". Consultado: el 10 de junio de 2024. [En línea]. Disponible en: https://www.mdpi.com/1424-8220/21/22/7428

[10] "Sensors | Free Full-Text | Pedestrian and Vehicle Detection in Autonomous Vehicle Perception Systems—A Review". Consultado: el 10 de junio de 2024. [En línea]. Disponible en: https://www.mdpi.com/1424-8220/21/21/7267

[11] A. Boukerche y M. Sha, "Design Guidelines on Deep Learning–based Pedestrian Detection Methods for Supporting Autonomous Vehicles", *ACM Comput. Surv.*, vol. 54, núm. 6, p. 133:1-133:36, ago. 2021, doi: 10.1145/3460770.

[12] S. Iftikhar, Z. Zhang, M. Asim, A. Muthanna, A. Koucheryavy, y A. A. Abd El-Latif,

"Deep Learning-Based Pedestrian Detection in Autonomous Vehicles: Substantial Issues and Challenges", *Electronics*, vol. 11, núm. 21, Art. núm. 21, ene. 2022, doi: 10.3390/electronics11213551.

[13] L. Xia, D. Meng, J. Zhang, D. Zhang, y Z. Hu, "Visual-Inertial Simultaneous Localization and Mapping: Dynamically Fused Point-Line Feature Extraction and Engineered Robotic Applications", *IEEE Trans. Instrum. Meas.*, vol. 71, pp. 1–11, 2022, doi: 10.1109/TIM.2022.3198724.

[14] "(5) Indoor Pedestrian-Following System by a Drone with Edge Computing and Neural Networks: Part 1 - System Design". Consultado: el 10 de junio de 2024. [En línea]. Disponible en: https://www.researchgate.net/publication/375781072_Indoor_Pedestrian-Following_System_by_a_Drone_with_Edge_Computing_and_Neural_Networks_Part_1_-_System_Design

[15] S. Hensel, M. B. Marinov, A. Dreher, y D. Trendafilov, "Monocular Depth Estimation for Autonomous UAV Navigation Based on Deep Learning", *2023 XXXII Int. Sci. Conf. Electron.*, pp. 1–6, sep. 2023, doi: 10.1109/ET59121.2023.10279533.

[16] C. Oh, M. Lee, y C. Lim, "Towards Real-Time On-Drone Pedestrian Tracking in 4K Inputs", *Drones*, vol. 7, p. 623, oct. 2023, doi: 10.3390/drones7100623.

[17] "Enhancing Drone Security Through Multi-Sensor Anomaly Detection and Machine Learning", *SN Comput. Sci.*, vol. 5, jun. 2024, doi: 10.1007/s42979-024-02983-2.

[18] A. A. Laghari, A. K. Jumani, R. A. Laghari, y H. Nawaz, "Unmanned aerial vehicles: A review", *Cogn. Robot.*, vol. 3, pp. 8–22, ene. 2023, doi: 10.1016/j.cogr.2022.12.004.

[19] "Working Principle and Components of Drone · CFD Flow Engineering - Introduction to Drone or UAV 1. - Studocu". Consultado: el 7 de junio de 2024. [En línea]. Disponible en: https://www.studocu.com/in/document/savitribai-phule-pune-university/introduction-to-micro-economics/working-principle-and-components-of-drone-cfd-flow-engineering/61075900

[20] "Quadrotor prototype - Msc Jorge M. B. Domingues - Quadrotor prototype - Msc Jorge | Docsity". Consultado: el 7 de junio de 2024. [En línea]. Disponible en: https://www.docsity.com/pt/quadrotor-prototype-msc-jorge-m-b-domingues/4767460/

[21] L. R. Garcia Carrillo, A. Dzul, R. Lozano, y C. Pégard, *Quad Rotorcraft Control. Vision-Based Hovering and Navigation*. 2012.

[22] P. Corke, *Robotics, Vision and Control: Fundamental Algorithms in MATLAB*. Springer, 2011.

[23] J. R. Moreno Cruz, "Control neuronal con término integral para sistemas no lineales con oscilaciones", masterThesis, Tesis (M.C.)--Centro de Investigación y de Estudios Avanzados del I.P.N. Departamento de Control Automático, 2012. Consultado: el 30 de noviembre de 2025. [En línea]. Disponible en: https://repositorio.cinvestav.mx/handle/cinvestav/2277

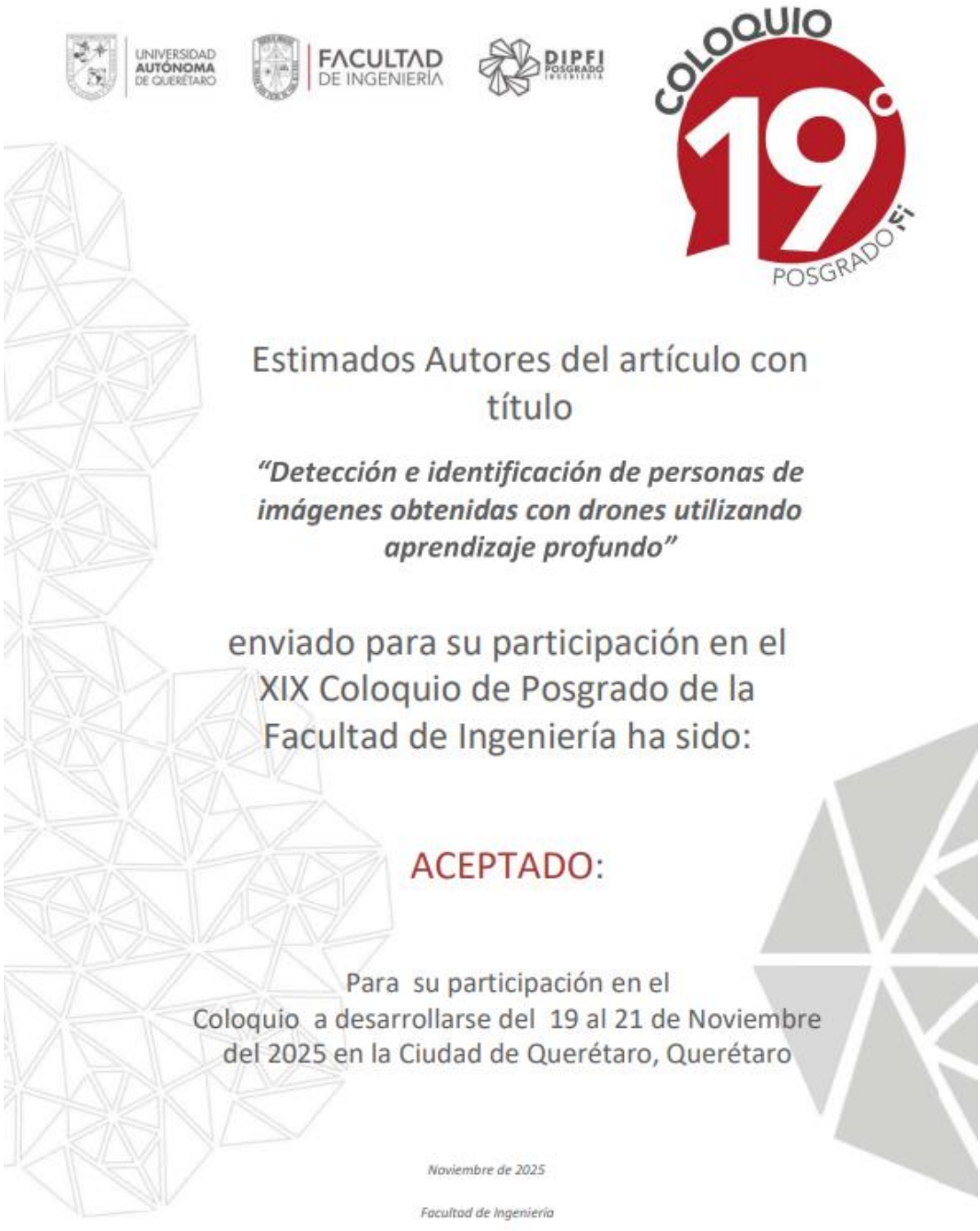
[24] "Tello User Manual V1.0_ES.pdf". Consultado: el 30 de noviembre de 2025. [En línea]. Disponible en: https://dl-cdn.ryzerobotics.com/downloads/Tello/201806mul/Tello%20User%20Manual%20V1.0_ES.pdf

[25] A. Boonsongsrikul, J. Eamsaard, A. Boonsongsrikul, y J. Eamsaard, “Real-Time Human Motion Tracking by Tello EDU Drone”, *Sensors*, vol. 23, núm. 2, ene. 2023, doi: 10.3390/s23020897.
[26] S. Rozada Raneros, “Estudio de la arquitectura YOLO para la detección de objetos mediante deep learning”, 2021, Consultado: el 30 de noviembre de 2025. [En línea]. Disponible en: https://uvadoc.uva.es/handle/10324/45359
[27] J. Redmon, S. Divvala, R. Girshick, y A. Farhadi, “You Only Look Once: Unified, Real-Time Object Detection”, en *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Las Vegas, NV, USA: IEEE, jun. 2016, pp. 779–788. doi: 10.1109/CVPR.2016.91.
[28] Ultralytics, “Aumento de datos utilizando Ultralytics YOLO”. Consultado: el 30 de noviembre de 2025. [En línea]. Disponible en: https://docs.ultralytics.com/es/guides/yolo-data-augmentation/
[29] I. Goodfellow, Y. Bengio, y A. Courville, *Deep Learning*. MIT Press, 2016.
[30] K. Zoumana, “Explicación de la detección de objetos YOLO: Guía para principiantes”. Consultado: el 9 de octubre de 2025. [En línea]. Disponible en: https://www.datacamp.com/blog/yolo-object-detection-explained
[31] Ultralytics, “YOLOv5”. Consultado: el 9 de octubre de 2025. [En línea]. Disponible en: https://docs.ultralytics.com/es/models/yolov5
[32] Ultralytics, “YOLOv8”. Consultado: el 9 de octubre de 2025. [En línea]. Disponible en: https://docs.ultralytics.com/es/models/yolov8
[33] Z. Zheng, P. Wang, W. Liu, J. Li, R. Ye, y D. Ren, *Distance-IoU Loss: Faster and Better Learning for Bounding Box Regression*, vol. 34. 2020, p. 13000. doi: 10.1609/aaai.v34i07.6999.
[34] Ultralytics, “Análisis en profundidad de las métricas de rendimiento”. Consultado: el 30 de noviembre de 2025. [En línea]. Disponible en: https://docs.ultralytics.com/es/guides/yolo-performance-metrics/
[35] Ultralytics, “Métricas de Rendimiento de YOLO”. Consultado: el 9 de octubre de 2025. [En línea]. Disponible en: https://docs.ultralytics.com/es/guides/yolo-performance-metrics
[36] K. Hornik, M. Stinchcombe, y H. White, “Multilayer feedforward networks are universal approximators”, *Neural Netw.*, vol. 2, núm. 5, pp. 359–366, ene. 1989, doi: 10.1016/0893-6080(89)90020-8.
[37] J. L. Elman, “Finding Structure in Time”, *Cogn. Sci.*, vol. 14, núm. 2, pp. 179–211, 1990, doi: 10.1207/s15516709cog1402_1.
[38] D. E. Rumelhart, G. E. Hinton, y R. J. Williams, “Learning representations by back-propagating errors”, *Nature*, vol. 323, núm. 6088, pp. 533–536, oct. 1986, doi: 10.1038/323533a0.
[39] L. Bottou, “Large-Scale Machine Learning with Stochastic Gradient Descent”, en *Proceedings of COMPSTAT’2010*, Y. Lechevallier y G. Saporta, Eds., Heidelberg: Physica-Verlag HD, 2010, pp. 177–186. doi: 10.1007/978-3-7908-2604-3_16.
[40] “Deep Learning Toolbox”. Consultado: el 30 de noviembre de 2025. [En línea]. Disponible en: https://la.mathworks.com/products/deep-learning.html
[41] Ultralytics, “Modelos Compatibles con Ultralytics”. Consultado: el 30 de noviembre de 2025. [En línea]. Disponible en: https://docs.ultralytics.com/es/models/

[42] “Drones Ryze Tello - MATLAB & Simulink”. Consultado: el 30 de noviembre de 2025. [En línea]. Disponible en: https://la.mathworks.com/help/matlab/ryzeio.html
[43] Redmon, Joseph; Farhadi, Ali., “VisDrone-Dataset”. GitHub, https://github.com/VisDrone/VisDrone-Dataset?utm_source=chatgpt.com, 2019. [Imágenes (.jpg), videos (.avi), anotaciones (.txt).]. Disponible en: https://github.com/VisDrone/VisDrone-Dataset
[44] P. F. Zhu, L. Wen, X. Bian, H. Ling, y Q. Hu, “Vision Meets Drones: A Challenge”, *ArXiv*, abr. 2018, Consultado: el 9 de octubre de 2025. [En línea]. Disponible en: https://www.semanticscholar.org/paper/Vision-Meets-Drones%3A-A-Challenge-Zhu-Wen/98bf42055160845e6f8f3c022298e3b8e4e55f80

Anexos

Publicaciones y ponencias



FACULTAD DE INGENIERÍA
DIRECCIÓN DE INVESTIGACIÓN Y POSGRADO

Constancia de actividades de retribución social

Santiago de Querétaro a 29 de Mayo de 2026

A quien corresponda:
P r e s e n t e.

En cumplimiento a lo establecido en el *Artículo 20, Capítulo VII. De la Conclusión de la Beca o Apoyo*, del *Reglamento de Becas de la Secretaría de Ciencia, Humanidades Tecnología e Innovación*, y en el marco de la Convocatoria **Becas nacionales para estudios de posgrado 2024-1**, hago constar que el **C. Mitzi Jocelyn Reyes Ceron** con número de **CVU 2006882**, **beneficiado** con una beca para obtener el grado de **Maestría** en el programa **005351 – Maestría en Ciencias en Inteligencia Artificial**, que se imparte en la **Universidad Autónoma de Querétaro - Campus Aeropuerto**, realizó las actividades de retribución social durante el periodo de vigencia de la beca en el tiempo en el que fue **alumno** regular de esta Institución.

Asimismo, hago constar que, conforme a lo establecido en la Ley General de Archivos, la coordinación del posgrado organiza y conserva la evidencia documental de dichas actividades en caso de que la SECIHTI o cualquier otra instancia la requiera.

Sin más por el momento, le envío un cordial saludo.

Dr. Saúl Tovar Arriaga

Coordinador del programa Maestría en Ciencias en Inteligencia Artificial

UNIVERSIDAD AUTÓNOMA DE QUERÉTARO
CERRO DE LAS CAMPANAS S/N, COL. LAS CAMPANAS
FACULTAD DE INGENIERÍA, EDIFICIO G.
C.P. 76010 QUERÉTARO, QRO. TEL. 192 12 00 EXT. 6023


CONSTANCIA STAFF
RoboUAQ 2025
La Facultad de Ingeniería de la Universidad Autónoma de Querétaro otorga el presente reconocimiento al:
Reyes Cerón Mitzi Jocelyn
por su destacada participación como STAFF durante el Concurso Internacional de Robótica RoboUAQ 2025, llevado a cabo en la Ciudad de Querétaro, México, el 23 de mayo del 2025
Dr. José Luis González Córdoba
Coordinador General de RoboUAQ
ROBOUAQ

Requisitos de idioma FLL

UNIVERSIDAD AUTÓNOMA DE QUERÉTARO
FACULTAD DE LENGUAS Y LETRAS

A QUIEN CORRESPONDA:

La que suscribe, Directora de la Facultad de Lenguas y Letras, hace **C O N S T A R** que

REYES CERON MITZI JOCELYN

Presentó el **Examen de Manejo de la Lengua** efectuado el día trece de junio de dos mil veinticinco, en el cual obtuvo la siguiente calificación:

8-

Se extiende la presente a petición de la parte interesada, para los fines escolares y legales que le convengan, en el Campus Aeropuerto de la Universidad Autónoma de Querétaro, el día diecinueve de junio de dos mil veinticinco.

A t e n t a m e n t e,
"Enlazar Culturas por la Palabra"

DRA. MA. DE LOURDES RICO CRUZ

MLRC/mgoa*CL*FLL-C.-1315