



UNIVERSIDAD TÉCNICA DE AMBATO

**FACULTAD DE INGENIERÍA EN SISTEMAS, ELECTRÓNICA E
INDUSTRIAL**

CARRERA DE TELECOMUNICACIONES

Tema:

**SISTEMA DE CONTROL DE CALIDAD DE CULTIVO DE FRUTA DE
TEMPORADA PARA ETAPA DE PRECOSECHA EMPLEANDO
ROBÓTICA AÉREA CON PLANIFICACIÓN DE TRAYECTORIAS Y
VISIÓN ARTIFICIAL**

Trabajo de titulación modalidad Proyecto de Investigación, presentado previo a la
obtención del título de Ingeniero en Telecomunicaciones

ÁREA: Comunicaciones

LÍNEA DE INVESTIGACIÓN: Tecnología de la información y Sistemas de
control

AUTOR: Michelle Ivette Ashqui Balseca

Bryan Paul Aucatoma Matias

TUTOR: Ing. Edgar Patricio Córdova Córdova, Mg.

Ambato - Ecuador

agosto – 2024

APROBACIÓN DEL TUTOR

En calidad de tutor del trabajo de titulación con el tema: SISTEMA DE CONTROL DE CALIDAD DE CULTIVO DE FRUTA DE TEMPORADA PARA ETAPA DE PRECOSECHA EMPLEANDO ROBÓTICA AÉREA CON PLANIFICACIÓN DE TRAYECTORIAS Y VISIÓN ARTIFICIAL desarrollado bajo la modalidad Proyecto de Investigación por la señorita Michelle Ivette Ashqui Balseca y el señor Bryan Paul Aucatoma Matias, estudiantes de la Carrera de Telecomunicaciones, de la Facultad de Ingeniería en Sistemas, Electrónica e Industrial, de la Universidad Técnica de Ambato, me permito indicar que los estudiantes han sido tutorados durante todo el desarrollo del trabajo hasta su conclusión, de acuerdo a lo dispuesto en el Artículo 17 del Reglamento para la Titulación de Grado en la Universidad Técnica de Ambato y el numeral 6.3 del instructivo del reglamento referido.

Ambato, agosto 2024.



Ing. Edgar Patricio Córdova Córdova, Mg.

TUTOR

AUTORÍA

El presente trabajo de titulación con el tema: SISTEMA DE CONTROL DE CALIDAD DE CULTIVO DE FRUTA DE TEMPORADA EN ETAPA DE PRECOSECHA EMPLEANDO ROBÓTICA AÉREA CON PLANIFICACIÓN DE TRAYECTORIAS Y VISIÓN ARTIFICIAL es absolutamente original, auténtico y personal y ha observado los preceptos establecidos en la Disposición General Quinta del Reglamento para la Titulación de Grado en la Universidad Técnica de Ambato. En tal virtud, el contenido, efectos legales y académicos que se desprenden del mismo son de exclusiva responsabilidad del autor.

Ambato, agosto 2024.



Michelle Ivette Ashqui Balseca

C.C. 1804733382

AUTOR



Bryan Paul Aucatoma Matias

C.C. 1805242235

AUTOR

DERECHOS DE AUTOR

Autorizo a la Universidad Técnica de Ambato para que reproduzca total o parcialmente este trabajo de titulación dentro de las regulaciones legales e institucionales correspondientes. Además, cedo todos mis derechos de autor a favor de la institución con el propósito de su difusión pública, por lo tanto, autorizo su publicación en el repositorio virtual institucional como un documento disponible para la lectura y uso con fines académicos e investigativos de acuerdo con la Disposición General Cuarta del Reglamento para la Titulación de Grado en la Universidad Técnica de Ambato.

Ambato, agosto 2024.



Michelle Ivette Ashqui Balseca

C.C. 1804733382

AUTOR



Bryan Paul Aucatoma Matias

C.C. 1805242235

AUTOR

APROBACIÓN DEL TRIBUNAL DE GRADO

En calidad de par calificador del informe final del trabajo de titulación presentado por la señorita Michelle Ivette Ashqui Balseca y el señor Bryan Paul Aucatoma Matias, estudiantes de la Carrera de Telecomunicaciones, de la Facultad de Ingeniería en Sistemas, Electrónica e Industrial, bajo la Modalidad Proyecto de Investigación, titulado SISTEMA DE CONTROL DE CALIDAD DE CULTIVO DE FRUTA DE TEMPORADA EMPLEANDO ROBÓTICA AÉREA CON PLANIFICACIÓN DE TRAYECTORIAS Y VISIÓN ARTIFICIAL, nos permitimos informar que el trabajo ha sido revisado y calificado de acuerdo al Artículo 19 del Reglamento para la Titulación de Grado en la Universidad Técnica de Ambato y el numeral 6.4 del instructivo del reglamento referido. Para cuya constancia suscribimos, conjuntamente con la señora Presidente del Tribunal.

Ambato, agosto 2024.



Firmado electrónicamente por:
ELSA PILAR URRUTIA
URRUTIA

Ing. Elsa Pilar Urrutia Urrutia, Mg.
PRESIDENTE DEL TRIBUNAL



Firmado electrónicamente por:
ANA PAMELA CASTRO
MARTIN

Ing. Ana Pamela Castro Martin, MSc.

PROFESOR CALIFICADOR



Firmado electrónicamente por:
JESUS ISRAEL GUAMAN
MOLINA

Ing. Jesús Israel Guamán Molina, Mg.

PROFESOR CALIFICADOR

DEDICATORIA

El presente proyecto de investigación se lo dedico a Dios. Él ha sido mi fortaleza en momentos de incertidumbre, mi refugio cuando me sentía vulnerable y mi principal fuente de inspiración en cada paso de este arduo y gratificante proceso. "Confía en el Señor de todo corazón, y no en tu propia inteligencia. Reconócelo en todos tus caminos, y él allanará tus sendas." Consciente de que, sin su guía y gracia, nada de esto habría sido posible, le dedico todo el mérito y la gloria de este logro.

A mis queridos padres, cuya fe inquebrantable en mis capacidades y constante apoyo han sido fundamentales para alcanzar este hito en mi vida. Quiero dedicar un reconocimiento especial a mi madre, Amarilis, una mujer que siempre ha deseado que sus hijos sean tan buenos como ella aspiró a ser, cuyo amor incondicional, sabias enseñanzas y ternura han dejado una huella imborrable en mi vida. Esta tesis es un humilde testimonio de mi profunda gratitud por todo lo que has hecho por mí, por ser mi faro de luz y mi fuente constante de ánimo.

A mis amados hermanos Josué, Mateo, Génesis y Sabrina, quienes han sido mi inspiración y motivo de lucha constante. Su presencia en mi vida es un regalo invaluable, y su amor y apoyo me han impulsado a superar cada obstáculo. Este logro es tanto mío como de ustedes, pues hemos trabajado juntos en espíritu y corazón para alcanzarlo.

Michelle Ashqui

AGRADECIMIENTO

A Dios, por concederme la salud, la sabiduría y la fortaleza necesaria para culminar mi carrera.

A mis queridos padres y hermanos, por su constante apoyo, amor incondicional y fe en mí a lo largo de este arduo camino.

A mis abuelitos, por su amor incondicional, su sabiduría y sus valiosos consejos. Gracias por enseñarme con su ejemplo que todo es posible y por ser mi inspiración en cada paso de este camino.

A mis tíos, por su constante motivación y por estar presentes en cada una de las etapas importantes de mi vida. Un agradecimiento especial a Ruth y Micaela, cuyo apoyo ha sido especialmente significativo.

A mis amigos David, Antony, Christopher y Rick, por darme la fuerza y la motivación necesarias para seguir adelante en este camino. Su apoyo incondicional y su amistad han sido pilares fundamentales en mi vida.

A Steven , por su incondicional ayuda, infinita paciencia y valiosas enseñanzas a lo largo de este camino. Su apoyo ha sido esencial para superar los desafíos y alcanzar este logro.

A mi amigo Paul Aucatoma, quien me acompañó durante esta investigación, aportando todo su conocimiento y colaborando estrechamente para cumplir esta meta. Tu apoyo ha sido invaluable.

A mi tutor de tesis, el Ingeniero Patricio Córdova, por el tiempo dedicado a apoyarme y guiarme para alcanzar este logro. Su orientación y sabiduría han sido fundamentales en este proceso, y le estoy profundamente agradecido.

Michelle Ashqui

DEDICATORIA

Este proyecto de investigación va dedicado en primer lugar a Dios y la Virgencita de El Cisne por otorgarme las fuerzas necesarias y sabiduría para concluir con éxito mi carrera profesional.

A mis amados padres, Joselito y Rosita que con su enorme esfuerzo y apoyo incondicional han sido mi fuente constante de motivación y fortaleza. Gracias por creer en mí y por brindarme las herramientas para alcanzar mis sueños.

A mi querida esposa, Jessenia, por su amor, paciencia y apoyo incansable. Tu comprensión y sacrificio han sido esenciales para lograr este objetivo. Gracias por creer en mí.

A mi adorada hija, Cielo, quien con su ternura y alegría ilumina mis días. Eres mi mayor inspiración y el motivo por el cual me esfuerzo cada día para ser mejor. Tu sonrisa y amor me han dado la fuerza para seguir adelante.

A mis hermanas, Michelle y Katherine, mi abuelita Antonia, a la memoria de mi abuelita Hermelinda y demás familiares, quienes han depositado toda su fe en mí y me han brindado todo su apoyo y cariño a lo largo de este camino.

Este logro no es solo mío, sino de todos quienes han caminado a mi lado y han creído en mis sueños. Hoy, más que nunca, siento que cada sacrificio ha valido la pena, y cada esfuerzo ha tenido su recompensa. Este trabajo es el fruto de un sueño compartido, y lleva en cada página un pedazo del amor y apoyo que me han brindado. Gracias por ser mi inspiración y mi fortaleza.

Con cariño, Paul Aucatoma

AGRADECIMIENTO

Quiero expresar mi más profundo y sincero agradecimiento a todas las personas que de diversas maneras, han contribuido a la realización de esta tesis. Su apoyo y colaboración han sido esenciales para alcanzar este logro.

A mis queridos padres, hermanas, esposa e hija. Su aliento constante y su fe en mis capacidades han sido el pilar sobre el cual he construido mis logros.

A mi tutor de tesis, Ing. Edgar Patricio Córdova, cuyas valiosas enseñanzas y guía han sido fundamentales en mi desarrollo académico. Cada lección impartida ha sido un verdadero tesoro que va más allá de las páginas de este trabajo.

A mis amigos, Jonathan, Estefanía, Alexis, Rick, Fer y Kevin por su apoyo moral y por estar siempre presentes, tanto en los momentos de éxito como en los de dificultad. Su amistad ha sido una fuente constante de motivación y fortaleza, y su presencia ha hecho que este viaje sea más llevadero y significativo.

A mi amiga y compañera de tesis, Ivette, por su invaluable apoyo y dedicación. Tu colaboración ha sido crucial para completar este proyecto con éxito. Valoraré siempre nuestra amistad y la experiencia enriquecedora de trabajar juntos hacia esta meta académica.

A los docentes de la FISEI, por compartir su vasto conocimiento, experiencia, tiempo y paciencia. Sus enseñanzas han moldeado mi pensamiento crítico y mi enfoque hacia la investigación. Su dedicación y compromiso con mi formación han sido una fuente constante de inspiración y motivación.

Gracias infinitas, Paul Aucatoma

ÍNDICE GENERAL DE CONTENIDOS

PORTADA.....	i
APROBACIÓN DEL TUTOR.....	ii
AUTORÍA.....	iii
DERECHOS DE AUTOR	iv
APROBACIÓN DEL TRIBUNAL DE GRADO.....	v
DEDICATORIA.....	vi
AGRADECIMIENTO	vii
DEDICATORIA.....	viii
AGRADECIMIENTO	ix
ÍNDICE GENERAL DE CONTENIDOS	x
ÍNDICE DE TABLAS.....	xiii
ÍNDICE DE FIGURAS.....	xvi
ÍNDICE DE ANEXOS.....	xxi
RESUMEN EJECUTIVO	xxii
ABSTRACT	xxiii
CAPÍTULO I. MARCO TEÓRICO.....	1
1.1 Tema de investigación.....	1
1.2 Planteamiento del problema.....	1
1.3 Antecedentes investigativos	3

1.4 Fundamentación teórica	6
1.4.1 Agricultura 4.0	6
1.4.2 Cultivo de frutas	10
1.4.3 Robótica	13
1.4.4 Vehículos Aéreos no Tripulados	15
1.4.5 Inteligencia Artificial	26
1.4.6 Visión Artificial	29
1.4.7 Procesamiento digital de imágenes	29
1.4.8 Adquisición y representación de imágenes digitales	30
1.4.9 Algoritmo de procesamiento de imágenes	32
1.4.10 Modelado de imágenes	34
1.4.11 Visión Artificial aplicada a la Agricultura	35
1.4.12 Control de calidad empleando visión artificial	36
1.4.13 Clasificación de frutos	36
1.4.14 Microcomputador	38
1.5 Objetivos	40
1.5.1 Objetivo general	40
1.5.2 Objetivos específicos	40
CAPÍTULO II. METODOLOGÍA	41
2.1 Materiales	41
2.2 Métodos	42

2.2.1 Modalidad de la investigación	42
2.2.2 Recolección de información.....	43
2.2.3 Procesamiento y análisis de datos	43
CAPÍTULO III. RESULTADOS Y DISCUSIÓN.....	44
3.1 Desarrollo de la propuesta.....	44
3.1.1 Definición de fruta de temporada para la propuesta	44
3.1.2 Atributos para la evaluación de la calidad de manzanas	48
3.1.3 Arquitectura del sistema.....	51
3.1.4 Selección de tecnologías	55
3.1.5 Diagrama del sistema	67
3.1.6 Diseño del sistema.....	67
3.1.7 Implementación del sistema.....	107
3.1.8 Resultados de las pruebas de funcionamiento.....	116
3.1.9 Costo computacional.....	119
3.1.10 Presupuesto	120
CAPÍTULO IV. CONCLUSIONES Y RECOMENDACIONES	123
4.1 Conclusiones	123
4.2 Recomendaciones.....	125
REFERENCIAS BIBLIOGRÁFICAS.....	127
ANEXOS.....	135

ÍNDICE DE TABLAS

Tabla 1. Aspectos que incorpora la agricultura	7
Tabla 2. Tecnologías usadas en la agricultura	9
Tabla 3. Factores intrínsecos y extrínsecos para control de calidad.	12
Tabla 4. Herramientas para evaluación de calidad.....	12
Tabla 5. Prácticas agrícolas y ambientales en la calidad.	13
Tabla 6. Características de controladores de cuadricópteros.	19
Tabla 7. Desventajas de usar un UAV.	21
Tabla 8. Sistemas de UAV en Ecuador.....	23
Tabla 9. Consideraciones de obstáculos.	24
Tabla 10. Consideraciones para la optimización de trayectorias.	24
Tabla 11. Áreas de la Inteligencia Artificial.	27
Tabla 12. Algoritmos de aprendizaje automático.	27
Tabla 13. Etapas del procesamiento de imágenes.....	34
Tabla 14. Proceso de clasificación de frutos.....	37
Tabla 15. Características del hardware seleccionado.....	41
Tabla 16. Característica de software seleccionado	42
Tabla 17: Criterios de recolección de información	43
Tabla 18. Frutas por temporada en el Ecuador.	45
Tabla 19. Variedades de manzana en la sierra ecuatoriana.....	47
Tabla 20. Variedades de manzanas seleccionadas.	48

Tabla 21. Atributos para evaluar la calidad de manzanas.....	49
Tabla 22. Factores para la evaluación de la calidad de manzanas.	49
Tabla 23. Defectos que afectan en la calidad de la manzana.	50
Tabla 24. Grados de calidad para clasificación de manzanas.	51
Tabla 25. Comparación de UAV.....	56
Tabla 26. Comparación de microcomputadores.....	57
Tabla 27. Comparación de lenguajes de programación usados en Inteligencia Artificial.	58
Tabla 28. Comparación de plataformas basadas en la nube.....	60
Tabla 29. Comparación de Redes Neuronales Convolucionales.	61
Tabla 30. Comparación de versiones de YOLO.	62
Tabla 31. Comparación SSD MobileNet vs. YOLOv8	63
Tabla 32. Comparación de sistemas de gestión de base de datos.	65
Tabla 33. Comparación de frameworks para interfaz web.	66
Tabla 34. Bibliotecas requeridas para el sistema	68
Tabla 35. Comandos de control DJI Tello.	74
Tabla 36. Características data set para detección.....	82
Tabla 37. Características data set para control de calidad	82
Tabla 38. Cronograma de creación de dataset	82
Tabla 39. Parámetros de maquina usada en entrenamiento.	86
Tabla 40. Parámetros YOLOv8.....	89

Tabla 41. Resultados de entrenamiento con distintas arquitecturas de YOLOv8.....	92
Tabla 42. Matriz de correlación por modelo entrenado	97
Tabla 43. Matriz de confusión por modelo entrenado	98
Tabla 44. Comandos de instalación para librerías requeridas.....	109
Tabla 45. Resultados de velocidad de preprocesamiento.....	116
Tabla 46. Pruebas de funcionamiento.	117
Tabla 47. Eficacia del sistema.....	118
Tabla 48. Costo computacional.....	120
Tabla 49. Presupuesto del sistema.	121

ÍNDICE DE FIGURAS

Figura 1. Arquitectura general de agricultura 4.0	8
Figura 2. Comparación entre explotación agrícola a pequeña escala y una inteligente	9
Figura 3. Ciclo de gestión de información.	10
Figura 4. Conceptualización de agricultura 4.0 en operación en el sector agrícola... ..	11
Figura 5. Monitoreo de cultivo.....	12
Figura 6. Robot agrícola que deshierba con un haz de luz.	14
Figura 7. Ejemplo de robot agrícola Agerris para control de maleza alimentado por energía solar	15
Figura 8. Dron multirotor perteneciente a la marca Heliceo.....	17
Figura 9. RPA de ala rotatoria.....	17
Figura 10. Modelos de dron de ala fija.	18
Figura 11. Diagrama de un sistema de control en lazo cerrado.	18
Figura 12. Objetivo de control punto a punto.	21
Figura 13. Seguimiento de trayectorias.....	22
Figura 14. Evasión de obstáculos.....	22
Figura 15. AR. Drone 2.0.....	25
Figura 16. DJI Mavic 2 Pro.....	25
Figura 17. DJI Tello Edu.....	26
Figura 18. Diagrama de bloques de las etapas típicas en un sistema de visión artificial.	29

Figura 19. En la fila superior se presenta la misma imagen, siempre a 256 niveles de intensidad, usando diferentes resoluciones espaciales. En la fila inferior se mantiene la resolución espacial y se reduce el nivel de cuantización.	31
Figura 20. Esquema de funcionamiento de un filtro.	32
Figura 21. Algoritmo de procesamiento de imágenes.	33
Figura 22. Raspberry Pi 5.	38
Figura 23. NVIDIA Jetson Nano.	39
Figura 24. AI-Motion K210.	40
Figura 25. Grado de calidad manzana GrannySmith; a) Calidad normal, b) Media calidad, c) Mala calidad.	51
Figura 26. Arquitectura del sistema	52
Figura 27. Arquitectura de adquisición de imágenes	52
Figura 28. Imágenes capturadas por el dron.	53
Figura 29. Arquitectura de la fase de entrenamiento	54
Figura 30. Arquitectura de las etapas de procesamiento y visualización del sistema	55
Figura 31. Diagrama general del proceso de experimentación del sistema	67
Figura 32. Diagrama de bloques del sistema	67
Figura 33. Diagrama de flujo del proceso de adquisición de imágenes	70
Figura 34. Mapa de ubicación del cultivo de manzana.	71
Figura 35. Cultivo donde se realizará la planeación de trayectoria.	72
Figura 36. Cultivo de manzana del sector.	72
Figura 37. Trayectoria realizada por el dron en el cultivo.	73

Figura 38. Comandos registrados en un archivo .txt de la trayectoria.....	75
Figura 39. Diagrama de flujo del procesamiento de imágenes	76
Figura 40. Imágenes para procesamiento.....	77
Figura 41. Proceso de redimensionamiento de imagen.....	77
Figura 42. Proceso de transformación de imágenes.....	78
Figura 43. Etiquetado en el software LabelMe	79
Figura 44. Arquitectura de la base de datos	81
Figura 45. Arquitectura de YOLOv8.	83
Figura 46. Capas del modelo de IA.....	84
Figura 47. Variantes del modelo YOLOv8.....	85
Figura 48. Arquitectura de red neuronal YOLOv8x	86
Figura 49. Resultados de instalación de Ultralytics	87
Figura 50. Espacio de color HSV.....	89
Figura 51. Vista condensada de mapa de colores HSV	89
Figura 52. Ejecución del entrenamiento por épocas modelo 2	91
Figura 53. Ejecución del entrenamiento por épocas modelo 3	93
Figura 54. Métricas de rendimiento del modelo entrenado (Detección de Manzanas)	94
Figura 55. Métricas de rendimiento del modelo entrenado (Tipo de manzanas).....	95
Figura 56. Métricas de rendimiento del modelo entrenado (Clasificación de manzanas por grado de calidad).....	96
Figura 57. Predicción por lotes modelo de detección de manzanas.....	99

Figura 58. Predicción por lotes modelo de clasificación de manzanas.....	99
Figura 59. Predicción por lotes modelo de clasificación de manzanas por grado de calidad.	100
Figura 60. Precisión del modelo 1.....	101
Figura 61. Precisión del modelo 2.....	101
Figura 62. Precisión del modelo 3.....	102
Figura 63. Tabla de autenticación en SQLite.....	103
Figura 64. Proceso de autenticación de usuario.	104
Figura 65. Tabla para almacenamiento de datos de detección y clasificación de manzanas en SQLite.....	105
Figura 66. Interfaz (Autenticación de usuario)	106
Figura 67. Interfaz de usuario para visualización de los datos entregados por los modelos en funcionamiento.	106
Figura 68. Prototipo del sistema.	107
Figura 69. Flasheo de sistema operativo con herramienta Etcher.....	108
Figura 70. Primer inicio Jetson Nano.....	108
Figura 71. OpenCV con CUDA para Jetson Nano.	109
Figura 72. Interfaz web integrada a Jetson Nano.	113
Figura 73. Detección y clasificación (Red Delicious)	113
Figura 74. Detección y clasificación (Granny Smith).....	113
Figura 75. Detección y clasificación (Golden Delicious).	114
Figura 76. Detección y clasificación (Fuji).....	114

Figura 77. Detección y clasificación 2 (Fuji).....	115
--	-----

ÍNDICE DE ANEXOS

Anexo A. Categorización de variables	135
Anexo B. DJI Tello EDU	136
Anexo C. NVIDIA JETSON NANO	137
Anexo D. Manual de usuario DJI Tello EDU	138
Anexo E. Hardware Adquirido.....	139
Anexo F. Visitas al cultivo de manzana.....	140
Anexo G. Pruebas del sistema.....	141
Anexo H. Código del sistema	142

RESUMEN EJECUTIVO

En la actualidad, la agricultura desempeña un papel fundamental en la economía mundial. Con el fin de satisfacer la creciente demanda alimentaria, se han integrado herramientas tecnológicas avanzadas para optimizar las prácticas agrícolas, conocidas como Agricultura de Precisión (PA). Estas herramientas ofrecen soluciones que mitigaran las dificultades a las que se enfrentan los agricultores en sus tareas cotidianas.

En este contexto, se llevó a cabo un estudio con el objetivo de implementar un sistema de control de calidad de cultivo de fruta de temporada para etapa de precosecha empleando robótica aérea con planificación de trayectorias y visión artificial. La importancia del control de calidad de las frutas en Ecuador radica en su alta demanda tanto en el mercado nacional como internacional. La norma técnica NTE INEN 1872 establece criterios de clasificación de la manzana mediante grados de calidad para exportación y consumo. Este sistema se fundamenta en el uso de YOLOv8, una herramienta de aprendizaje profundo que evalúa el grado de calidad y clasifica los diferentes tipos de manzana.

El sistema consta de cuatro etapas: adquisición, procesamiento, entrenamiento y visualización. En la etapa de adquisición, se utilizó el dron Dji Tello para capturar imágenes o videos en tiempo real. Los datos adquiridos tienen un preprocesamiento mediante OpenCV. Se emplea una red neuronal para entrenar un modelo capaz de reconocer de manera precisa el tipo y grado de calidad en manzanas. Para la visualización se diseñó una interfaz gráfica intuitiva que permita la representación visual de los datos derivados del modelo entrenado. El algoritmo del sistema se desarrolló en Python debido a sus múltiples librerías.

Palabras clave: UAV, Agricultura de precisión, inteligencia artificial, Deep learning, YOLO.

ABSTRACT

Currently, agriculture plays a fundamental role in the global economy. To meet the growing food demand, advanced technological tools have been integrated to optimize agricultural practices, known as Precision Agriculture (PA).

These tools offer solutions that will mitigate the difficulties faced by farmers in their daily tasks. In this context, a study was carried out with the aim of implementing a quality control system for seasonal fruit crops for the pre-harvest stage using aerial robotics with trajectory planning and artificial vision. The importance of fruit quality control in Ecuador lies in its high demand both in the national and international markets. Technical standard NTE INEN 1872 establishes criteria for classifying apples according to quality grades for export and consumption. This system is based on the use of YOLOv8, a deep learning tool that evaluates the quality grade and classifies the different types of apples.

The system consists of four stages: acquisition, processing, training, and visualization. In the acquisition stage, the Dji Tello drone was used to capture images or videos in real-time. The acquired data undergo preprocessing using OpenCV. A neural network was employed to train a model capable of accurately recognizing the type and quality grade of apples. For visualization, an intuitive graphical interface was designed to allow visual representation of the data derived from the trained model. The system algorithm was developed in Python due to its multiple libraries.

Keywords: UAV, Precision Agriculture, artificial intelligence, Deep Learning, YOLO.

CAPÍTULO I. MARCO TEÓRICO

1.1 Tema de investigación

SISTEMA DE CONTROL DE CALIDAD DE CULTIVO DE FRUTA DE TEMPORADA EMPLEANDO ROBÓTICA AÉREA CON PLANIFICACIÓN DE TRAYECTORIAS Y VISIÓN ARTIFICIAL

1.2 Planteamiento del problema

A nivel mundial la adopción de tecnologías digitales en la agricultura se caracteriza por su lentitud y desigualdad. Esto se debe a diversos factores, como la variedad de actores involucrados y las barreras específicas para la digitalización. Estas barreras varían según las regiones y los grupos de agricultores, lo que subraya la necesidad de abordarlas de manera específica para impulsar la digitalización en el sector. Pesé a que el sector agrícola ha tenido muchos avances, en la capacidad para detectar, comprender y responder a problemas en la producción de alimentos, todavía existen desafíos en la adquisición y análisis de datos en entornos agrícolas. Estos desafíos pueden limitar la optimización de prácticas agrícolas, como la identificación de productos de calidad, la estimación de rendimiento, la detección temprana de enfermedades en los cultivos y la clasificación de la madurez de la cosecha[1].

La incorporación de nuevas tecnologías agrícolas, como el uso de los drones en la agricultura no solo se limita a tener mapas del suelo o aplicar agroquímicos, también se usan para el monitoreo de cultivos, especialmente en cultivos de áreas medianas y pequeñas. China es el país del mundo con más uso de drones en agricultura y su uso agrícola está en expansión en América Latina. La rápida adopción de los drones en cultivos exige el análisis de la regulación sobre el uso de estos vehículos no tripulados para su uso en el sector agrícola, a pesar de su potencial para promover la sostenibilidad en este sector, su uso se ve amenazado por la falta de marcos regulatorios actualizados y la oposición infundada, lo que puede limitar el acceso de los agricultores a estas tecnologías y afectar la productividad agrícola y la sostenibilidad[2].

En Ecuador del 64% de la producción agrícola nacional está en manos de pequeños productores esto sumado a que el sector agrícola no ha alcanzado un nivel de desarrollo significativo principalmente en la adopción de nuevas tecnologías que pueden facilitar el trabajo del agricultor, puede ser un gran problema para la productividad y el comercio de productos por parte de los agricultores, debido a que estos carecen de los datos y los recursos esenciales para incorporar nuevas herramientas tecnológicas en sus prácticas agrícolas diarias[3]. La adopción de tecnologías avanzadas en la agricultura, como IA para el diagnóstico de plagas o enfermedades y la introducción de drones en la gestión de cultivos tiene ciertas complicaciones tales como costos iniciales elevados, falta de capacitación, necesidad de establecer estándares de calidad, preocupaciones sobre la privacidad de datos y regulaciones agrícolas obsoletas. Superar estos obstáculos es fundamental para garantizar la implementación exitosa de tecnologías que contribuyan a mejorar la calidad de los cultivos y la sostenibilidad de la agricultura.

En la producción de ciertos cultivos de fruta de temporada se puede llegar a cometer ciertos errores al momento de cosecharlas, ya que la inspección visual en la etapa de precosecha puede ser ineficiente y propensa a errores humanos, lo que conduce a una recolección de productos de baja calidad y a la pérdida de frutas que podrían haber sido aptas para la comercialización en el control de calidad del cultivo. Produciendo una disminución de la rentabilidad de la producción, ya que se desperdician recursos valiosos en la recolección y manejo de frutas que no cumplen con los estándares de calidad. Además, la falta de detección visual precisa puede llevar a la recolección prematura o tardía de la fruta, lo que afecta negativamente la calidad y el valor de la producción.

La inspección visual realizada por operadores humanos está sujeta a errores y subjetividad en la evaluación de la calidad, el tamaño, la madurez y la presencia de defectos en la producción. Por lo que puede existir problemas de calidad, como frutas con imperfecciones o signos tempranos de enfermedades que pueden ser pasados por alto. Como resultado, el producto cosechado puede llegar al mercado en esas

condiciones, lo que afecta la calidad general de la producción y puede dar lugar a la insatisfacción de los consumidores y a la reducción de los precios de venta.

La evaluación manual de la producción frutal requiere una inversión significativa en recursos humanos, lo que implica costos laborales y puede ser una tarea intensiva en términos de mano de obra por tanto aumenta los costos operativos y puede ser ineficiente en términos de tiempo. Requiere la contratación y capacitación de personal adicional, lo que representa un gasto significativo para los productores. Además, la dependencia de la mano de obra humana puede llevar a una mayor variabilidad en la evaluación de la calidad de las frutas, lo que afecta la consistencia y confiabilidad del proceso de control de calidad.

1.3 Antecedentes investigativos

A continuación, se presentan los antecedentes investigativos recopilados a partir de la revisión bibliográfica llevada a cabo en diversos repositorios universitarios nacionales e internacionales, así como en bases de datos en línea que contienen artículos científicos. Estos antecedentes ofrecen una base sólida para enriquecer la investigación sobre el tema.

En el año 2019 los autores Jhony Paul Tacuri Auquilla y Wilson Jhonatan Olmedo Carrillo de la Escuela Superior Politécnica de Chimborazo, en su investigación titulada “IMPLEMENTACIÓN DE UN DRON ANTICOLISIONES CAPAZ DE SEGUIR UNA TRAYECTORIA PRE ESTABLECIDA PARA LA ELABORACIÓN DE UN MAPA DE LA TASA DE OCUPACION VEGETAL EN UNA ZONA.”, desarrollaron un sistema de vuelo no tripulado con el propósito de asistir en la evaluación de la ocupación vegetal en áreas específicas. El desarrollo del prototipo constó de tres etapas. En la primera fase, se implementaron los elementos necesarios utilizando el kit de desarrollo Intel Aero y se incorporó un sistema anticollisiones con Arduino y sensor ultrasónico. La segunda etapa involucró la calibración del dron y la captura de imágenes en vuelo mediante QGround Control, utilizando una cámara deportiva. Finalmente, en la tercera fase, se reconstruyeron las imágenes con el módulo Image-Stitching de Matlab, aplicando un algoritmo de segmentación para evaluar la

ocupación vegetal en la imagen resultante. Como resultado de las pruebas realizadas en un terreno de 200 metros cuadrados destinado al cultivo de alfalfa reveló una ocupación vegetal promedio del 65%, equivalente a 139 metros cuadrados de vegetación. A través de imágenes aéreas capturadas por un dron, se identificaron áreas con menor densidad de plantas, lo que permitió recomendar la aplicación adicional de recursos agrícolas en sectores específicos del cultivo[4].

En el año 2020 los autores Bryan Cristhian Martin Cuya Marzal y Martin Guillermo Ramos Lugo de la Universidad de Lima, Perú, presentaron su “SISTEMA DE CONTROL DE CALIDAD UTILIZANDO REDES NEURONALES PARA LA CLASIFICACIÓN DEL ESTADO DE LA GRANADILLA”, en el cual se desarrolló un prototipo que utiliza tecnología avanzada, específicamente Internet de las Cosas (IoT) y redes neuronales convolucionales, para mejorar la calidad y eficiencia en la producción agrícola de granadillas, utilizó tecnología disruptiva, interconexión y comunicación entre un ordenador y una cámara como dispositivo de captura de imágenes basándose en la interconexión de una cámara web que pudiera enviar las capturas de imágenes de manera inalámbrica a un ordenador utilizando la librería OpenCV y su función de VideoCapture. Además, se hizo uso de machine learning realizando distintos modelos de redes neuronales convolucionales utilizando la librería PyTorch. De esta manera, se obtuvo una lección o aprendizaje para la elaboración de una red final donde se utiliza 4 capas convolucionales, 4 pooling y 2 fully-connected. El proyecto fue desarrollado mediante el uso de cuatro redes diseñadas que permite el aprendizaje y mejoramiento de clasificación de frutas, ajustando parámetros y arquitecturas para obtener una mayor precisión y capacidad de generalización[5].

En el año 2021 el autor Juan Gabriel Castro Vega de la Universidad Militar Nueva Granada, en su investigación titulada “ALGORITMO CON IA PARA PLANEACIÓN DE TRAYECTORIAS DE UN QUADROTOR PARA DETECCIÓN DE PLAGAS Y ESTADO NUTRICIONAL DE CULTIVOS DE ANNONA MURICATA” proponen desarrollar sistemas de optimización de trayectorias, sistemas de supervisión y procesamiento de imágenes para un cultivo de Annona Muricata con el fin de identificar problemas tanto nutricionales como de plagas, haciendo uso de algoritmos con IA el desarrollo de un algoritmo con IA para la planeación de trayectorias del

quadrotor, con el fin de minimizar la distancia que este debía recorrer en el cultivo por su limitada capacidad y duración de batería. Del cual para determinar el estado nutricional del cultivo se clasificaron las imágenes de los frutos en 2 grupos, “frutos en buenas condiciones” y “frutos en malas condiciones”, para ser clasificados en alguna de estas categorías se tuvo en cuenta si estos mostraban signos de la plaga “avispa” o de la enfermedad de la Antracnosis, se realiza un algoritmo basado en el método SIFT, que permite obtener las características principales de la imagen, además, poder clasificar estas imágenes según la información extraída. Dando como resultado en el estudio de algoritmo genético y el segundo el algoritmo A*, el más apropiado es A* ya que es aproximadamente 30% mejor a comparación del algoritmo genético[6].

En el año 2022 el autor Andrés Alejandro Garcés de la Universidad Técnica de Ambato, en su investigación titulada “CLASIFICACIÓN DE CALIDAD DE MANZANA PARA MONITOREO DE COSECHABILIDAD UTILIZANDO VISIÓN POR COMPUTADOR Y ALGORITMOS DE APRENDIZAJE PROFUNDO” proponen el desarrollo de un sistema portable de detección de calidad para la identificación del estado cosechable de la manzana haciendo uso de un sistema de visión por computador y aprendizaje de máquina. Basándose en método de detección del tipo de variedad de manzana que permita el conteo de fruta y otro método de segmentación que permita medir el nivel de calidad para su aceptabilidad de consumo. Se emplea un modelo de red SSD-MobileNet siendo modificado para alcanzar un mejor rendimiento en imágenes de entrada. Además, para el segundo método se usa una red convolucional rápida FCN-ResNet 18 con el objetivo de ser implementada sobre hardware de limitadas capacidades. Como resultado de las pruebas, se evidencia que el sistema es funcional para verificar la calidad de manzanas, pero sugiere el uso en sistemas en aplicaciones reales donde la identificación de la calidad de fruto sea usualmente manual [7].

En el año 2022 la autora Ana Victoria Canterla Martín de la Universidad de Sevilla, en su investigación titulada “PLANIFICACIÓN EFICIENTE DE TRAYECTORIAS LIBRES DE COLISIONES PARA UN EQUIPO DE DRONES. COMPARACIÓN DE ALGORITMOS” muestra un análisis comparativo entre dos métodos basados en evitación de obstáculos para vehículos aéreos no tripulados para lo cual, se han

diseñado cuatro escenarios para comparar las técnicas CA-nk y ORCA, junto con 100 escenarios aleatorios para una prueba ampliada. El lenguaje de programación utilizado para distintas pruebas es Python, en un escenario bidimensional el cual posee como parámetros de entrada la posición inicial del dron (bidimensional), la velocidad preferida, el radio del dron, su dirección inicial, el punto objetivo (bidimensional), la distancia mínima del objetivo a la que se considera que el dron ha llegado a su destino. Como resultado de la comparación realizada en este trabajo, se concluyó decir que CA-nk da buenos resultados cuando el objetivo es realizar planificación de trayectorias de manera eficiente con respecto a la energía requerida, el tiempo en ejecutar la tarea y el tiempo de computación necesario para dar las soluciones. Todo esto, asumiendo un número reducido de drones. Por el contrario, ORCA sería una buena elección si el objetivo es evitar colisiones por encima de la eficiencia de la tarea y de la cantidad de drones que coexistan en el escenario[8].

1.4 Fundamentación teórica

1.4.1 Agricultura 4.0

a. ¿Qué es la agricultura 4.0?

La agricultura 4.0 hace referencia a la implementación de nuevas tecnologías agrícolas avanzadas que están diseñadas para ofrecer soluciones innovadoras a los procesos actuales en la industria agrícola. Una de las razones importantes para su adopción es la calidad de los datos que estas tecnologías pueden generar. Estos datos facilitan la toma de decisiones para los productores y otros trabajadores, mejorando la eficiencia en la realización de tareas tradicionales de producción de alimentos, las cuales históricamente demandan una gran cantidad de trabajo manual y visual[9] .

La nueva revolución agrícola abarca diversos aspectos a nivel global e incorpora diversas perspectivas sobre el concepto de agricultura 4.0 que se detallan en la Tabla 1:

Tabla 1. Aspectos que incorpora la agricultura 4.0[10].

Concepto	Descripción
Agricultura de precisión	Empleo de tecnologías avanzadas para optimizar el uso de recursos y maximizar el rendimiento en cultivos.
Tecnología digital	Uso de herramientas digitales como sensores, drones y sistemas de información geográfica para mejorar la eficiencia y precisión de operaciones agrícolas.
Rentabilidad y sostenibilidad	Foco en aumentar la rentabilidad de operaciones agrícolas asegurando la sostenibilidad ambiental y económica a largo plazo.
Toma de decisiones informada	Uso de los datos en tiempo real y análisis predictivo para toma de decisiones estratégicas en gestión agrícola.
Industria 4.0	Integración de tecnologías inteligentes, análisis de datos en procesos agrícolas y automatización.

El objetivo de la agricultura 4.0 es proporcionar un apoyo más integral y preciso a los agricultores en su proceso de toma de decisiones, mejorando la relación con otros componentes de la cadena de suministro. Además, busca promover la sostenibilidad ambiental, social y económica, así como la rentabilidad de prácticas agrícolas [9].

b. Principios de la agricultura 4.0

Las prácticas agrícolas han evolucionado significativamente con la implementación de principios clave e innovadores en este sector detallados a continuación[11]:

- Digitalización e integración de datos
- Automatización y Robótica
- Inteligencia Artificial y Aprendizaje Autónomo
- Conectividad y redes
- Sostenibilidad y gestión de recursos

Cada uno de estos contribuye a la conectividad del sistema agrícola y a su vez a la eficiencia del sistema siendo así capaz de adaptarse a los desafíos actuales, optimizando recursos y mejorando decisiones a través de datos en tiempo real y análisis predictivos[11].

c. *Arquitectura de la agricultura 4.0*

Se integra de varios elementos y tecnologías interconectadas para optimizar las operaciones del sector agrícola en la Figura 1, se proporciona una visión general de como los diferentes componentes trabajan conjuntamente para formar un sistema agrícola inteligente aprovechando las tecnologías modernas para mejorar la productividad, sostenibilidad y gestión en el sector agrícola[12].



Figura 1. Arquitectura general de agricultura 4.0[12]

d. *Tecnologías utilizadas en la agricultura 4.0*

La agricultura 4.0 representa una revolución en el sector agrícola mediante la integración de tecnologías avanzadas. Este enfoque está basado en el uso de herramientas digitales y sistemas automatizados para optimizar la producción, mejorar la eficiencia y reducir el impacto ambiental en el sector agrícola. Entre las tecnologías clave que impulsan la agricultura 4.0 se detallan en la Tabla 2[9].

En la Figura 2, se detalla como la agricultura 4.0 mejora lo que hoy en día se conoce como agricultura convencional al implementar las tecnologías antes mencionadas[9].

Las mejoras técnicas de las nuevas tecnologías deberían[9]:

- Optimizar la eficiencia

- Optimizar la calidad
- Minimizar el impacto ambiental
- Minimizar los riesgos relacionados con la producción

Tabla 2. Tecnologías usadas en la agricultura 4.0[9].

Tecnología	Descripción
Inteligencia Artificial	Aplicación de algoritmos para mejora de toma de decisiones facilitando la gestión optimizada de cultivos y respuesta a condiciones cambiantes del entorno.
Teledetección	Es la ciencia que permite la obtención de información sobre objetos o zonas que se encuentran a distancias considerables, normalmente desde aeronaves o satélites.
Tecnología de cadena de bloques	Sistema en el que una lista creciente de registros, conocido como bloques, se vinculan mediante criptografía.
Internet de las cosas (IoT)	Infraestructura global de red mundial donde objetos físicos y virtuales con identidades únicas se descubren e integran de manera fluida en la red de información asociada.
Tecnología de la información y la comunicación (TIC)	Diversos tipos de tecnologías que comunican información a los usuarios mediante telecomunicaciones.
LIDAR o sistema lidar	Combina diferentes sensores de varias frecuencias para medir distancias y después pueden ser usadas para crear imágenes 3D.
Big Data y Análisis de datos	Recolecta grandes volúmenes de datos para identificar patrones y tendencias.



Figura 2. Comparación entre explotación agrícola a pequeña escala y una inteligente[9].

Las industrias agrícolas y agricultores juegan un gran papel para optar por estas tecnologías ya que no solo requiere de un enfoque agronómico sino también de

infraestructura y conocimientos especializados, implicando que el agricultor se convierta en un gestor de tecnología de información trabajando más desde una oficina o dispositivo digital en lugar de ser un operador manual en el campo agrícola[9].

En la Figura 3, se muestra cómo se realiza la gestión de información para que el agricultor tome decisiones en base a los datos recopilados de los cultivos[9].

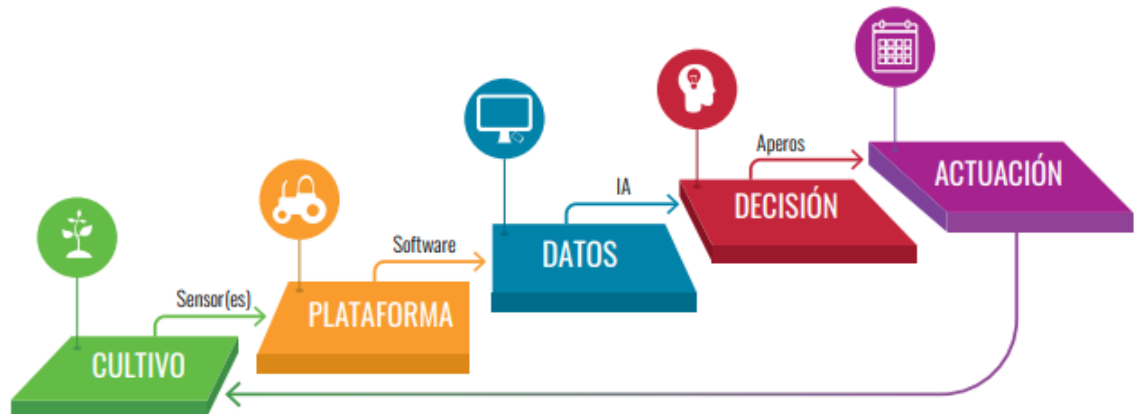


Figura 3. Ciclo de gestión de información [9].

La Agricultura 4.0 además de las tecnologías mencionadas, aprovecha la tecnología aérea como drones y plataformas de detección en tiempo real permitiendo así al agricultor monitorear el estado del cultivo, salud del suelo, presencia de plagas y otros parámetros que son relevantes al momento de iniciar con el proceso de cultivo de distintos productos[9].

Este enfoque implica integrar las tecnologías tradicionales y autónomas adaptándose a las necesidades del agricultor y de cada cultivo, en la se ilustra la interacción de estas tecnologías en operaciones del sector agrícola[9].

1.4.2 Cultivo de frutas

Cultivar especies frutales ha sido desde su inicio una actividad ligada a la economía de un país siendo el fator de calidad un parámetro prioritario para el análisis de la producción total para su exportación. Factores como el tamaño, madurez, sabor, aroma, acidez, etc., conforman parámetros de suma importancia al momento de analizar el nivel de calidad de los frutos[13].

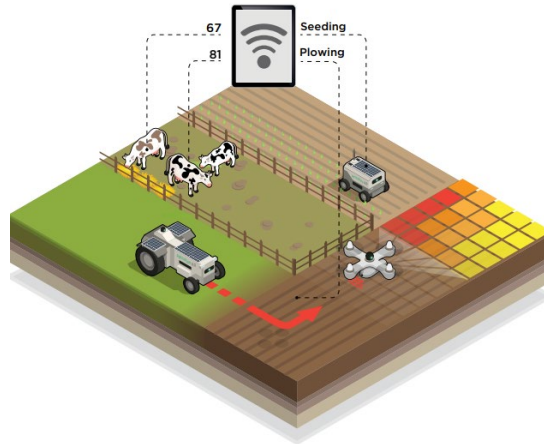


Figura 4. Conceptualización de agricultura 4.0 en operación en el sector agrícola[9].

a. Control de calidad

La percepción de la calidad está estrechamente vinculada con cumplir minuciosamente las especificaciones definidas por las leyes pertinentes, un producto es considerado de alta calidad siempre y cuando este cumpla con las normativas vigentes y cubra satisfactoriamente los requisitos impuestos por el consumidor[14].

El proceso de control de calidad de frutas y verduras implica el análisis de sus propiedades sensoriales y composición fisicoquímica, con el objetivo de proporcionar ventajas tanto para los productos alimenticios como para los consumidores[15].

b. Factores internos y externos que contribuyen al control de calidad

Para evaluar la calidad de las frutas, es crucial analizar una serie de factores tanto internos como externos que influyen en su aspecto, sabor, textura y composición nutricional que se detallan en Tabla 3[16].

c. Técnicas y herramientas para la evaluación de la calidad en etapa de precosecha

El control de calidad de frutas requiere el uso de una variedad de técnicas y herramientas especializadas como el monitoreo del cultivo, sensorización de nutrientes de suelo o niveles de agua en la plantación, que permiten analizar diferentes aspectos de manera detallada y eficiente.

Tabla 3. Factores intrínsecos y extrínsecos para el control de calidad[16].

Factor	Descripción
Internos	
Azúcares	Contenido de azúcares que son solubles y que afectan el sabor
Ácidos orgánicos	Niveles de ácido en los que se encuentran pueden ser ácido cítrico y málico que influyen en el sabor y conservación
Compuestos fenólicos	Afectan al sabor, color y propiedades antioxidantes.
Externos	
Color	Indicador visual de madurez y calidad, influenciado por pigmentos como carotenoides y antocianinas.
Tamaño y Forma	Caracterizado por que afecta la aceptación del consumidor y clasificación comercial.
Textura	Indicador de firmeza y crocancia que afectan la percepción del consumidor, medida mediante técnicas de penetración y compresión.

A continuación, en la Tabla 4 se detallan las herramientas fundamentales para garantizar la calidad y seguridad alimentaria en producción y comercialización de productos agrícolas[17].

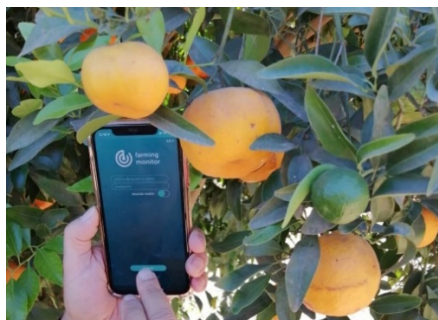


Figura 5. Monitoreo de cultivo

Tabla 4. Herramientas para evaluación de calidad[18].

Técnica/Herramienta	Descripción
Monitoreo del crecimiento	Se utilizan sensores y dispositivos para medir parámetros como tamaño, peso y desarrollo de la fruta.
Análisis Foliar	Analiza las hojas de las plantas para determinar la disponibilidad de nutrientes y salud de la planta.
Texturómetro	Mide la firmeza y textura de la fruta a través de pruebas de penetración y compresión.
Colorimetría	Mide índice de refracción para determinar el contenido de sólidos solubles.
Evaluación visual	Observación directa de apariencia de la fruta para detección de enfermedades daños o madurez
Tacto	Evaluación de la firmeza y textura de la fruta mediante palpación manual.

d. Influencia de prácticas agrícolas y ambientales en la calidad

Es importante conocer la influencia de los parámetros establecidos que son presentados en la Tabla 5 ya que de ello depende el crecimiento adecuado de la fruta y el impacto ambiental de la misma en etapa de precosecha.

Tabla 5. Prácticas agrícolas y ambientales en la calidad[19].

Práctica	Influencia en la calidad
Riego	La frecuencia de riego y cantidad afecta el tamaño, firmeza y contenido de azúcares de las frutas.
Fertilización	Los niveles de nutrientes en el suelo influyen en el crecimiento, sabor y contenido nutricional de las frutas.
Manejo de plagas y enfermedades	El control de plagas y enfermedades es crucial para mantener la calidad y apariencia de la fruta.
Condiciones climáticas	Factores como la humedad, luz solar y temperatura afectan el desarrollo, madurez y calidad final de las frutas.
Práctica de poda	La poda adecuada en el tiempo establecido puede mejorar la calidad de frutas y optimizar la exposición al sol y ventilación.

1.4.3 Robótica

La robótica es una rama de la ingeniería y la ciencia que se ocupa del diseño, construcción, operación y uso de robots. Un robot es una máquina programable que puede realizar tareas específicas de manera autónoma o semiautónoma. Estas tareas pueden incluir movimientos físicos, interacción con el entorno, procesamiento de información y toma de decisiones. Algunas características clave de la robótica incluyen [9]:

- Programación y Control
- Sensores y Percepción
- Movilidad
- Autonomía
- Interacción Hombre-Máquina
- Desarrollo de Inteligencia Artificial

a. Robótica Agrícola

No hay una definición clara de lo que es un robot agrícola, pero se lo asocia a un dispositivo móvil autónomo capaz de tomar decisiones en las tareas agrícolas bajo supervisión de un humano, pero sin intervenir en su labor. Estos robots pueden realizar diversas tareas como preparación del suelo, trasplante, siembra, deshierbe, control de plagas y cosecha. Según la FAO en la se ilustra la tarea que desempeña un robot agrícola o agrobot que no es más que una adaptación del concepto de máquinas autónomas[9].

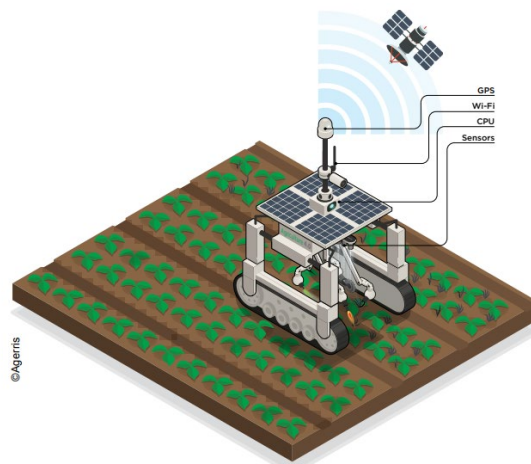


Figura 6. Robot agrícola que deshierba con un haz de luz[9].

• *Uso de robots agrícolas*

Pueden realizar varias tareas que ayuda a minimizar costos de operación y disminuir pérdidas de rendimiento debido a la detección tardía de cualquier plaga o enfermedad del cultivo, algunas tareas que realizan estos dispositivos son[9]:

- Eliminación de malezas
- Vigilancia de plagas
- Vigilancia de enfermedades
- Cosecha de cultivos especializados



Figura 7. Ejemplo de robot agrícola Agerris para control de maleza alimentado por energía solar[9].

1.4.4 Vehículos Aéreos no Tripulados

Es considerado un vehículo aéreo no tripulado por sus siglas en inglés (UAV Unmanned Aerial Vehicle), pero conocido en español como VANT, es autónomo el cual es capaz de volar sin la necesidad de un piloto. Se denominó UAV por los militares de los EE. UU. Estas aeronaves poseen sistemas que combinan información que recopilan de los sistemas de posicionamiento como GPS, navegación mediante GIS, servomecanismo, etc[20].

La CPU que lleva a bordo se encarga de pilotar sin la necesidad de que un ser humano esté a bordo de la nave. A pesar de ser desarrollado inicialmente para uso militar, actualmente los VANT tienen aplicaciones en el campo civil, ya que puede usarse para dar seguimiento a un incendio, o para catástrofes de cualquier tipo, en la agronomía, en la seguridad y otros usos [20]

Gracias a los avances en telecomunicaciones, los VANT se muestran particularmente útiles en cuanto a obtener información, en el manejo y transmisión de esta, y así conseguir comunicaciones que sean difíciles de interferir y más seguras[20].

Los VANT se pueden clasificar por la misión en los siguientes tipos[20]:

- De blanco: simulan aviones o ataques enemigos en los sistemas de defensa de tierra o aire.

- Reconocimiento: Enviando información militar.
- Combate: Llevado a misiones que suelen ser peligrosas.
- Logísticas: Diseñados para llevar carga.
- Investigación y desarrollo: En ellos se prueban e investigan los sistemas en desarrollo.
- Comerciales y civiles: Son diseñados para propósitos civiles

a. Clasificación de los vehículos aéreos no tripulados

El uso de los drones permite la planificación y elaboración de estrategias previo un análisis de la salud de los cultivos. Actualmente existen dos tipos de drones aplicados a la agricultura de precisión los cuales pueden ser equipados con cámaras especiales[20].

Drones de ala rotatoria o multirrotores: Presentan una menor autonomía entorno a los 30 minutos de vuelo y distancias por debajo de los 10 km. Estos drones presentan entre 4 y 8 hélices con longitudes de ala que superan el metro. Pueden soportar una carga de peso de más de dos kilos y a diferencia de los drones de ala fija, pueden girar sobre sí mismos en distancias muy cortas, lo que los hace especialmente versátiles para trabajos verticales. En función del estudio a realizar, topográfico, geofísico o geológico, cada dron puede aportar diferentes recursos y capacidades que deben ser planificadas con anterioridad a los vuelos para asegurar los mejores resultados[20].

• *Modelo de un dron multirroto*

Son aquellas que se sostiene en el aire gracias a las hélices como se puede observar en la Figura 8, a estas se las llama multirrotores o helicópteros, sus características más importantes son[20]:

- Despegue y aterrizaje vertical
- Posibilidad de volar a puntos fijos o a muy baja velocidad.

- Mayor maniobrabilidad y precisión de vuelo.
- Permiten manejar mayores cargas útiles.



Figura 8. Dron multirotor perteneciente a la marca Heliceo[20].

- ***Modelo de un RPA de ala rotatoria***

Son usados en implementación de un sistema de comunicación inalámbrico entre un UAV/RPA con su estación terrena para la transmisión de imágenes térmicas en tiempo real y su utilización en el control de incendios[20].



Figura 9. RPA de ala rotatoria[20].

- ***Drones de ala fija***

El modelo de este dron es semejante al modelo de un avión. Se caracterizan por alcanzar grandes distancias de vuelo con autonomías de hasta 60 km y una hora de vuelo. Estos modelos pueden superar los 500 metros de altura y velocidades comprendidas entre 50 y 70 km/h. Debido a las características del aparato, los despegues se realizan desde una plataforma metálica inclinada un cierto ángulo, mientras que los aterrizajes son por impacto directo contra el suelo. A diferencia de los drones de ala rotatoria, presentan la cámara en posición fija, pueden incorporar

hasta 2.5 kg de peso como carga extra de material, sin incluir la estructura básica que permita volar al dron, esto es, baterías y motores. Estas características los hacen especialmente versátiles para la elaboración de cartografías y toma de orto imágenes de superficies extensas[20].



Figura 10. Modelos de dron de ala fija[20].

b. Algoritmos de control para un UAV

Un algoritmo de control es un conjunto de instrucciones o reglas lógicas diseñadas para dirigir el comportamiento de un sistema dinámico hacia un objetivo específico. Estos algoritmos se utilizan comúnmente en el campo de la ingeniería de control, que se ocupa de regular el comportamiento de sistemas dinámicos para que sigan ciertos patrones deseados. Un sistema dinámico puede ser cualquier entidad que evoluciona con el tiempo, y el objetivo del algoritmo de control es influir en su comportamiento para lograr ciertos resultados. Los algoritmos de control se aplican en una variedad de áreas, desde sistemas mecánicos y eléctricos hasta sistemas biológicos y de software[21].

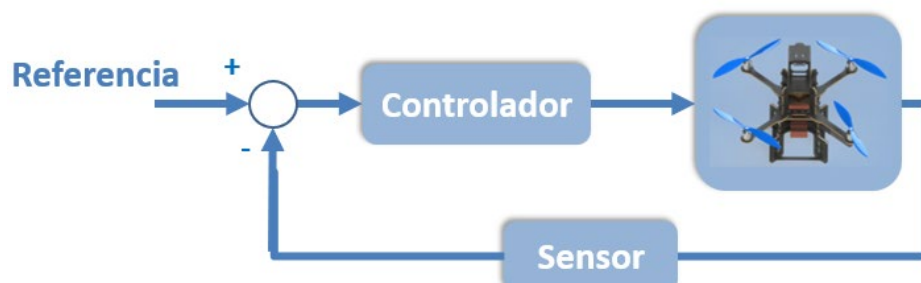


Figura 11. Diagrama de un sistema de control en lazo cerrado[21].

Se clasifican en dos tipos:

- ***Controladores Lineales***

Un sistema de control se considera lineal si sigue el principio de superposición y homogeneidad. Esto significa que si la respuesta del sistema a una entrada es $y_1(t)$ y a otra entrada es $y_2(t)$, entonces la respuesta a la suma ponderada de esas dos entradas (por ejemplo, $ay_1(t) + by_2(t)$) será igual a la suma ponderada de las respuestas individuales ($ay_1(t) + by_2(t)$). Estos controladores son simples y fáciles de programar e implementar, pero su desempeño se ve afectado debido a perturbaciones[21].

- ***Controladores no lineales***

Un sistema de control se considera no lineal si no cumple con el principio de superposición y homogeneidad. En otras palabras, la relación entre las entradas y las salidas no es proporcional y no puede ser representada por ecuaciones lineales. Un sistema de control no lineal presenta un mejor desempeño aun con perturbaciones, pero son más complejos de programar e implementar que los lineales[21].

En la Tabla 6 se muestran las características principales de los controladores mencionados anteriormente.

Tabla 6. Características de controladores de cuadricópteros[21].

CONTROLADORES LINEALES	CONTROLADORES NO LINEALES
Control PD, PI, PID	PD robusto
Regulador Cuadrático lineal (RCL)	Control Adaptable
Realimentación de estados completa	Control Backstepping combinado con control H_∞
Desigualdades matriciales lineales	Control por Modos deslizantes (CMD)

c. Ventajas y Desventajas del uso de UAV

Entre las varias ventajas del uso de UAV, podemos mencionar que son una alternativa mucho más económica y ecológica al uso de los helicópteros y aviones en determinadas tareas. Al no llevar tripulación permite llevar a cabo misiones peligrosas sin poner en riesgo al piloto[20].

El reducido tamaño y el peso de algunos UAV, puede volar sobre personas o propiedades sin crear molestias o ponerlos en riesgos, por lo que lo vuelve ideal para tareas policiales y de observación de la naturaleza. Todos los días se descubren nuevas aplicaciones para los UAV, se trata de una nueva tecnología que está empezando a encontrar usos prácticos[20].

El pilotaje es parecido al de una aeronave convencional, el piloto se encuentra en tierra con la misma instrumentación y de los mismos sistemas de navegación empleados en estas aeronaves por medio de un control remoto[20].

El sistema control remoto tiene un rango de unos cuantos kilómetros alrededor del control dependiendo del tipo de aeronave que se diseñó, no debe existir ningún obstáculo que se interponga con el centro de control y el UAV, para aquello debe tomar la orografía antes de programar los vuelos[20].

La función de esta tecnología se basa en que cada área del campo puede gestionarse según sus características, donde las tareas agrícolas se toman conociendo la localización exacta de las zonas para identificar variables como vigor del cultivo, infestaciones de malas hierbas, aumentar determinada cantidad de fertilizante, o en las que se requiere implementar más o menos riego[20].

Se puede considerar que la gestión localizada de un cultivo está definida por un ciclo de cuatro fases[20]:

- Monitorización para la detección y mapeo de las variables que estén afectando el cultivo[20].
- Toma de decisiones y elaboración del mapa de tratamiento para planificación de qué aplicar, cómo, cuándo y dónde para mejorar el rendimiento de nuestro cultivo[20].
- Actuación en campo o ejecución del manejo localizado que se ha decidido[20].
- Evaluación de la rentabilidad económica y medioambiental de las operaciones realizadas en el cultivo, el cual permitirá conocer las mejoras que se realizan en cada planificación[20].

- **Desventajas**

Las principales desventajas al momento de usar un UAV se detallan a continuación en Tabla 7 [20].

Tabla 7. Desventajas de usar un UAV[20].

Desventaja	Descripción
Inversión Inicial:	La adquisición de drones y equipos relacionados puede representar una inversión significativa
Costos de Mantenimiento	Los gastos para el mantenimiento, reparación y actualización de los drones también deben considerarse.
Capacitación y Experiencia Requerida	El uso efectivo de drones en la agricultura requiere que el personal esté capacitado adecuadamente y a la vanguardia de nuevas tecnologías.
Limitaciones de Carga Útil	La capacidad de carga útil de los drones puede ser limitada lo que restringe la cantidad de sensores o equipos que se puedan transportar simultáneamente.
Capacidad de Carga	La capacidad de carga útil de los drones puede ser limitada, lo que restringe la cantidad de sensores o equipos que se pueden transportar simultáneamente.
Autonomía y Tiempo de Vuelo	La autonomía de vuelo de los drones con batería limita el tiempo de operación antes de que sea necesario recargar o cambiar la batería.
Duración de la Batería	La autonomía de vuelo de los drones está limitada por la capacidad de sus baterías, medida en miliamperios-hora (mh) o vatios-hora (Wh)
Legislación y Regulaciones	El uso de drones está sujeto a diversas leyes y regulaciones que varían según cada región. En Ecuador, la regulación de uso de drones está a cargo de la Dirección General de Aviación Civil (DGAC).
Integración con Sistemas Existentes	La compatibilidad puede requerir inversiones adicionales en software y hardware para asegurar que los datos recolectados por los drones puedan ser usados eficazmente con los sistemas de gestión agrícola ya en uso.
Compatibilidad	Integrar tecnologías de drones con los sistemas agrícolas existentes puede ser un desafío y puede requerir inversiones adicionales.

d. Tipos de objetivos de control

- **Control punto a punto**

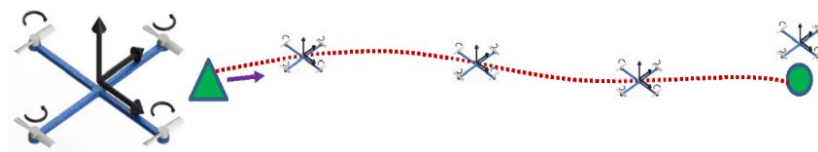


Figura 12. Objetivo de control punto a punto[21].

- ***Seguimiento de trayectorias***

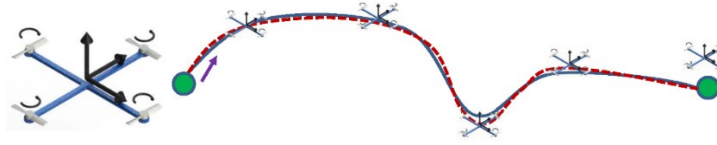


Figura 13. Seguimiento de trayectorias[21].

- ***Evasión de obstáculos***

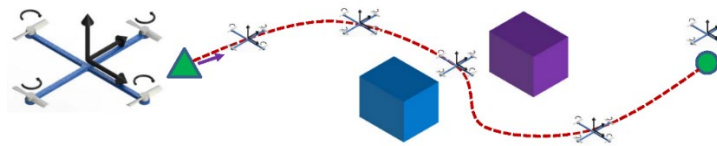


Figura 14. Evasión de obstáculos[21].

e. Planificación de trayectorias

Es el proceso de determinar una serie de puntos y rutas que el dron seguirá para cumplir con ciertos objetivos, como llegar a un destino específico, evitar obstáculos o realizar una tarea particular. Este proceso implica la generación de una trayectoria suave y eficiente que tenga en cuenta las restricciones y condiciones del entorno en el que opera el dron[21].

- ***Ventajas de la etapa de planeación de trayectorias***

Entre las principales ventajas de la etapa de planeación de trayectorias se tiene[21]:

- Aumento del nivel de autonomía de la aeronave.
- Habilidad de evitar obstáculos
- Rastreo de objetivos
- Capacidad de movimiento con seguridad de un punto a otro en el espacio de configuración.

f. Sistemas de UAV en Ecuador

En el Ecuador los sistemas de UAV han encontrado diversas aplicaciones, tanto en el ámbito de seguridad nacional como en el desarrollo tecnológico, agrícola y usos

comerciales A continuación en la Tabla 8 , se presentan dos ejemplos de UAVs utilizados en el país, detallando su procedencia, características y usos específicos. Estos sistemas representan un esfuerzo continuo en la integración de tecnología avanzada para la mejorar la seguridad y eficiencia en distintos ámbitos de operación dentro del territorio ecuatoriano[20].

Tabla 8. Sistemas de UAV en Ecuador[20].

Nombre	Procedencia-Fabricación	Características	Operación / Uso
Heron y Searcher	Procedencia Israelí	Sistema de mando remoto operado por 5 personas, Manta-Ecuador, 2009, Aeronave Radio controlada	Vigilancia y reconocimiento
Gavilán-2	Fabricado y diseñado por el Centro de CID-FAE. Modelos: UAV-0 Gavilán (prueba y demostración), Fénix y UAV-2 Gavilán.	Desarrollo Universitario: ESPOL, EPN, Universidad de las Fuerzas Armadas ESPE. En Ecuador, diferentes instituciones utilizan drones de uso comercial (Bomberos Quito, Gama TV, Policía Nacional).	Actualmente operativo para vigilancia de la frontera Norte

g. Diseño de trayectorias

El diseño de este sistema de control UAV utiliza el control ardupilot con aplicación Mission Planner, con el objetivo de controlar y diseñar la trayectoria de vuelo en el avión para que la aeronave pueda volar en modo piloto automático y que se puede activar manualmente desde el mando a distancia[22].

- ***Generación de Trayectorias***

Para generar trayectorias se debe tomar en cuenta dos parámetros[22]:

- **Puntos de Referencia:** Selecciona puntos clave en el espacio que el dron debe alcanzar, como puntos de destino o lugares para realizar tareas específicas.
- **Interpolación:** Crea una ruta suave y continua entre estos puntos.

- ***Consideraciones de Obstáculos***

En la Tabla 9 se describen las consideraciones que se debe tomar con respecto a los obstáculos[22].

- **Optimización de Trayectorias**

Los aspectos por considerar para la optimización de trayectorias se muestran en la Tabla 10.

Tabla 9. Consideraciones de obstáculos[22].

Consideración	Descripción
Detección de Obstáculos	Utiliza sensores (cámaras, lidar, etc.) para detectar obstáculos en la trayectoria y ajusta dinámicamente la ruta para evitar colisiones.
Algoritmos de Evitación de Obstáculos	Implementa algoritmos que permitan al dron esquivar obstáculos de manera eficiente, como RRT (Rapidly Exploring Random Tree) o algoritmos de campo de potencial, técnicas de interpolación, como splines, para evitar movimientos bruscos.

Tabla 10. Consideraciones para la optimización de trayectorias[22].

Consideraciones	Descripción
Consideraciones de Eficiencia	Minimizan el tiempo de vuelo, el consumo de energía o cumplir con otros objetivos específicos de la misión.
Optimización Numérica	Se utilizan métodos numéricos para optimizar la trayectoria en función de los objetivos y restricciones establecidos.
Adaptabilidad a Condiciones Cambiantes	Sistemas de Navegación y Localización: Incorpora sistemas que permitan al dron adaptarse a cambios en el entorno o en su posición, como GPS, IMU (Unidad de Medición Inercial), y otros sensores.

h. Sistemas UAV usados en agricultura

A continuación, se describe los UAV que se compararan posteriormente para la selección del dispositivo a usar en el desarrollo del proyecto.

- **AR Drone 2.0**

El AR-Drone 2.0 cuenta con cuatro rotores conectados a una estructura en forma de cruz. Esta estructura sostiene la batería y el hardware de RF en sus extremos. Funciona con motores sin escobillas con tres fases de corriente cuyo control lo hace un microcontrolador, utiliza una carga de 1000 mAh y baterías de 11.1 V LiPo. Además, cuenta con sensores de inclinación y presión, usa una cámara de frontal de HD 720p-30fps. Tiene un puerto USB maestro con conector USB-A estándar para la grabación de video mediante cables USB, puede ser controlado desde cualquier dispositivo cliente que admita WiFi. Ampliamente utilizado para la evaluación de algoritmos de

control, tecnologías de detección y aplicaciones relacionadas a la robótica e Inteligencia Artificial (IA)[23].



Figura 15. AR. Drone 2.0 [23].

- ***DJI MAVIC 2***

Mavic 2 Pro es un dron topográfico que cuenta con un tiempo de vuelo de 29 minutos a una distancia máxima de vuelo de 18km, la capacidad de almacenamiento interno de este cuadricóptero es de 8GB y lleva integrado una cámara fotogramétrica con el fin de obtener fotografías aéreas en HD. Además, integra un sistema de detección de obstáculos con resolución de video de 720p a 30fps y 1080p a 30fps, admite bandas de frecuencia de 2,4 GHz y 5,8 GHz, presenta una mayor resistencia a las interferencias y capacidades de conmutación automática para ofrecer transmisiones de video de 1080p de 10 km. No se admite control mediante WiFi solo puede ser controlado mediante control remoto [24].



Figura 16. DJI Mavic 2 Pro [24].

- ***DJI Tello***

El dron Tello de DJI es un quadcopter con un alcance máximo de vuelo de 100 metros y una velocidad máxima de 8 m/s. Se conecta a través de wifi (IEEE 802.11n) en la banda de frecuencia de 2.4GHz. Su cámara captura fotos de 5 megapíxeles y graba

video en una resolución máxima de 1280x720p a 30fps en formato MP4, con un ángulo de visión de 82. 6°. Dispone de funciones automáticas de despegue y aterrizaje[25].

En cuanto a la tecnología de su batería, utiliza polímero de ion-litio, con una capacidad de 1100mAh/4.1Wh y 3.8V, permitiendo una autonomía de vuelo de 13 minutos. Sus dimensiones son 92mm de ancho, 98mm de profundidad y 41mm de altura, con rotores de 7,62cm de diámetro y un peso total de 80g con la batería incluida. La caja del dron incluye una herramienta específica para montar o desmontar las hélices, junto con dos hélices de repuesto y un manual básico de instrucciones[25].

También se suministra una batería adicional, totalizando dos baterías en el paquete. Además, cuenta con cuatro protectores para los rotores y hélices, que se pueden acoplar fácilmente con un clic en cada uno de los rotores[25] .



Figura 17. DJI Tello Edu [25].

1.4.5 Inteligencia Artificial

Definir la inteligencia artificial representa un reto debido a la naturaleza imprecisa del concepto de inteligencia en sí, la IA representa un conjunto de capacidades que permite a una máquina poder emular funciones cognitivas humanas tales como: creatividad, aprendizaje, comprensión, percepción del entorno y el uso del lenguaje. Un subcampo en auge de la IA es el aprendizaje computacional, donde los sistemas aprenden a realizar tareas a partir de ejemplos o mediante ensayo y error[26].

En la Tabla 11 se detallan algunas áreas de inteligencia artificial.

a. Aprendizaje automático

El aprendizaje automático, también conocido como machine learning en inglés, constituye una de las vertientes fundamentales de la inteligencia artificial. En esencia, se refiere a una disciplina informática en la que las computadoras o máquinas adquieren la capacidad de aprender sin necesidad de ser explícitamente programadas para ello. Como resultado, este proceso permite generar sugerencias o realizar predicciones en contextos específicos[26].

Tabla 11. Áreas de la Inteligencia Artificial[27].

Área	Descripción
Tratamiento de lenguajes naturales	En este campo se incluyen ciertas aplicaciones que abarcan desde traducciones entre distintos idiomas hasta interfaces hombre-máquina que posibilitan la consulta de bases de datos o la emisión de órdenes a un sistema operativo.
Sistemas expertos	En este ámbito se integra el conocimiento de expertos cualificados en sistemas con el fin de lograr inferencias más precisas y cercanas a la realidad.
Robótica	En este campo también se incluyen aplicaciones como la navegación de robots móviles, el control de brazos robóticos y el ensamblaje de piezas, entre otros sistemas automatizados, donde la inteligencia artificial se emplea para mejorar la eficiencia, la precisión y la adaptabilidad de dichos procesos.
Problemas de percepción	En este ámbito, que abarca tanto la visión como el habla, se desarrollan tecnologías para el reconocimiento de objetos y la identificación del habla, así como la detección de defectos en piezas mediante sistemas de visión, entre otras aplicaciones.
Aprendizaje	Otra denominación para esta disciplina es "aprendizaje automático", que consiste en desarrollar técnicas que permiten a las computadoras aprender a resolver problemas mediante el análisis de datos y la identificación de patrones.

A continuación, en la Tabla 12 se describen los algoritmos de aprendizaje automático.

Tabla 12. Algoritmos de aprendizaje automático[26].

Tipos de Aprendizaje Automático	Descripción
Aprendizaje supervisado	Se emplean conjuntos de datos previamente etiquetados u organizados para instruir al sistema sobre cómo clasificar nueva información. Este enfoque implica la participación humana para proporcionar retroalimentación.
Aprendizaje no supervisado	En contraste con los enfoques convencionales, los algoritmos en cuestión no se basan en datos previamente etiquetados u organizados para guiar la clasificación de nueva información, sino que deben desarrollar sus propios métodos de clasificación. Por lo tanto, este método prescinde de la intervención humana, ya que los algoritmos asumen la responsabilidad de categorizar la información por sí mismos.
Aprendizaje de refuerzo	Desde una perspectiva técnica, los algoritmos adquieren conocimiento a través de la experiencia, lo que implica que reciben un estímulo positivo cada vez que logran una tarea exitosamente.

b. Aprendizaje profundo

Una de las ramas más potentes y en constante expansión de la inteligencia artificial es el aprendizaje profundo, conocido como deep learning en inglés. Este subcampo del aprendizaje automático se emplea para abordar problemas altamente complejos que suelen involucrar grandes volúmenes de datos[26].

El aprendizaje profundo se lleva a cabo mediante el uso de redes neuronales organizadas en múltiples capas, las cuales son capaces de identificar relaciones y patrones complejos en los datos. Su implementación requiere conjuntos de datos masivos y una capacidad de procesamiento considerable. Actualmente, se utiliza en una variedad de aplicaciones como reconocimiento de voz, procesamiento de lenguaje natural, visión artificial y sistemas de asistencia al conductor para la identificación de vehículos[26].

c. Redes Neuronales

Una red neuronal artificial (ANN) es un modelo de computación distribuida que toma inspiración de la estructura del sistema nervioso humano. La arquitectura de una red neuronal se construye conectando múltiples procesadores elementales, formando así un sistema adaptativo que cuenta con un algoritmo para ajustar sus pesos (parámetros libres) con el fin de satisfacer los requisitos de rendimiento del problema utilizando muestras representativas[28].

Una ANN es un sistema de computación que tienen las siguientes características[28]:

- Grupo de unidades básicas con capacidades de procesamiento limitadas.
- Una estructura altamente interconectada que utiliza conexiones ponderadas.
- Parámetros ajustables que necesitan ser modificados para cumplir con los criterios de rendimiento.
- Un elevado nivel de ejecución simultánea.

- **Redes neuronales profundas (DNN)**

Las redes neuronales profundas, conocidas como DNN, tienen la capacidad de adquirir características de un nivel de complejidad mayor en comparación con las redes neuronales menos profundas. En el ámbito de la Visión por Computadora, un tipo específico de estas redes profundas son las Redes Neuronales Convolucionales (CNN), las cuales son ampliamente utilizadas para el procesamiento de datos a nivel de píxeles. Se ha sugerido a los investigadores que utilicen arquitecturas populares para construir nuevos modelos, incluyendo ejemplos como Yolo, AlexNet, CaffeNet, VGG, ResNet, entre otros[29].

1.4.6 Visión Artificial

La visión artificial tiene como finalidad la extracción de información del mundo físico a partir de imágenes, utilizando para ello un computador. Se trata de un objetivo ambicioso y complejo que actualmente se encuentra en una etapa primitiva[30].

Actúa sobre una representación de una realidad que le proporciona información sobre brillo, colores, formas, etcétera. Estas representaciones suelen estar en forma de imágenes estáticas, escenas tridimensionales o imágenes en movimiento[30].

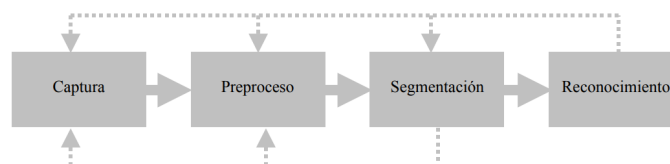


Figura 18. Diagrama de bloques de las etapas típicas en un sistema de visión artificial [30].

1.4.7 Procesamiento digital de imágenes

Proceso mediante el cual se toma una imagen y se produce una versión modificada de esta imagen[31].

El procesamiento digital de imágenes es una disciplina de la informática que se ocupa de manipular imágenes digitales mediante el uso de algoritmos y técnicas computacionales. Consiste en aplicar una serie de operaciones a una imagen digital

para mejorarla, analizarla o extraer información específica. Las imágenes digitales son representaciones numéricas de imágenes o escenas y se almacenan en formato digital, generalmente como una matriz de píxeles[32].

Las principales áreas de aplicación del procesamiento digital de imágenes incluyen la mejora de imágenes, la segmentación de objetos, el reconocimiento de patrones, la restauración de imágenes, la detección de bordes, la compresión de imágenes, entre otros. Algunas de las operaciones comunes en el procesamiento digital de imágenes son[30]:

- Filtrado y Mejora de Imágenes
- Transformaciones Geométricas
- Segmentación de Imágenes
- Detección de Bordes
- Reconocimiento de Patrones
- Compresión de Imágenes
- Restauración de Imágenes
- Análisis de Color
- Procesamiento Morfológico
- Extracción de Características

1.4.8 Adquisición y representación de imágenes digitales

a. Captura de imágenes

Las imágenes digitales son “señales” discretas, que suelen tener origen en una “señal” continua. Por ejemplo, una cámara digital toma imágenes del mundo real que es continuo (tanto el espacio, como el espectro emitido por los objetos se consideran continuos); otro ejemplo es el de un escáner, el cual digitaliza imágenes procedentes de documentos o fotografías que a efectos prácticos también se consideran continuos[30].

En el proceso de obtención de imágenes digitales se distinguen dos etapas. La primera, conocida como captura, utiliza un dispositivo, generalmente óptico, con el que obtiene información relativa a una escena. En la segunda etapa, que se conoce como digitalización, se transforma esta información, que es una señal con una o varias componentes continuas, en la imagen digital, que es una señal con todas sus componentes discretas[30].

- ***Modelos de captura de imágenes***

Al momento de realizar la captura de imágenes existen distintos modelos que pueden usarse, siendo estos[30]:

- El modelo de lente fina
- La cámara oscura
- El escaneo

- ***La digitalización***

Es el proceso de paso del mundo continuo (o analógico) al mundo discreto (o digital). En la digitalización normalmente se distinguen dos procesos: el muestreo (“sampling”) y la cuantización (“quantization”)[30].

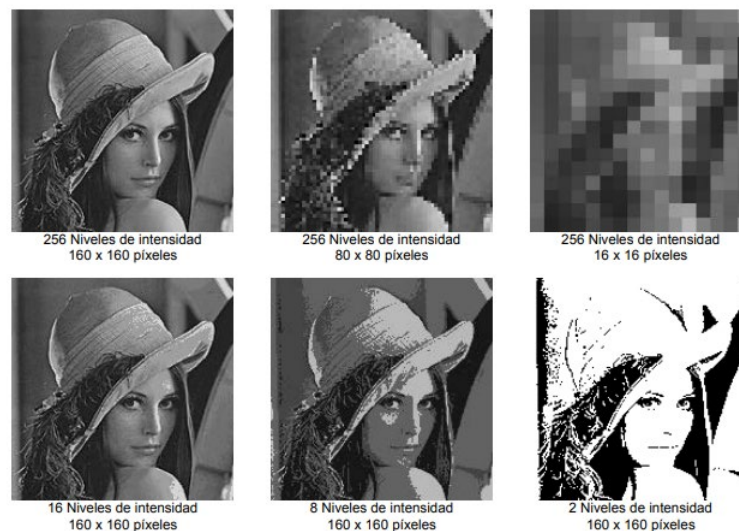


Figura 19. En la fila superior se presenta la misma imagen, siempre a 256 niveles de intensidad, usando diferentes resoluciones espaciales. En la fila inferior se mantiene la resolución espacial y se reduce el nivel de cuantización [30].

Una vez digitalizada una imagen bidimensional queda constituida por un conjunto de elementos llamados píxeles. Cada píxel ofrece cierta información sobre una región elemental de la imagen. En imágenes en blanco y negro esta información es el brillo. En imágenes en color, la información corresponde a la intensidad de cada una de las componentes de una base de color (por ejemplo, RGB)[30].

b. Filtrado, realizado y segmentación de imágenes

Un filtro puede verse como un mecanismo de cambio o transformación de una señal de entrada a la que se le aplica una función, conocida como función de transferencia, para obtener una señal de salida. En este contexto se entiende por señal una función de una o varias variables independientes. Los sonidos y las imágenes son ejemplos típicos de señales[30].

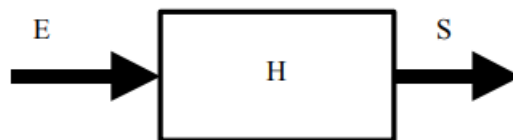


Figura 20. Esquema de funcionamiento de un filtro [30].

Se representa el esquema general de funcionamiento de un filtro, siendo E la función de entrada, S la de salida y H la función de transferencia del filtro. Todas estas señales y funciones pueden ser discretas o continuas, y aunque en el tratamiento de imágenes se procesan señales y funciones discretas, suele recurrirse al caso continuo para explicar sus comportamientos, ya que sobre las funciones continuas es posible emplear herramientas más potentes de cálculo matemático[30].

1.4.9 Algoritmo de procesamiento de imágenes

Un algoritmo de procesamiento de imágenes es un conjunto de pasos o instrucciones diseñadas para realizar operaciones específicas en una imagen con el objetivo de obtener información o mejorar sus características. Estos algoritmos son esenciales en diversas aplicaciones, como visión por computadora, reconocimiento de patrones, análisis médico, entre otros[33].

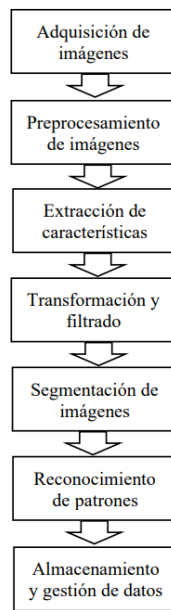


Figura 21. Algoritmo de procesamiento de imágenes[33]

Las diversas etapas de un algoritmo de procesamiento de imágenes desempeñan roles cruciales en la obtención de información significativa a partir de datos visuales. La etapa de adquisición establece la base, asegurando la disponibilidad de una imagen de calidad para el análisis. El preprocesamiento suaviza imperfecciones y mejora la coherencia de los datos, facilitando análisis más precisos[32].

La segmentación permite la identificación de áreas clave, mientras que la extracción de características destaca aspectos distintivos para una representación más efectiva. El procesamiento principal modifica la imagen según objetivos específicos, y el post-procesamiento refina resultados, asegurando coherencia. Estas etapas convergen en la fase de análisis, donde la interpretación de datos visuales se traduce en la toma de decisiones informadas, destacando la importancia integral de cada fase para obtener resultados confiables y valiosos en diversas aplicaciones[32].

La capacidad de los algoritmos de procesamiento de imágenes para interpretar y extraer información valiosa de los datos visuales ha llevado a avances revolucionarios en campos tan diversos como la medicina, la seguridad, la industria automotriz y la inteligencia artificial.[32].

Tabla 13. Etapas del procesamiento de imágenes[33].

Etapas	Descripción
Adquisición de imágenes	Captura de imágenes utilizando cámaras, escáneres u otros dispositivos de entrada.
Preprocesamiento de imágenes	Mejora de la calidad y claridad de las imágenes mediante ajuste de contraste, eliminación de ruido, etc.
Extracción de características	Identificación y extracción de características relevantes de las imágenes, como bordes, texturas y formas.
Transformación y filtrado	Aplicación de técnicas de transformación y filtrado para modificar y mejorar las imágenes.
Segmentación de imágenes	Separación de regiones de interés u objetos del fondo mediante algoritmos de segmentación.
Reconocimiento de patrones	Identificación y clasificación de elementos o patrones específicos dentro de una imagen
Almacenamiento y gestión de datos	Almacenamiento y gestión eficiente de imágenes y datos relacionados en sistemas de almacenamiento adecuados

1.4.10 Modelado de imágenes

El modelado de imágenes se refiere al proceso de crear representaciones matemáticas o computacionales de imágenes para capturar sus características esenciales y facilitar su análisis, manipulación o procesamiento. Este proceso es fundamental en campos como la visión por computadora, el procesamiento de imágenes y la inteligencia artificial[34].

El objetivo del modelado de imágenes es convertir datos visuales en una forma que sea más fácil de comprender, analizar y utilizar en aplicaciones específicas. Algunas técnicas comunes de modelado de imágenes incluyen[34].

- Modelado Geométrico
- Modelado de Color
- Modelado de Texturas
- Modelado Estadístico
- Modelado de Superficies
- Modelado de Iluminación

1.4.11 Visión Artificial aplicada a la Agricultura

Los sistemas de visión artificial se emplean cada vez más en la industria agroalimentaria para realizar las inspecciones de calidad, ya que ofrecen el potencial necesario para automatizar las prácticas manuales de selección a la vez que estandarizan las técnicas y eliminan las costosas tareas humanas de inspección. La visión artificial incluye la captura, el procesamiento y el análisis de imágenes, facilitando la valoración objetiva y no destructiva de las características visuales de calidad en los productos alimentarios a la vez que permite la toma de decisiones, posterior al reconocimiento[35].

Estos sistemas se pueden combinar con otras tecnologías innovadoras, como el Internet de las cosas (IoT), sensores, sistemas de geoposicionamiento, Big Data, drones y robots. Debido a la heterogeneidad y estacionalidad de los productos, la automatización de los procesos agroalimentarios debe ser extremadamente precisa y flexible para poder anticipar ineficiencias o anomalías en la producción con su consecuente impacto en la reducción de costes y la mejora de los tiempos productivos[35].

Por ejemplo, la visión artificial permite: Automatizar la maquinaria agrícola y los procesos de cosecha. Digitalizar los procesos de cultivo de invernadero. Optimizar la explotación agrícola. Mecanizar la clasificación e inspección de semillas y frutos. Optimizar la cría de animales en granjas[35].

a. Ventajas

La visión artificial combina hardware y software con la capacidad de capturar, procesar y analizar imágenes para tomar decisiones y actuar en consecuencia. Estos son algunos ejemplos de su aplicación en el sector agroalimentario[36].

- Imágenes aéreas multiespectrales
- Análisis del punto de maduración de frutas y hortalizas
- Detección de malezas
- Control de plagas y enfermedades
- Categorización y precio de compra para materias primas naturales

- Supervisión y diagnóstico de la salud del ganado

1.4.12 Control de calidad empleando visión artificial

El control de calidad de la producción de alimentos permite limitar los desperdicios, garantizar un producto final que cumpla con la normativa vigente, proteger a la empresa de posibles reclamaciones y de la retirada de productos no reglamentarios[36].

Sin embargo, los desafíos son múltiples y complejos. La inspección debe ayudar a identificar productos contaminados con cuerpos extraños, empaques dañados o alimentos en mal estado[36].

La utilización de visión artificial contribuye a disminuir errores en la elección y empaquetado de alimentos, mejorando eficientemente todo el procedimiento. Dispositivos de captura de imágenes de alta velocidad, como cámaras de vídeo equipadas con tecnología InGaAs o CMOS, escanean los productos. Mientras que las primeras identifican defectos que se encuentran debajo de la superficie del producto, las segundas identifican los defectos visibles. etiquetados y alimentos en mal estado[36].

1.4.13 Clasificación de frutos

El proceso de clasificación de frutos mediante un algoritmo de procesamiento de imágenes es esencial para la optimización y eficiencia en la producción agrícola. La adquisición de imágenes de los frutos proporciona la base necesaria para la evaluación de la calidad del producto, en la Tabla 14 se resume el proceso a seguir al momento de clasificar frutos haciendo uso de algoritmos de procesamiento de imágenes[34].

Tabla 14. Proceso de clasificación de frutos[34], [37].

Paso	Descripción
1. Adquisición de Datos	Recopilación un conjunto diverso de imágenes representativas de las frutas que desees clasificar.
2. Preprocesamiento de Imágenes	Mejorar la calidad de las imágenes mediante operaciones como corrección de color, ajuste de tamaño y eliminación de fondos no deseados.
3. Segmentación	Separar las frutas del fondo utilizando técnicas de segmentación, como la umbralización.
4. Extracción de Características	Identificar y extraer características relevantes como color, textura, forma y tamaño de las frutas.
5. Entrenamiento del Modelo	Usar un algoritmo de aprendizaje automático (por ejemplo, SVM, CNN) para entrenar el modelo de clasificación con un conjunto de entrenamiento.
6. División de Datos	Dividir el conjunto de datos en conjuntos de entrenamiento y prueba para evaluar la eficacia del modelo.
7. Entrenamiento del Modelo	Ajustar los parámetros del modelo alimentándolo con las imágenes de entrenamiento.
8. Validación y Ajuste	Utilizar el conjunto de prueba para validar el rendimiento del modelo y realizar ajustes según sea necesario.
9. Pruebas en Tiempo Real (Opcional)	Implementar el modelo entrenado en un sistema para clasificar frutas en tiempo real si es necesario.

a. Tarea de postrecolección

La tarea de postrecolección en agricultura abarca una serie de actividades realizadas después de la cosecha con el objetivo de preservar la calidad de los productos, prepararlos para su comercialización y minimizar pérdidas[36].

- Clasificación y Selección
- Limpieza y Desinfección
- Enfriamiento
- Empaque
- Monitoreo y Control de Calidad
- Procesamiento Postcosecha
- Gestión de Residuos
- Implementación de Tecnologías

1.4.14 Microcomputador

Es un sistema informático que emplea un microprocesador como CPU y se caracteriza por su tamaño compacto y versatilidad. Incorpora componentes como memoria RAM, unidades de almacenamiento e interfaces de entrada y salida. Su diseño permite ejecutar una amplia gama de aplicaciones, desde software de propósito general hasta aplicaciones específicas.

a. Raspberry Pi 5

La Raspberry Pi 5 está basada en arquitectura ARM Cortex-A76, cuenta con un procesador de cuatro núcleos y 2,4 GHz de 64 bits. Ofrece una amplia gama de opciones de conectividad, incluyendo puertos Gigabit Ethernet, USB (3.0 y 2.0) y HDMI dual 4kp60 con soporte HDR, así como soporte para Wi-Fi de doble banda y Bluetooth 5.0 / Bluetooth de bajo consumo integrados. La Raspberry Pi 5 también incorpora mejoras en memoria RAM (SDRAM SKU de 4 GB y 8 GB) y almacenamiento, con opciones de ampliación disponibles al integrar una ranura para tarjeta microSD, compatible con el modo SDR104 de alta velocidad. Con su capacidad para ejecutar sistemas operativos completos como Linux, la Raspberry Pi 5 es una potente herramienta para proyectos de desarrollo de software, IoT, educación, IA y más[38].

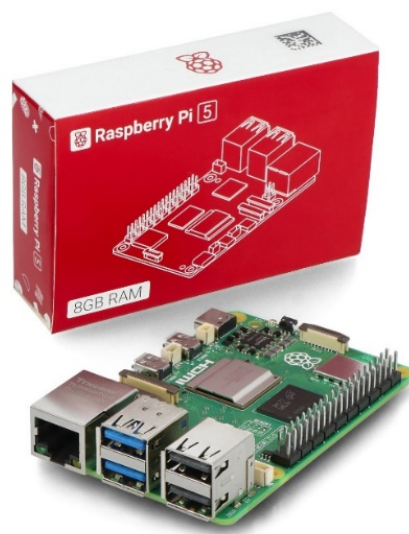


Figura 22. Raspberry Pi 5 [38].

b. NVIDIA Jetson Nano

La Jetson Nano es una computadora de placa única (SBC) desarrollada por NVIDIA, diseñada específicamente para aplicaciones de inteligencia artificial (IA) y aprendizaje profundo en el borde. Está equipada con un potente procesador ARM Cortex-A57 de cuatro núcleos y un GPU NVIDIA Maxwell con 128 núcleos CUDA, lo que le permite realizar tareas de procesamiento de datos intensivas con eficiencia energética. La Jetson Nano cuenta con 4 GB de memoria RAM LPDDR4 y ofrece múltiples opciones de conectividad, incluyendo puertos USB 3.0, Gigabit Ethernet, HDMI y ranura para tarjetas microSD. Además, tiene soporte para cámaras CSI y DisplayPort para aplicaciones de visión artificial y visualización. Esta combinación de potencia de computación y tamaño compacto la hace ideal para una amplia variedad de aplicaciones, como robots autónomos, sistemas de vigilancia, drones y más[39].



Figura 23. NVIDIA Jetson Nano [39].

c. AI-Motion K210

El kit de desarrollo AI-Motion es una plataforma integral diseñada para proyectos avanzados de inteligencia artificial centrados en el análisis y detección de movimiento. Este kit está compuesto por un conjunto de sensores de alta precisión, como cámaras de visión, LIDAR (Light Detection and Ranging) y/o acelerómetros, junto con una potente unidad de procesamiento, posiblemente una GPU o una NPU (Neural Processing Unit). Está optimizado para ejecutar algoritmos de aprendizaje profundo y visión por computadora de manera eficiente. Además, proporciona interfaces de programación y bibliotecas de software específicamente diseñadas para el

procesamiento y análisis de datos de movimiento, facilitando así el desarrollo de aplicaciones avanzadas en este ámbito[40].



Figura 24. AI-Motion K210 [40].

1.5 Objetivos

1.5.1 Objetivo general

Implementar un sistema de control de calidad de cultivo de fruta de temporada para etapa de precosecha empleando robótica aérea con planificación de trayectorias y visión artificial.

1.5.2 Objetivos específicos

- Analizar los métodos de clasificación y control de calidad en la producción de fruta de temporada.
- Determinar los algoritmos de visión artificial para la clasificación de la producción de fruta y planificación de trayectorias.
- Elaborar una interfaz web que permita la interpretación y gestión de información de la producción del cultivo de fruta de temporada.

CAPÍTULO II. METODOLOGÍA

2.1 Materiales

Para el desarrollo del proyecto de investigación se han empleado diversos dispositivos relacionados al uso de inteligencia artificial que mejor se adapten a lo requerido. Se ha desarrollado un sistema de control de calidad de cultivo de manzanas en la etapa de precosecha utilizando inteligencia artificial, basado en el lenguaje de programación Python. Se implementó la arquitectura YOLO (You Only Look Once) para la detección de imágenes, y la herramienta Labelme se utilizó para el etiquetado de datos.

El modelo se entrenó con un conjunto de datos personalizado que contiene 1500 imágenes de manzanas en diferentes estados de madurez y con daños que afectan la calidad. El entrenamiento de la red neuronal se realizó en Google Colab para aprovechar su capacidad de procesamiento. Para el procesamiento eficiente de estos datos en tiempo real, se emplearon una NVIDIA Jetson Nano y la cámara del UAV DJI TELLO. La planificación de trayectorias del UAV permite una cobertura completa del cultivo para la captura de imágenes.

Los datos procesados en tiempo real se dirigen a una base de datos SQLite y se gestionan mediante una aplicación web desarrollada con Flask. En las Tabla 15 y Tabla 16 se detallan el hardware y software utilizados en el sistema.

Tabla 15. Características del hardware seleccionado

Material	Característica	Descripción
NVIDIA Jetson Nano	Sistema de desarrollo para aplicaciones de Inteligencia Artificial	Potente ordenador compatible con el software de IA de NVIDIA diseñada para acelerar aplicaciones de IA ya que cuenta con un procesador potente incluyendo GPU.
DJI Tello Edu	Cámara HDR Programable Ligero	UAV educativo diseñado para la experimentación y aprendizaje con IA. Está equipado con una cámara HD y capacidades de programación ideal para la captura de imágenes y videos de alta calidad en aplicaciones de monitoreo agrícola.

Tabla 16. Característica de software seleccionado

Material	Característica	Descripción
Flask	Microframework web ligero	Diseñado para facilitar el desarrollo de aplicaciones web rápidas y seguras, ideal para proyectos de IA y microservicios.
SQLite	Motor de base de datos SQL pequeño, rápido, autónomo y de alta confiabilidad	Motor de base de datos SQL embebido, sin necesidad de configuración y altamente eficiente con aplicaciones de almacenamiento local.
Google Colab	Entorno de desarrollo en la nube	Plataforma de Google que permite ejecutar código de Python en notebooks Jupyter en la nube, con acceso a TPUs y GPUs para tareas de entrenamiento intensivo.
Python	Lenguaje de programación	Lenguaje de programación de alto nivel, ampliamente utilizado en desarrollo web, análisis de datos e inteligencia artificial.
Yolov8n	Arquitectura de visión artificial	Basada en una arquitectura de red neuronal profunda, ofrece mejoras significativas en la extracción de características y eficiencia del modelo.

2.2 Métodos

2.2.1 Modalidad de la investigación

A continuación, se describe la modalidad de investigación que se aplicó para el desarrollo del proyecto:

El proyecto tiene la característica de ser una investigación aplicada, ya que se emplaron conocimientos ya existentes para desarrollar el sistema de gestión y control de calidad de la producción de fruta de temporada en etapa de precosecha.

La investigación es de tipo bibliográfica, ya que fue fundamentada en la recopilación de información proveniente de libros, artículos científicos, datasheets de los dispositivos usados, publicaciones en internet y tesis relacionadas con la implementación del sistema.

Al ser un proyecto de investigación de carácter práctico, se empleó además la investigación de tipo experimental, enfocada en recopilar datos fundamentales basados en la experimentación para la validación del sistema de control de calidad, contribuyendo así a la mejora continua y la optimización de las tecnologías empleadas en la etapa de precosecha de los cultivos frutales.

2.2.2 Recolección de información

Para la obtención de información se hizo uso de los distintos repositorios de universidades del país, libros, revistas de difusión científica en línea, y demás bases de datos que sean confiables con la finalidad que puedan aportar para el desarrollo del proyecto.

Tabla 17: Criterios de recolección de información

Criterio	Descripción
¿Para qué?	Determinar el estado actual del proceso de control de calidad en etapa de precosecha.
¿De qué personas u objetos?	Del cultivo de fruta de temporada.
¿Cómo?	A través de la evaluación del sistema de control de calidad de cultivos frutales.
¿Con qué?	Informes y reportes del sistema de gestión de información.
¿Cuándo?	En el periodo académico marzo – agosto 2024
¿Dónde?	En la provincia de Tungurahua, cantón Cevallos, sector Manzana de Oro.

2.2.3 Procesamiento y análisis de datos

Con la información recopilada se procedió de la siguiente manera:

- Recolección de la información.
- Revisión de la información obtenida.
- Análisis de la fundamentación teórica de la información sobre las variables de investigación.
- Deducción de la información más relevante y útil para el proyecto de investigación.

CAPÍTULO III. RESULTADOS Y DISCUSIÓN

3.1 Desarrollo de la propuesta

Para la ejecución del proyecto, se llevó a cabo una modalidad investigativa, bibliográfica y de campo, enfocada inicialmente en encontrar cultivos con gran índice frutal en la ciudad de Ambato siendo el cultivo de manzana el seleccionado para el desarrollo del proyecto.

La implementación del sistema se desarrolla a través de un modelo implementado en Python, aprovechando su versatilidad y la disponibilidad de librerías con inteligencia artificial. Para el entrenamiento del sistema de detección y clasificación de instancias se seleccionó el modelo YOLOv8. El entrenamiento se realizó en la plataforma Google Colab con YOLOv8 Ultralytics. El proceso de etiquetado de datos se realizó mediante las herramientas LabelMe y LabelImg para el proceso de elaboración del DataSet.

3.1.1 Definición de fruta de temporada para la propuesta

a. Frutas en el Ecuador

En el Ecuador existen una gran variedad de frutas que se cultivan en diferentes regiones gracias a la biodiversidad y variados climas del país. La diversidad geográfica del país permite el cultivo de frutas tropicales en la Costa y Amazonía, así como de frutas subtropicales en la Sierra. A continuación, en la Tabla 18 se detalla las frutas cultivadas en Ecuador y las temporadas en las que estas están en producción[41].

b. Selección de fruta para estudio

El objeto de estudio para esta investigación es la ciudad de Ambato, ubicada dentro de la zona 3 Central del Ecuador. Esta región, con una extensión significativa de 44.899 km², constituye aproximadamente el 18% del territorio nacional y está administrativamente compuesta por cuatro provincias: Cotopaxi, Chimborazo, Pastaza y Tungurahua.

En Ambato, al igual que en otras áreas de esta zona, se ha establecido una red de 19 distritos y 142 circuitos para mejorar la prestación de servicios y acercarlos a la población. Ambato, además de ser reconocida por su importancia agropecuaria y manufacturera, se destaca por su estratégica ubicación geográfica, sirviendo como punto de conexión vital entre la Sierra, la Costa y la Amazonía ecuatoriana.

La región de la sierra ecuatoriana se distingue por sus montañas, donde el clima fresco y las condiciones óptimas impulsan la agricultura de una variedad de frutas y verduras. Entre las frutas habitualmente cultivadas en esta zona se encuentran manzanas, peras, arándanos, fresas y moras, las cuales tienden a ser particularmente frescas y jugosas debido a las condiciones climáticas favorables que promueven su desarrollo.

Tabla 18. Frutas por temporada en el Ecuador[41], [42].

Fruta	Región	Temporada
Aguacate	Sierra	Todo el año
Arazá	Costa	Todo el año
Babaco	Sierra	Todo el año
Borojó	Costa	Todo el año
Chirimoya	Sierra	Octubre-Marzo
Cítricos	Sierra	Mayo-Octubre
Copouazú	Amazonia	Todo el año
Durazno	Sierra	Octubre-Marzo
Granadilla	Costa	Mayo-Octubre
Guanábana	Amazonia	Todo el año
Mango	Costa	Septiembre-Enero
Maracuyá	Costa	Todo el año
Manzana	Sierra	Abril-Mayo
Mora	Sierra	Todo el año
Naranjilla	Sierra	Todo el año
Taxo	Sierra	Todo el año
Tomate de árbol	Sierra	Todo el año
Uva	Todas	Enero-Junio
Uvilla	Todas	Todo el año
Pera	Sierra	Agosto - Octubre
Pera blanca	Sierra	Abril - Mayo

En la Tabla 18, también se presentan las frutas más destacadas de la región sierra junto a la temporada en la que se encuentran disponibles. En Ambato y en el resto de la región sierra centro del país cerca de la temporada de Carnaval en el primer trimestre de cada año se comercializan frutas de temporada como: claudia, pera, uvilla y duraznos, y, entre los meses de abril y mayo los productores comercializan la pera blanca y manzanas de la región [43].

Además, teniendo en cuenta la relevancia de la manzana como el fruto más consumido del país, es importante explorar cómo la producción local en Ambato y en la región en general enfrenta desafíos para satisfacer la demanda interna, lo que conlleva a la dependencia de importaciones y a una salida de divisas. Cabe destacar que, a pesar de su larga historia en la región desde la época colonial, el cultivo de manzana en Ambato y sus alrededores ha sido tradicionalmente gestionado con tecnologías ancestrales, lo que también puede influir en su producción y calidad[44].

A partir de un análisis en los mercados realizado para determinar la fruta de temporada en la ciudad de Ambato, se ha optado por la manzana debido a la temporada de disponibilidad de la producción que es acorde al periodo en el que se desarrolla esta investigación. La manzana en cuanto a demanda y tendencias del mercado de frutas tiene un bajo nivel de venta, preferencias del consumidor y precio. Esto se debe a la falta de control de calidad de esta y por lo cual no se puede mitigar el estado de esta.

- ***Condiciones Ambientales donde se produce la fruta seleccionada (manzana)***

Altitud

Los cultivos de manzana se producen en la zona alta del Ecuador, localizadas en las provincias de Tungurahua, Chimborazo, Cañar, Cotopaxi, Azuay y Loja en altitudes que van de 2650 a 3200 msnm[45].

Clima

Para asegurar un rendimiento óptimo del cultivo de esta fruta en las zonas elevadas del Ecuador, es necesario contar con un período caracterizado por condiciones climáticas relativamente cálidas y lluviosas, que promuevan el crecimiento de la vegetación y la producción de frutos. Además, se requiere otro período frío, seco y ventoso, con cierta presencia de nubosidad, que facilite el proceso de reposo vegetativo y la inducción de la floración[45].

Una temperatura media de alrededor de 13°C es ideal para acumular las horas de frío necesarias, que generalmente oscilan entre 400 y 600 horas, para un desarrollo óptimo del cultivo[45].

Suelo

El cultivo de manzana se adapta a una amplia gama de tipos de suelo, desde arenosos hasta arcillosos, aunque muestra una preferencia por suelos sueltos para evitar complicaciones radiculares debido al exceso de humedad. Se recomiendan especialmente suelos franco-arenosos, profundos y ricos en materia orgánica, con un pH que oscila entre 5.8 y 7.5. Además, se sugiere que los suelos tengan una inclinación del pH entre 5 y 10%, con el objetivo de mitigar los efectos adversos de las heladas. También se prefieren terrenos planos para el cultivo de manzana[45].

- ***Variedades de manzana***

En esta región, se encuentran disponibles variedades de manzana como se detalla en la Tabla 19, con diferentes períodos de maduración: tempranas, que son cosechadas en diciembre; intermedias, que se recolectan entre febrero y marzo; y tardías, que pueden ser cosechadas hasta junio[45].

Tabla 19. Variedades de manzana en la sierra ecuatoriana[45].

Nombre de la variedad	Altitudes msnm	Requerimientos Horas-frío	Época de cosecha	Desarrollo del árbol
Gravenstein	2.650 - 2800	600	Diciembre– Enero	Vigor medio
Emilia	2.650 - 3200	700	Febrero – Abril	Muy vigoroso
Gala	2.650 – 2800	650	Marzo - Abril	Vigoroso
Royal Gala	2.650 – 3000	650	Marzo - Abril	Vigoroso
Rome Beauty	2.650 – 3200	700	Febrero - Marzo	Vigoroso
W. Banana	2.650 – 3000	600	Enero - Febrero	Vigoroso
Jona Gold	2.650 - 3000	700	Abril - Mayo	Vigoroso
G. Delicius	2.650 – 3200	800	Abril - Mayo	Vigor medio
Belgolden	2.650 – 3000	800	Abril - Mayo	Vigor medio
Granny Smith	2.650 – 3200	850	Mayo Junio	Vigoroso

- ***Selección de variedades de manzana***

Para la elección de las variedades de manzanas a evaluar durante el proyecto se idéntico aquellas que predominan en el periodo que se realiza la investigación incluyendo una variedad de manzana que es conocida a nivel mundial como los es la Manzana Fuji, y, en base a la Tabla 19 se han seleccionado las siguientes:

Tabla 20. Variedades de manzanas seleccionadas.

Nombre	Color y Forma Representativa	
Red Delicious	Es de forma cónica, alargada y tiene un tamaño promedio, exhibiendo un tono rojo oscuro.	
Granny Smith	Posee una pigmentación verde intensa y homogénea, con una estructura elongada y simétrica.	
Fuji	Exhibe una tonalidad combinada de rojo y rosa, con presencia de tonos amarillos, y una forma redonda.	
Golden Delicious	Presenta un color dorado amarillento con matices verdosos, y su morfología es alargada y redonda.	

3.1.2 Atributos para la evaluación de la calidad de manzanas

La evaluación de la calidad de las manzanas es un proceso fundamental en la industria agrícola, donde diversos atributos internos y externos son cuidadosamente examinados para garantizar la excelencia del producto final. Estos atributos, tanto visibles como medibles, desempeñan un papel crucial en la determinación de la calidad y el valor comercial de las manzanas en la etapa precosecha. Cada atributo proporciona información valiosa sobre la madurez, el sabor y la textura de las manzanas. A continuación, en la Tabla 21 se presentan los atributos comúnmente utilizados para la evaluación de la calidad de las manzanas[46].

Además, la calidad de las manzanas en la etapa de precosecha es determinada por una serie de factores interconectados que incluyen aspectos intrínsecos de la planta y elementos externos. Los factores intrínsecos están relacionados con la fisiología de la

planta, como la adecuada distribución y asimilación de recursos energéticos, hídricos y nutricionales[47].

Tabla 21. Atributos para evaluar la calidad de las manzanas[46].

Atributo	Descripción
Color	Tonalidad de la piel de la manzana, que puede variar según la variedad y el grado de maduración.
Diámetro	Medida de la anchura de la manzana, que puede indicar su tamaño y potencial comercial.
Grados Brix	Nivel de concentración de azúcares en el jugo de la manzana, que influye en su dulzura y sabor.
Acidez	Concentración de ácido málico y otros ácidos en la pulpa de la manzana, que afecta su sabor y frescura.
Almidones	Contenido de almidón en la pulpa de la manzana, que disminuye a medida que la fruta madura, indicando su estado de madurez y tiempo óptimo de consumo.
Firmeza	Resistencia de la pulpa de la manzana al ser presionada, que puede indicar su frescura y textura.

En contraste, los factores externos abarcan elementos genéticos, ambientales y agronómicos que pueden influir en el desarrollo y la calidad final del fruto. La comprensión de estos factores es crucial para optimizar las prácticas de manejo y mejorar la calidad de la cosecha[47].

En la Tabla 22 se detallan estos factores, clasificándolos en factores intrínsecos y extrínsecos para facilitar la evaluación de su impacto en la calidad de las manzanas en la fase de precosecha:

Tabla 22. Factores para la evaluación de la calidad de manzanas[47], [48].

Categoría	Factores Intrínsecos	Factores Extrínsecos
Energía	Eficiencia fotosintética; metabolismo energético	Influencia del clima en la actividad fotosintética
Agua	Balance hídrico; eficiencia de uso del agua	Patrones de lluvia; prácticas de irrigación
Nutrientes	Metabolismo de nutrientes; almacenamiento	Fertilidad del suelo; aplicación de fertilizantes
Genéticos	Variabilidad genética; resistencia a enfermedades	Programas de mejoramiento genético
Ambientales	Resistencia a estrés abiótico (sequía, salinidad)	Temperatura; humedad; exposición al sol
Agronómicos	-	Técnicas de poda; control de malezas
Fisiológicos	Maduración y senescencia; producción de etileno	Efectos del microclima en la fisiología del fruto
Sanidad	Defensas naturales contra patógenos	Manejo integrado de plagas y enfermedades

Estos factores no solo influyen en la calidad de las manzanas, sino que también los defectos que pueden surgir durante su desarrollo y manejo. Desde el cultivo hasta el consumidor final, estos defectos pueden comprometer la apariencia, el sabor y la frescura de la fruta, generando preocupaciones tanto para los productores como para los consumidores. En la Tabla 23 se resumen los defectos más comunes que afectan la calidad de las manzanas en la etapa precosecha[47].

En base a la Tabla 21, Tabla 22 y Tabla 23, se han identificado una serie de atributos clave para la evaluación de la calidad del cultivo de manzana, estos atributos detallados a continuación han sido seleccionados cuidadosamente debido a su importancia para el proceso de detección y evaluación de calidad en la etapa de precosecha:

- Color (maduración)
- Factores fisiológicos (daños, pudrición y manchas de corcho)

Tabla 23. Defectos que afectan en la calidad de la manzana[46], [47], [48].

Defecto	Descripción
Daños	Magulladuras o daños causados durante el crecimiento, transporte o almacenamiento de las manzanas.
Deshidratación	Pérdida de humedad debido a condiciones ambientales adversas, lo que afecta el sabor y la calidad de la fruta.
Plagas	Infestaciones de áfidos, polillas, larvas, entre otros, que afectan la calidad de las manzanas y requieren tratamientos con pesticidas.
Defectos en la piel	Golpe de sol, manchas rojizas u otros daños en la epidermis de la fruta, reduciendo su resistencia y calidad.
Magulladuras	Hematomas o marcas en la superficie de la manzana que afectan su sabor y calidad.
Pudrición	Descomposición de la fruta debido a la maduración, infecciones fúngicas u otros factores.
Escaldaduras y quemaduras solares	Lesiones causadas por exposición prolongada a la luz solar directa, que afectan la firmeza y calidad de la fruta.
Moho	Infecciones fúngicas que pueden desarrollarse durante el almacenamiento o transporte de las manzanas.
Manchas de corcho	Pequeños granos o agujeros verdes en la piel de la manzana que afectan su aspecto y sabor.
Corazón acuoso	Acumulación de líquido entre las células de la fruta, resultando en una apariencia traslúcida.
Descomposición interna	Oxidación del líquido interno de la manzana, provocando su descomposición completa.
Tamaño pequeño	Manzanas más pequeñas de lo normal, que pueden afectar su precio en el mercado.

Estos atributos son críticos para determinar la calidad y la frescura de las manzanas, y su detección temprana puede ayudar a prevenir pérdidas significativas durante el

proceso de almacenamiento y comercialización. Además, son fácilmente observables y medibles en el campo, lo que los hace ideales para la implementación de técnicas de inteligencia artificial y visión por computadora.

a. Categorías para la cuantificación de calidad de manzanas

En base a los criterios de coloración y factores fisiológicos, para la cuantificación de la calidad de manzanas se han definido tres categorías que se detallan en la Tabla 24 y se muestran en la Figura 25.

Tabla 24. Grados de calidad para clasificación de manzanas[7].

Grado de Calidad	Descripción
Normal	Están libres de daños visibles, corteza intacta y en buen estado
Media	Presentan daños superficiales mínimos que no afectan a su calidad, no presenta podredumbre o enfermedades.
Mala	Presencia de daños evidentes como cortes, abolladuras, señales claras de podredumbre y enfermedades.

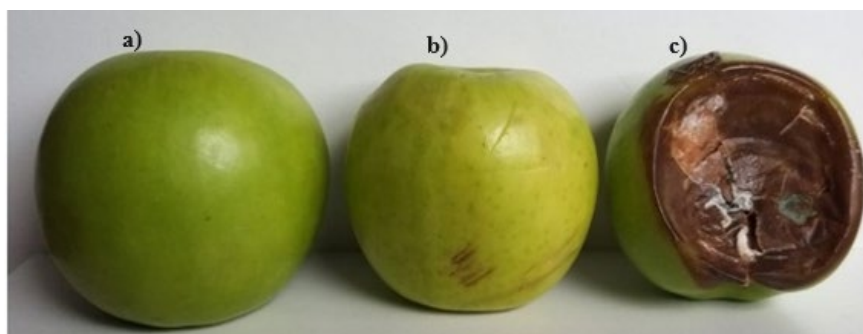


Figura 25. Grado de calidad manzana GrannySmith; a) Calidad normal, b) Media calidad, c) Mala calidad.

3.1.3 Arquitectura del sistema

En la Figura 26, se muestra la arquitectura del sistema que consta de 4 etapas, adquisición de datos, entrenamiento, procesamiento y visualización, detallados a continuación:

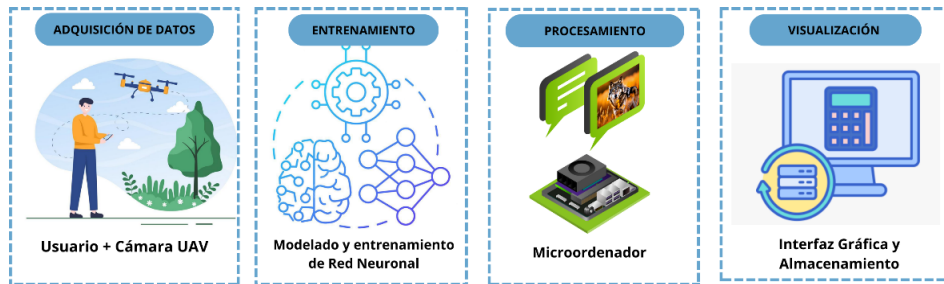


Figura 26. Arquitectura del sistema

a. Etapa 1: Adquisición de datos

Para obtener las imágenes necesarias para el sistema, se utilizó un UAV especializado, equipado con una cámara de alta resolución. Este UAV opera bajo una planificación de trayectorias, que optimiza varios aspectos críticos que garantiza una operación segura, eficiente y efectiva. Los aspectos optimizados incluyen:

- Tiempos de vuelo y velocidad: Se adecuan los tiempos de vuelo, se programan vuelos más lentos para que el dron se estabilice y capture imágenes de mayor enfoque y calidad.
- Seguridad: La planificación asegura la prevención de colisiones y evita que el UAV entre en zonas de exclusión aérea.
- Precisión de movimiento: La planificación se basa en comandos predefinidos, mejorando la precisión en la ejecución de movimientos y programando las distancias para ahorrar energía y tiempo, considerando múltiples puntos de interés.
- Adaptabilidad: El UAV se adapta a cambios en el entorno o a la topografía, evitando zonas peligrosas y ajustando su trayectoria según las condiciones presentes.

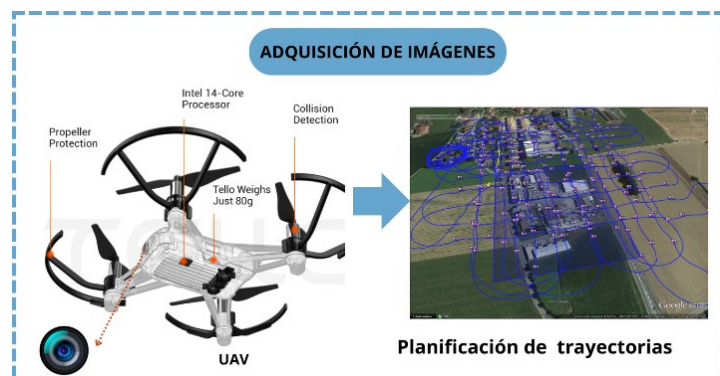


Figura 27. Arquitectura de adquisición de imágenes

Esta planificación de trayectorias permite al UAV capturar imágenes originales sin procesar para su posterior manipulación o ajuste de tamaño, recorte u otra transformación. A continuación, se muestra en la Figura 28, las imágenes obtenidas por el UAV.



Figura 28. Imágenes capturadas por el dron.

b. Etapa 2: Fase de entrenamiento

En la creación del Dataset, las imágenes obtenidas a través del UAV se procesan y etiquetan cuidadosamente para identificar características clave como la madurez de las manzanas, posibles defectos y signos de enfermedad, conformando así un conjunto de datos robusto para el entrenamiento de un modelo de inteligencia artificial. Se utilizó recursos computacionales disponibles en la nube para acelerar de manera significativa el entrenamiento de complejas redes neuronales.

Estos algoritmos de aprendizaje profundo, diseñados con múltiples capas, como las convolucionales y de pooling, se entrenan para clasificar y detectar la calidad de las manzanas directamente desde las imágenes recopiladas, lo que permite una evaluación precisa y automatizada de la salud del cultivo en etapas tempranas.

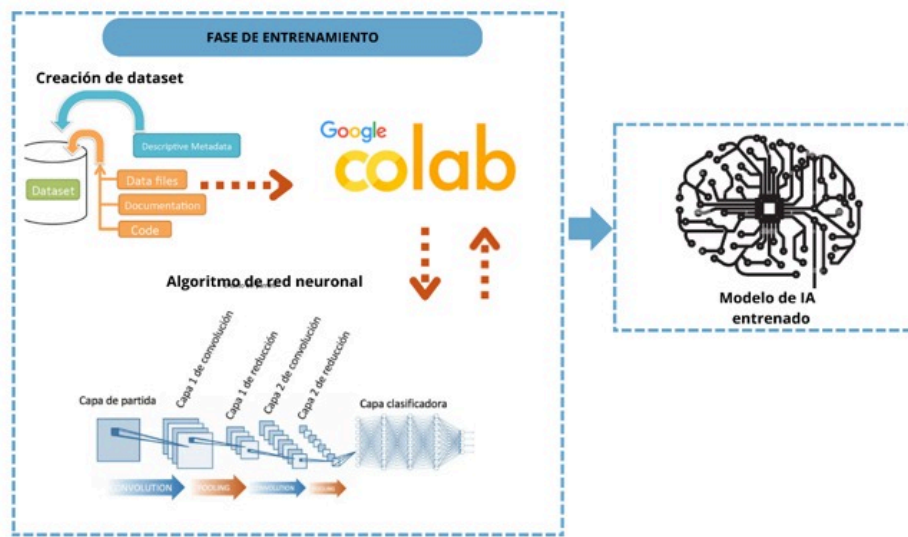


Figura 29. Arquitectura de la fase de entrenamiento

c. *Etapa 3: Procesamiento de datos*

Una vez completado el entrenamiento, el modelo de inteligencia artificial se desplegará en un dispositivo de computación de borde, que está optimizado específicamente para aplicaciones de inteligencia artificial en situaciones de campo. Este dispositivo compacto tiene la capacidad de ejecutar algoritmos de aprendizaje profundo avanzados, permitiendo la implementación de tareas de visión por computadora de manera autónoma y eficiente. Al operar en el borde de la red, el dispositivo minimiza la latencia, reduce la necesidad de ancho de banda y asegura la privacidad de los datos, al procesar la información localmente y entregar resultados de análisis en tiempo real. Esto es crucial para aplicaciones agrícolas donde las decisiones rápidas y basadas en datos pueden optimizar la gestión de los cultivos y mejorar los resultados de la precosecha.

d. *Etapa 4: Visualización y almacenamiento de datos*

En la fase de clasificación y detección, el modelo de inteligencia artificial, finamente ajustado, se aplica sobre el conjunto de imágenes adquiridas para realizar una evaluación exhaustiva de la calidad de las manzanas. Utilizando técnicas avanzadas de visión computacional, el modelo examina minuciosamente cada imagen para identificar y clasificar atributos críticos como la tonalidad y la uniformidad del color, así como para detectar y catalogar anomalías o defectos superficiales en la fruta.

Posteriormente, los datos procesados y las métricas de calidad obtenidas se sincronizan con una infraestructura de alojamiento web, donde se almacenan de manera segura en bases de datos cifradas. La información es accesible a través de una interfaz web protegida, garantizando la confidencialidad e integridad de los datos durante la transmisión. Esta interfaz proporciona capacidades analíticas interactivas y visualización en tiempo real, lo que permite a los usuarios finales, como agrónomos o gerentes de producción, tomar decisiones informadas basadas en datos precisos y actualizados sobre el estado de sus cultivos.

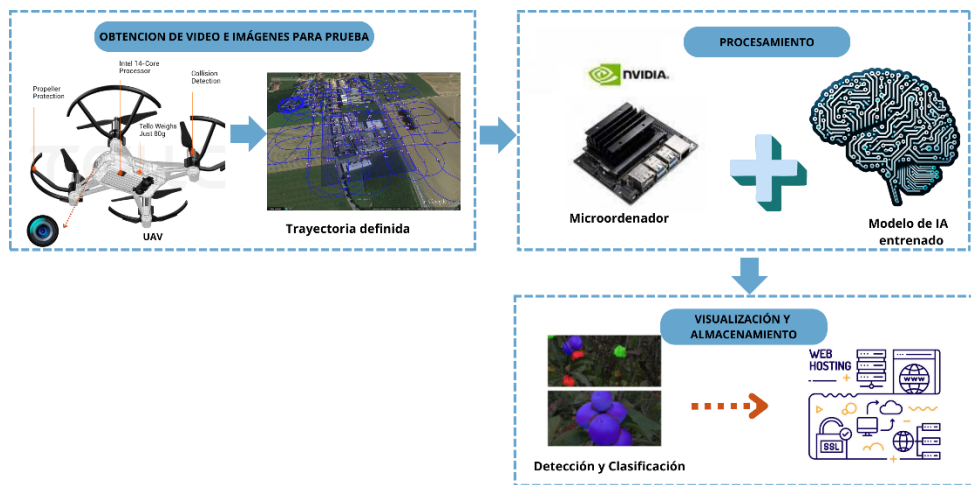


Figura 30. Arquitectura de las etapas de procesamiento y visualización del sistema

3.1.4 Selección de tecnologías

a. Selección de UAV

A continuación, en la Tabla 25 se realiza una comparación entre los diferentes UAV analizados anteriormente en el Capítulo I (Sistemas UAV usados en agricultura).

Para el sistema de control de calidad de cultivo de fruta de temporada en la etapa de precosecha que emplea visión artificial y planificación de trayectorias, el dron DJI Tello se ha seleccionado como la aeronave preferida en primera instancia por el precio y disponibilidad en el mercado ecuatoriano, además debido a su programabilidad y ligereza.

El Tello destaca por su capacidad de ser programado para realizar vuelos autónomos y tareas específicas, lo cual es esencial para la implementación de algoritmos de

control y visión artificial personalizados que son necesarios en el monitoreo preciso de cultivos. Además, su peso ligero de solo 80 gramos minimiza el riesgo de daño a los cultivos durante el vuelo bajo y las maniobras entre los árboles, lo que es crucial para la recolección de datos en la precosecha sin perturbar la integridad del fruto.

Tabla 25. Comparación de UAV [23], [24], [25].

Especificaciones	AR Drone 2.0	DJI Mavic 2	DJI Tello
Estructura	Cuadricóptero con estructura en cruz	Cuadricóptero	Cuadricóptero
Batería	1000 mAh, 11.1 V LiPo	No especificado, vuelo de 29 min	1100mAh/4.1Wh, 3.8V Li-ion polímero
Sensores	Inclinación y presión	Sistema de detección de obstáculos	Sensor de vision, acelerómetro, presión, ultrasonido
Cámara	HD 720p-30fps	Fotogramétrica HD	720p-30fps en MP4
Conectividad	WiFi	Control remoto, no WiFi	WiFi (IEEE 802.11n)
Almacenamiento	No dispone	8GB internos	No dispone
Tiempo de Vuelo	12 minutos	29 minutos	13 minutos
Alcance Máximo	50 metros	18 km	100 metros
Velocidad Máxima	5 m/s	20 m/s	8 m/s
Resolución de Video	720p a 30fps	720p a 30fps y 1080p a 30fps	720p a 30fps
Frecuencia	2.4 GHz	2.4 GHz y 5.8 GHz	2.4 GHz
Aplicaciones	Robótica e IA	Topografía y fotografía aérea HD	Recreacional, educativo
Dimensiones y Peso	525x515x121 mm, 420g	322x242x84 mm, 907g	92x98x41 mm, 80g con batería
Costo	\$ 300	\$ 1825	\$ 180

Estas características hacen del DJI Tello una herramienta valiosa para la recolección de datos agrícolas y la ejecución eficiente de tareas de control de calidad en un entorno de campo dinámico y en constante cambio.

b. *Selección de microcomputador*

A continuación, en la Tabla 26, se realiza la comparación de las características más importantes de los microcomputadores investigados en el Capítulo I (Microcomputador).

La elección de la NVIDIA Jetson Nano para el sistema de control de calidad de cultivos de fruta de temporada en la etapa de precosecha es estratégica debido a su capacidad para llevar a cabo tareas de inteligencia y visión artificial directamente en el campo. La Jetson Nano se destaca por su procesador ARM Cortex-A57 de cuatro núcleos y su

potente GPU NVIDIA Maxwell con 128 núcleos CUDA, proporcionando la capacidad de procesamiento necesaria para ejecutar complejos algoritmos de aprendizaje profundo y análisis de imágenes en tiempo real.

Además, su diseño optimizado para eficiencia energética es fundamental en entornos agrícolas donde el acceso a la electricidad puede ser limitado. La Jetson Nano es capaz de operar de forma autónoma, procesando datos a nivel local sin necesidad de conectividad constante a la nube, lo que es ideal en zonas rurales con acceso restringido a internet. Su soporte para cámaras CSI y diversas opciones de conectividad facilitan la integración con una gama de sensores y cámaras a bordo de UAVs, lo cual es crucial para la recolección y análisis de datos visuales en la precosecha.

Tabla 26. Comparación de microcomputadores [38], [39], [40].

Especificaciones	Raspberry Pi 5	NVIDIA Jetson Nano	AI-Motion K210
Arquitectura del Procesador	ARM Cortex-A76	ARM Cortex-A57	RISC-V Kendryte K210
Núcleos del Procesador	4 núcleos	4 núcleos	4 núcleos
Frecuencia del Procesador	2.4 GHz	1.43 GHz	800 MHz
Memoria RAM	4 GB y 8 GB SDRAM SKU	4 GB LPDDR4	No especificado
GPU	No especificado	NVIDIA Maxwell con 128 núcleos CUDA	No especificado (posiblemente GPU o NPU)
Conectividad	Gigabit Ethernet, Wi-Fi de doble banda, Bluetooth 5.0/LE, USB 3.0 y 2.0, HDMI dual	USB 3.0, Gigabit Ethernet, HDMI, microSD, CSI, DisplayPort	No especificado (sensores de movimiento incluidos)
Almacenamiento	Ranura para tarjeta microSD	Ranura para tarjeta microSD	Ranura para tarjeta microSD
Puertos de Video	HDMI dual 4kp60 con soporte HDR	HDMI, DisplayPort	No especificado
Sistemas Operativos Soportados	Sistemas completos como Linux	Linux (Ubuntu)	Linux Android FreeRTOS
Aplicaciones	Desarrollo de software, IoT, educación, IA	Inteligencia artificial, aprendizaje profundo, robots, drones	Análisis de movimiento, aprendizaje profundo, visión por computadora
Potencia de Cómputo	No especificado	GPU NVIDIA Maxwell con 128 núcleos CUDA	Procesador Quad-Core de 64 bits a 800 MHz
Funcionalidades Especiales	Amplia gama de opciones de conectividad, soporte para Wi-Fi y Bluetooth	Soporte para cámaras CSI, eficiencia energética en tareas de IA	Sensores de alta precisión, optimización para algoritmos de movimiento

c. Periféricos de entrada y salida

Monitor: Periférico de salida que muestra visualmente la información generada por la computadora, incluyendo texto, imágenes, gráficos y vídeos, proporcionando una interfaz visual para la interacción del usuario con la máquina.

Mouse: Periférico de entrada que detecta movimientos físicos y los traduce en movimientos del cursor en la pantalla, ofreciendo una interfaz intuitiva para la interacción con la computadora.

Teclado: Dispositivo de entrada que permite al usuario introducir datos mediante la pulsación de teclas, incluyendo letras, números, símbolos y comandos, facilitando la comunicación con la computadora.

d. Software de programación

En la selección de un lenguaje de programación para el control de calidad de cultivos de fruta de temporada utilizando robótica aérea y visión artificial en la etapa de precosecha, Python, JavaScript, y MATLAB representan tres opciones viables, cada una con sus propias fortalezas en el contexto de inteligencia artificial.

Tabla 27. Comparación de lenguajes de programación usados en Inteligencia Artificial[49], [50], [51].

Características	Python	JavaScript	MATLAB
Enfoque Principal	Versátil en ciencia de datos y IA	Desarrollo web y aplicaciones interactivas	Ingeniería y análisis numérico
Bibliotecas/Frameworks en IA	Excelente (TensorFlow, PyTorch)	Suficiente para tareas ligeras (TensorFlow.js)	Bueno para análisis numérico (Toolboxes especializadas)
Facilidad de Uso	Muy alta	Media	Alta en aplicaciones de ingeniería
Aplicaciones Comunes	Aprendizaje profundo, visión por computadora	Aplicaciones web en tiempo real	Simulaciones, procesamiento de señales
Integración de Sistemas	Excelente con hardware de IA	Limitada a web y Node.js	Buena con hardware y sistemas embebidos
Costo	Gratuito	Gratuito	Licencia costosa

Python fue seleccionado como el lenguaje de programación para el sistema de control de calidad de cultivos de fruta de temporada, aprovechando su robustez en el manejo de visión artificial y aprendizaje automático. Con librerías como OpenCV,

TensorFlow y PyTorch, Python facilita desde el procesamiento y análisis de imágenes capturadas por drones hasta la implementación de complejos modelos de redes neuronales que clasifican la calidad de las frutas. Esta capacidad es crucial para identificar características visuales detalladas, tales como tamaño, color y defectos, que son determinantes en la evaluación de la calidad. Además, Python ofrece herramientas que permiten la manipulación eficiente de grandes volúmenes de datos, esencial para el procesamiento en tiempo real necesario en la precosecha.

La integración de Python con hardware de procesamiento en el borde, como la NVIDIA Jetson Nano, permite ejecuciones autónomas de modelos de IA directamente en el campo, eliminando la dependencia de conectividad continua a la nube y reduciendo latencias. Su sintaxis intuitiva y la amplia comunidad de desarrolladores aceleran el desarrollo y facilitan el mantenimiento y la escalabilidad del sistema, mientras que su naturaleza de código abierto ofrece una solución coste-efectiva. Estas características hacen de Python una opción ideal para desarrollar un sistema avanzado y eficiente que mejora significativamente la precisión y la confiabilidad del control de calidad de cultivos utilizando tecnologías avanzadas de robótica aérea.

e. Plataforma basada en la nube

Para el desarrollo del proyecto se requiere de una plataforma basada en la nube para el entrenamiento del modelo de inteligencia artificial para la detección y clasificación del cultivo de manzana, estas plataformas son un entorno de computación que proporcionan servicios y recursos a través de internet, en lugar de depender de hardware o infraestructura local. En este contexto, los servicios y recursos, como el almacenamiento de datos, el procesamiento de datos, el desarrollo de aplicaciones y el alojamiento de sitios web se ofrecen y gestionan de forma remota, además, permiten a los usuarios acceder a sus aplicaciones y datos desde cualquier lugar con conexión a internet y suelen ofrecer una mayor flexibilidad, escalabilidad y eficiencia en comparación con las soluciones locales tradicionales. Ejemplos de plataformas basadas en la nube incluyendo Amazon Web Services (AWS), Microsoft Azure, Google Cloud Platform mismas que sus características principales se detallan en la Tabla 28.

Se ha optado por Google Colab ya que ofrece acceso gratuito a recursos computacionales de alto rendimiento, incluyendo GPUs y TPUs, lo cual es esencial para el entrenamiento y la implementación eficiente de modelos de visión artificial y aprendizaje profundo. La integración nativa de Colab con bibliotecas populares de aprendizaje automático como TensorFlow, PyTorch y OpenCV facilita el desarrollo de algoritmos de visión artificial necesarios para analizar las imágenes capturadas. La compatibilidad de Colab con Google Drive permite un almacenamiento y gestión de datos simplificados, asegurando que las grandes cantidades de datos de imágenes recolectadas pueden ser procesadas y almacenadas de manera eficiente.

Tabla 28. Comparación de plataformas basadas en la nube[52], [53], [54].

Característica	AWS	Microsoft Azure	Google Cloud Platform (GCP)
Entorno de Desarrollo	AWS Cloud9, Amazon SageMaker Studio	Azure Notebooks, Azure Machine Learning Notebooks	Google Colab, AI Platform Notebooks
Servicios de Machine Learning	Amazon SageMaker	Azure Machine Learning	Google Cloud Machine Learning Engine
Frameworks de Aprendizaje	TensorFlow, PyTorch, MXNet, entre otros	TensorFlow, PyTorch, ONNX Runtime, entre otros	TensorFlow, PyTorch, Scikit-learn, entre otros
Integración con otras herramientas	Integración con otros servicios de AWS como S3, EC2, y Redshift	Integración con otros servicios de Azure como Azure Blob Storage, Azure Databricks, y Azure Kubernetes Service	Integración con otros servicios de GCP como Google Cloud Storage, BigQuery, y Google Kubernetes Engine
Herramientas de automatización	AWS AutoML, AWS Glue, AWS Lambda, entre otros	Azure AutoML, Azure Databricks, Azure Functions, entre otros	Cloud AutoML, AI Platform Pipelines, entre otros
Acceso gratuito a GPU/ Colaboración en Tiempo real	Si	Si	Si
Almacenamiento en la nube	Amazon S3 (Simple Storage Service)	Azure Blob Storage	Google Cloud Storage

f. Arquitectura de visión artificial

Las Redes Neuronales Convolucionales (CNNs por sus siglas en inglés de Convolutional Neural Networks) son un tipo de redes neuronales profundas que se utilizan principalmente para procesar datos que tienen una topología de cuadrícula, como imágenes. Una CNN efectivamente captura la dependencia espacial entre los

pixeles al aprender características visuales jerárquicamente a través de sus diversas capas[34].

La selección de una arquitectura óptima de red neuronal para aplicaciones de visión artificial, como el sistema de control de calidad de cultivo de manzanas en la etapa de precosecha utilizando robótica aérea con planificación de trayectorias, es crucial debido a las exigencias específicas de precisión y velocidad en la captura y procesamiento de datos visuales. En este contexto, el equilibrio entre velocidad y precisión determina la eficacia con la que el sistema puede identificar problemas de calidad en los cultivos y ejecutar acciones correctivas en tiempo real. A continuación, en la Tabla 29 se detallan características de las principales CNN a analizar.

Tabla 29. Comparación de Redes Neuronales Convolucionales[55], [56], [57].

Característica	YOLO (You Only Look Once)	SSD (Single Shot MultiBox Detector)	Faster R-CNN
Velocidad	Muy rápida, adecuada para aplicaciones en tiempo real.	Rápida, casi comparable a YOLO.	Más lenta debido al paso de propuestas de región
Precisión	Alta, con mejoras significativas en versiones recientes.	Alta, especialmente eficaz en detectar objetos de tamaños variados.	Muy alta, considerada como uno de los benchmarks de precisión.
Uso de Recursos	Moderado, optimizado para análisis rápido.	Moderado, más eficiente que Faster R-CNN en tiempo real.	Alto, debido a la necesidad de computar propuestas de regiones.
Complejidad	Menor que Faster R-CNN, manejo simple de cuadrículas.	Complicada por el uso de múltiples cajas y escalas.	Alta, involucra múltiples etapas y refinamiento de propuestas.
Detección en Tiempo Real	Excelente para tiempo real debido a su velocidad.	Muy buena, con capacidad de uso en tiempo real.	No ideal para tiempo real debido a su velocidad de procesamiento.
Aplicaciones Típicas	Vigilancia, vehículos autónomos, robótica.	Robótica móvil, sistemas embebidos, aplicaciones IoT.	Investigación académica, aplicaciones que requieren alta precisión.
Facilidad de Entrenamiento	Relativamente fácil de entrenar en comparación con Faster R-CNN.	Requiere ajuste fino dependiendo de los tamaños de los objetos.	Requiere más tiempo debido a la necesidad de ajustar propuestas de regiones.

El enfoque en arquitecturas como YOLO (You Only Look Once) y SSD (Single Shot MultiBox Detector) es crucial para aplicaciones de detección de objetos en tiempo real debido a su capacidad de procesar imágenes rápidamente y con alta precisión utilizando un solo paso adelante por imagen. YOLO redefine el problema de detección

de objetos como una regresión simple, directa desde la imagen completa a las coordenadas del cuadro delimitador y las probabilidades de las clases, lo que permite que la detección sea extremadamente rápida y adecuada para aplicaciones en tiempo real como la vigilancia por video y los vehículos autónomos.

Las versiones del algoritmo YOLO representan avances significativos en detección de objetos en imágenes y videos en tiempo real. Desde su primera versión, YOLO ha evolucionado continuamente para mejorar la precisión, eficiencia y velocidad de detección. Cada versión introduce mejoras en la arquitectura del modelo y técnicas avanzadas de procesamiento detalladas en la Tabla 30.

Tabla 30. Comparación de versiones de YOLO[55].

Versión	Año de lanzamiento	Principales Mejoras	Descripción
YOLO	2015	Alta velocidad y precisión.	Modelo de detección y segmentación de objetos desarrollado por Joseph Redmon y Ali Farhadi en la Universidad de Washington
YOLOv2	2016	Normalización por lotes, cajas de anclaje, agrupaciones de dimensiones.	Mejoras en el modelo original incorporando normalización por lotes, cajas de anclaje y agrupaciones de dimensiones.
YOLOv3	2018	Red de backbone más eficiente, múltiples anclajes y agrupación de pirámide espacial.	Mejora el rendimiento del modelo utilizando una red de backbone más eficiente, múltiples anclajes y agrupación de pirámide espacial.
YOLOv4	2020	Aumento de datos Mosaic, cabeza de detección sin anclaje, nueva función de pérdida.	Introduce innovaciones como el aumento de datos Mosaic, una nueva cabeza de detección sin anclaje y una nueva función de pérdida.
YOLOv5	2020	Optimización de hiperparámetros, seguimiento de experimentos integrado, exportación automática a formatos populares.	Mejora el rendimiento del modelo e incluye nuevas características como la optimización de hiperparámetros, el seguimiento de experimentos integrado y la exportación automática a formatos populares.
YOLOv6	2022	Uso en robots de entrega autónomos de Meituan.	Versión de código abierto utilizada en muchos de los robots de entrega autónomos de Meituan.
YOLOv7	2022	Estimación de pose en el conjunto de datos de puntos clave COCO.	Añade tareas adicionales como la estimación de pose en el conjunto de datos de puntos clave COCO.
YOLOv8	2023	Mejoras en rendimiento, flexibilidad y eficiencia, soporte para tareas de IA de visión completas.	Última versión de YOLO por Ultralytics. Modelo de vanguardia que introduce nuevas características y mejoras para un rendimiento, flexibilidad y eficiencia mejorados. Soporta una gama completa de tareas de visión AI, incluyendo detección, segmentación, estimación de pose, seguimiento y clasificación.

Por su parte, SSD mejora la precisión en la detección sobre YOLO al utilizar múltiples cajas de referencia en varias capas de la red, lo que permite detectar objetos de diferentes tamaños de manera más efectiva y eficiente. Ambas arquitecturas optimizan el equilibrio entre velocidad y precisión, haciéndolas ideales para escenarios donde los recursos computacionales son limitados y la respuesta rápida es esencial, en la Tabla 31 se analizan las características principales de estas dos arquitecturas.

YOLOv8 es la elección ideal para el sistema de control de calidad de cultivos en la etapa de precosecha debido a su excepcional precisión en la detección de objetos, capacidad para operar en tiempo real y eficacia en la identificación de detalles pequeños, como defectos y signos de enfermedad en las frutas.

Tabla 31. Comparación SSD MobileNet vs. YOLOv8 [55], [56].

Aspecto	SSD MobileNet	YOLOv8
Estructura de Red	Detector de múltiples cajas en una sola pasada con MobileNet	Octava versión de You Only Look Once
Fecha de Introducción	SSD en 2016 y MobileNet v1 en 2017	2023
Exactitud	Elevada, mejora con ajustes adicionales	Excepcionalmente alta, destacando en evaluaciones
Rapidez	Alta, diseñada para dispositivos móviles	Alta, calibrada para operaciones en tiempo real
Demanda de Hardware	Reducida, optimizada para dispositivos con recursos limitados	Moderada, necesita unidades de procesamiento gráfico robustas
Presencia en el Mercado	Predominantemente en dispositivos móviles	Seleccionado para tareas de precisión crítica
Plataformas Compatibles	Compatible con TensorFlow, Keras y PyTorch (mediante conversiones)	Principalmente PyTorch, compatible con TensorFlow
Utilización	Implementado para detección en tiempo real en móviles e IoT	Empleado en sistemas de vigilancia y vehículos autónomos en tiempo real
Peso del Modelo	Ligero, generalmente bajo los 30 MB	Pesado, en el rango de los cientos de MB
Proceso de Entrenamiento	Requiere relativamente pocos datos, entrenamiento rápido	Demanda una cantidad considerable de datos y más tiempo para el entrenamiento
Capacidad de Detección	Competente en identificar objetos, con algunas restricciones en objetos diminutos	Extremadamente competente, eficaz con objetos de pequeño tamaño
Optimización	Afinado para maximizar la velocidad y minimizar el tamaño del modelo	Afinado para una precisión y velocidad óptimas

Su compatibilidad con plataformas populares como PyTorch y TensorFlow, junto con su optimización para un equilibrio óptimo entre velocidad y precisión, lo hacen especialmente adecuado para aplicaciones agrícolas automatizadas.

Además, aunque requiere hardware más robusto, su capacidad para aprovechar GPUs permite un análisis detallado y eficiente, garantizando un control de calidad altamente fiable y rápido.

g. Bibliotecas y Algoritmos de IA utilizados con YOLOv8

Para el diseño de la red neuronal destinada al sistema de control de calidad de cultivo de fruta de temporada, se han aplicado los siguientes algoritmos.

- ***Ultralytics.*** proporciona la tecnología subyacente para la detección y clasificación de objetos mediante el uso de modelos YOLO avanzados, lo cual es fundamental para las tareas de visión por computadora que estás ejecutando[55].
- ***Simple Online and Realtime Tracking (SORT).*** Es un algoritmo de seguimiento de objetos en tiempo real que utiliza las detecciones de objetos (en este caso, proporcionadas por YOLOv8) para seguir sus movimientos a través de los cuadros de un vídeo. Utiliza un enfoque simple basado en la asociación de datos de cuadro a cuadro usando la superposición de cajas delimitadoras y la estimación del estado del objeto[55].
- ***Segmentación por color HSV.*** Es un modelo de color que representa colores en términos de su tono (Hue), saturación (Saturation) y valor (Value, que a veces se refiere a la luminosidad o brillo). Este modelo es particularmente útil para tareas de procesamiento de imágenes que involucran colores porque separa la información de color (tono) de la información de intensidad (valor), facilitando la manipulación y segmentación basada en el color[58].
- ***OpenCV.*** Es una biblioteca de visión por computadora que proporciona herramientas para el procesamiento de imágenes y vídeo, incluyendo la lectura y escritura de imágenes, transformaciones geométricas, conversión de espacio de color, y muchas otras operaciones. Se usa para manipulaciones de imagen como convertir imágenes a espacio de color HSV, aplicar máscaras de color, y visualizar los resultados de las detecciones y clasificaciones[59].
- ***Numpy.*** Aunque no es una biblioteca de IA, NumPy es esencial para el manejo eficiente de matrices numéricas en Python, que es una parte fundamental de casi todos

los procesos de visión por computadora y aprendizaje automático. Se usa para definir los rangos de color en el espacio HSV mediante arrays de NumPy. Estos arrays especifican los valores mínimos y máximos para el tono, la saturación y el brillo (valor) que se utilizan para segmentar las manzanas por color[60].

h. Base de datos

En la Tabla 32, se presenta una comparación de distintos sistemas de gestión de datos destacando SQLite, MySQL y MongoDB. La comparación se enfoca en varios criterios relevantes como el lenguaje que se usa y la compatibilidad con aplicaciones de Inteligencia Artificial y aprendizaje automático.

Tabla 32. Comparación de sistemas de gestión de base de datos[61], [62], [63].

Característica	SQLite	MySQL	MongoDB
Tipo de DBMS	Embebido	Relacional	NoSQL
Lenguaje de implementación	C	C, C++	C++
Lenguaje de consulta	SQL	SQL	MQL
Flexibilidad	Alta, sin configuración de servidor	Moderada, requiere configuración de servidor	Esquema Flexible
Escalabilidad	Ideal para aplicaciones pequeñas	Soporta grandes números de datos	Excelente manejo de big data
Uso de recursos	Bajo	Alto	Muy Alto
Compatibilidad	Integración con Python y librerías de IA/ML	Integración de bibliotecas de IA/ML	Compatible con frameworks de Inteligencia Artificial

SQLite es la elección óptima para gestionar los datos en un sistema de control de calidad de manzanas con IA implementado en un Jetson Nano debido a su naturaleza de DBMS embebido, que elimina la necesidad de un servidor de base de datos externo y minimiza la complejidad de configuración. Esta característica es especialmente relevante en entornos de hardware limitado como el Jetson Nano, donde la eficiencia y la simplicidad son críticas. Además, SQLite ofrece una integración fluida con Python y bibliotecas de IA, permitiendo un desarrollo y despliegue eficientes de modelos de IA en la plataforma Jetson Nano. Su bajo uso de recursos garantiza un rendimiento óptimo y una gestión segura de los datos, mientras que su esquema flexible y dinámico proporciona la versatilidad necesaria para adaptarse a las necesidades cambiantes de un sistema de control de calidad en evolución.

i. Plataforma de desarrollo de interfaz web

En la selección de una plataforma de desarrollo de interfaz web para el control de calidad de cultivos de fruta de temporada utilizando robótica aérea y visión artificial en la etapa de precosecha, Flask, Django y FastAPI representan tres opciones viables, cada una con sus propias fortalezas en el contexto de inteligencia artificial

Tabla 33. Comparación de frameworks para interfaz web[64], [65], [66].

Característica	Flask	Django	FastAPI
Tipo de framework	Microframework para desarrollo web en Python, enfocado en simplicidad y extensibilidad.	Framework de alto nivel para desarrollo web en Python, ofrece un conjunto completo de características y funcionalidades.	Microframework para el desarrollo rápido de APIs web en Python, con un enfoque en la velocidad y el soporte a estándares modernos.
Lenguaje de programación	Python	Python	Python
Rendimiento	Bueno, adecuado para aplicaciones de tamaño mediano y tráfico moderado.	Bueno, aunque puede ser más lento en comparación con microframeworks debido a su mayor complejidad.	Muy bueno, optimizado para rendimiento y velocidad, especialmente para aplicaciones de API.
Autenticación	Flask-Login, Flask-HTTPAuth	Django Allauth, Django Rest Framework	FastAPI-Auth, PyJWT
Servidor de Aplicaciones	Werkzeug, uWSGI, Gunicorn	Django server	Uvicorn, Hypercorn
Plantillas	HTML Jinja2	Django Templates	HTML Jinja2
Base de Datos Relacionales	SQLite, MySQL, PostgreSQL	SQLite, MySQL, PostgreSQL	SQLite, MySQL, PostgreSQL

Para el desarrollo de del proyecto se elige Flask debido a su simplicidad y flexibilidad. Como microframework, Flask permite una configuración rápida y personalizada, adaptándose fácilmente a las necesidades específicas del proyecto. Su integración fluida con las principales bibliotecas de IA y ML de Python, como TensorFlow y scikit-learn, facilita la implementación y despliegue de modelos de inteligencia artificial. Flask permite una configuración sencilla y personalizada, adaptándose a las necesidades específicas del proyecto sin imponer estructuras rígidas. Su integración con herramientas de front-end como HTML, CSS y JavaScript es fluida, lo que permite a los desarrolladores construir interfaces de usuario interactivas y atractivas.

3.1.5 Diagrama del sistema

En la Figura 31 se muestra el diagrama de procesos para el entrenamiento del modelo de inteligencia artificial.

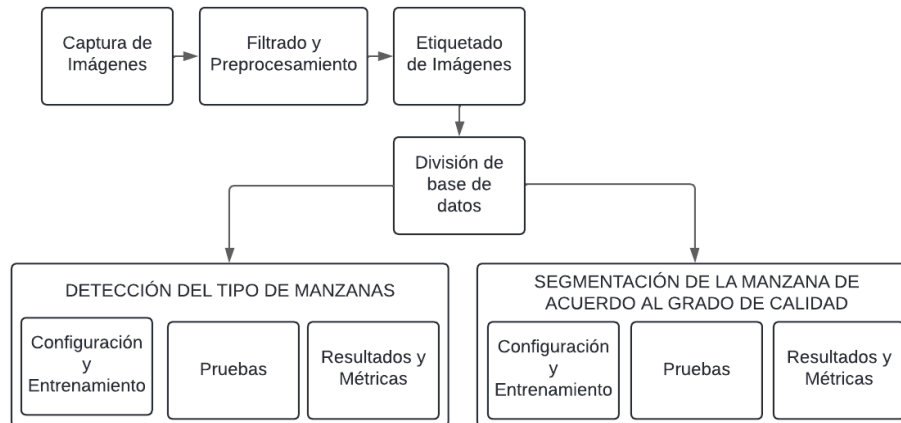


Figura 31. Diagrama general del proceso de experimentación del sistema

En la Figura 32 se detallan los bloques del sistema implementado:

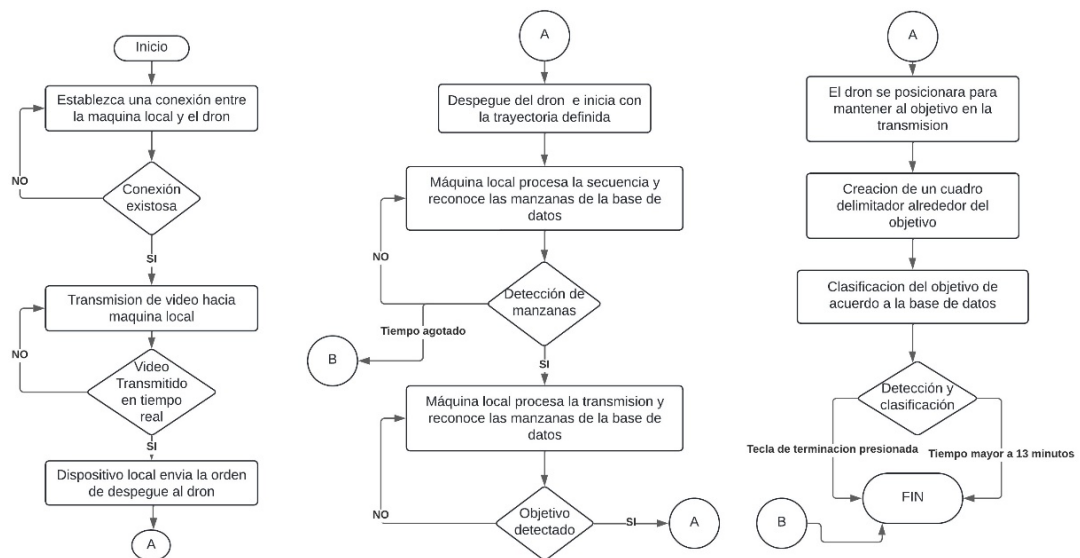


Figura 32. Diagrama de bloques del sistema

3.1.6 Diseño del sistema

Se utilizó Python como lenguaje de programación para implementar el modelo entrenado, con el entorno de desarrollo Visual Studio Code. A continuación, se

presenta un análisis detallado de las acciones realizadas en el código para la programación del sistema.

a. Bibliotecas requeridas

Las bibliotecas utilizadas para la implementación del sistema se detallan en la Tabla 34, presentada a continuación:

Tabla 34. Bibliotecas requeridas para el sistema

Biblioteca	Propósito Principal	Funciones Utilizadas	Características
Ultralytics	Detección de objetos mediante modelos YOLO	YOLO	- Implementación basada en PyTorch de YOLOv8 - Soporte para diferentes configuraciones de modelo y tamaños - Optimización para GPU con CUDA - Soporte para transformaciones de datos y aumento en tiempo de ejecución
OpenCV	Procesamiento de imágenes y vídeo	cv2.VideoCapture, cv2.cvtColor, cv2.inRange, cv2.bitwise_and, cv2.rectangle, cv2.putText	Extensa colección de algoritmos para procesamiento de imágenes, incluyendo algoritmos de filtrado, transformaciones geométricas, y análisis de imágenes. Su arquitectura basada en C/C++ garantiza un rendimiento óptimo.
NumPy	Manipulación y operaciones numéricas con arreglos y matrices	np.array, np.mean	- Biblioteca de alto rendimiento para cálculo numérico - Soporta operaciones vectorizadas y broadcasting - Gestión eficiente de memoria para grandes arrays numéricos
SORT	Seguimiento de objetos en tiempo real utilizando detecciones previas	tracker.update	Utiliza un filtro de Kalman para la predicción y actualización del estado de los objetos, minimizando el costo global de asignación para el seguimiento de múltiples objetos. Su diseño permite una integración eficiente con sistemas de detección.
djitellopy	Control y comunicación con drones DJI Tello	takeoff, land, streamon, get_frame_read	La biblioteca djitellopy ofrece una API simple y efectiva, diseñada para ser accesible para desarrolladores sin experiencia previa en drones. Soporta comandos para movimientos básicos y avanzados del dron, transmisión de video, y otras funcionalidades útiles en un contexto educativo o de desarrollo.

- *Inclusión de bibliotecas en la programación*

```
import threading
import time
from djitellopy import Tello
```

```
import cv2
from ultralytics import YOLO
import numpy as np
from sort import Sort
import os
import sqlite3
from flask import Flask, render_template, Response
```

b. Adquisición de imágenes

Este proceso constituye la fase de adquisición de datos, que es la primera etapa crítica en el flujo de trabajo de aplicaciones de inteligencia artificial, ya que provee el conjunto de datos que será utilizado para el entrenamiento del modelo de IA.

Para que el sistema capte imágenes están serán procesadas en tiempo real haciendo uso del UAV seleccionado siguiendo una serie de pasos que se muestra en la Figura 33 .

Se utiliza un dron equipado con protección de hélices, un procesador Intel de 14 núcleos y detección de colisiones. El dron pesa solo 1.8 kg. Se utiliza la planificación de trayectorias sobre el área de objetivo en este caso la plantación de manzanas ya que el UAV está equipado con una cámara de 5 megapíxeles y resolución 720HD para capturar imágenes.

- ***Inclusión de Bibliotecas***

```
from djitellopy import Tello
from threading import Thread
import cv2
import pygame
import numpy as np
import time
```

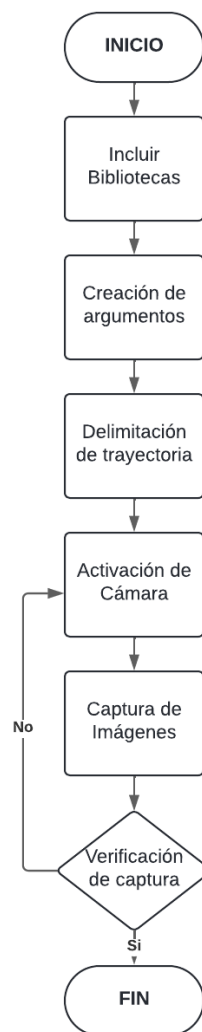


Figura 33. Diagrama de flujo del proceso de adquisición de imágenes

- ***Creación de argumentos***

Para la adquisición de imágenes se maneja con la biblioteca djitellopy, que es una interfaz para el dron Tello, y el módulo cv2, que es parte de OpenCV. Los métodos clave para la captura de cuadros y la grabación de video son:

- **self.tello.get_frame_read()**: Obtiene el objeto lector de cuadros.
- **frame_read.frame**: Accede al cuadro de video actual del dron.
- **self.start_video_recording()**: Inicia la grabación de video.

- **self.record_video_thread()**: Se ejecuta en un hilo separado para capturar cuadros y escribirlos en un archivo de video.

El método `record_video_thread()` codifica el video usando el códec XVID, establece los FPS y el tamaño del cuadro a (960, 720), y luego escribe los cuadros en el archivo 'videotello1.avi'. Cuando se detiene la grabación, se libera el objeto **VideoWriter**, finalizando el archivo de video.

- ***Delimitación de trayectoria***

- ***Ubicación***

En primera instancia el proyecto tiene lugar en un cultivo de manzana ubicado en el sector de la Manzana de Oro de la parroquia de Huachi Grande de la ciudad de Ambato, la ubicación aproximada se puede observar en la Figura 34.

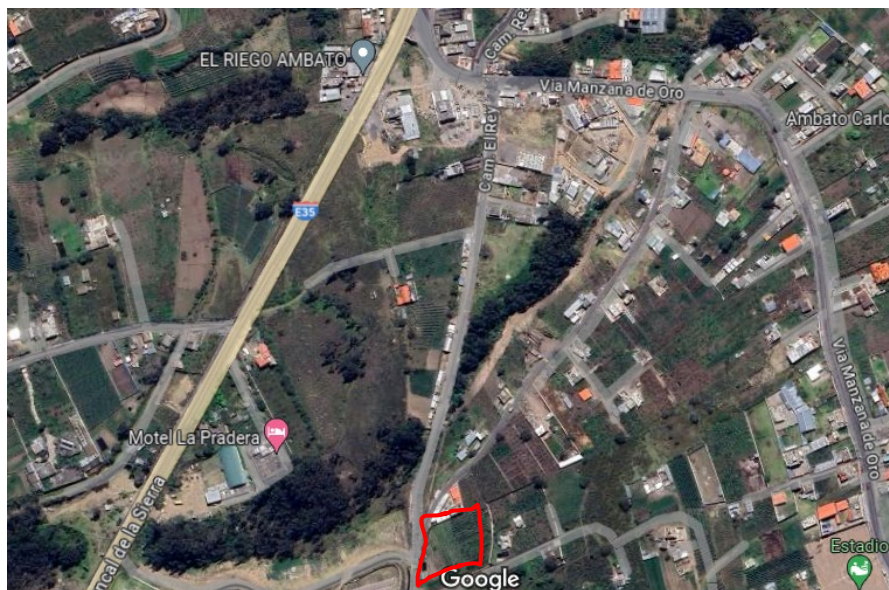


Figura 34. Mapa de ubicación del cultivo de manzana.

El cultivo de manzanas únicamente disponía de árboles con dos tipos de manzana, Red Delicious y Golden Delicious, pero con la particular característica que existían manzanas con los tres tipos de grado de calidad para analizar por lo que las imágenes capturadas por el dron fueron en su mayoría empleadas para el entrenamiento del modelo de clasificación de manzanas por grado de calidad.



Figura 35. Cultivo donde se realizará la planeación de trayectoria.



Figura 36. Cultivo de manzana del sector.

- *Mapeo de vuelo*

Para el mapeo de vuelo previamente se ha realizado una inspección visual del sector donde se encuentra el cultivo de manzana, se ha seleccionado una parte de toda la proporción topográfica y a su vez dentro de esto se ha delimitado la plantación donde se realizará la planeación de la trayectoria. Se ha marcado el área de interés donde se cubrirá parte de la plantación de manera eficiente como se observa en la Figura 37 , una vez que se tiene claro de forma visual cómo será la trayectoria esperada que el dron realice, se procedió a ajustar distancias y giros acorde a la topografía y ubicación de las parcelas por donde se ha planificado el vuelo.

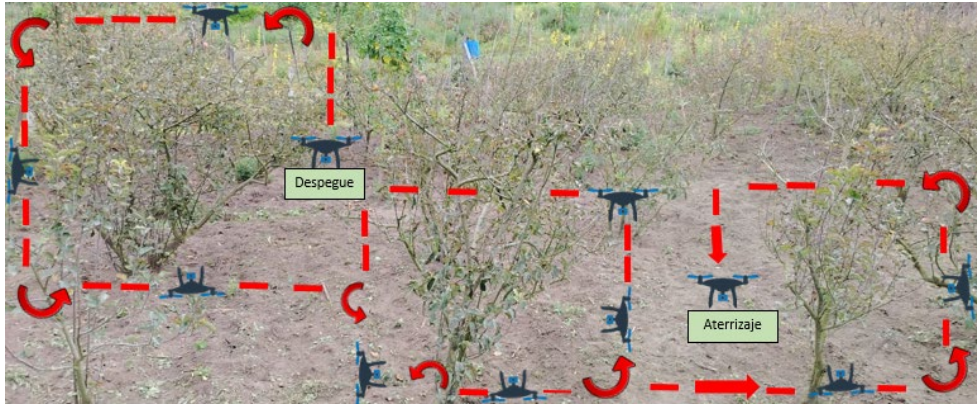


Figura 37. Trayectoria realizada por el dron en el cultivo.

- *Ajuste de distancias y giros para la trayectoria*

El dron no tenía una GPU para realizar cálculos de aprendizaje automático y análisis de imágenes. En su lugar, un ordenador local conectado al dron a través de WiFi ejecutaba un script de Python que automatiza el vuelo, no requiriendo intervención humana durante toda su duración. El SDK de Tello estaba conectado al dron a través de un puerto UDP de WiFi.

La conexión y control del dron Dji Tello se inicia mediante la red Wi-Fi generada por el propio dron, a la cual el dispositivo controlador se conecta. Utilizando la biblioteca djitellopy en Python, se establece la comunicación con el dron a través del comando `tello.connect()`. Tras un intervalo de espera de 5 segundos para asegurar una conexión estable, el dron despegue con el comando `tello.takeoff()`. El programa luego abre un archivo de texto denominado 'archivo.txt' y, mediante la lectura línea por línea, ejecuta comandos específicos basados en palabras clave (como 'ADELANTE', 'ATRÁS', 'DERECHA', 'IZQUIERDA', 'ROTIZ', 'ROTDER', 'ATERRIZAR'). Cada comando de movimiento es enviado al dron con pausa inmediata de 2 segundos para asegurar la correcta ejecución de cada acción. Si ocurre un error durante el proceso, se captura y se imprime un mensaje de error detallado.

Estructura del comando de desplazamiento y rotación

```
send_rc_control(left_right_velocity, forward_backward_velocity, up_down_velocity,
yaw velocity)
```

- **left_right_velocity:** Controla la velocidad hacia la izquierda o derecha. El valor positivo mueve al dron hacia la derecha, mientras que un valor negativo mueve el dron hacia la izquierda y el valor 0 indica que no se moverá en dicha dirección.
- **forward_backward_velocity:** Controla la velocidad hacia delante o hacia atrás. Un valor positivo mueve al dron hacia adelante, mientras que un valor negativo lo moverá hacia atrás.
- **up_down_velocity:** Controla la velocidad hacia arriba o hacia abajo, Un valor positivo mueve el dron hacia arriba, mientras que un valor negativo lo mueve hacia abajo.
- **yaw velocity:** Controla la velocidad de rotación (yaw) hacia la izquierda o hacia la derecha. Un valor positivo rota el dron hacia la derecha en sentido de las agujas del reloj, mientras que un valor negativo lo rota hacia la izquierda en sentido contrario a las manecillas del reloj.

En la Tabla 35 se detalla comandos importantes que controla el Dji Tello en la ruta establecida:

Tabla 35. Comandos de control DJI Tello.

Comando	Acción	Detalles
takeoff	Despegue	El dron despegue y comienza a volar
land	Aterrizaje	El dron aterriza y finaliza el vuelo
send_rc_control(0, 50, 0, 0)	Adelante	El dron se mueve hacia adelante a 50cm/s
send_rc_control(0, -50, 0, 0)	Atrás	El dron se mueve hacia atrás a -50 cm/s
send_rc_control(50, 0, 0, 0)	Derecha	El dron se mueve hacia la derecha a 50 cm/s
send_rc_control(-50, 0, 0, 0)	Izquierda	El dron se mueve hacia la izquierda a -50 cm/s
send_rc_control(0, 0, 50, 0)	Arriba	El dron se mueve hacia arriba a 50 cm/s
send_rc_control(0, 0, -50, 0)	Abajo	El dron se mueve hacia abajo a -50 cm/s
send_rc_control(0, 0, 0, -45)	Rotación Izquierda	El dron se mueve hacia la izquierda a -45 grados/s
send_rc_control(0, 0, 0, 45)	Rotación Derecha	El dron se mueve hacia la derecha a 45 grados/s

Para calcular la distancia que recorre en cada tramo el dron se hace uso de la siguiente formula:

$$\textit{Distancia} = \textit{Velocidad} \times \textit{Tiempo} \quad (1)$$

Velocidad lateral (eje Y) = 50cm/s

Tiempo de ejecución = 2s

$$\textit{Distancia} = \frac{50 \text{ cm}}{\text{s}} \times 2 \text{ s} = 100 \text{ cm}$$

Por lo que la distancia recorrida lateralmente es 100 cm en cada tramo.

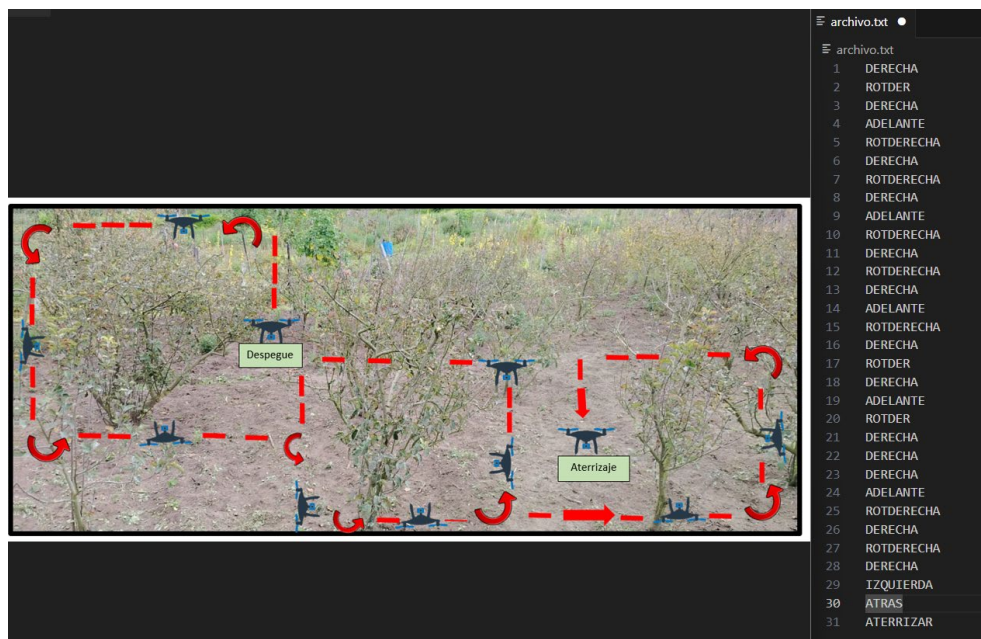


Figura 38. Comandos registrados en un archivo .txt de la trayectoria.

- **Activación de la cámara**

La cámara del dron Tello se activa mediante el método **streamon()** de la biblioteca **djitellopy**. La secuencia para la activación de la cámara es la siguiente:

```
self.tello.connect()

self.tello.set_speed(self.speed)
```

```
self.tello.streamoff()

self.tello.streamon()

frame_read = self.tello.get_frame_read()
```

Se establece la conexión con el dron usando `self.tello.connect()`. Antes de activar la transmisión de video, se asegura que cualquier flujo anterior esté desactivado con `self.tello.streamoff()`. La cámara se activa con `self.tello.streamon()`. Esto inicia el flujo de video desde la cámara del dron.

- ***Captura de imágenes***

```
frame_read = self.tello.get_frame_read()
```

Para leer los cuadros (frames) transmitidos, se utiliza `self.tello.get_frame_read()`, que devuelve un objeto a través del cual se pueden obtener continuamente los cuadros del video en un hilo aparte, sin bloquear el programa principal.

- c. Procesamiento de imágenes***

Para que el sistema procese las imágenes capturadas por el dron que serán de uso primordial para el entrenamiento del modelo de inteligencia artificial a usar, se sigue una serie de pasos como se muestra en la Figura 39. para la detección del tipo de manzana y la determinación del grado de calidad.

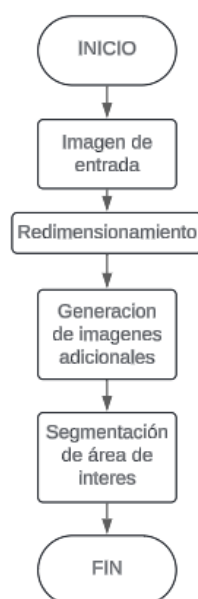


Figura 39. Diagrama de flujo del procesamiento de imágenes

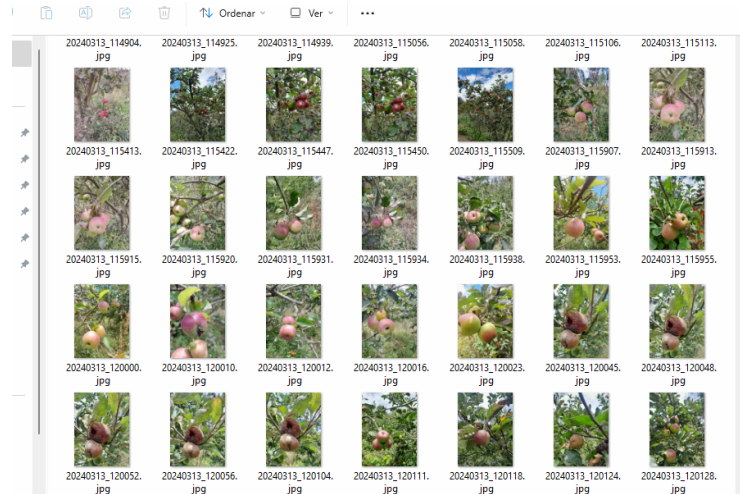


Figura 40. Imágenes para procesamiento.

- **Redimensionamiento**

Para no sobrecargar los recursos computacionales al momento de realizar el entrenamiento del modelo de IA se redimensionan las imágenes capturadas a un formato de 640 x 640 pixeles como se detalla en la Figura 41 .

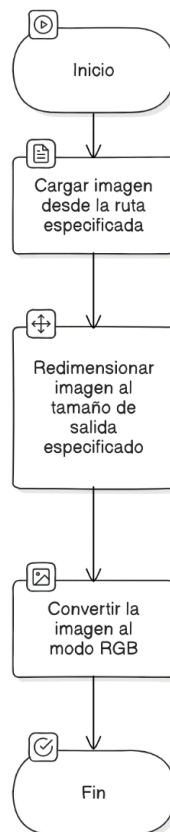


Figura 41. Proceso de redimensionamiento de imagen

- **Generación de imágenes adicionales**

Un proceso adicional dentro del programa de redimensionamiento de imágenes se aplican generadores de imagen a partir de la original cambiando características como rotación, brillo, saturación, contraste, etc., para la creación de un data set extenso para un mejor entrenamiento.

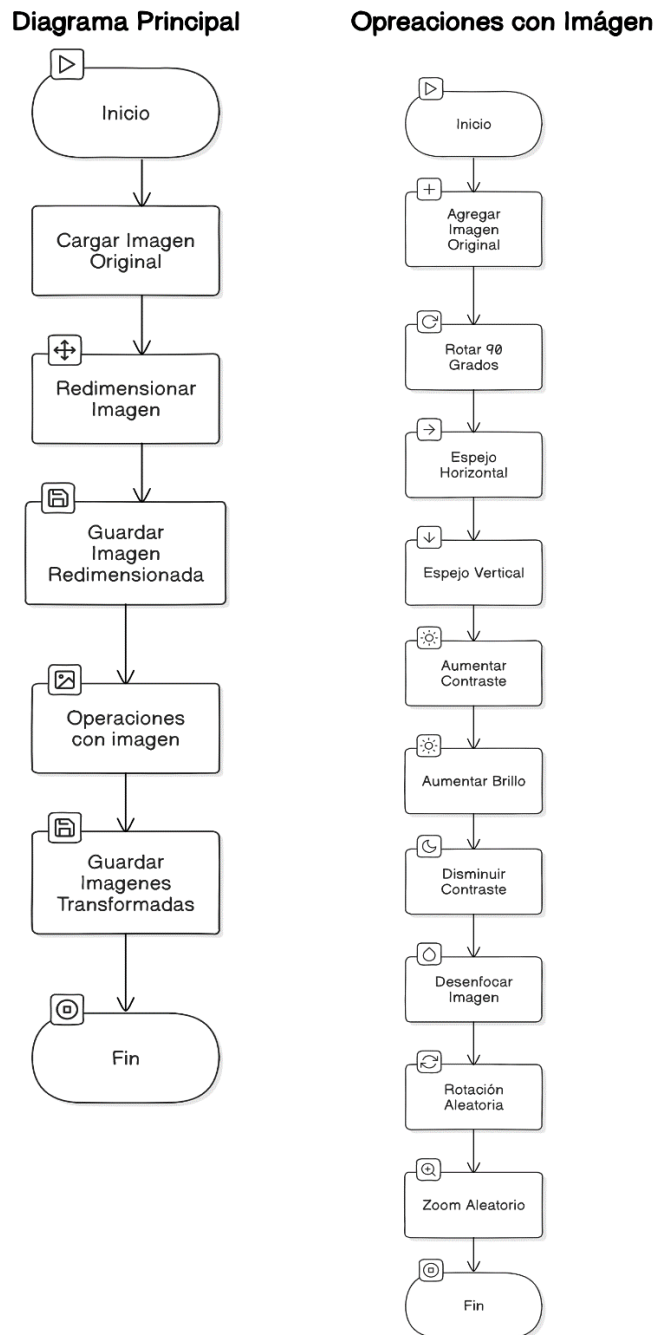


Figura 42. Proceso de transformación de imágenes.

- ***Segmentar área de interés***

En esta sección se hizo uso de LabelMe que es una herramienta de anotación gráfica de imágenes que permite a los usuarios etiquetar objetos en imágenes con un contorno poligonal. El proceso de segmentación en LabelMe es bastante interactivo y detallado, lo que permite una precisión considerable al definir los límites de los objetos. Como se puede observar en la Figura 43 el área de interés se limita a daños que afectan en el grado de calidad del fruto de manzana.

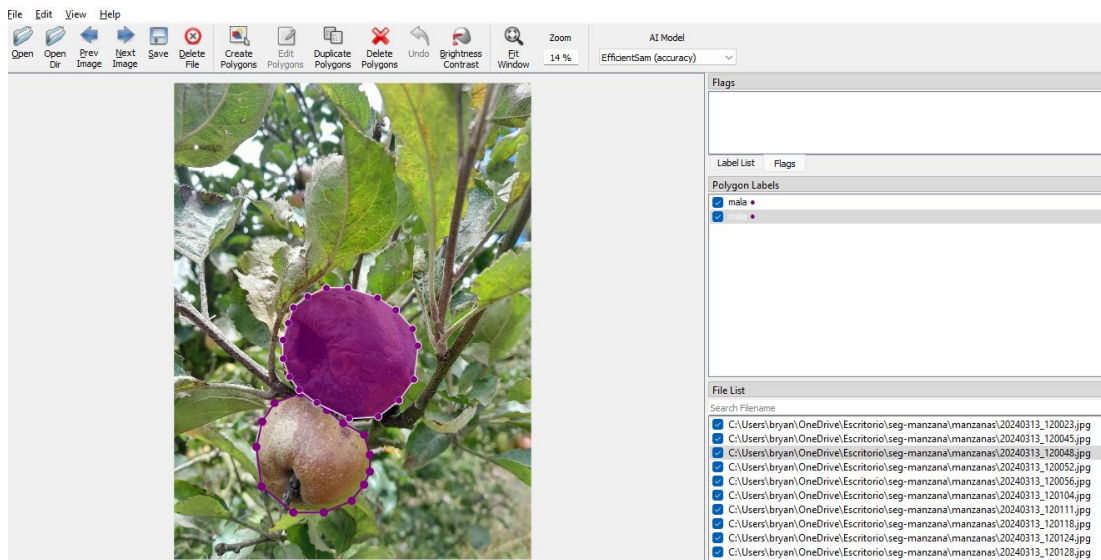


Figura 43. Etiquetado en el software LabelMe

d. Modelo de inteligencia Artificial: YOLO V8

- ***Modelo No Secuencial de Alto Rendimiento.*** A diferencia de un modelo secuencial típico, YOLOv8 implementa una estructura de red neuronal compleja que integra elementos de diferentes arquitecturas avanzadas. Este modelo es conocido por su rendimiento óptimo en tareas de detección de objetos en tiempo real, capaz de procesar imágenes y vídeo a alta velocidad con una precisión notable.

- ***Modelo Convencional Profundamente Conectado.*** YOLOv8 utiliza una arquitectura de red neuronal convolucional (CNN) que comprende múltiples capas, incluyendo capas de convolución, capas de pooling y capas densas al final para la clasificación y localización precisas de objetos. Esta estructura multicapa permite que

el modelo procese desde la entrada visual hasta la salida de las coordenadas de las cajas delimitadoras y las probabilidades de las clases en una sola pasada, haciendo uso de la inferencia directa sin la necesidad de etapas de post-procesamiento separadas.

- ***Aprendizaje Profundo Avanzado.*** La elección de utilizar aprendizaje profundo en YOLOv8 se justifica por su capacidad para construir y entrenar redes neuronales que manejan tareas complejas de detección de objetos. Las redes profundas en YOLOv8 son capaces de extraer y aprender representaciones ricas y detalladas de los datos visuales, lo cual es esencial en campos que requieren un análisis visual intensivo como la visión por computadora y el reconocimiento de patrones. Esto incluye, pero no se limita a, aplicaciones en seguridad, sistemas autónomos de navegación, análisis de tráfico y muchas otras áreas de la tecnología y la investigación.

- ***Optimización y Compatibilidad.*** YOLOv8 ha sido diseñado para ser compatible con una amplia gama de herramientas y plataformas de aprendizaje profundo, principalmente PyTorch. La arquitectura del modelo está optimizada para hacer uso eficiente de los recursos computacionales, lo que permite implementaciones en dispositivos con capacidades limitadas como smartphones y cámaras de seguridad, además de sistemas más robustos como servidores y plataformas en la nube.

e. Data set

En el contexto del desarrollo de un sistema para la detección y clasificación según el grado de calidad de cultivos de manzanas, se llevó a cabo una meticulosa búsqueda y selección de un repositorio de datos óptimo. Se dio prioridad a aquellos conjuntos de datos que ofrecen una amplia diversidad de imágenes, particularmente aquellas pertinentes a la evaluación de calidad de las manzanas. Este enfoque se adoptó con el objetivo de optimizar la capacidad del modelo para identificar de manera precisa y efectiva las características específicas de las manzanas, mejorando así su desempeño en tareas de reconocimiento y clasificación. La arquitectura de las bases de datos seleccionadas está específicamente organizada, como se detalla en la Figura 44, para soportar las etapas críticas del proceso de entrenamiento. Esta estructuración meticulosa facilita la implementación eficaz y el manejo de datos durante las fases esenciales de aprendizaje del modelo.

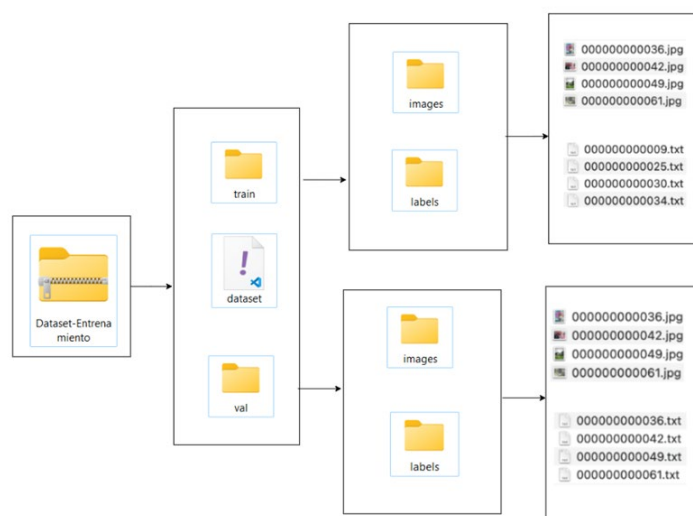


Figura 44. Arquitectura de la base de datos

La arquitectura de la base de datos está diseñada para facilitar el entrenamiento de modelos de aprendizaje automático y comprende dos componentes principales estructurados de forma específica. El primer componente, denominado "train", se desglosa en dos subdirectorios: "images" y "labels". En el subdirectorio "images", se almacenan representaciones visuales de entidades de interés, que sirven como entrada para el entrenamiento del modelo. Por otro lado, el subdirectorio "labels" incluye metadatos anotados, principalmente las coordenadas de las cajas delimitadoras, que son generadas mediante el uso de la herramienta de anotación Labelme, proporcionando así las etiquetas de verdad de campo necesarias para el aprendizaje supervisado.

El segundo componente es el directorio "val", que replica la estructura de "train" con subdirectorios "images" y "labels". Sin embargo, a diferencia del directorio de entrenamiento, las imágenes en "val" son exclusivamente para validación y no se exponen al modelo durante la fase de entrenamiento. Estas imágenes de validación son cruciales para evaluar la habilidad del modelo para generalizar a nuevos datos, actuando como un control para medir el desempeño del modelo bajo condiciones no observadas previamente. Este esquema de validación proporciona un indicador esencial para la evaluación objetiva y rigurosa de la capacidad predictiva y la robustez del modelo en escenarios operativos reales.

- **Características del primer data set para detección de manzana.** Durante el entrenamiento de la red neuronal convolucional, se identificó y utilizó un conjunto de datos específico, caracterizado por atributos claramente definidos que sirve para la detección de manzanas. Este conjunto de datos es reconocido como “COCO”. Este conjunto de datos tiene las siguientes características representadas en la Tabla 36.

Tabla 36. Características data set para detección

Características	Detalle
Numero de imágenes	Está compuesta por un total de 13000 imágenes
Dimensión	640X640
Parámetros	La base de datos contiene 41 clases de las cuales se usa 1 clase de manzanas

- **Características del segundo data set para control de calidad de manzana.** Con la finalidad de discernir la calidad de las manzanas, se ha creado una base de datos propia que abarca las siguientes categorías de daños: daños leves, daños moderados y daños severos. Este conjunto de datos tiene las siguientes características detalladas en la Tabla 37 y la información del cronograma para la obtención del data set se muestra en la Tabla 38.

Tabla 37. Características data set para control de calidad

Características	Detalle
Numero de imágenes	Esta compuesta por un total de 1500 imágenes
Resolución	640X640
Parámetros	Evalúa pequeños, medianos y grandes defectos

Tabla 38. Cronograma de creación de dataset

Periodo recolección de información	Fecha	Días	Hora
Marzo-Abril	13/03/2024 30/04/2024	3 veces por semana Lunes-Miercoles-Viernes	8:00 14:00 18:00

- **Rutas de entrenamiento del data set**

```
dataset_path = '/content/drive/MyDrive/data/dataset.yamll'

# Check if the dataset path exists
if not os.path.exists(dataset_path):
```

```
raise FileNotFoundError(f"Dataset path '{dataset_path}' does not exist.")
```

f. Arquitectura del modelo YOLOv8

Se ha usado la arquitectura de la red neuronal convolucional YOLOv8, en la Figura 45 se ilustra la arquitectura de esta red desglosada en sus componentes principales que son: Backbone, Head (usados para la detección de objetos) y Details. El Backbone utiliza diversas capas convolucionales y módulos C2f organizados en resoluciones de P1 a P5 para extraer características de imágenes, dentro de este existe el modulo SPPF que maneja diferentes escales y resoluciones. El Head integra y refina las características extraídas en el Backbone usando concatenaciones y upsampling para mejorar la precisión de la detección de objetos a múltiples escalas. Details describe capas específicas usadas en la arquitectura como convoluciones, normalización, concatenación, max pooling optimizando el flujo de gradientes y la robustez del modelo[67].

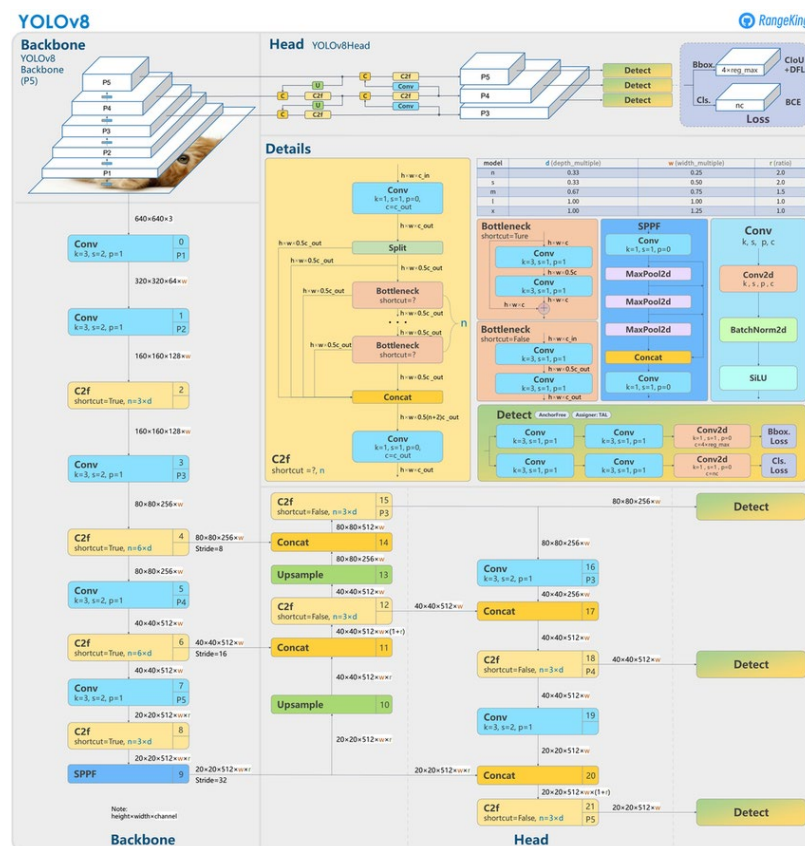


Figura 45. Arquitectura de YOLOv8.

- **Capas del modelo de red neuronal convolucional**

La arquitectura de la red neuronal convolucional (CNN) está meticulosamente estructurada en varias capas, comenzando con la capa de entrada y progresando a través de múltiples capas ocultas hasta llegar a la capa de salida. La capa de entrada recibe directamente los datos en forma de frames o imágenes capturadas por la cámara del dron.

En la Figura 46 se muestra una arquitectura de red neuronal para el sistema basada en YOLOv8, el flujo del proceso inicia con la entrada de datos que incluye información sobre la manzana, color y nivel de daño, además, contiene una capa oculta donde se realizan convoluciones para la extracción de características, una capa de salida y las salidas del modelo de red diseñado para la clasificación y evaluación del estado de las manzanas.

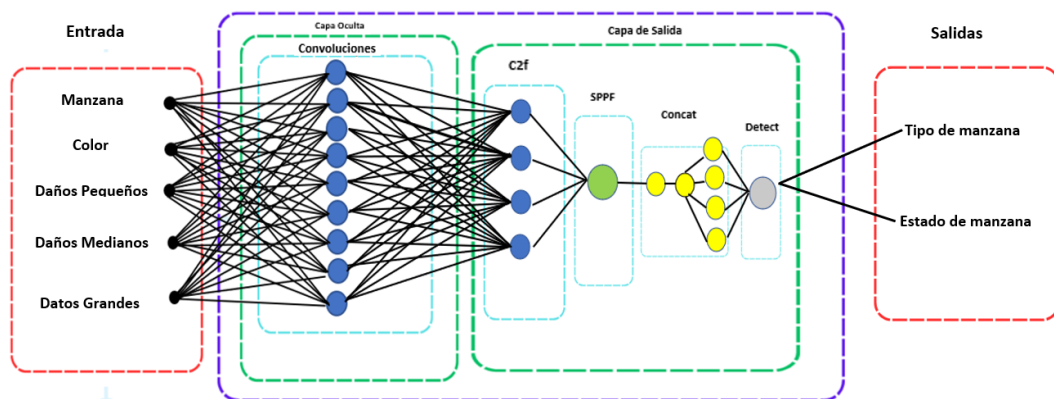


Figura 46. Capas del modelo de IA

- **Capa oculta (convoluciones):** se aplican varias capas de convolución actuando como filtros para la extracción de características, identificando patrones esenciales como bordes, texturas y formas de las imágenes de entrada.
- **Capa C2f (CSP Bottleneck with 2 convolutions):** se implementó 4 capas C2f que ayuda a mejorar la representación de las características extraídas reduciendo la complejidad computacional sin perder precisión.
- **Capa SPPF (Spatial Pyramid Pooling-Fast):** esta capa permite el manejo de imágenes de diferentes tamaños sin necesidad de que sean redimensionadas

agregando robustez a la red al combinar características de diferentes escalas y resoluciones.

- **Capa Concat:** se empleo 6 capas de concatenación para la fusión de las características extraídas de las capas anteriores para una mejor representación
- **Capa Detect:** en esta capa se realiza la tarea de identificar y clasificar objetos en la imagen de entrada.

g. Entrenamiento del modelo

El entrenamiento es una de las partes más importantes de la red neuronal, en el desarrollo del sistema, se empleó el modelo YOLOv8, proporcionado por Ultralytics. Este modelo utiliza técnicas avanzadas de visión por computadora para realizar tareas de detección en tiempo real. YOLOv8 cuenta con cinco variantes de modelo como se observa en la Figura 47, cada uno con diferentes modelos de precisión y velocidad, permitiendo seleccionar el modelo que más se adapte a las necesidades específicas de la aplicación[68].

Modelo	tamaño (pixeles)	mAPval 50-95	Velocidad CPU ONNX (ms)	Velocidad A100 TensorRT (ms)	parámetros (M)	FLOPs (B)
YOLOv8n	640	37.3	80.4	0.99	3.2	8.7
YOLOv8s	640	44.9	128.4	1.20	11.2	28.6
YOLOv8m	640	50.2	234.7	1.83	25.9	78.9
YOLOv8l	640	52.9	375.2	2.39	43.7	165.2
YOLOv8x	640	53.9	479.1	3.53	68.2	257.8

Figura 47. Variantes del modelo YOLOv8[68].

El proyecto demanda un cierto grado de complejidad y se pretende alcanzar la mayor eficiencia posible del sistema, se requiere más de un modelo para realizar la correcta detección y clasificación de manzanas, la arquitectura para el sistema será YOLOv8x, este modelo destaca por su alta precisión y capacidad para captar características detalladas y complejas. Esta versión está compuesta por 365 capas con un total de 68154534 parámetros[55].

```

0          from n  params module arguments
1          -1 1    2320 ultralytics.nn.modules.conv.Conv [3, 80, 3, 2]
2          -1 1   115520 ultralytics.nn.modules.conv.Conv [80, 160, 3, 2]
3          -1 3   436800 ultralytics.nn.modules.block.C2f [160, 160, 3, True]
4          -1 1   461440 ultralytics.nn.modules.conv.Conv [160, 320, 3, 2]
5          -1 6   3281920 ultralytics.nn.modules.block.C2f [320, 320, 6, True]
6          -1 1   1844480 ultralytics.nn.modules.conv.Conv [320, 640, 3, 2]
7          -1 6   13117440 ultralytics.nn.modules.block.C2f [640, 640, 6, True]
8          -1 1   3687680 ultralytics.nn.modules.conv.Conv [640, 640, 3, 2]
9          -1 3   6969600 ultralytics.nn.modules.block.C2f [640, 640, 3, True]
10         -1 1   1025920 ultralytics.nn.modules.block.SPPF [640, 640, 5]
11         -1 1      0 torch.nn.modules.upsampling.Upsample [None, 2, 'nearest']
12        [-1, 6] 1      0 ultralytics.nn.modules.conv.Concat [1]
13         -1 3   7379200 ultralytics.nn.modules.block.C2f [1280, 640, 3]
14         -1 1      0 torch.nn.modules.upsampling.Upsample [None, 2, 'nearest']
15        [-1, 4] 1      0 ultralytics.nn.modules.conv.Concat [1]
16         -1 3   1948800 ultralytics.nn.modules.block.C2f [960, 320, 3]
17         -1 1   922240 ultralytics.nn.modules.conv.Conv [320, 320, 3, 2]
18        [-1, 12] 1      0 ultralytics.nn.modules.conv.Concat [1]
19         -1 3   7174400 ultralytics.nn.modules.block.C2f [960, 640, 3]
20         -1 1   3687680 ultralytics.nn.modules.conv.Conv [640, 640, 3, 2]
21        [-1, 9] 1      0 ultralytics.nn.modules.conv.Concat [1]
22         -1 3   7379200 ultralytics.nn.modules.block.C2f [1280, 640, 3]
23        [15, 18, 21] 1 8719894 ultralytics.nn.modules.head.Detect [2, [320, 640, 640]]
YOLOv8x summary: 365 layers, 68154534 parameters, 68154518 gradients, 258.1 GFLOPs

```

Figura 48. Arquitectura de red neuronal YOLOv8x

- **Característica de máquina usada para entrenamiento**

Para lograr el mejor rendimiento del entrenamiento y en el menor tiempo posible se usó la herramienta Google Colab en su versión gratuita, que proporciona un entorno de notebook de Jupyter ejecutado en la nube y cuenta con una RAM de GPU de 15 GB, para ello se inició creando un nuevo notebook en este entorno asociado a GoogleDrive. En la Tabla 39, se detallan las características de la máquina de Google Colab usada para el entrenamiento.

Tabla 39. Parámetros de máquina usada en entrenamiento[69].

Parámetro	Especificación
CPU	CPU Intel Xeon a 2,20 GHz
RAM	13 GB
GPU	NVIDIA Tesla T4 o P100 (varía)
TPU	CPU Intel Xeon a 2,30 GHz
Memoria de GPU	15 GB
Sistema Operativo	Linux (Ubuntu)
Almacenamiento	78.2 GB
Librerías preinstaladas	NumPy, Pandas, Matplotlib, PyTorch, TensorFlow, Keras, etc.
Tiempo máximo de ejecución	12 horas

- **Código para el entrenamiento**

Como primer paso se cargó el dataset en formato “.rar” a la carpeta de GoogleDrive lo que asegura que el proceso de entrenamiento sea eficaz permitiendo si en determinado

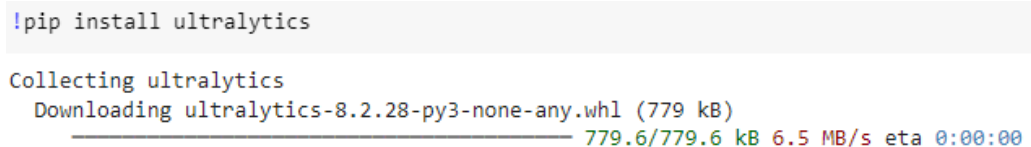
caso es necesario volver a ejecutar el notebook se pueda acceder al dataset directamente.

Una vez iniciado el notebook fue necesario ejecutar la siguiente celda para montar la unidad en Google Drive y así tener acceso a los datos almacenados en la nube.

```
from google.colab import drive
drive.mount('/content/drive')

import os
os.chdir('/content/drive/MyDrive')
```

Una vez montado el drive a Colab, con el comando **!pip install ultralytics** se instala la biblioteca de ultralytics, las librerías que este contiene y su instalación se detallan en la Figura 49 .



```
!pip install ultralytics

Collecting ultralytics
  Downloading ultralytics-8.2.28-py3-none-any.whl (779 kB)
    779.6/779.6 kB 6.5 MB/s eta 0:00:00
```

Figura 49. Resultados de instalación de Ultralytics

Para descomprimir el archivo .rar perteneciente al dataset se instala la dependencia usando el comando **!sudo apt-get install unrar** en una celda de ejecución del notebook, se procede con la descompresión y almacenamiento en Google Drive para posterior uso.

```
import os

# Nombre del archivo RAR
rar_file_name = "data.rar"

# Ruta de destino para la extracción
extract_path = "/content/drive/MyDrive"

# Descomprimir el archivo RAR
os.system(f"unrar x {rar_file_name} {extract_path}")
```

Antes de iniciar el entrenamiento es fundamental evitar errores relacionados a memoria en este proceso, por eso en otra celda de ejecución se importó la dependencia de Pytorch juntamente con CUDA usándolo para gestionar la memoria de GPU, el comando **torch.cuda.empty_cache()** libera la memoria GPU retenida que no se está usando previo al entrenamiento.

```
import torch
torch.cuda.empty_cache()
torch.cuda.memory_summary(device=None, abbreviated=False)
```

- ***Modelo 1 para detección de manzanas***

El proceso de entrenamiento del primer modelo a usar se lleva a cabo con el dataset detallado en la Tabla 36 utilizando una división de este con una proporción del 80% de imágenes para entrenamiento y un 20% para validación según las recomendaciones para YOLOv8[70].

A continuación, se muestra la celda ejecutada con la línea de comandos usada durante la inicialización del proceso de entrenamiento del modelo YOLOv8x. Se ha empleado la función integrada de la clase YOLO de Ultralytics con los siguientes parámetros:

- Modelo inicial: Se parte de yolov8x.pt, un modelo pre entrenado.
- Numero de épocas (epochs) = 70
- Tamaño de imagen: 640x640 píxeles (imgsz=640).

```
from ultralytics import YOLO

# Load a model
model = YOLO("yolov8x.pt") # load a pretrained model
                             (recommended for training)

# Train the model
results = model.train(data="coco.yaml", epochs=70, imgsz=640)
```

El resto de los parámetros del modelo se han dejado en valores por defecto, algunos de estos parámetros se describen en la Tabla 40:

Tabla 40. Parámetros YOLOv8[55].

Parámetro	Configuración por defecto	Descripción
batch	16	Numero de tamaño de lote para el proceso por lotes
save	True	Guarda los puntos de control de entrenamiento.
val	True	Activa la validación durante el entrenamiento
plots	True	Genera y guarda gráficos de las métricas de entrenamiento

- **Modelo 2 para tipos de manzanas**

Para este modelo se ha usado el filtrado HSV (Hue Saturation and Value) proceso mediante el cual aísla y extrae colores específicos de una imagen en función de un umbral definido con la finalidad de identificar diferentes tipos de manzanas mediante sus características de color en el espacio[58].

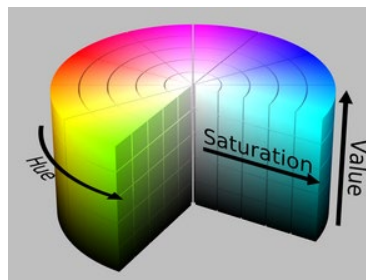


Figura 50. Espacio de color HSV[58].

Para definir el umbral para cada tipo de manzana se usa como referencia la Figura 51, que usa un determinado rango de valores HSV, para Hue de 0-180, Saturation y Value de 0-255[71].

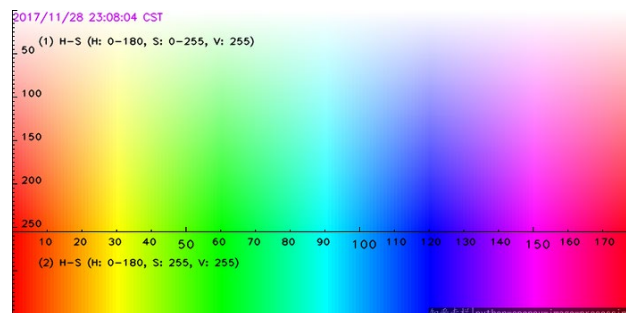


Figura 51. Vista condensada de mapa de colores HSV[71]

Para el entrenamiento de este modelo se va a usar 4 tipos de manzana definidas en anteriormente en la Tabla 20, por ello cada región de la imagen de la manzana debe ser convertida desde el espacio de color BGR hacia el espacio de color HSV facilitando así la diferenciación y segmentación de colores específicos.

El comando que permite la conversión de BGR a HSV es:

```
hsv_region = cv2.cvtColor(region_manzana, cv2.COLOR_BGR2HSV)
```

Una vez realizada la conversión, se definen los rangos de colores para la variedad de manzanas que se desea segmentar:

- Red Delicious: rango [0, 100, 200] a [1, 255, 255]
- Granny Smith: rango [30, 100, 100] a [40, 255, 255]
- Golden Delicious: rango [20, 100, 100] a [30, 255, 255]
- Fuji: rango [3, 100, 100] a [15, 255, 255]

Una vez definido el rango de colores para cada tipo de manzana, se crean mascarar binarias produciendo una imagen en escala de grises donde los pixeles que caen dentro del rango especificado se establecen en color blanco (255) y el resto de los pixeles en color negro (0, esto se realiza con los siguientes comandos:

```
mask_red = cv2.inRange(hsv_region, lower_red, upper_red)
mask_green = cv2.inRange(hsv_region, lower_green, upper_green)
mask_yellow = cv2.inRange(hsv_region, lower_yellow, upper_yellow)
mask_fuji = cv2.inRange(hsv_region, lower_fuji, upper_fuji)
```

Se configura el modelo seleccionado, en este se hace uso de un modelo preentrenado con arquitectura YOLOv8x estableciendo los hiperparametros necesarios para el entrenamiento, inicialmente el entrenamiento se configuro con 25 épocas, pero para lograr una mejor precisión en el modelo se optó por entrenar con 50 épocas con un tamaño por lote de 16 para el proceso por lotes e imágenes con una resolución de 640x640.

```

dataset_path = '/content/drive/MyDrive/data/dataset.yaml'

# Check if the dataset path exists
if not os.path.exists(dataset_path):
    raise FileNotFoundError(f"Dataset path '{dataset_path}' does not exist.")
from ultralytics import YOLO

# Load a model
model = YOLO('yolov8x.pt') # load a pretrained model
                             (recommended for training)
model = YOLO('yolov8x.yaml')
model = YOLO('yolov8x.yaml').load('yolov8x.pt')
# Train the model with 2 GPUs
results = model.train(data=dataset_path, epochs=50,
imgsz=640,batch=16)

```

Durante la ejecución del entrenamiento de las 50 épocas cada una de estas consume un tiempo promedio aproximado de 53 segundos con una utilización de GPU de 14.2 Gbytes como se observa en la Figura 52.

```

Starting training for 50 epochs...

```

Epoch	GPU_mem	box_loss	cls_loss	dfl_loss	Instances	Size
1/50	14.3G	2.021	3.4	2.453	18	640: 100% 42 [00:53<00:00, 1.78s/it]
	Class	Images	Instances	Box(P)	R	mAP50 mAP50-95): 100% 5/5 [00:00<00:00, 1.75s/it]
2/50	14.2G	1.789	2.198	2.046	7	640: 100% 42 [00:54<00:00, 1.1s/it]
	Class	Images	Instances	Box(P)	R	mAP50 mAP50-95): 100% 5/5 [00:04<00:00, 1.06it/s]
3/50	13.9G	1.845	2.091	1.992	7	640: 100% 42 [00:53<00:00, 1.38s/it]
	Class	Images	Instances	Box(P)	R	mAP50 mAP50-95): 100% 5/5 [00:05<00:00, 1.09s/it]
4/50	14.2G	1.869	1.823	1.966	10	640: 100% 42 [00:54<00:00, 1.9s/it]
	Class	Images	Instances	Box(P)	R	mAP50 mAP50-95): 100% 5/5 [00:05<00:00, 1.01s/it]
	all	130	157	0.548	0.55	0.523 0.238

Figura 52. Ejecución del entrenamiento por épocas modelo 2

Una vez completado el entrenamiento de este modelo los resultados se muestran en la salida de consola de la celda en ejecución, en específico para este modelo al entorno de ejecución GoogleColab le tomo 51.3 minutos, los archivos generados como resultado del entrenamiento se almacenan en la unidad de Google Drive y de la misma manera los archivos “best.pt” y “last.pt” representando los pesos del modelo guardados en distintos momentos del entrenamiento.

- ***Modelo 3 para clasificación de manzanas por grado de calidad.***

Para este modelo se han entrenado en distintas arquitecturas con los mismo hiperparametros y dataset detallado en la Tabla 37, los resultados que se han obtenido en la validación de cada entrenamiento se muestran en la Tabla 41.

Tabla 41. Resultados de entrenamiento con distintas arquitecturas de YOLOv8.

Modelo	Épocas	Precision	Recall	mAP50	mPA50-95
Yolov8n	60	0.855	0.89	0.407	0.21
Yolov8l	100	0.877	0.716	0.493	0.30
Yolov8x	25	0.845	0.86	0.616	0.375

Los resultados muestran una mejora significativa para el modelo YOLOv8x ya que con tan solo 25 épocas de entrenamiento se acerca a la precisión de las otras arquitecturas entrenadas con más épocas y se muestra una mejora significativa en las métricas de rendimiento.

Determinada la arquitectura a usar se procedió con dos fases de entrenamiento distintas: una con una duración de 25 épocas entrenada para la elección de la arquitectura y otra extendida a 50 épocas. La rotación de imágenes y aplicación de filtrado de imágenes en un conjunto de datos previo al entrenamiento resulta altamente ventajosa, dado que constituye una forma de aumento de datos. Estas técnicas contribuyen significativamente a la versatilidad del modelo, ya que, al incorporar imágenes desde diversas orientaciones y con diversos filtros, el modelo se capacita para identificar patrones específicos bajo diferentes ángulos y el ajuste de luminosidad, mejorando su rendimiento y previniendo el fenómeno de sobreajuste.

El entrenamiento se realizó mediante el siguiente algoritmo, usando el modelo 2 entrenado para los tipos de manzanas con los mismos hiperparametros para los dos entrenamientos por épocas mencionadas.

```
dataset_path = '/content/drive/MyDrive/data/dataset.yaml'
```

```
# Check if the dataset path exists
if not os.path.exists(dataset_path):
    raise FileNotFoundError(f"Dataset path '{dataset_path}' does not exist.")
from ultralytics import YOLO
```

```
# Load a model
model = YOLO('best.pt') # load a pretrained model (recommended
for training)
model = YOLO('yolov8x.yaml')
model = YOLO('yolov8x.yaml').load('best.pt')
# Train the model with 2 GPUs
results = model.train(data=dataset_path, epochs=50,
imgsz=640,batch=16)
```

Al ejecutar el entrenamiento del modelo el tiempo consumido por épocas tiene un promedio aproximado de 1:37 minutos con un uso de memoria de GPU de 14 Gbytes como se muestra en la Figura 53.

Epoch	GPU_mem	box_loss	cls_loss	dfl_loss	Instances	Size
1/50	14.3G	1.773	2.873	1.687	40	640: 100% [01:37:00:00, 1.19s/it]
	Class	Images	Instances	Box(P)	R	mAP50 mAP50-95: 100% [01:37:00:00, 1.19s/it]
Epoch	GPU_mem	box_loss	cls_loss	dfl_loss	Instances	Size
2/50	14G	1.828	2.178	1.72	41	640: 100% [01:39:00:00, 1.02s/it]
	Class	Images	Instances	Box(P)	R	mAP50 mAP50-95: 100% [01:39:00:00, 1.02s/it]
Epoch	GPU_mem	box_loss	cls_loss	dfl_loss	Instances	Size
3/50	14.3G	1.881	2.176	1.751	40	640: 100% [01:35:00:00, 1.26s/it]
	Class	Images	Instances	Box(P)	R	mAP50 mAP50-95: 100% [01:35:00:00, 1.26s/it]
Epoch	GPU_mem	box_loss	cls_loss	dfl_loss	Instances	Size
4/50	14.2G	1.871	2.162	1.758	24	640: 100% [01:37:00:00, 1.03it/s]
	Class	Images	Instances	Box(P)	R	mAP50 mAP50-95: 100% [01:37:00:00, 1.03it/s]

Figura 53. Ejecución del entrenamiento por épocas modelo 3

Una vez completado el entrenamiento en específico para este modelo al entorno de ejecución GoogleColab le tomo 95.4 minutos, los archivos generados como resultado del entrenamiento se almacenan en la unidad de Google Drive y de la misma manera los archivos “best.pt” y “last.pt”.

h. Resultados de entrenamiento

• Métricas de rendimiento

Muestran una colección de gráficos que rastrean el progreso del modelo entrenado, YOLOv8x, durante su fase de entrenamiento y validación. Los gráficos cubren varias métricas importantes como las pérdidas durante el entrenamiento y la validación y medidas de precisión y exactitud para diferentes tamaños de objetos. Estos gráficos son herramientas esenciales para entender cómo el modelo mejora con el tiempo y cómo se comporta en pruebas prácticas, ayudando a ajustar y mejorar su rendimiento general.

Para el primer modelo, la Figura 54 muestra una evolución de diversas pérdidas y métricas evaluadas durante el entrenamiento y validación de un modelo de aprendizaje profundo para detección de objetos a lo largo de 70 épocas. Las pérdidas de entrenamiento para las cajas delimitadoras (train/box_loss), clasificación (train/cls_loss), y distribución focal(train/dfl_loss) disminuye consistentemente de aproximadamente 1.0, 1.6 y 1.3 a 0.7, 1.1 y 1.0 respectivamente, lo que indicó que el modelo mejoró en la predicción de las posiciones y tamaños de cajas que encierran los objetos, el modelo fue más preciso en identificación de las clases de objetos y mejoró la precisión de asignación de pesos a diferentes áreas dentro de las cajas delimitadoras lo que contribuyó a mejorar la precisión general de las predicciones. Las métricas de precisión (B) y recall (metric/recall(B)) aumentaron de 0.6 a 0.9 y de 0.5 a 0.75, lo que indicó que el modelo redujo la tasa de falsos positivos al identificar correctamente más objetos sin equivocaciones, y la tasa de falsos negativos al identificar más objetos presentes en la imagen. En validación, se observó una tendencia similar, con las pérdidas de validación (val/box_loss, val/cls_loss, val/dfl_loss) bajando de 0.9, 1.6 y 1.2 a 0.7, .2 y 0.9, respectivamente. Las métricas de precisión media promedio al umbral de IoU de 50%(metrics/Map50(B)) y entre 50% y 95% (metrics/Map50-95(B)) mostraron una recuperación notable después de caídas iniciales, finalizando en 0.8 y 0.6 lo que demostró que el modelo mejoró su rendimiento general en detección de objetos a diferentes umbrales y se ajustó correctamente a los datos de entrenamiento.

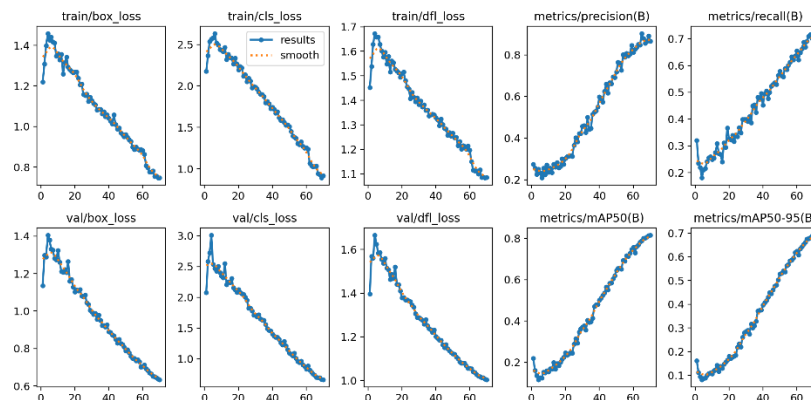


Figura 54. Métricas de rendimiento del modelo entrenado (Detección de Manzanas)

La Figura 55 ilustra la evolución de pérdidas y métricas durante el entrenamiento y validación del modelo de clasificación de tipo de manzana a lo largo de 50 épocas.

Las pérdidas de entrenamiento para las cajas delimitadoras, clasificación de etiquetas, clasificación de objetos y distribución focal descendieron consistentemente de valores iniciales de aproximadamente 1.75, 0.8, 12.5 y 2.1 a valores finales de 1.2, 0.1, 2.2 y 1.0 respectivamente, indicando mejoras en la precisión de la predicción de posiciones y tamaños de cajas, segmentación de etiquetas, clasificación de objetos y ponderación dentro de las cajas. Las métricas de precisión y recall aumentaron significativamente reflejando una reducción en los falsos positivos y negativos. Las métricas de precisión media promedio a umbrales de IoU del 50% y entre 50% y 95% exhibieron mejoras notables, alcanzando valores finales de 0.85 y 0.65 respectivamente. Estos resultados confirman la mejora general del rendimiento del modelo.

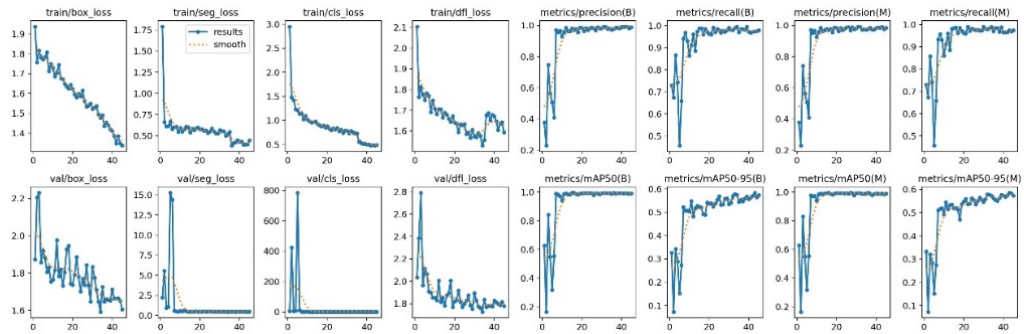


Figura 55. Métricas de rendimiento del modelo entrenado (Tipo de manzanas)

La Figura 56 ilustra la evolución de pérdidas y métricas durante el entrenamiento y validación del modelo de clasificación de manzana por grado de calidad a lo largo de 50 épocas. Las pérdidas de entrenamiento para las cajas delimitadoras, clasificación de etiquetas, clasificación de objetos y distribución focal descendieron consistentemente de valores iniciales de aproximadamente 1.4, 1.2, 1.35 y 2.1 a valores finales de 1.1, 0.8 y 0.525 respectivamente, indicando mejoras en la precisión de la predicción de posiciones y tamaños de cajas, segmentación de etiquetas, clasificación de objetos y ponderación dentro de las cajas. Las métricas de precisión y recall aumentaron significativamente reflejando una reducción en los falsos positivos y negativos. Las métricas de precisión media promedio a umbrales de IoU del 50% y entre 50% y 95% exhibieron mejoras notables, alcanzando valores finales de 0.66 y 0.46 respectivamente. Estos resultados confirman la mejora general del rendimiento del modelo.

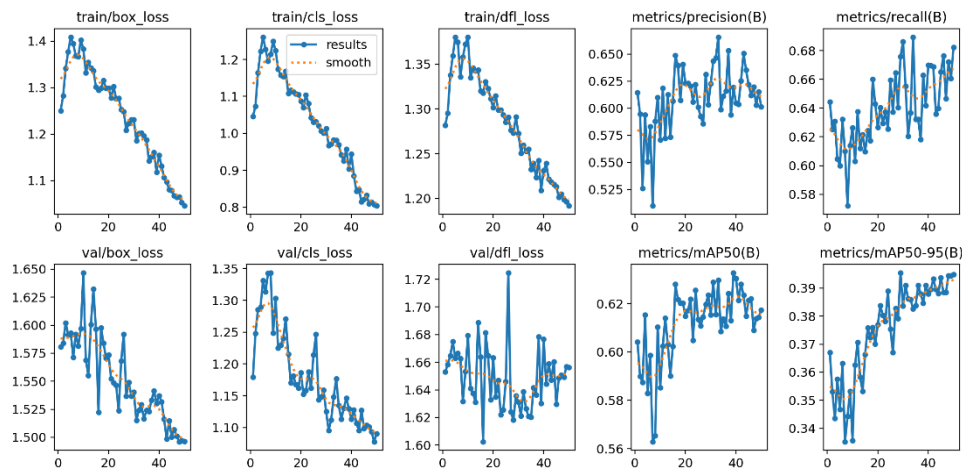


Figura 56. Métricas de rendimiento del modelo entrenado (Clasificación de manzanas por grado de calidad)

- **Matriz de correlación**

A partir del archivo results.csv generado después de cada entrenamiento se obtiene las matrices de correlación, en la Tabla 42 se visualiza las relaciones entre las métricas de pérdidas de entrenamiento y validación para cada modelo entrenado. Cada celda en la matriz representa el coeficiente de correlación entre diferentes tipos de pérdidas durante las fases de entrenamiento y validación, donde valores más altos (representados en tonos rojos) indican una fuerte correlación positiva y valores más bajos (en tonos azules) denotan correlaciones débiles.

Esta herramienta es crucial para evaluar la similitud en el comportamiento de las pérdidas y para identificar posibles sobreajustes o dinámicas inconsistentes entre el entrenamiento y la validación.

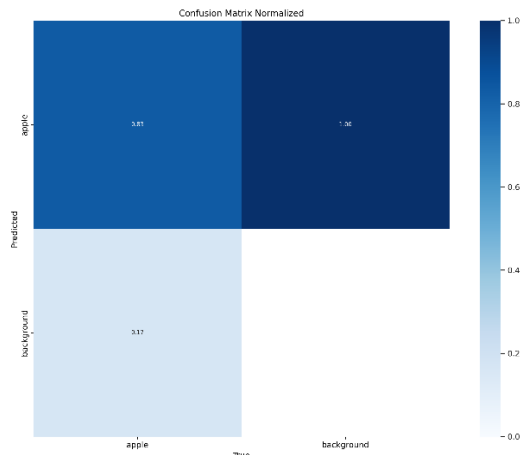
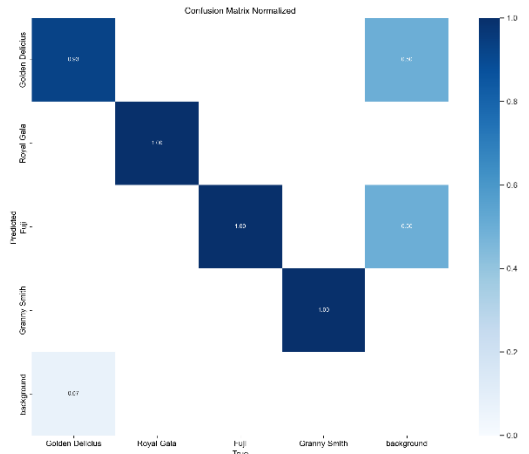
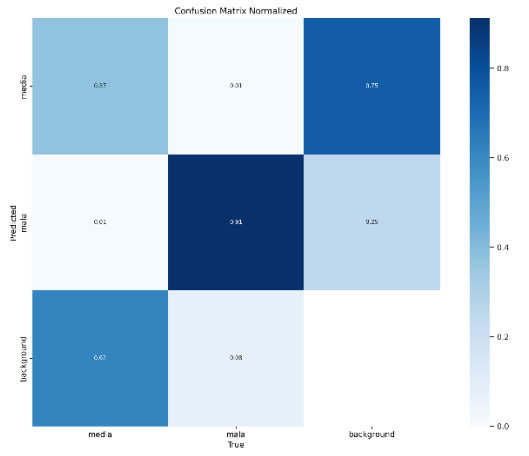
- **Matriz de confusión.**

A partir de los resultados obtenidos, se llevó a cabo la elaboración de una matriz de correlación por cada modelo entrenado como se muestra en la Tabla 43 entre los conjuntos de datos de entrada, abarcando tanto el total de valores de entrenamiento como la totalidad de los valores asociados a cada una de las variables, la función principal es medir la relación lineal entre cada variable y sus valores pasados.

Tabla 42. Matriz de correlación por modelo entrenado

Modelo	Matriz de correlación																																																																																																																									
1. Detección de manzanas: La matriz de correlación muestra una alta correlación entre las pérdidas de entrenamiento y validación, así como entre estas pérdidas y las métricas de precisión. Para este caso la perdida de cajas de entrenamiento tiene una correlación de 0.99 con la pérdida de clasificación de etiquetas de entrenamiento y una correlación inversa de -0.97 con la métrica de precisión (metrics/precision(B)). Esto indica que las mejoras en las perdidas están directamente relacionadas con mejoras en las métricas de precisión del modelo.	Correlation Matrix of Training and Validation Losses <table><tr><th></th><th>train/box_loss</th><th>train/cls_loss</th><th>train/dfl_loss</th><th>val/box_loss</th><th>val/cls_loss</th><th>val/dfl_loss</th><th>metrics/precision(B)</th><th>metrics/recall(B)</th><th>metrics/mAP50(B)</th><th>metrics/mAP50-95(B)</th></tr><tr><th>train/box_loss</th><td>1.00</td><td>0.99</td><td>0.99</td><td>0.98</td><td>0.97</td><td>0.98</td><td>-0.95</td><td>-0.97</td><td>-0.96</td><td>-0.97</td></tr><tr><th>train/cls_loss</th><td>0.99</td><td>1.00</td><td>0.99</td><td>0.99</td><td>0.97</td><td>0.98</td><td>-0.97</td><td>-0.98</td><td>-0.97</td><td>-0.98</td></tr><tr><th>train/dfl_loss</th><td>0.99</td><td>0.99</td><td>1.00</td><td>0.97</td><td>0.96</td><td>0.98</td><td>-0.95</td><td>-0.97</td><td>-0.96</td><td>-0.97</td></tr><tr><th>val/box_loss</th><td>0.98</td><td>0.99</td><td>0.97</td><td>1.00</td><td>0.98</td><td>1.00</td><td>-0.98</td><td>-0.99</td><td>-0.99</td><td>-1.00</td></tr><tr><th>val/cls_loss</th><td>0.97</td><td>0.97</td><td>0.96</td><td>0.98</td><td>1.00</td><td>0.99</td><td>-0.98</td><td>-0.98</td><td>-0.99</td><td>-0.99</td></tr><tr><th>val/dfl_loss</th><td>0.98</td><td>0.98</td><td>0.98</td><td>1.00</td><td>0.99</td><td>1.00</td><td>-0.98</td><td>-0.99</td><td>-0.99</td><td>-0.99</td></tr><tr><th>metrics/precision(B)</th><td>-0.95</td><td>-0.97</td><td>-0.95</td><td>-0.98</td><td>-0.98</td><td>-0.98</td><td>1.00</td><td>0.97</td><td>0.99</td><td>0.99</td></tr><tr><th>metrics/recall(B)</th><td>-0.97</td><td>-0.98</td><td>-0.97</td><td>-0.99</td><td>-0.98</td><td>-0.99</td><td>0.97</td><td>1.00</td><td>0.99</td><td>0.99</td></tr><tr><th>metrics/mAP50(B)</th><td>-0.96</td><td>-0.97</td><td>-0.96</td><td>-0.99</td><td>-0.99</td><td>-0.99</td><td>0.99</td><td>0.99</td><td>1.00</td><td>1.00</td></tr><tr><th>metrics/mAP50-95(B)</th><td>-0.97</td><td>-0.98</td><td>-0.97</td><td>-1.00</td><td>-0.99</td><td>-0.99</td><td>0.99</td><td>0.99</td><td>1.00</td><td>1.00</td></tr></table>		train/box_loss	train/cls_loss	train/dfl_loss	val/box_loss	val/cls_loss	val/dfl_loss	metrics/precision(B)	metrics/recall(B)	metrics/mAP50(B)	metrics/mAP50-95(B)	train/box_loss	1.00	0.99	0.99	0.98	0.97	0.98	-0.95	-0.97	-0.96	-0.97	train/cls_loss	0.99	1.00	0.99	0.99	0.97	0.98	-0.97	-0.98	-0.97	-0.98	train/dfl_loss	0.99	0.99	1.00	0.97	0.96	0.98	-0.95	-0.97	-0.96	-0.97	val/box_loss	0.98	0.99	0.97	1.00	0.98	1.00	-0.98	-0.99	-0.99	-1.00	val/cls_loss	0.97	0.97	0.96	0.98	1.00	0.99	-0.98	-0.98	-0.99	-0.99	val/dfl_loss	0.98	0.98	0.98	1.00	0.99	1.00	-0.98	-0.99	-0.99	-0.99	metrics/precision(B)	-0.95	-0.97	-0.95	-0.98	-0.98	-0.98	1.00	0.97	0.99	0.99	metrics/recall(B)	-0.97	-0.98	-0.97	-0.99	-0.98	-0.99	0.97	1.00	0.99	0.99	metrics/mAP50(B)	-0.96	-0.97	-0.96	-0.99	-0.99	-0.99	0.99	0.99	1.00	1.00	metrics/mAP50-95(B)	-0.97	-0.98	-0.97	-1.00	-0.99	-0.99	0.99	0.99	1.00	1.00
	train/box_loss	train/cls_loss	train/dfl_loss	val/box_loss	val/cls_loss	val/dfl_loss	metrics/precision(B)	metrics/recall(B)	metrics/mAP50(B)	metrics/mAP50-95(B)																																																																																																																
train/box_loss	1.00	0.99	0.99	0.98	0.97	0.98	-0.95	-0.97	-0.96	-0.97																																																																																																																
train/cls_loss	0.99	1.00	0.99	0.99	0.97	0.98	-0.97	-0.98	-0.97	-0.98																																																																																																																
train/dfl_loss	0.99	0.99	1.00	0.97	0.96	0.98	-0.95	-0.97	-0.96	-0.97																																																																																																																
val/box_loss	0.98	0.99	0.97	1.00	0.98	1.00	-0.98	-0.99	-0.99	-1.00																																																																																																																
val/cls_loss	0.97	0.97	0.96	0.98	1.00	0.99	-0.98	-0.98	-0.99	-0.99																																																																																																																
val/dfl_loss	0.98	0.98	0.98	1.00	0.99	1.00	-0.98	-0.99	-0.99	-0.99																																																																																																																
metrics/precision(B)	-0.95	-0.97	-0.95	-0.98	-0.98	-0.98	1.00	0.97	0.99	0.99																																																																																																																
metrics/recall(B)	-0.97	-0.98	-0.97	-0.99	-0.98	-0.99	0.97	1.00	0.99	0.99																																																																																																																
metrics/mAP50(B)	-0.96	-0.97	-0.96	-0.99	-0.99	-0.99	0.99	0.99	1.00	1.00																																																																																																																
metrics/mAP50-95(B)	-0.97	-0.98	-0.97	-1.00	-0.99	-0.99	0.99	0.99	1.00	1.00																																																																																																																
2. Determinación de tipo de manzanas: La pérdida de cajas de entrenamiento (train/box/loss) tiene una alta correlación con la perdida de clasificación (train/cls_loss)(0.90) y con la perdida de distribución focal (train/dfl_loss)(0.79). Las métricas de precisión tienen correlaciones negativas con las pérdidas de clasificación y distribución focal de validación, con valores como -0.79 y -0.85 respectivamente. Además, las métricas de precisión y recall (0.99) están altamente correlacionadas indicando consistencia en la evaluación del modelo. Esto indica que las mejoras en las perdidas están directamente relacionadas con mejoras en las métricas de precisión del modelo.	Correlation Matrix of Training and Validation Losses <table><tr><th></th><th>train/box_loss</th><th>train/cls_loss</th><th>train/dfl_loss</th><th>val/box_loss</th><th>val/cls_loss</th><th>val/dfl_loss</th><th>metrics/precision(B)</th><th>metrics/recall(B)</th><th>metrics/mAP50(B)</th><th>metrics/mAP50-95(B)</th></tr><tr><th>train/box_loss</th><td>1.00</td><td>0.90</td><td>0.79</td><td>0.39</td><td>0.54</td><td>0.38</td><td>-0.67</td><td>-0.68</td><td>-0.67</td><td>-0.72</td></tr><tr><th>train/cls_loss</th><td>0.90</td><td>1.00</td><td>0.92</td><td>0.58</td><td>0.69</td><td>0.60</td><td>-0.87</td><td>-0.87</td><td>-0.87</td><td>-0.89</td></tr><tr><th>train/dfl_loss</th><td>0.79</td><td>0.92</td><td>1.00</td><td>0.57</td><td>0.68</td><td>0.58</td><td>-0.86</td><td>-0.85</td><td>-0.85</td><td>-0.86</td></tr><tr><th>val/box_loss</th><td>0.39</td><td>0.58</td><td>0.57</td><td>1.00</td><td>0.50</td><td>0.99</td><td>-0.81</td><td>-0.81</td><td>-0.82</td><td>-0.80</td></tr><tr><th>val/cls_loss</th><td>0.54</td><td>0.69</td><td>0.68</td><td>0.50</td><td>1.00</td><td>0.51</td><td>-0.72</td><td>-0.74</td><td>-0.76</td><td>-0.77</td></tr><tr><th>val/dfl_loss</th><td>0.38</td><td>0.60</td><td>0.58</td><td>0.99</td><td>0.51</td><td>1.00</td><td>-0.83</td><td>-0.82</td><td>-0.83</td><td>-0.80</td></tr><tr><th>metrics/precision(B)</th><td>-0.67</td><td>-0.87</td><td>-0.86</td><td>-0.81</td><td>-0.72</td><td>-0.83</td><td>1.00</td><td>0.99</td><td>0.99</td><td>0.98</td></tr><tr><th>metrics/recall(B)</th><td>-0.68</td><td>-0.87</td><td>-0.85</td><td>-0.81</td><td>-0.74</td><td>-0.82</td><td>0.99</td><td>1.00</td><td>1.00</td><td>0.99</td></tr><tr><th>metrics/mAP50(B)</th><td>-0.67</td><td>-0.87</td><td>-0.85</td><td>-0.82</td><td>-0.76</td><td>-0.83</td><td>0.99</td><td>1.00</td><td>1.00</td><td>0.99</td></tr><tr><th>metrics/mAP50-95(B)</th><td>-0.72</td><td>-0.89</td><td>-0.86</td><td>-0.80</td><td>-0.77</td><td>-0.80</td><td>0.98</td><td>0.99</td><td>0.99</td><td>1.00</td></tr></table>		train/box_loss	train/cls_loss	train/dfl_loss	val/box_loss	val/cls_loss	val/dfl_loss	metrics/precision(B)	metrics/recall(B)	metrics/mAP50(B)	metrics/mAP50-95(B)	train/box_loss	1.00	0.90	0.79	0.39	0.54	0.38	-0.67	-0.68	-0.67	-0.72	train/cls_loss	0.90	1.00	0.92	0.58	0.69	0.60	-0.87	-0.87	-0.87	-0.89	train/dfl_loss	0.79	0.92	1.00	0.57	0.68	0.58	-0.86	-0.85	-0.85	-0.86	val/box_loss	0.39	0.58	0.57	1.00	0.50	0.99	-0.81	-0.81	-0.82	-0.80	val/cls_loss	0.54	0.69	0.68	0.50	1.00	0.51	-0.72	-0.74	-0.76	-0.77	val/dfl_loss	0.38	0.60	0.58	0.99	0.51	1.00	-0.83	-0.82	-0.83	-0.80	metrics/precision(B)	-0.67	-0.87	-0.86	-0.81	-0.72	-0.83	1.00	0.99	0.99	0.98	metrics/recall(B)	-0.68	-0.87	-0.85	-0.81	-0.74	-0.82	0.99	1.00	1.00	0.99	metrics/mAP50(B)	-0.67	-0.87	-0.85	-0.82	-0.76	-0.83	0.99	1.00	1.00	0.99	metrics/mAP50-95(B)	-0.72	-0.89	-0.86	-0.80	-0.77	-0.80	0.98	0.99	0.99	1.00
	train/box_loss	train/cls_loss	train/dfl_loss	val/box_loss	val/cls_loss	val/dfl_loss	metrics/precision(B)	metrics/recall(B)	metrics/mAP50(B)	metrics/mAP50-95(B)																																																																																																																
train/box_loss	1.00	0.90	0.79	0.39	0.54	0.38	-0.67	-0.68	-0.67	-0.72																																																																																																																
train/cls_loss	0.90	1.00	0.92	0.58	0.69	0.60	-0.87	-0.87	-0.87	-0.89																																																																																																																
train/dfl_loss	0.79	0.92	1.00	0.57	0.68	0.58	-0.86	-0.85	-0.85	-0.86																																																																																																																
val/box_loss	0.39	0.58	0.57	1.00	0.50	0.99	-0.81	-0.81	-0.82	-0.80																																																																																																																
val/cls_loss	0.54	0.69	0.68	0.50	1.00	0.51	-0.72	-0.74	-0.76	-0.77																																																																																																																
val/dfl_loss	0.38	0.60	0.58	0.99	0.51	1.00	-0.83	-0.82	-0.83	-0.80																																																																																																																
metrics/precision(B)	-0.67	-0.87	-0.86	-0.81	-0.72	-0.83	1.00	0.99	0.99	0.98																																																																																																																
metrics/recall(B)	-0.68	-0.87	-0.85	-0.81	-0.74	-0.82	0.99	1.00	1.00	0.99																																																																																																																
metrics/mAP50(B)	-0.67	-0.87	-0.85	-0.82	-0.76	-0.83	0.99	1.00	1.00	0.99																																																																																																																
metrics/mAP50-95(B)	-0.72	-0.89	-0.86	-0.80	-0.77	-0.80	0.98	0.99	0.99	1.00																																																																																																																
3. Clasificación de manzanas por grado de calidad: La pérdida de cajas de entrenamiento (train/box/loss) tiene una alta correlación con la perdida de clasificación (train/cls_loss)(0.96) y con la perdida de distribución foca (train/dfl_loss)(0.98). Las métricas de precisión tienen correlaciones negativas con las pérdidas de clasificación y distribución focal de validación, con valores como -0.44 y -0.47 respectivamente. Además, las métricas de precisión y recall (0.89) están altamente correlacionadas indicando consistencia en la evaluación del modelo. Esto indica que las mejoras en las perdidas están directamente relacionadas con mejoras en las métricas de precisión del modelo.	Correlation Matrix of Training and Validation Losses <table><tr><th></th><th>train/box_loss</th><th>train/cls_loss</th><th>train/dfl_loss</th><th>val/box_loss</th><th>val/cls_loss</th><th>val/dfl_loss</th><th>metrics/precision(B)</th><th>metrics/recall(B)</th><th>metrics/mAP50(B)</th><th>metrics/mAP50-95(B)</th></tr><tr><th>train/box_loss</th><td>1.00</td><td>0.96</td><td>0.98</td><td>0.63</td><td>0.45</td><td>0.40</td><td>-0.64</td><td>-0.82</td><td>-0.93</td><td>-0.96</td></tr><tr><th>train/cls_loss</th><td>0.96</td><td>1.00</td><td>0.95</td><td>0.65</td><td>0.43</td><td>0.41</td><td>-0.70</td><td>-0.90</td><td>-0.97</td><td>-0.98</td></tr><tr><th>train/dfl_loss</th><td>0.98</td><td>0.95</td><td>1.00</td><td>0.68</td><td>0.53</td><td>0.45</td><td>-0.67</td><td>-0.82</td><td>-0.95</td><td>-0.96</td></tr><tr><th>val/box_loss</th><td>0.63</td><td>0.65</td><td>0.68</td><td>1.00</td><td>0.50</td><td>0.96</td><td>-0.62</td><td>-0.76</td><td>-0.73</td><td>-0.67</td></tr><tr><th>val/cls_loss</th><td>0.45</td><td>0.43</td><td>0.53</td><td>0.50</td><td>1.00</td><td>0.74</td><td>-0.44</td><td>-0.10</td><td>-0.56</td><td>-0.50</td></tr><tr><th>val/dfl_loss</th><td>0.40</td><td>0.41</td><td>0.45</td><td>0.96</td><td>0.74</td><td>1.00</td><td>-0.47</td><td>-0.61</td><td>-0.51</td><td>-0.43</td></tr><tr><th>metrics/precision(B)</th><td>-0.64</td><td>-0.70</td><td>-0.67</td><td>-0.62</td><td>-0.44</td><td>-0.47</td><td>1.00</td><td>0.93</td><td>0.75</td><td>0.73</td></tr><tr><th>metrics/recall(B)</th><td>-0.82</td><td>-0.90</td><td>-0.82</td><td>-0.76</td><td>-0.10</td><td>-0.61</td><td>0.93</td><td>1.00</td><td>0.89</td><td>0.88</td></tr><tr><th>metrics/mAP50(B)</th><td>-0.93</td><td>-0.97</td><td>-0.95</td><td>-0.73</td><td>-0.56</td><td>-0.51</td><td>0.75</td><td>0.89</td><td>1.00</td><td>0.99</td></tr><tr><th>metrics/mAP50-95(B)</th><td>-0.96</td><td>-0.98</td><td>-0.96</td><td>-0.67</td><td>-0.50</td><td>-0.43</td><td>0.73</td><td>0.88</td><td>0.99</td><td>1.00</td></tr></table>		train/box_loss	train/cls_loss	train/dfl_loss	val/box_loss	val/cls_loss	val/dfl_loss	metrics/precision(B)	metrics/recall(B)	metrics/mAP50(B)	metrics/mAP50-95(B)	train/box_loss	1.00	0.96	0.98	0.63	0.45	0.40	-0.64	-0.82	-0.93	-0.96	train/cls_loss	0.96	1.00	0.95	0.65	0.43	0.41	-0.70	-0.90	-0.97	-0.98	train/dfl_loss	0.98	0.95	1.00	0.68	0.53	0.45	-0.67	-0.82	-0.95	-0.96	val/box_loss	0.63	0.65	0.68	1.00	0.50	0.96	-0.62	-0.76	-0.73	-0.67	val/cls_loss	0.45	0.43	0.53	0.50	1.00	0.74	-0.44	-0.10	-0.56	-0.50	val/dfl_loss	0.40	0.41	0.45	0.96	0.74	1.00	-0.47	-0.61	-0.51	-0.43	metrics/precision(B)	-0.64	-0.70	-0.67	-0.62	-0.44	-0.47	1.00	0.93	0.75	0.73	metrics/recall(B)	-0.82	-0.90	-0.82	-0.76	-0.10	-0.61	0.93	1.00	0.89	0.88	metrics/mAP50(B)	-0.93	-0.97	-0.95	-0.73	-0.56	-0.51	0.75	0.89	1.00	0.99	metrics/mAP50-95(B)	-0.96	-0.98	-0.96	-0.67	-0.50	-0.43	0.73	0.88	0.99	1.00
	train/box_loss	train/cls_loss	train/dfl_loss	val/box_loss	val/cls_loss	val/dfl_loss	metrics/precision(B)	metrics/recall(B)	metrics/mAP50(B)	metrics/mAP50-95(B)																																																																																																																
train/box_loss	1.00	0.96	0.98	0.63	0.45	0.40	-0.64	-0.82	-0.93	-0.96																																																																																																																
train/cls_loss	0.96	1.00	0.95	0.65	0.43	0.41	-0.70	-0.90	-0.97	-0.98																																																																																																																
train/dfl_loss	0.98	0.95	1.00	0.68	0.53	0.45	-0.67	-0.82	-0.95	-0.96																																																																																																																
val/box_loss	0.63	0.65	0.68	1.00	0.50	0.96	-0.62	-0.76	-0.73	-0.67																																																																																																																
val/cls_loss	0.45	0.43	0.53	0.50	1.00	0.74	-0.44	-0.10	-0.56	-0.50																																																																																																																
val/dfl_loss	0.40	0.41	0.45	0.96	0.74	1.00	-0.47	-0.61	-0.51	-0.43																																																																																																																
metrics/precision(B)	-0.64	-0.70	-0.67	-0.62	-0.44	-0.47	1.00	0.93	0.75	0.73																																																																																																																
metrics/recall(B)	-0.82	-0.90	-0.82	-0.76	-0.10	-0.61	0.93	1.00	0.89	0.88																																																																																																																
metrics/mAP50(B)	-0.93	-0.97	-0.95	-0.73	-0.56	-0.51	0.75	0.89	1.00	0.99																																																																																																																
metrics/mAP50-95(B)	-0.96	-0.98	-0.96	-0.67	-0.50	-0.43	0.73	0.88	0.99	1.00																																																																																																																

Tabla 43. Matriz de confusión por modelo entrenado

Modelo	Matriz de confusión																																				
1. Detección de manzanas: La matriz de confusión normalizada evalúa el desempeño de un modelo de clasificación en la diferenciación entre “manzana” y “fondo”. En la matriz, el eje vertical representa las predicciones del modelo, mientras que en el eje horizontal muestra las clases reales. El valor de 0.89 en la celda superior izquierda indica que el modelo predice correctamente las manzanas el 83% del tiempo. Sin embargo, hay un 17% de manzanas que son incorrectamente clasificadas como fondo. La celda superior derecha muestra un valor de 1.00, indicando que todas las predicciones de fondo son correctas.	 <table><caption>Confusion Matrix Normalized (Apple vs Background)</caption><tr><th>Predicted \ True</th><th>apple</th><th>background</th></tr><tr><th>apple</th><td>0.89</td><td>0.00</td></tr><tr><th>background</th><td>0.17</td><td>1.00</td></tr></table>	Predicted \ True	apple	background	apple	0.89	0.00	background	0.17	1.00																											
Predicted \ True	apple	background																																			
apple	0.89	0.00																																			
background	0.17	1.00																																			
2. Determinación de tipo de manzanas: La matriz de confusión normalizada presentada evalúa el desempeño de modelo de clasificación en la identificación de diferentes variedades de manzana (Golden Delicious, Royal Gala, Fuji, Granny Smith) y fondo. Golden Delicious tiene una predicción del 93% y un 7% que se clasifica incorrectamente como fondo. Royal Gala y Fuji son identificadas correctamente el 100% del tiempo. Granny Smith también tiene una precisión del 100% pero el fondo se clasifica incorrectamente como Granny Smith el 50% del tiempo. El fondo se clasifica correctamente el 50% del tiempo.	 <table><caption>Confusion Matrix Normalized (Apple Varieties vs Background)</caption><tr><th>Predicted \ True</th><th>Golden Delicious</th><th>Royal Gala</th><th>Fuji</th><th>Granny Smith</th><th>background</th></tr><tr><th>Golden Delicious</th><td>0.93</td><td>0.00</td><td>0.00</td><td>0.00</td><td>0.07</td></tr><tr><th>Royal Gala</th><td>0.00</td><td>1.00</td><td>0.00</td><td>0.00</td><td>0.00</td></tr><tr><th>Fuji</th><td>0.00</td><td>0.00</td><td>1.00</td><td>0.00</td><td>0.00</td></tr><tr><th>Granny Smith</th><td>0.00</td><td>0.00</td><td>0.00</td><td>1.00</td><td>0.50</td></tr><tr><th>background</th><td>0.07</td><td>0.00</td><td>0.00</td><td>0.50</td><td>0.50</td></tr></table>	Predicted \ True	Golden Delicious	Royal Gala	Fuji	Granny Smith	background	Golden Delicious	0.93	0.00	0.00	0.00	0.07	Royal Gala	0.00	1.00	0.00	0.00	0.00	Fuji	0.00	0.00	1.00	0.00	0.00	Granny Smith	0.00	0.00	0.00	1.00	0.50	background	0.07	0.00	0.00	0.50	0.50
Predicted \ True	Golden Delicious	Royal Gala	Fuji	Granny Smith	background																																
Golden Delicious	0.93	0.00	0.00	0.00	0.07																																
Royal Gala	0.00	1.00	0.00	0.00	0.00																																
Fuji	0.00	0.00	1.00	0.00	0.00																																
Granny Smith	0.00	0.00	0.00	1.00	0.50																																
background	0.07	0.00	0.00	0.50	0.50																																
3. Clasificación de manzanas por grado de calidad: La matriz muestra el desempeño de un modelo para tres clases: media, mala y fondo. El modelo clasifica correctamente el 37% de los ejemplos de media, pero confunde el 26% con mala o fondo. La clase mala se identifica con alta precisión (92%) mientras que la clase fondo tiene la mayor tasa de error, con solo un 25% de aciertos y un 70% confundido con media o mala.	 <table><caption>Confusion Matrix Normalized (Apple Quality vs Background)</caption><tr><th>Predicted \ True</th><th>media</th><th>mala</th><th>background</th></tr><tr><th>media</th><td>0.37</td><td>0.26</td><td>0.37</td></tr><tr><th>mala</th><td>0.00</td><td>0.92</td><td>0.08</td></tr><tr><th>background</th><td>0.63</td><td>0.08</td><td>0.25</td></tr></table>	Predicted \ True	media	mala	background	media	0.37	0.26	0.37	mala	0.00	0.92	0.08	background	0.63	0.08	0.25																				
Predicted \ True	media	mala	background																																		
media	0.37	0.26	0.37																																		
mala	0.00	0.92	0.08																																		
background	0.63	0.08	0.25																																		

- **Predicción por lote de validación**

La predicción por lotes de validación implica tomar un conjunto de datos de validación y alimentarlo al modelo en lotes, en lugar de alimentarlo con un solo ejemplo a la vez. Esto puede acelerar el proceso de evaluación y proporcionar una medida más precisa del rendimiento del modelo en el conjunto de datos de validación. Para YOLO la validación por lotes por defectos es 16 pero si no se tiene certeza de cuanta memoria de GPU se dispone para el entrenamiento se coloca -1 para que se ajuste a los requerimientos del dispositivo donde se entrenará el modelo. En la Figura 57, Figura 58, y Figura 59, se muestra las predicciones de la red neuronal para cada modelo entrenado.

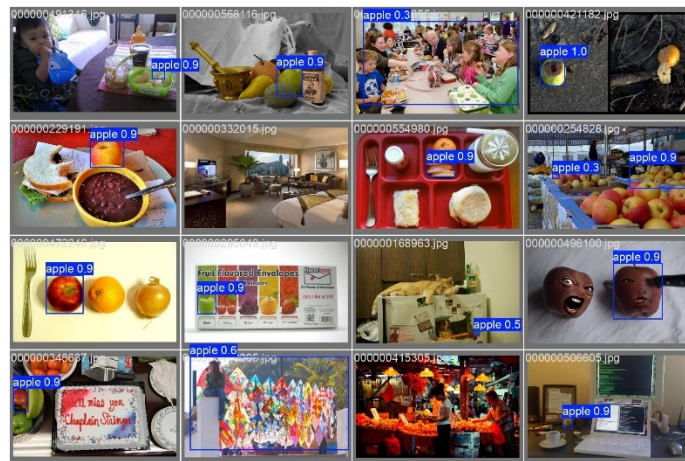


Figura 57. Predicción por lotes modelo de detección de manzanas.



Figura 58. Predicción por lotes modelo de clasificación de manzanas.

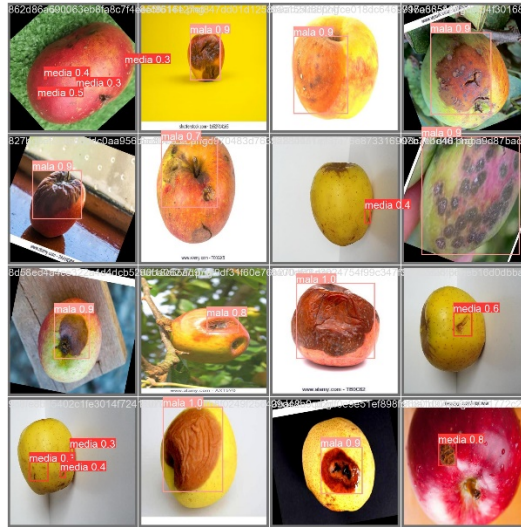


Figura 59. Predicción por lotes modelo de clasificación de manzanas por grado de calidad.

- ***Precisión del entrenamiento***

El gráfico de la curva de precisión-confianza (Precision-Confidence Curve) obtenido del entrenamiento del modelo YOLOv8 es una herramienta crucial para evaluar el rendimiento predictivo del modelo a distintos niveles de confianza. En el eje X se representa la confianza de las predicciones, mientras que en el eje Y se muestra la precisión correspondiente.

La curva resultante revela cómo varía la precisión del modelo a medida que se ajusta el umbral de confianza para las predicciones. La línea azul gruesa indica la precisión global para todas las clases, alcanzando una precisión perfecta (1.00) a un nivel de confianza de 0.946. La curva más delgada en azul claro representa la precisión específica para la clase "apple". Este análisis es vital para determinar el umbral óptimo de confianza, balanceando entre precisión y el número de predicciones, y para identificar y corregir posibles deficiencias en el modelo. La Figura 60, permite una comprensión detallada del comportamiento del modelo, facilitando la toma de decisiones informadas para mejorar su rendimiento en aplicaciones prácticas.

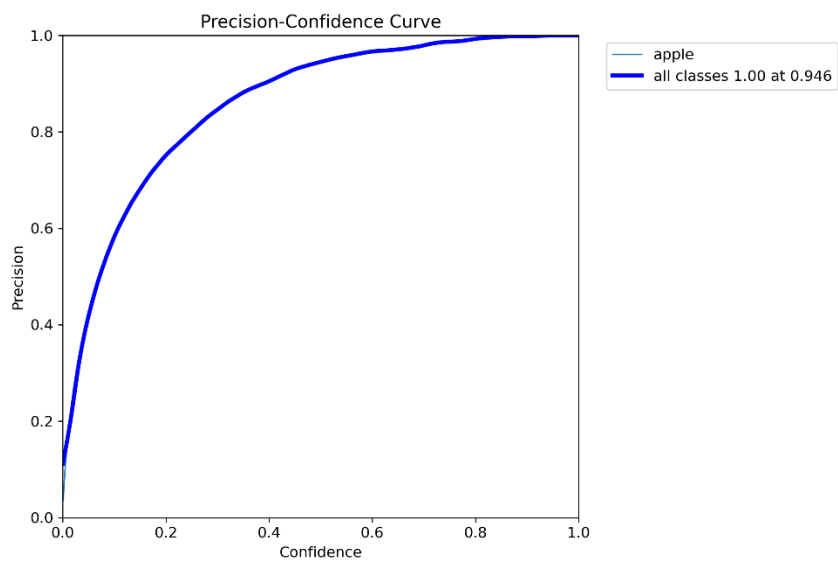


Figura 60. Precisión del modelo 1.

La curva azul gruesa de la Figura 61 indica la precisión global para todas las clases, alcanzando una precisión perfecta (1.00) a un nivel de confianza de 0.776. Las curvas de colores más claros representan la precisión específica para las clases "Golden Delicious", "Royal Gala", "Fuji" y "Granny Smith".

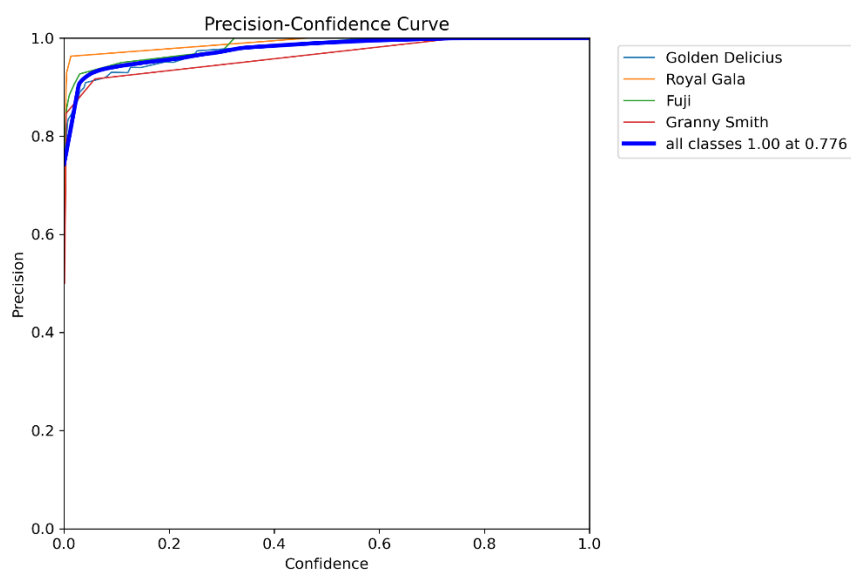


Figura 61. Precisión del modelo 2.

En la Figura 62, la curva presenta varias líneas, cada una correspondiente a diferentes clases: la línea azul gruesa representa la precisión global para todas las clases, alcanzando una precisión perfecta (1.00) a un nivel de confianza de 0.945, mientras que las líneas de colores más claros (naranja para "mala" y azul claro para "media") muestran la precisión específica para estas clases individuales. Este gráfico revela cómo varía la precisión del modelo a medida que se ajusta el umbral de confianza para las predicciones. La línea para "mala" muestra una mayor precisión en general, alcanzando casi una precisión perfecta en niveles de confianza más bajos en comparación con "media", que tiene una precisión menor y más variable a lo largo de la gama de confianza.

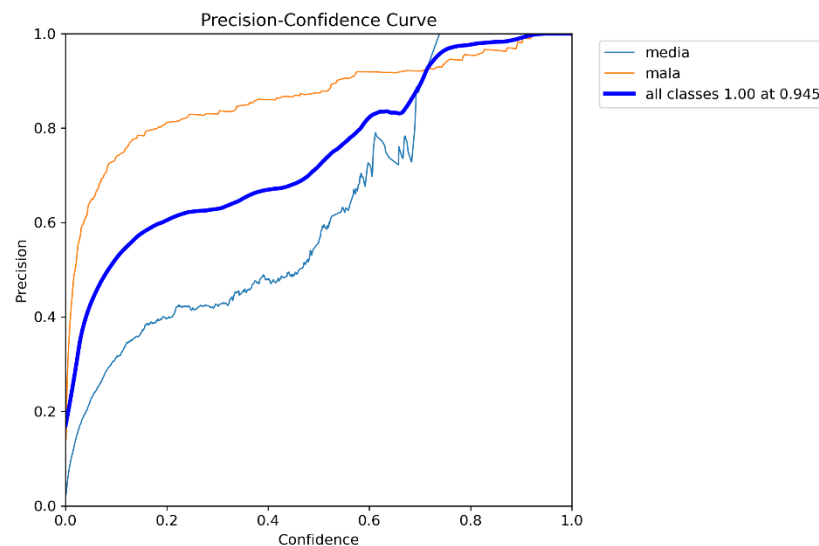


Figura 62. Precisión del modelo 3.

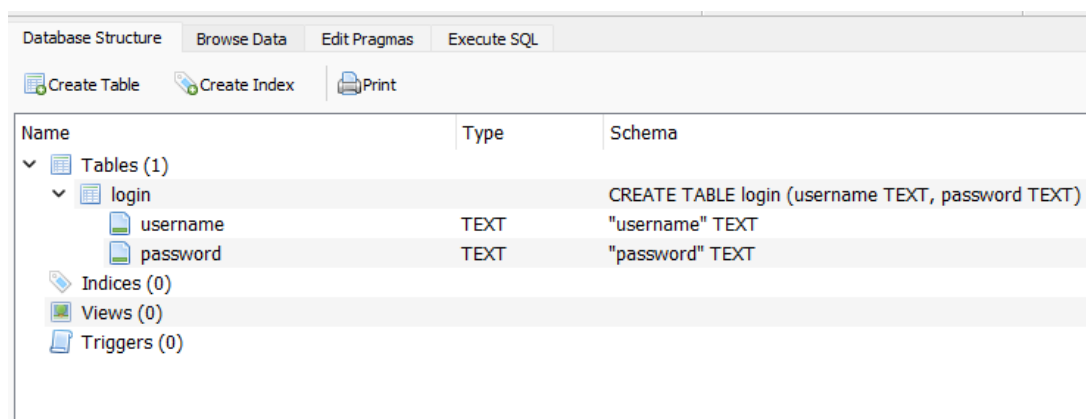
i. Visualización y Almacenamiento

En el sistema de control de calidad de manzanas, se ha implementado una solución basada en Flask y SQLite para la visualización y el almacenamiento de datos. Flask, un microframework web en Python, se utilizó debido a su simplicidad y flexibilidad, permitiendo el desarrollo rápido y eficiente de la interfaz web del sistema. Este framework facilitó la creación de rutas HTTP y la renderización de plantillas HTML, proporcionando una interfaz amigable para los usuarios donde pueden visualizar los resultados del control de calidad.

Para el almacenamiento de datos, se optó por SQLite, una base de datos ligera y sin servidor que es ideal para aplicaciones de tamaño pequeño a mediano. SQLite se integra perfectamente con Flask mediante la extensión Flask-SQLAlchemy, permitiendo un manejo sencillo de las operaciones CRUD (crear, leer, actualizar, eliminar) sobre los datos. Esta elección proporcionó una solución robusta y eficiente para almacenar los resultados del análisis de calidad de las manzanas, incluyendo las métricas de precisión y confianza generadas por el modelo de visión artificial. La combinación de Flask y SQLite ofreció una infraestructura sólida para el desarrollo de un sistema de control de calidad escalable y fácil de mantener.

- ***Autenticación de usuarios***

Para la autenticación de los usuarios con la finalidad de que existan credenciales de ingreso al sistema para la generación de reportes, se crea una tabla dentro de SQLite con los campos username y password, el proceso a realizar se puede observar en la Figura 63 y el resultado del mismo en la Figura 64 .



Name	Type	Schema
Tables (1)		
login		CREATE TABLE login (username TEXT, password TEXT)
username	TEXT	"username" TEXT
password	TEXT	"password" TEXT
Indices (0)		
Views (0)		
Triggers (0)		

Figura 63. Tabla de autenticación en SQLite.

Diagrama de Flujo de Autenticación de Usuario

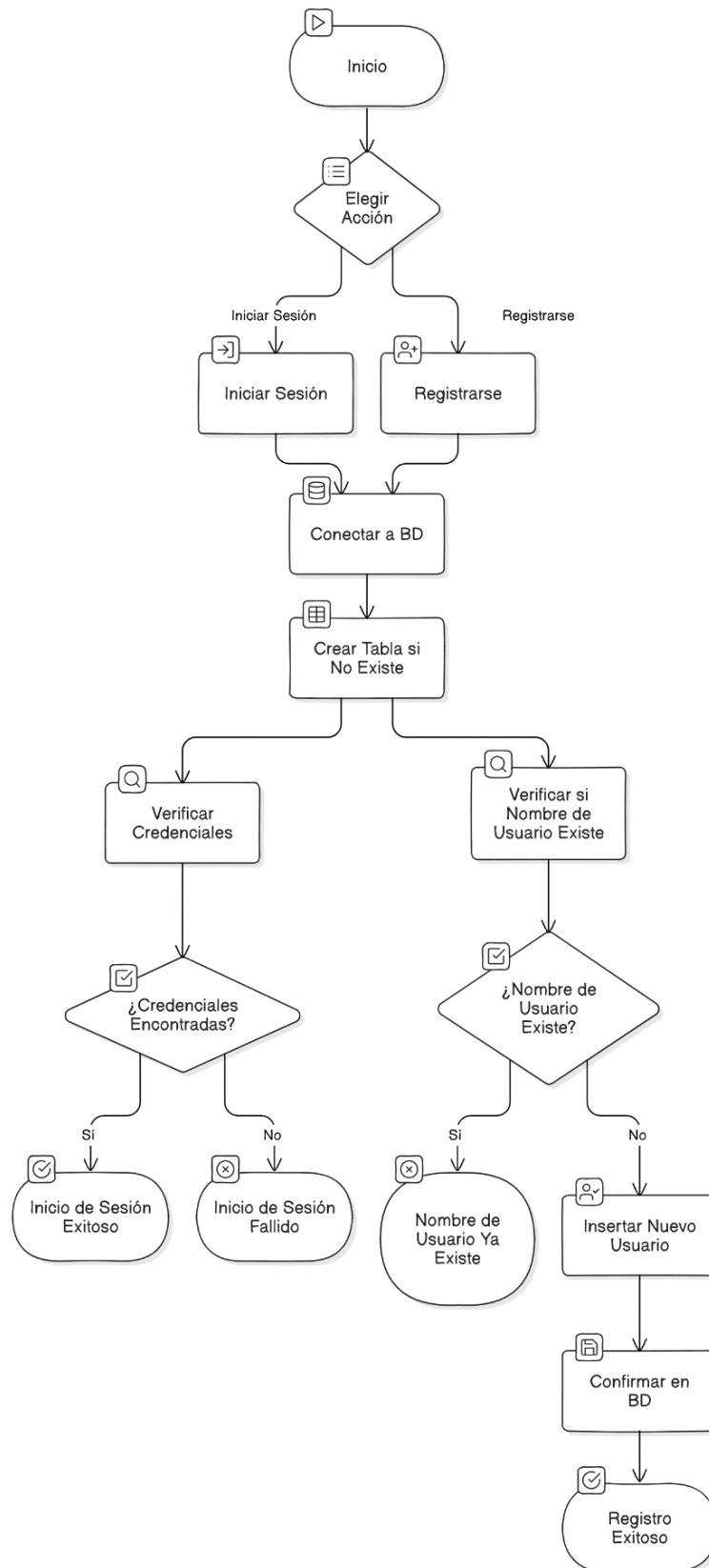
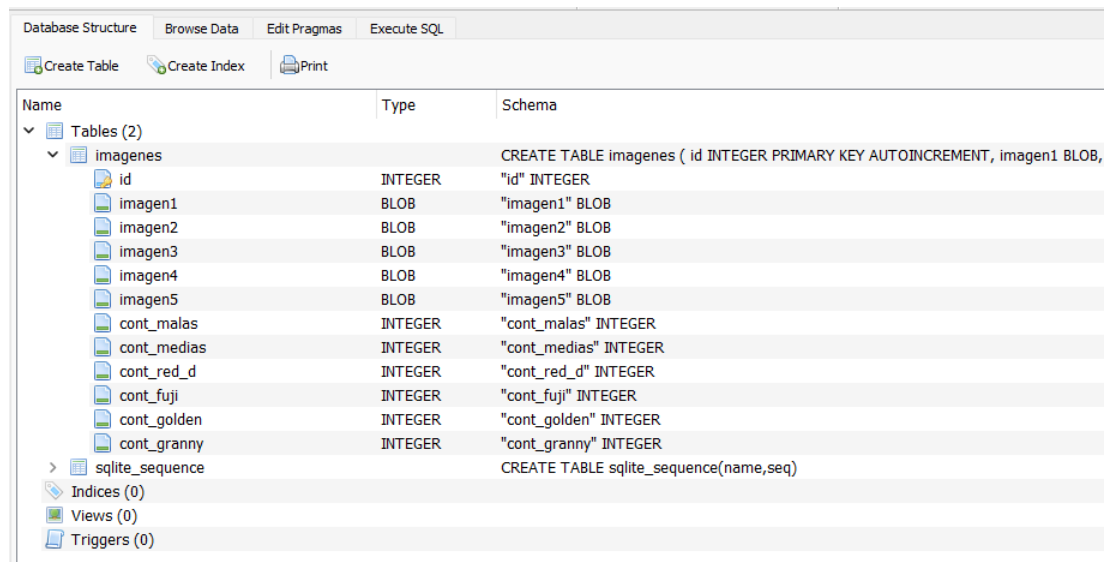


Figura 64. Proceso de autenticación de usuario.

Además, también se almacena los datos cuando los modelos entrenados empiezan a funcionar, guardando datos de la detección de manzanas, clasificación por tipo y grado de calidad, contabilización del total de manzanas clasificadas e imágenes de las primeras 5 detecciones que realiza el programa en marcha, esta estructura se puede observar en la Figura 65.



Name	Type	Schema
▼ Tables (2)		
▼ imagenes		CREATE TABLE imagenes (id INTEGER PRIMARY KEY AUTOINCREMENT, imagen1 BLOB,
id	INTEGER	"id" INTEGER
imagen1	BLOB	"imagen1" BLOB
imagen2	BLOB	"imagen2" BLOB
imagen3	BLOB	"imagen3" BLOB
imagen4	BLOB	"imagen4" BLOB
imagen5	BLOB	"imagen5" BLOB
cont_malas	INTEGER	"cont_malas" INTEGER
cont_medias	INTEGER	"cont_medias" INTEGER
cont_red_d	INTEGER	"cont_red_d" INTEGER
cont_fuji	INTEGER	"cont_fuji" INTEGER
cont_golden	INTEGER	"cont_golden" INTEGER
cont_granny	INTEGER	"cont_granny" INTEGER
sqlite_sequence		CREATE TABLE sqlite_sequence(name,seq)
Indices (0)		
Views (0)		
Triggers (0)		

Figura 65. Tabla para almacenamiento de datos de detección y clasificación de manzanas en SQLite.

- ***Interfaz de usuario***

Una vez con los datos almacenados en una base de datos, se ha desarrollado una interfaz de usuario para la visualización en el microordenador NVIDIA Jetson Nano. Por seguridad del sistema como se observa en la Figura 66, para iniciar con el funcionamiento del sistema se debe autenticar el usuario a ingresar o registrarse si aún no lo ha hecho.

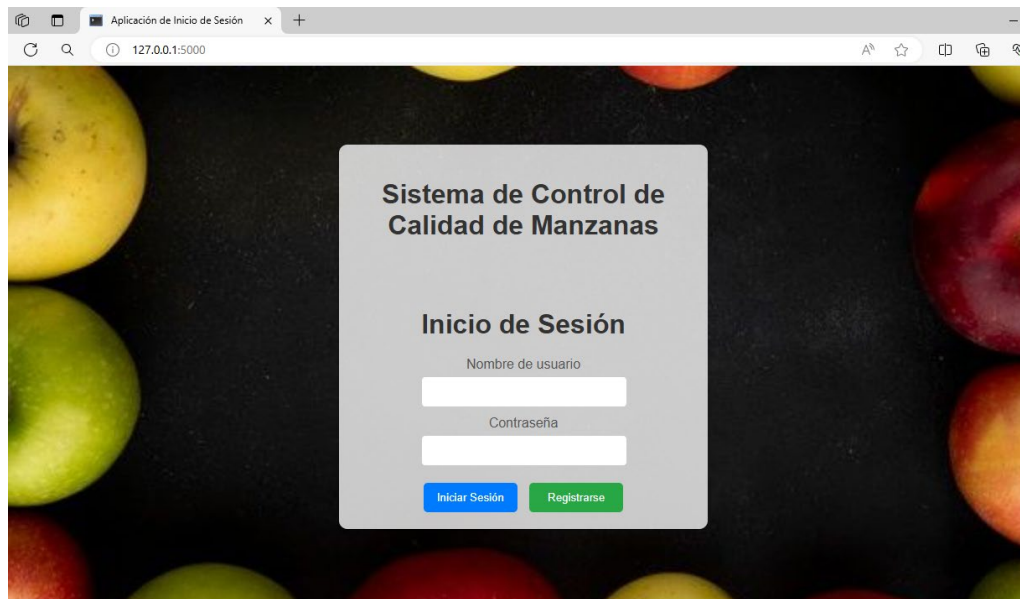


Figura 66. Interfaz (Autenticación de usuario)

Después de que el usuario ha ingresado las credenciales correctamente se le dirige dentro de la misma ventana a lo que es la visualización del sistema en general, donde podrá observar el video en tiempo real transmitido por el dron, iniciar ruta o detener el proceso y todos los datos de detección, conteo y clasificación por tipo/grado de calidad del cultivo de manzana como se observa en la Figura 67.



Figura 67. Interfaz de usuario para visualización de los datos entregados por los modelos en funcionamiento.

3.1.7 Implementación del sistema

Para la implementación del sistema que maneja video en tiempo real transmitido por el dron para detectar el tipo de manzana y clasificarlo de acuerdo con su grado de calidad que se observa en la Figura 68, fue necesario seguir los siguientes pasos detallados a continuación:

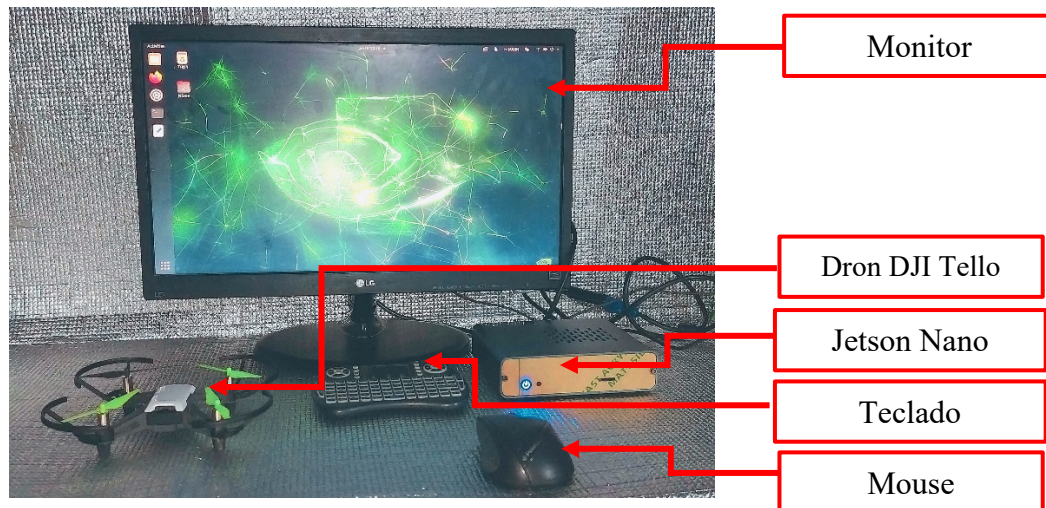


Figura 68. Prototipo del sistema.

a. Instalación del sistema operativo

Para instalar el sistema operativo en el Jetson Nano, primero se descarga la última versión de NVIDIA JetPack, que incluye el sistema operativo Ubuntu, desde el sitio web de NVIDIA. Luego, se utiliza una herramienta como Etcher para grabar la imagen descargada en una tarjeta microSD. Una vez hecho esto, se inserta la tarjeta microSD en el Jetson Nano, se conecta un monitor, teclado, ratón y la fuente de alimentación, y se enciende el dispositivo.

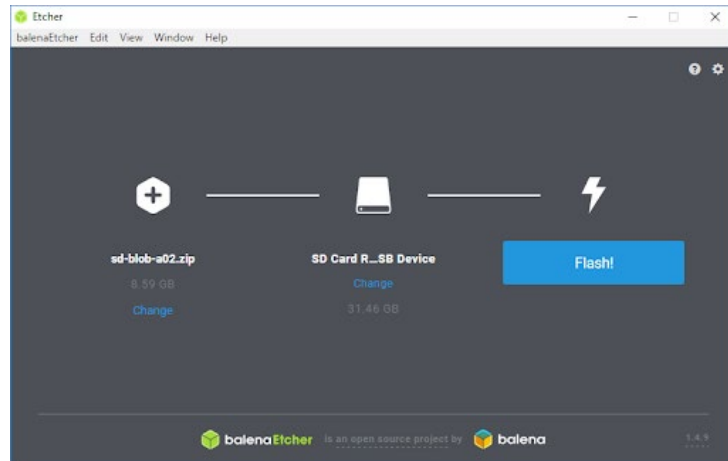


Figura 69. Flasheo de sistema operativo con herramienta Etcher [72].



Figura 70. Primer inicio Jetson Nano[72].

b. Configuración del Entorno de Desarrollo

El primer paso para configurar el entorno de desarrollo para el despliegue de los modelos entrenados juntamente con la base de datos e interfaz web es la activación de CUDA que es una tecnología que permite aprovechar el potencial de las GPUs NVIDIA para la ejecución de algoritmos de visión artificial. Por defecto CUDA no viene habilitado en la Jetson Nano y para habilitarla se procede con los siguientes comandos:

Con el comando **git clone https://github.com/mdegans/nano_build_opencv.git** se clona el repositorio para la instacion de OpenCV con soporte para CUDA.

Una vez clonado el repositorio dentro de la carpeta “nano_build_opencv”se edita la línea **-D CUDA_ARCH_BIN=5.3,6.2,7.2,8.7**, eliminando la versión 8.7 que no es

compatible con la arquitectura de la Jetson Nano y en la línea **-D CUDNN_VERSION='8.0'**, se debe cambiar a la versión que viene por defecto en el microordenador por lo general para la Jetson Nano es la 8.2., con el comando **./build_opencv.sh 4.8.0** se ejecuta el script de compilación e instalación de Opencv con CUDA aproximadamente este proceso tarda 7 horas en completarse para que finalmente se pueda comprobar que CUDA ha sido activado como se muestra en la Figura 71.

```

jtop 4.2.1 - (c) 2023, Raffaello Bonghi [raffaello@rnext.it]
Website: https://rnext.it/jetson_stats

Platform
Machine: aarch64
System: Linux
Distribution: Ubuntu 20.04 focal
Release: 4.9.253-tegra
Python: 3.8.10

Serial Number: [s]XX CLICK TO READ XXX]
Hardware
Model: NVIDIA Jetson Nano Developer Kit
699-level Part Number: 699-13448-0000-402 M.0
P-Number: p3448-0000
BoardIDs: p3448
Module: NVIDIA Jetson Nano (4 GB ram)
SoC: tegra210
CUDA Arch BIN: 5.3
Codename: Porg
L4T: 32.6.1
Jetpack: 4.6

Libraries
CUDA: 10.2.300
cuDNN: 8.2.1.32
TensorRT: 8.0.1.6
VPI: 1.1.15
Vulkan: 1.2.141
OpenCV: 4.8.0 with CUDA: YES

Hostname: nano
Interfaces
wlan0: 192.168.100.100
docker0: 172.17.0.1

```

Figura 71. OpenCV con CUDA para Jetson Nano.

- **Instalación de bibliotecas requeridas**

En la Tabla 44, se muestran los comandos de instalación de las librerías a usar para el despliegue de los modelos en la Jetson Nano.

Tabla 44. Comandos de instalación para librerías requeridas

Librería	Comando	Descripción
FilterPy	Sudo pip3 install filterpy	Instala FilterPy, una biblioteca de Python para la implementación de filtros de Kalman y otros algoritmos de filtrado.
Ultralytics	pip3 install ultralytics	Instala Ultralytics, una biblioteca de detección de objetos de alto rendimiento y fácil de usar basada en YOLO.
Flask	pip3 install flask	Instala Flask, un framework web ligero y flexible para Python.
SQLite 3	Parte de la biblioteca de Python	SQLite3 es un sistema de gestión de bases de datos relacional ligero que se utiliza a menudo para aplicaciones web y móviles.
DJITelloPy	pip3 install djitellopy	Instala DJITelloPy, una biblioteca Python para controlar el dron DJI Tello.

- ***Integración de programas para despliegue del sistema***

Ya con las librerías instaladas se puede hacer uso de todos los scripts creados a lo largo del desarrollo del proyecto, entre ellos se tiene el script principal (Anexo D) de Python donde se cargan los modelos entrenados con la arquitectura YOLOv8x.

```
model = YOLO("yolov8x.pt")
model1 = YOLO("best.pt")
model2 = YOLO("bestcalidad.pt")
```

- **Tracking:** Se inicializa el rastreador SORT para el seguimiento del objeto detectado a través de los frames.

```
tracker = Sort()
```

- **Captura de fotograma:** Se realiza la conexión mediante Wi-Fi con el dron para la transmisión y captura de video para lectura de un fotograma.

```
frame = tello.get_frame_read().frame
if frame is not None:
    frame = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)
```

- **Detección de objetos:** Utiliza un modelo YOLO (model) para detectar objetos en un frame de video (frame). Los parámetros especifican que se deben mostrar los resultados (show=False), pero no las cajas delimitadoras ni las etiquetas (show_boxes=False, show_labels=False). Se establece un umbral de confianza de detección del 30% (conf=0.3), y se especifica que el modelo debe procesar el video como un flujo continuo (stream=True). Además, se especifica que solo se deben detectar objetos de la clase 47 (classes=[47]).

```
results = model(frame, conf=0.3, show=False, show_boxes=False,
show_labels=False, stream=True, classes=[47])
for res in results:
    boxes = res.boxes.xyxy.cpu().numpy().astype(int)
    tracks = tracker.update(boxes)
    tracks = tracks.astype(int)
```

- **Determina el color de la manzana:** Utiliza la función determinar_color para determinar el color de la región de la manzana en el frame actual, utilizando el ID de seguimiento (track_id). El color calculado se asigna a la variable color_manzana y se guarda en el diccionario color_manzanas bajo la clave track_id

```
color_manzana = determinar_color(region_manzana, track_id)
color_manzanas[track_id] = color_manzana
```

- **Detecta el estado de la manzana:** Verifica si la región de la manzana es válida (`region_manzana is not None and region_manzana.size != 0`) y, si es así, realiza una segunda detección de objetos utilizando `model2`. Esta detección se realiza para clasificar el estado de la manzana en "mala", "medio" o "normal" en función de la presencia de ciertos objetos en la región de la manzana. Los resultados de esta detección se utilizan para determinar el estado de la manzana (`estado_manzanas[track_id]`) y actualizarlo en el diccionario `estado_manzanas`.

```
results2 = model2(region_manzana, show=False, show_boxes=False,
show_labels=False, stream=True, classes=[0, 1], conf=0.2)
for result2 in results2:
    reg = result2.bboxes.cls.cpu().numpy()
    num_unos, num_ceros = contar_elementos(reg)

    if num_unos >= 1:
        estado_manzanas[track_id] = "mala"
    elif num_ceros >= 1:
        estado_manzanas[track_id] = "medio"
```

- **Dibujar texto y rectángulos:** se utiliza la librería OpenCV para dibujar un rectángulo y añadir texto sobre un objeto detectado en un frame de video. El rectángulo se dibuja alrededor del objeto detectado, utilizando las coordenadas de la esquina superior izquierda y la esquina inferior derecha del rectángulo. El texto añadido proporciona información adicional sobre el objeto, como su ID de seguimiento, color y estado.

```
cv2.rectangle(frame, (xmin, ymin), (xmax, ymax), color, 3)
cv2.putText(frame, f"Manzana {track_id} - Color:
{color_manzana} Estado:{estado_manzanas.get(track_id, 'Normal')}",
(xmin, ymin - 10), cv2.FONT_HERSHEY_SIMPLEX, 1, color, 4)
```

- **Almacenamiento de datos:** se establece una conexión con una base de datos SQLite denominada 'imagenes.db' y crea una tabla llamada 'imagenes' si aún no existe. La tabla 'imagenes' está diseñada para almacenar información relacionada

con imágenes, como datos binarios de hasta cinco imágenes diferentes (imagen1, imagen2, imagen3, imagen4, imagen5), así como contadores para diferentes tipos de manzanas detectadas en las imágenes (cont_malas, cont_medias, cont_red_d, cont_fuji, cont_golden, cont_granny). La creación de esta tabla proporciona una estructura de almacenamiento adecuada para guardar y consultar información relevante relacionada con la detección y clasificación de manzanas.

```
conn = sqlite3.connect('imagenes.db')
cursor = conn.cursor()

# Crear tabla para almacenar imágenes y contadores
cursor.execute('''
CREATE TABLE IF NOT EXISTS imagenes (
    id INTEGER PRIMARY KEY AUTOINCREMENT,
    imagen1 BLOB,
    imagen2 BLOB,
    imagen3 BLOB,
    imagen4 BLOB,
    imagen5 BLOB,
    cont_malas INTEGER,
    cont_medias INTEGER,
    cont_red_d INTEGER,
    cont_fuji INTEGER,
    cont_golden INTEGER,
    cont_granny INTEGER
)
''')
```

c. Pruebas en tiempo real de cada grado de calidad de manzanas

En la Figura 72, se muestra la interfaz de usuario dentro del microordenador Jetson nano, posteriormente en la Figura 73, Figura 74, Figura 75, Figura 76 y Figura 77, se muestran capturas de imágenes de las manzanas en tiempo real con el respectivo reconocimiento del grado de calidad detectado. Para realizar las pruebas se usó manzanas de distinto tipo y con daños en la epidermis de la estructura.



Figura 72. Interfaz web integrada a Jetson Nano.



Figura 73. Detección y clasificación (Red Delicious)

Una vez finalizada la etapa de prueba 1, el modelo proporciona 4 ítems de información. En primer lugar, confirma la detección de una manzana, En segundo lugar, reporta la eficiencia que tiene el sistema en la detección del grado de calidad con un valor de precisión del 79.93%. En tercer lugar, identifica el tipo de manzana, especificando que se trata de una Red Delicious. Finalmente indica el estado de la manzana, en este caso, clasificándose como de mala calidad.



Figura 74. Detección y clasificación (Granny Smith)

Una vez finalizada la etapa de prueba 2, el modelo proporciona 4 ítems de información. En primer lugar, confirma la detección de una manzana, En segundo lugar, reporta la eficiencia que tiene el sistema en la detección del grado de calidad con un valor de precisión del 66.63%. En tercer lugar, identifica el tipo de manzana, especificando que se trata de una Granny Smith. Finalmente indica el estado de la manzana, en este caso, clasificándose como de media calidad.



Figura 75. Detección y clasificación (Golden Delicious).

Una vez finalizada la etapa de prueba 3, el modelo proporciona 4 ítems de información. En primer lugar, confirma la detección de una manzana, En segundo lugar, reporta la eficiencia que tiene el sistema en la detección del grado de calidad con un valor de precisión del 80.01%. En tercer lugar, identifica el tipo de manzana, especificando que se trata de una Golden Delicious. Finalmente indica el estado de la manzana, en este caso, clasificándola como de mala calidad.

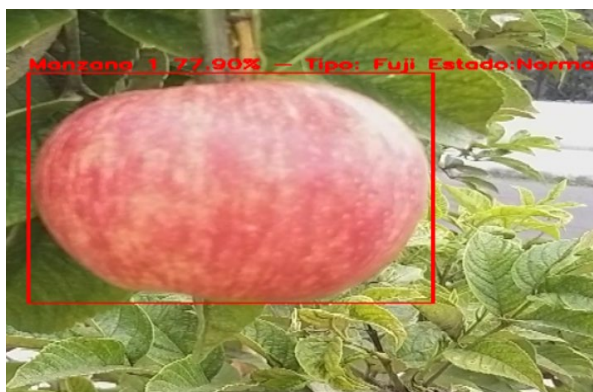


Figura 76. Detección y clasificación (Fuji).

Una vez finalizada la etapa de prueba 4, el modelo proporciona 4 ítems de información. En primer lugar, confirma la detección de una manzana, En segundo lugar, reporta la eficiencia que tiene el sistema en la detección del grado de calidad con un valor de precisión del 77.90%. En tercer lugar, identifica el tipo de manzana, especificando que se trata de una Fuji Finalmente indica el estado de la manzana, en este caso, clasificándola como de buena calidad.



Figura 77. Detección y clasificación 2 (Fuji).

Una vez finalizada la etapa de prueba 5, el modelo proporciona 4 ítems de información. En primer lugar, confirma la detección de una manzana, En segundo lugar, reporta la eficiencia que tiene el sistema en la detección del grado de calidad con un valor de precisión del 70.02%. En tercer lugar, identifica el tipo de manzana, especificando que se trata de una Fuji Finalmente indica el estado de la manzana, en este caso, clasificándola como de mala calidad

d. Prueba de velocidad de procesamiento

En el contexto del presente proyecto y las pruebas realizadas en el apartado anterior, se ha evaluado la eficiencia del modelo en términos de velocidad de inferencia, es decir, la rapidez con la que puede realizar predicciones sobre imágenes o videos. A continuación, en la Tabla 45 , se presenta los resultados del análisis de velocidad al procesar un video de prueba.

Tabla 45. Resultados de velocidad de preprocesamiento.

Detección	Shape				Interferencia	Preprocesamiento	Postprocesamiento	Tiempo Total(ms)
	Lote	Numero de canales	Altura de la imagen	Anchura de la imagen				
1 Detección	1	3	640	608	176.ms	5.8ms	5.3ms	187.1ms
1 Detección	1	3	640	384	84.4ms	1.0ms	1.0ms	86.4ms
No Detecta	1	3	640	640	84.8ms	3.0ms	1.0ms	88.8ms
No Detecta	1	3	640	384	84.3ms	2.0ms	1.0ms	87.3ms
1 Detección	1	3	576	640	75.8ms	3.0ms	2.0 ms	80.8 ms
1 Detección	1	3	640	384	84.3ms	1.9ms	2.0 ms	88.2 ms
1 Detección	1	3	608	640	75.5 ms	3.0 ms	1.0 ms	79.5 ms
1 Detección	1	3	640	384	84.7 ms	2.0 ms	1.0 ms	87.7ms
No Detecta	1	3	608	640	74.3ms	3.0ms	0.0ms	77.3ms
No Detecta	1	3	640	384	83.8ms	2.0ms	2.0ms	87.8ms

Mediante pruebas en el sistema de detección y clasificación YOLOv8 con el modelo YOLOv8x se obtuvo resultados de velocidad de preprocesamiento eficientes en términos de rapidez recopilando la información de una prueba en la que se muestra todos los tipos de manzana obtuvo un tiempo ponderado de 95.05ms. La capa de inferencia refleja la velocidad del modelo para analizar y hacer predicciones el cual tiene un valor ponderado de 90.79ms en la prueba lo cual se considera eficiente para aplicaciones en tiempo real ya que permite procesar más de 10 cuadros por segundo (FPS).

3.1.8 Resultados de las pruebas de funcionamiento

Para la interpretación de los resultados del sistema de control de calidad de fruta de temporada en etapa de precosecha se realizó una comparación de valores obtenidos con las pruebas realizadas mediante inspección visual e inspección artificial como se observa en la Tabla 46, para la determinación del grado de calidad de la manzana y eficiencia del sistema.

Tabla 46. Pruebas de funcionamiento.

Control de calidad			Nombre:	Rita Andrade						
			INSPECCIÓN HUMANA VISUAL				INSPECCIÓN ARTIFICIAL			
			GRADO DE CALIDAD				GRADO DE CALIDAD			
# Pruebas	# Manzanas	Hora	BAJA	MEDIA	ALTA	TPO	BAJA	MEDIA	ALTA	TPO
PRUEBA 1	5	09:00	3	1	1	Golden Delicious	3	1	1	Golden Delicious
PRUEBA 2	6	12:00	3	2	1	Granny Smith	3	2	1	Granny Smith
PRUEBA 3	8	14:00	4	3	1	Red Delicious	4	3	1	Red Delicious
PRUEBA 4	6	16:00	2	2	2	Fuji	2	2	2	Fuji
PRUEBA 6	5	09:00	2	2	1	Granny Smith	2	1	2	Granny Smith
PRUEBA 7	6	12:00	1	3	2	Red Delicious	1	2	3	Red Delicious
PRUEBA 8	4	14:00	3	0	1	Fuji	2	1	1	Fuji
PRUEBA 9	5	18:00	2	2	1	Golden Delicious	2	2	1	Golden Delicious
PRUEBA 10	5	09:00	2	2	1	Red Delicious	3	1	1	Red Delicious
PRUEBA 11	4	12:00	1	2	1	Golden Delicious	1	2	1	Golden Delicious
PRUEBA 12	6	14:00	2	3	1	Granny Smith	4	2	2	Granny Smith
PRUEBA 13	4	16:00	3	0	1	Fuji	3	0	1	Fuji
PRUEBA 14	4	09:00	2	2	0	Fuji	3	1	0	Fuji
PRUEBA 15	5	12:00	2	2	1	Granny Smith	2	2	1	Granny Smith
PRUEBA 16	6	14:00	2	3	1	Red Delicious	2	2	2	Red Delicious
PRUEBA 17	3	18:00	1	1	1	Golden Delicious	2	0	1	Golden Delicious
PRUEBA 18	5	09:00	1	2	2	Golden Delicious	2	1	2	Golden Delicious
PRUEBA 19	7	12:00	2	3	2	Granny Smith	2	2	3	Granny Smith
PRUEBA 20	6	14:00	2	2	2	Red Delicious	2	1	3	Red Delicious
PRUEBA 21	4	16:00	0	0	4	Fuji	0	0	4	Fuji
PRUEBA 22	5	08:00	2	2	1	Granny Smith	2	2	1	Granny Smith
PRUEBA 23	8	12:00	3	2	3	Red Delicious	3	2	3	Red Delicious
PRUEBA 24	7	14:00	2	4	1	Fuji	2	3	2	Fuji
PRUEBA 25	4	16:00	2	0	2	Golden Delicious	2	0	2	Golden Delicious
PRUEBA 26	8	09:00	3	4	1	Red Delicious	4	2	2	Red Delicious
PRUEBA 27	5	12:00	2	2	1	Golden Delicious	2	2	1	Golden Delicious
PRUEBA 28	6	14:00	2	2	2	Granny Smith	2	1	3	Granny Smith
PRUEBA 29	7	18:00	2	4	1	Fuji	3	3	1	Fuji
PRUEBA 30	4	08:00	3	1	0	Granny Smith	3	1	0	Granny Smith
PRUEBA 31	3	12:00	1	1	1	Golden Delicious	1	0	2	Golden Delicious
PRUEBA 32	6	14:00	2	3	1	Fuji	3	2	1	Fuji
PRUEBA 33	5	16:00	1	2	2	Golden Delicious	2	1	2	Golden Delicious
PRUEBA 34	4	09:00	0	2	2	Red Delicious	0	1	3	Red Delicious
PRUEBA 35	5	12:00	1	1	3	Golden Delicious	1	1	3	Golden Delicious
PRUEBA 36	8	14:00	2	5	1	Fuji	5	2	1	Fuji
PRUEBA 37	7	16:00	3	2	2	Granny Smith	4	1	2	Granny Smith
PRUEBA 38	4	09:00	0	2	2	Golden Delicious	0	2	2	Golden Delicious

PRUEBA 39	6	12:00	3	2	1	Fuji	3	1	2	Fuji
PRUEBA 40	8	14:00	4	2	2	Granny Smith	4	2	2	Granny Smith
PRUEBA 41	7	18:00	0	6	1	Red Delicious	2	3	2	Red Delicious
PRUEBA 42	6	08:00	2	2	2	Fuji	3	1	2	Fuji
PRUEBA 43	6	12:00	2	3	1	Golden Delicious	2	3	1	Golden Delicious
PRUEBA 44	8	14:00	4	1	3	Fuji	5	0	3	Fuji
PRUEBA 45	7	16:00	3	4	0	Red Delicious	3	3	1	Red Delicious
PRUEBA 46	4	09:00	0	2	2	Red Delicious	1	1	2	Red Delicious
PRUEBA 47	8	12:00	4	3	1	Granny Smith	4	2	2	Granny Smith
PRUEBA 48	7	14:00	2	2	3	Fuji	3	1	3	Fuji
PRUEBA 49	6	18:00	2	2	2	Golden Delicious	2	1	3	Golden Delicious
PRUEBA 50	5	15:00	2	1	2	Fuji	3	0	2	Fuji

a. Cálculo del error

El cálculo del error se realizó colocando las manzanas en diferentes estados en el prototipo en cada prueba realizada de esta manera se hace un análisis de las detectadas correctamente para ello se emplearon los valores en cada grado de calidad.

Se determina el error absoluto de cada prueba mediante lo siguiente:

$$\text{Error absoluto} = \text{Detectadas correctamente} - \text{Colocadas} \quad (2)$$

La determinación del error relativo se lleva a cabo mediante la siguiente formula:

$$ER = \left| \frac{\text{Valor Teorico} - \text{Valor medido}}{\text{Valor Teorico}} \right| \times 100 = 21.05\% \quad (3)$$

Tabla 47. Eficacia del sistema

Grado de calidad	Manzanas Colocadas	Detetadas correctamente	Error Absoluto	Error Relativo	Error relativo porcentual %
Baja	99	119	20	0.2020	20.20
Media	106	72	-34	0.3207	32.07
Alta	73	89	16	0.2191	21.91
PROMEDIO					24.73

Con el error relativo porcentual calculado se obtiene la eficiencia que tiene el sistema para determinar el grado de calidad en la manzana como resultado de este se tiene;

Baja Calidad: 79.79%, Media Calidad: 67.93%, Alta Calidad: 72.08%. En función de las pruebas realizadas se determinó las categorías de alta y baja calidad como categorías de decisión para la evaluación de confidencialidad del sistema. Estas categorías fueron utilizadas durante el entrenamiento del modelo de visión artificial sirviendo como parámetros evaluadores principales. Por lo que se definió la categoría de media calidad como una clasificación de descarte para aquellos casos que no cumplen con criterios de alta ni baja calidad. Este enfoque ha sido validado con un error relativo porcentual del 21.05%.

b. Confiabilidad del prototipo

La confiabilidad del sistema se establece mediante el promedio ponderado de errores porcentuales individuales.

$$\textbf{Confiabilidad}_{del\ sistema} = 100\% - \textbf{error}_{promedio\ porcentual\ del\ sistema} \quad (4)$$

$$\textit{Confiabilidad}_{del\ sistema} = 100\% - 21.05\%$$

$$\textit{Confiabilidad}_{del\ sistema} = 78.95\%$$

La confiabilidad del prototipo se basa en el rendimiento observado durante el entrenamiento y la precisión alcanzada por el algoritmo de visión artificial YOLOv8, empleado para clasificar los grados de calidad de la manzana. La validación del algoritmo demuestra que el sistema logra una tasa de precisión del 78.95%, lo que garantiza que el prototipo sea fiable para su aplicación en investigaciones agrícolas utilizando técnicas avanzadas de visión artificial y uso de UAV.

3.1.9 Costo computacional

El uso de Google Colab con GPU acelera significativamente el entrenamiento de modelos de aprendizaje profundo, gracias a su capacidad de procesamiento paralelo. A continuación, en la Tabla 48 se presenta una estimación del costo computacional al

entrenar tres modelos de redes neuronales convolucionales (CNN) durante distintas épocas cada uno.

Tabla 48. Costo computacional

Parámetro		Descripción	
MÁQUINA	CPU	CPU Intel Xeon a 2,20 GHz	
	RAM	13 GB	
	GPU	NVIDIA Tesla T4 o P100 (varía)	
	TPU	CPU Intel Xeon a 2,30 GHz	
	Memoria de GPU	15 GB	
	Sistema Operativo	Linux (Ubuntu)	
	Almacenamiento	78.2 GB	
Tipo de modelo		YOLOv8x	
Tamaño del Dataset			
Modelo 1	Modelo 2	Modelo 3	
13000 imágenes	800 imágenes	1500 imágenes	
Número de épocas			
Modelo 1	Modelo 2	Modelo 3	Total
70	50	50	170
Tiempo de entrenamiento			
Modelo 1	Modelo 2	Modelo 3	Total
151.8 min	51.3 min	95.4 min	298.5 min / ~5 horas
Uso de GPU		~14.2 GB/15GB	90-100%
Memoria RAM utilizada		~5.1/13 GB	40-70%

3.1.10 Presupuesto

El sistema de control de calidad de manzanas tiene un presupuesto que se calcula considerando los costos de diseño y fabricación. Para el diseño, se han tenido en cuenta las horas dedicadas a su fabricación y el salario de un Ingeniero en Telecomunicaciones, que, según el Ministerio de Trabajo de Ecuador, parte de 925 dólares. Para esto, se consideran 20 días laborables para determinar el salario diario.

$$Salario_{xdía} = \frac{Salario_{xmes}}{Días laborables} \quad (5)$$

$$Salario_{xdía} = \frac{925}{20}$$

$$Salario \times día = 46,25\$$$

Para determinar el salario por hora se toma a consideración la jornada laboral por día que son 8 horas en relación con el salario por día:

$$\text{Salario}_{x\text{hora}} = \frac{\text{salario}_{\text{dia}}}{\text{Horas}_{\text{laborables}}} \quad (6)$$

$$\text{Salario}_{x\text{hora}} = \frac{46,25 \text{ dólares}}{8 \text{ horas}} = 5,78 \text{ dólares}$$

Se calcula un lapso de 150 horas destinado al estudio, diseño, ejecución y verificación de funcionamiento, dando como resultado el presupuesto del diseño.

$$\text{Presupuesto}_{\text{diseño}} = \text{Horas}_{\text{ximplementación}} \times \text{Salario}_{x\text{hora}} \quad (7)$$

$$\text{Presupuesto}_{\text{total diseño}} = 150 \text{ horas} \times 5,78 \text{ dólares} = 867,18 \text{ dólares}$$

En la Tabla 49 se detalla, costos relacionados a los materiales utilizados para la implementación del sistema.

Tabla 49. Presupuesto del sistema.

Ítem	Recursos	Cantidad	Precio Unitario	Precio Total
1	Kit de desarrollador Jetson Nano	1	\$ 210	\$210
2	DJI Tello Quadcopter Drone Boost Combo	1	\$ 250	\$250
3	Teclado	1	\$ 5	\$ 5
4	Monitor	1	\$ 80	\$ 80
5	Mouse	1	\$ 6	\$ 6
6	Adaptador HDMI a VGA	1	\$ 10	\$10
7	RCstyle 2CW 2CCW Rotación Coreless 8520 Motos Compatible con DJI Tello Drone Piezas de Repuesto (2 Pares)	1	\$ 15	\$15
8	28PCS 7 Color Hélice para DJI Tello RC Quadcopter	1	\$ 12	\$12
9	Tarjeta microSD de 64 GB con adaptador SD	1	\$ 10	\$ 10
			Subtotal	\$ 604
			IVA 13%	\$ 78.52
			Imprevistos 10%	\$ 68.25
			Total	\$ 750.77

El presupuesto total para la implementación del sistema se ha de terminado con la siguiente ecuación:

$$\begin{aligned} \mathbf{Presupuesto}_{total} & \qquad \qquad \qquad (8) \\ & = \mathbf{Presupuesto}_{total\ diseño} + \mathbf{Presupuesto}_{implementacion} \end{aligned}$$

$$Presupuesto_{total} = 867,18 \text{ dólares} + 750,77 \text{ dólares}$$

$$Presupuesto_{total} = 1617,95 \text{ dólares}$$

CAPÍTULO IV. CONCLUSIONES Y RECOMENDACIONES

4.1 Conclusiones

- En el desarrollo del proyecto se ha llevado a cabo un análisis de los métodos de clasificación y control de calidad de la producción de la fruta de temporada, como caso de estudio la manzana. En Ecuador, los métodos de clasificación y control de calidad de las manzanas destinadas a la comercialización y exportación se basan en técnicas rigurosas que aseguran la calidad del producto final. La clasificación comienza con la inspección visual, donde se evalúan características como el color, tamaño, forma y ausencia de defectos externos. Esta fase inicial es crucial, ya que determina la primera impresión del producto en el mercado. La evaluación de calidad se basó en la norma técnica NTE INEN 1872 que define tres grados de calidad para las manzanas. La clasificación también considera el calibre de las manzanas, es decir, su tamaño, lo que permite agruparlas según las preferencias de los consumidores y los requisitos de los mercados de destino.
- Los algoritmos de visión artificial, como YOLOv8, se basan en redes neuronales convolucionales (CNN), que son modelos de aprendizaje profundo diseñados para procesar imágenes y extraer características relevantes para la clasificación de objetos. Estos algoritmos utilizan capas convolucionales para detectar patrones visuales, como bordes, texturas y capas de pooling para reducir la dimensionalidad de las características extraídas. En el contexto de la producción de manzanas, estos algoritmos pueden entrenarse con un conjunto de datos que incluya imágenes de diferentes variedades de manzanas y sus posibles defectos, tales como manchas, golpes o deformidades, eliminando la necesidad de inspecciones visuales humanas antes de la cosecha. El modelo usado es YOLOv8x debido a su alta precisión, rapidez en la inferencia, mejoras en la arquitectura, flexibilidad y escalabilidad, fue entrenado con 25 épocas tiene una precisión de 0.845, un recall de 0.86, un mAP50 de 0.616 y un mAP50-95 de 0.375, comparada con el modelo YOLOv8n, entrenado durante 60 épocas que presentó una precisión de 0.855, un recall de 0.89, un

mAP50 y un mAP 50-95 de 0.21 y el modelo YOLOv8l con 100 épocas de entrenamiento, que tiene una precisión de 0.845, un recall de 0.86, un mAP50 de 0.493 y un mAP50-95 de 0.30. YOLOv8x es el mejor modelo en términos de rendimiento global especialmente en métricas de mAP50 y mAP50-95 indicando que su precisión para detectar objetos a un umbral de IoU del 50% es superior a otros. El sistema permite procesar una imagen en 1.59ms, con un promedio ponderado de inferencia de 95.05ms y un post procesamiento de 1.81ms dando como resultado el tiempo total de 98.45ms para procesar cada frame del video. El algoritmo implementado en conjunto con el DJI Tello EDU para la planificación de trayectorias y la inspección automatizada de la calidad de las manzanas en la producción frutícola de Ecuador, representa un avance significativo en la optimización de los procesos agrícolas. La capacidad de planificación de trayectoria del DJI Tello EDU, combinada con la precisión y eficiencia de los algoritmos de visión artificial, optimiza el proceso de despliegue de manera autónoma del dron a las áreas definidas en el algoritmo de planificación de trayectoria. Este enfoque reduce significativamente el tiempo y los recursos necesarios para la inspección manual, al tiempo que mejora la calidad del producto final al garantizar la selección precisa de manzanas aptas para la comercialización y exportación.

- La interfaz web desarrollada utilizando SQLite como base de datos y Flask como framework ha demostrado ser efectiva para la interpretación y gestión de información con un enfoque en la detección y clasificación de manzanas mediante la transmisión de vídeo desde un UAV en tiempo real. La interfaz presenta información en forma de tablas que muestran los diferentes tipos de manzanas y su calidad por defectos, facilitando la interpretación de los datos. Además, cuenta con un espacio dedicado para visualizar el video en tiempo real transmitido por el UAV, lo que permite a los usuarios observar la detección y clasificación de manzanas de manera interactiva. La integración de SQLite y Flask ha sido fundamental para el desarrollo de esta interfaz, ya que SQLite proporciona una base de datos ligera y eficiente para almacenar y gestionar los datos de la producción de manzanas, mientras que Flask facilita la creación de la interfaz web y la integración con el video en tiempo real.

- El sistema demostró una confiabilidad del 78.95%, consolidándose como un prototipo funcional que es capaz de evaluar la calidad y tipo de manzanas de manera eficiente. Esta capacidad contribuye a prevenir varios problemas como problemas fisiológicos, plagas y enfermedades ayudando a garantizar satisfacción al cliente, valor comercial, seguridad alimentaria y optimización de recursos.

4.2 Recomendaciones

- Para el entrenamiento de algoritmos de Deep Learning se recomienda tener un data set con un número de datos superior a 1500 imágenes, para mayor eficiencia en el modelo de control de calidad. De igual manera para el modelo de detección de manzanas se recomienda ampliar el data set de COCO con imágenes específicas del entorno donde vaya a ser aplicado tomando en cuenta todos los detalles posibles.
- Se sugiere desarrollar e implementar sistemas de monitoreo y control integrados basados en la interfaz web para proporcionar a los productores una herramienta completa para la gestión de la producción de manzanas. Estos sistemas podrían incluir capacidades de alerta temprana para condiciones ambientales adversas, seguimiento en tiempo real del estado de las manzanas en el campo y recomendaciones personalizadas para optimizar la calidad y el rendimiento de los cultivos.
- Se propone realizar una validación exhaustiva de los algoritmos de visión artificial desarrollados, utilizando conjuntos de datos representativos y evaluando su desempeño en condiciones reales de producción. Además, se recomienda implementar un ciclo de mejora continua para ajustar y optimizar los algoritmos en función de los resultados obtenidos y de las necesidades específicas de la industria frutícola.
- Es recomendable explorar el desarrollo de capacidades de análisis predictivo utilizando técnicas de aprendizaje automático para predecir tendencias futuras en la producción de manzanas y anticipar posibles problemas o necesidades.

Esto podría incluir la implementación de modelos predictivos para estimar la calidad de la fruta en función de factores como el clima, la ubicación y las prácticas agrícolas.

- Se sugiere investigar y evaluar la integración de tecnologías emergentes, como la Internet de las cosas (IoT) y la inteligencia artificial (IA) en la producción de manzanas. Estas tecnologías podrían utilizarse para mejorar la eficiencia de los procesos agrícolas, monitorear de forma remota las condiciones del cultivo y optimizar la gestión de recursos como el agua y los fertilizantes.
- Si el sector donde se pretende hacer volar el dron tiene como característica vientos moderadamente fuertes se recomienda no hacer volar el dron ya que no llega la referencia establecida con facilidad y se dificulta la planificación de trayectorias. Además, si la velocidad del viento supera los 2.5 m/s no se debe elevar el dron ya que los motores no tienen la fuerza necesaria para vencer al viento.
- Se sugiere que el dron se mantenga funcionando cuando la batería se encuentra entre un 100% - 60% caso contrario empieza a presentar fallos tanto en el vuelo como en la detección. Por lo que para trabajos futuros se recomienda optar por un dron de categoría superior caracterizado por un chasis más robusto, equipado con un sistema de estabilizador de última generación, cámara de alta resolución, rendimiento de batería y capacidad de programación.

REFERENCIAS BIBLIOGRÁFICAS

- [1] S. Santos Valle and J. Kienzle, “Agricultura 4.0 Robótica agrícola y equipos automatizados para la producción agrícola sostenible,” Roma, 2021.
- [2] CropLife Latinoamerica, “Drones en agricultura, una tecnología con potencial para apoyar especialmente a los pequeños agricultores,” Bogotá, 2021.
- [3] FAO Ecuador, “Ecuador en una mirada,” Quito, 2023.
- [4] J. P. Tacuri Auquilla and W. J. Olmedo Carrillo, “Implementación de un dron anticollisiones capaz de seguir una trayectoria pre establecida para la elaboración de un mapa de la tasa de ocupación vegetal en una zona.,” 2019, Accessed: Mar. 31, 2024. [Online]. Available: <http://dspace.esPOCH.edu.ec/handle/123456789/10898>
- [5] S. Bryan Cristhian Martin Cuya Marzal, M. Guillermo Ramos Lugo Código, A. Alvarez Valdivia, and E. Manuel Lima -Perú, “Sistema de control de calidad utilizando redes neuronales para la clasificación del estado de la granadilla,” *Repositorio Institucional - Ulima*, 2020, Accessed: Mar. 31, 2024. [Online]. Available: <https://repositorio.ulima.edu.pe/handle/20.500.12724/13854>
- [6] J. G. Castro Vega, “Algoritmo con IA para planeación de trayectorias de un quadrotor para detección de plagas y estado nutricional de cultivos de Annona Muricata.” Accessed: Mar. 31, 2024. [Online]. Available: <https://repository.unimilitar.edu.co/handle/10654/40391>
- [7] A. Alejandro *et al.*, “Clasificación de calidad de manzana para monitoreo de cosechabilidad utilizando visión por computador y algoritmos de aprendizaje profundo Apple quality grading for harvestability monitoring using computer vision and deep learning algorithms”, Accessed: Jun. 04, 2024. [Online]. Available: <https://orcid.org/0000-0001-7101-0319>
- [8] A. Victoria, C. Martín, and J. M. Díaz Báñez, “Planificación eficiente de trayectorias libres de colisiones para un equipo de drones. Comparación de algoritmos”.

- [9] S. 'Valle and J. 'Kienzle, *Agricultura 4.0: Robótica agrícola y equipos automatizados para la producción agrícola sostenible*, vol. N. 24. Roma: FAO, 2021.
- [10] G. E. 'Pat Money, *La insostenible Agricultura 4.0: Digitalización y poder corporativo en la cadena alimentaria*. Ciudad de México: Grupo ETC, 2019.
- [11] C. 'Chancellor, *EL FUTURO DE LA AGRICULTURA: DEL CONTROL DE LOS DATOS A LA SOBERANÍA ALIMENTARIA*, Noble Studio. Amigos de la Tierra, 2020.
- [12] V. Khanh Quy *et al.*, "IoT-Enabled Smart Agriculture: Architecture, Applications, and Challenges," 2022, doi: 10.3390/app12073396.
- [13] M. L. A. Saus and U. de C.-L. Mancha, *Evaluación de los parámetros físicos y químicos del fruto de melocotonero (variedad Cofrentes)[: influencia de la fertilización y orientación*. in Colección Tesis doctorales. Universidad de Castilla-La Mancha, 1997. [Online]. Available: <https://books.google.com.ec/books?id=6Z2PMT7YqF4C>
- [14] R. Díaz-Sobac and J. Vernon-Carter, "CYTA-Journal of Food INOCUIDAD MICROBIOLÓGICA DE FRUTAS FRESCAS Y MÍNIMAMENTE PROCESADAS MICROBIOLOGICAL SAFETY OF FRESH AND MINIMUM-PROCESSED FRUITS INOCUIDADE MICROBIOLÓGICA DE FRUTAS FRESCAS E MÍNIMAMENTE PROCESADAS," 2009, doi: 10.1080/11358129909487594.
- [15] "Control de calidad para frutas y hortalizas – Servicio de Acreditación Ecuatoriano." Accessed: Jun. 16, 2024. [Online]. Available: <https://www.acreditacion.gob.ec/control-calidad-para-frutas-y-hortalizas/>
- [16] B. Zhang *et al.*, "Principles, developments and applications of computer vision for external quality inspection of fruits and vegetables: A review," *Food Research International*, vol. 62, pp. 326–343, 2014, doi: <https://doi.org/10.1016/j.foodres.2014.03.012>.

- [17] B. Bruckner and R. Shewfelt, "Fruit and Vegetable Quality," *Fruit and Vegetable Quality*, Apr. 2000, doi: 10.1201/9781482293937/FRUIT-VEGETABLE-QUALITY-BERNHARD-BRUCKNER-ROBERT-SHEWFELT.
- [18] E. M. Yahia and H. C. H. C, *Fisiología y tecnología postcosecha de productos hortícolas*. Editorial Limusa, 1992. [Online]. Available: <https://books.google.com.ec/books?id=pyfovkiZkgsC>
- [19] R. M. J. Díaz and J. L. de Espinosa, *Agricultura sostenible*. Agrofuturo, 1998. [Online]. Available: <https://books.google.com.ec/books?id=BCdIAAAAYAAJ>
- [20] P. García, J. Avilés, V. Mendoza, J. Hechavarría, and A. Espín, *Drones para la agricultura de precisión, un enfoque al cultivo de pitahaya*. Guayaquil, 2022.
- [21] Miranda, Garrido, Aguilar, and Herrero, *Drones. Modelado y control de cuadricópteros*. Marcombo, 2020.
- [22] Miranda, Garrido, Aguilar, and Herrero, *Drones. Modelado y control de cuadricópteros*. Marcombo, 2020.
- [23] PARROT S.A., "AR.Drone 2.0 from Parrot." Accessed: May 13, 2024. [Online]. Available: <https://www.electronicsdatasheets.com/manufacturers/parrot/parts/ardrone-20>
- [24] DJI, "Mavic 2 - Product Information - DJI." Accessed: Mar. 22, 2024. [Online]. Available: <https://www.dji.com/global/mavic-2/info>
- [25] L. de la Clergerie, "DJI Ryze Tello - Drone - LDLC | ¡Musericordia!" Accessed: Mar. 22, 2024. [Online]. Available: <https://www.ldlc.com/es-es/ficha/PB00245643.html>
- [26] L. Rouhiainen, *INTELIGENCIA ARTIFICIAL*. España: Editorial Planeta, S.A, 2018.
- [27] M. Lozano, F. Escolano, M. Cazorla, and O. Colombina, *Inteligencia artificial (Modelos, Técnicas y Áreas de aplicación)*. Madrid, 2016.

- [28] H. Allende and C. Moraga, “ARTIFICIAL NEURAL NETWORKS IN TIME SERIES FORECASTING: A COMPARATIVE ANALYSIS,” *KYBERNETIKA*, pp. 1–3, 2002.
- [29] A. Kamilaris and F. Prenafeta, “Deep learning in agriculture: A survey,” *Computer and Electronics in agriculture*, vol. 147, pp. 70–90, 2018.
- [30] J. Vélez, A. Moreno, Á. Sánchez, and J. Sánchez, *Visión por Computador*. Madrid: DYKINSON, 2003.
- [31] M. Domingo, *Visión por Computador*. Santiago de Chile, 2004.
- [32] R. Gonzales and R. Woods, *Digital Image Processing*. Pearson Education, 2018.
- [33] O. J. Reyes-Ortiz, M. Mejia, and J. S. Useche-Castelblanco, “Técnicas de inteligencia artificial utilizadas en el procesamiento de imágenes y su aplicación en el análisis de pavimentos,” *Revista EIA*, vol. 16, pp. 189–207, 2019.
- [34] E. García, *Visión Artificial*. Catalunya: Fundació para la Universitat Oberta de Catalunya.
- [35] S. Costa, *AI y Blockchain en la agricultura: desafíos y oportunidades*. Santos Costa, 2023. [Online]. Available: <https://books.google.com.ec/books?id=cX3cEAAAQBAJ>
- [36] INFAIMON, *SOLUCIONES DE VISIÓN ARTIFICIAL PARA EL SECTOR AGROALIMENTARIO*. INFAIMON, 2021.
- [37] M. Shorif, *Computer Vision and Machine Learning in Agriculture*. Springer Nature, 2022.
- [38] I. TUCOWS, “Buy a Raspberry Pi 5 – Raspberry Pi.” Accessed: Mar. 22, 2024. [Online]. Available: <https://www.raspberrypi.com/products/raspberry-pi-5/>
- [39] J. Huang, “Jetson Nano Lleva el Poder de la IA Moderna a los Dispositivos en el Edge | NVIDIA.” Accessed: Mar. 22, 2024. [Online]. Available:

<https://www.nvidia.com/es-la/autonomous-machines/embedded-systems/jetson-nano/product-development/>

[40] Kubii, “AI-Motion K210 development kit for artificial intelligence.” Accessed: Mar. 22, 2024. [Online]. Available: <https://www.kubii.com/en/development-kit/3174-ai-motion-k210-development-kit-for-artificial-intelligence-3272496305014.html>

[41] W. Viera, “Fruticultura – Instituto Nacional de Investigaciones Agropecuarias.” Accessed: Jun. 19, 2024. [Online]. Available: <https://www.iniap.gob.ec/fruticultura/>

[42] Agromax Ecuador, “Frutas de Exportación - ECUADOR.” Accessed: Jun. 19, 2024. [Online]. Available: <https://agromaxecuador.com/index.php/2021/01/23/frutas-de-exportacion/>

[43] Gobierno Provincial de Tungurahua, “Frutas en el Mercado Central de Ambato.” Accessed: Mar. 30, 2024. [Online]. Available: <https://tungurahuaturismo.com/es-ec/tungurahua/ambato/gestores-gastronomicos/frutas-mercado-central-ambato-aigslqf5k>

[44] S. Espinosa, “Dirección Zonal de Planificación 3 – Secretaría Nacional de Planificación.” Accessed: Mar. 30, 2024. [Online]. Available: <https://www.planificacion.gob.ec/https-www-planificacion-gob-ec-coordinacion-zonal-planificacion-3/>

[45] Z. Alta Del Ecuador, “El cultivo del Manzano en la INIAP-Estación Experimental Santa Catalina”.

[46] A. Schwartz, R. Boyarski, and E. Mardix, “Control de calidad de las manzanas | Clarifruit.” Accessed: Jun. 17, 2024. [Online]. Available: <https://www.clarifruit.com/es/knowledge-base/categorias-de-productos-frescos/manzanas/>

[47] M. Gonzales, “Factores precosecha: en la calidad de la manzana,” *Unifrut*, vol. 21, Mar. 2023.

- [48] C. Suiza En Bolivia, “COSECHA, POSTCOSECHA Y COMERCIALIZACIÓN DE LA DE LA MANZANA”.
- [49] Python Software Foundation, “3.12.4 Documentation.” Accessed: Jun. 17, 2024. [Online]. Available: <https://docs.python.org/3/>
- [50] MDN contributors, “JavaScript documentation — DevDocs.” Accessed: Jun. 17, 2024. [Online]. Available: <https://devdocs.io/javascript/>
- [51] I. The MathWorks, “MATLAB Documentation - MathWorks América Latina.” Accessed: Jun. 17, 2024. [Online]. Available: <https://la.mathworks.com/help/matlab/>
- [52] I. Amazon Web Services, “Capa gratuita de AWS | Cloud computing gratis | AWS.” Accessed: Jun. 17, 2024. [Online]. Available: https://aws.amazon.com/es/free/?gclid=Cj0KCQjw4MSzBhC8ARIsAPFOuyWGX49Pgep9Rf1nIxJFg2RIg2SzA-u_HF8Ou4VQ5JoGLjmiOzqhTvEaAiONEALw_wcB&trk=ebd9fef4-469e-4e19-8d17-775ea6698635&sc_channel=ps&ef_id=Cj0KCQjw4MSzBhC8ARIsAPFOuyWGX49Pgep9Rf1nIxJFg2RIg2SzA-u_HF8Ou4VQ5JoGLjmiOzqhTvEaAiONEALw_wcB:G:s&s_kwid=AL!4422!3!648041763574!p!!g!!amazon%20web%20services!19685311598!147473978873&all-free-tier.sort-by=item.additionalFields.SortRank&all-free-tier.sort-order=asc&awsf.Free%20Tier%20Types=*all&awsf.Free%20Tier%20Categories=*all
- [53] I. Gartner, “Azure Kubernetes Service (AKS) | Microsoft Azure.” Accessed: Jun. 17, 2024. [Online]. Available: <https://azure.microsoft.com/es-es/products/kubernetes-service>
- [54] L. Upson, “Servicios de cloud computing | Google Cloud.” Accessed: Jun. 17, 2024. [Online]. Available: <https://cloud.google.com/>
- [55] I. Ultralytics, “YOLOv8 - Ultralytics YOLO Documentos.” Accessed: Jun. 13, 2024. [Online]. Available: <https://docs.ultralytics.com/es/models/yolov8/>

- [56] I. Roboflow, “MobileNet SSD v2 Object Detection Model: What is, How to Use.” Accessed: Jun. 17, 2024. [Online]. Available: <https://roboflow.com/model/mobilenet-ssd-v2>
- [57] R. Girshick, “Fast R-CNN”, Accessed: Jun. 17, 2024. [Online]. Available: <https://github.com/rbgirshick/>
- [58] N. Chapre, “Integrating HSV Filtering into our Detection model for Team Identification | by Nikhil Chapre | Medium.” Accessed: Jun. 13, 2024. [Online]. Available: <https://medium.com/@nikhilc2209/integrating-hsv-filtering-into-our-detection-model-for-team-identification-b9cf78cacfb0>
- [59] A. Senyayev, Smorkalov, G. Ser, and Skvark, “opencv-python · PyPI.” Accessed: Jun. 17, 2024. [Online]. Available: <https://pypi.org/project/opencv-python/>
- [60] NumPy Developers, “NumPy documentation — NumPy v2.0 Manual.” Accessed: Jun. 17, 2024. [Online]. Available: <https://numpy.org/doc/stable/>
- [61] R. Hipp, “SQLite Documentation.” Accessed: Jun. 17, 2024. [Online]. Available: <https://www.sqlite.org/docs.html>
- [62] I. Oracle, “MySQL :: MySQL 8.4 Reference Manual.” Accessed: Jun. 17, 2024. [Online]. Available: <https://dev.mysql.com/doc/refman/8.4/en/>
- [63] I. MongoDB, “MongoDB Atlas: Cloud Document Database | MongoDB.” Accessed: Jun. 17, 2024. [Online]. Available: https://www.mongodb.com/es/lp/cloud/atlas/try4?utm_source=google&utm_campaign=search_gs_pl_evergreen_atlas_core_retarget-brand_gic-null_amers-all_ps-all_desktop_eng_lead&utm_term=mongodb&utm_medium=cpc_paid_search&utm_ad=e&utm_ad_campaign_id=14412646314&adgroup=131761122172&cq_cmp=14412646314&gad_source=1&gclid=Cj0KCQjw4MSzBhC8ARIsAPFOuyW_-bzUU__HvCgxT9QI5O4c_qnW89jideEQpQuFXulj_A-HOt5TPIEaAlGyEALw_wcB?utm_source=google&utm_campaign=search_gs_pl_evergreen_atlas_core_retarget-brand_gic-null_amers-all_ps-all_desktop_eng_lead&utm_term=mongodb&utm_medium=cpc_paid_search&utm_ad=e&utm_ad_campaign_id=14412646314&adgroup=131761122172&cq_cmp=144

12646314&gad_source=1&gclid=Cj0KCQjw4MSzBhC8ARIsAPFOuyW_-
bzUU__HvCgxT9QI5O4c_qnW89jideEQpQuFXulj_A-
HOt5TPIEaAlGyEALw_wcB

[64] A. Ronacher, “Welcome to Flask — Flask Documentation (3.0.x).” Accessed: Jun. 17, 2024. [Online]. Available: <https://flask.palletsprojects.com/en/3.0.x/>

[65] K. Chaim, “The web framework for perfectionists with deadlines | Django.” Accessed: Jun. 17, 2024. [Online]. Available: <https://www.djangoproject.com/>

[66] S. Ramírez, “FastAPI.” Accessed: Jun. 17, 2024. [Online]. Available: <https://fastapi.tiangolo.com/>

[67] RangeKing, “Brief summary of YOLOv8 model structure · Issue #189 · ultralytics/ultralytics · GitHub.” Accessed: Jun. 12, 2024. [Online]. Available: <https://github.com/ultralytics/ultralytics/issues/189>

[68] I. Ultralytics, “COCO - Ultralytics YOLO Documentos.” Accessed: Jun. 12, 2024. [Online]. Available: <https://docs.ultralytics.com/es/datasets/detect/coco/>

[69] Google Colabs, “Te damos la bienvenida a Colaboratory - Colab.” Accessed: Jun. 23, 2024. [Online]. Available: <https://colab.research.google.com/>

[70] R. P. Abao, M. R. J. E. Estuar, and P. A. R. Abu, “Physical Distancing and Mask Wearing Behavior Dataset Generator from CCTV Footages Using YOLOv8,” in *Social, Cultural, and Behavioral Modeling*, R. Thomson, S. Al-khateeb, A. Burger, P. Park, and A. A. Pyke, Eds., Cham: Springer Nature Switzerland, 2023, pp. 295–304.

[71] F. Xu, B. Li, and S. Xu, “Accurate and Rapid Localization of Tea Bud Leaf Picking Point Based on YOLOv8,” in *Big Data and Social Computing*, X. Meng, Y. Chen, L. Suo, Q. Xuan, and Z.-K. Zhang, Eds., Singapore: Springer Nature Singapore, 2023, pp. 261–274.

[72] NVIDIA Corporation, “Get Started With Jetson Nano Developer Kit | NVIDIA Developer.” Accessed: Jun. 17, 2024. [Online]. Available: <https://developer.nvidia.com/embedded/learn/get-started-jetson-nano-devkit#write>

ANEXOS

Anexo A. Categorización de variables

En la Figura A1 se muestra la categorización de variables para el desarrollo del proyecto.

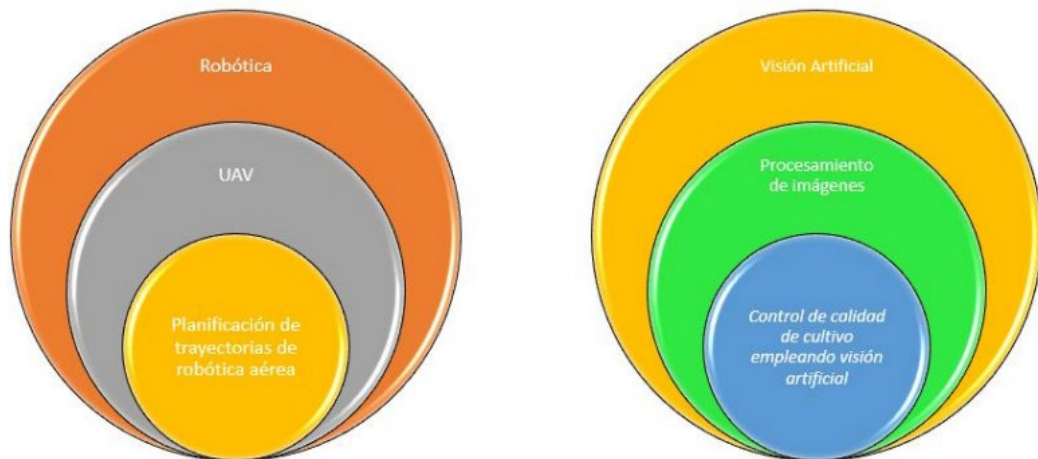


Figura A1. Categorización de variables

En la Figura A2 y Figura A3 se muestra la constelación de ideas para el desarrollo del proyecto.

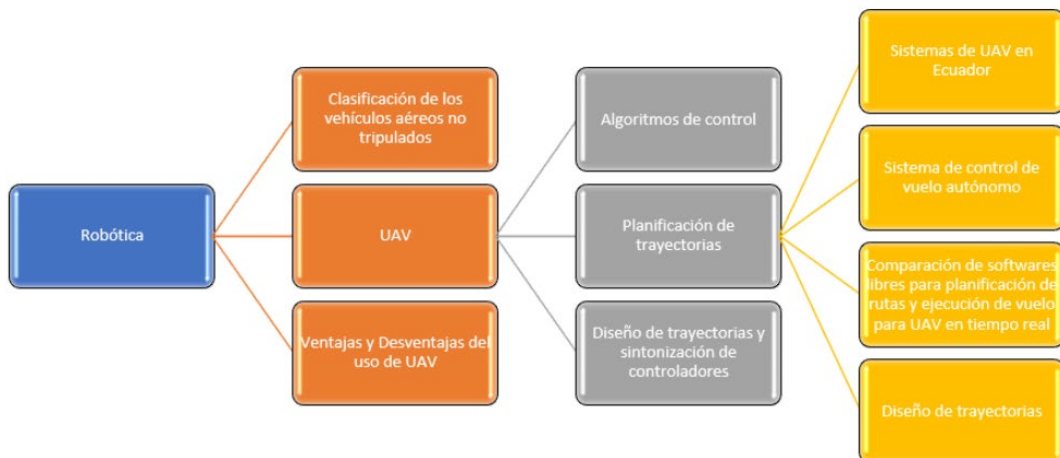


Figura A2. Constelación de ideas, variable 1

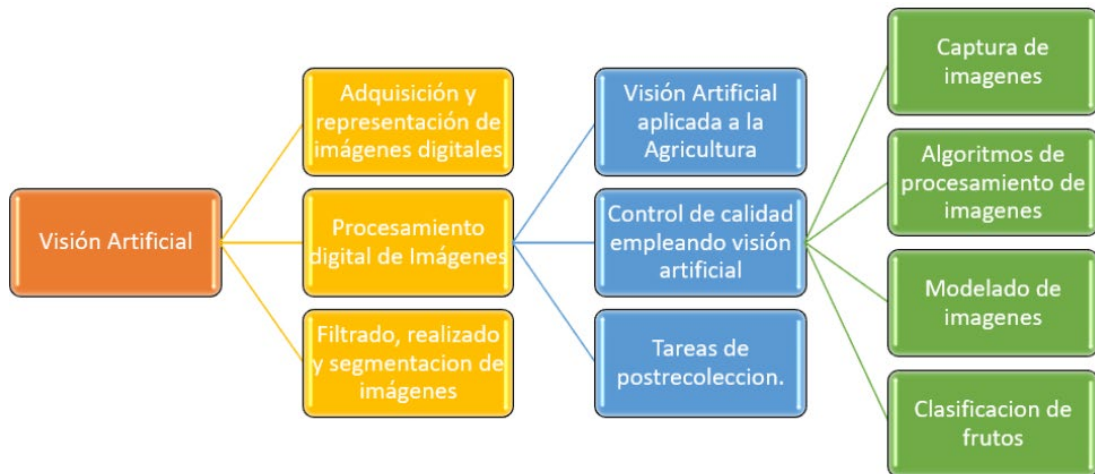


Figura A3. Constelación de ideas, variable 2

Anexo B. DJI Tello EDU

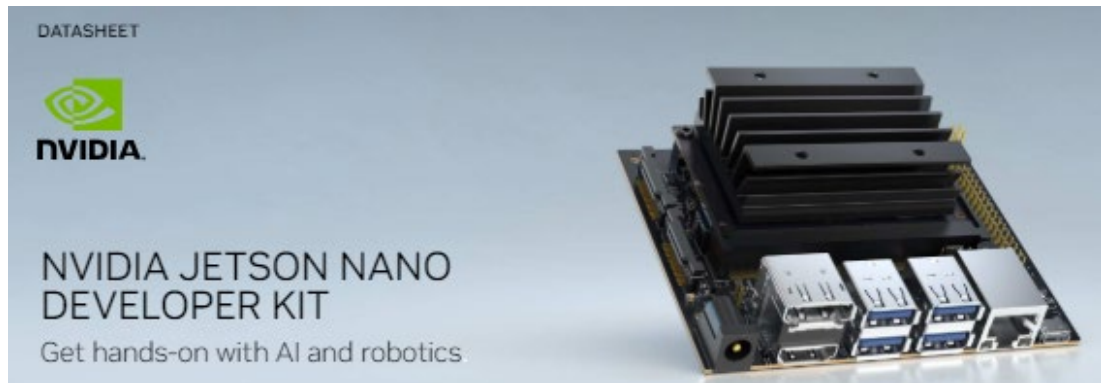
En la Figura B1 se muestra las especificaciones técnicas del UAV usado en el proyecto.

Specifications	
Product Code	DJI-TELLO
Aircraft	
Weight	Approximately 80 g (Propellers and Battery Included)
Dimensions	98mm*92.5mm*41mm
Propeller	3 inch
Built-In Functions	Range Finder, Barometer, LED, Vision System, WIFI 802.11n 2.4G, 720P Live View
Port	Micro USB Charging Port
Flight Performance	
Max Flight Distance	100m
Max Speed	8m/s
Max Flight Time	13min
Battery	
Detachable Battery	1.1Ah/3.8V
Camera	
Photo	5MP (2592x1936)
FOV	82.6°
Video	HD720P30
Format	JPG(Photo); MP4(Video)
EIS	Yes

Figura B1. Especificaciones técnicas DJI Tello EDU

Anexo C. NVIDIA JETSON NANO

En la Figura C1 se muestra las especificaciones técnicas del microordenador usado en el desarrollo del proyecto.



TECHNICAL SPECIFICATIONS

	Developer Kit
GPU	128-core NVIDIA Maxwell architecture GPU
CPU	Quad-core ARM® Cortex®-A57 MPCore processor
Memory	4GB 64-bit LPDDR4
Storage	microSD (Card not included)
Video Encoder	1x 4K30 2x 1080p60 4x 1080p30 9x 720p30 (H.264/H.265)
Video Decoder	1x 4K60 2x 4K30 4x 1080p60 8x 1080p30 18x 720p30 (H.264/H.265)
Camera	2x 15-pin 2-lane MIPI CSI-2 camera connectors
Connectivity	Gigabit Ethernet, M.2 Key E
Display	1x HDMI 2.0, 1x DP 1.2
USB	4x USB 3.0 Type-A connectors 1x USB 2.0 Micro-B connector
Others	40-pin header (UART, SPI, I2S, I2C, PWM, GPIO) 12-pin automation header 4-pin fan header 4-pin POE header DC power jack Power, Force Recovery, and Reset buttons
Mechanical	100mm x 79mm x 30.21mm (height includes carrier board, module, and thermal solution.)

Figura C1. Especificaciones técnicas NVIDIA JETSON NANO

Anexo D. Manual de usuario DJI Tello EDU

En la Figura D1 se muestra la portada del manual de usuario del dron utilizado que se lo puede visualizar en el siguiente link: https://dl-cdn.ryzerobotics.com/downloads/Tello/201806mul/Tello%20User%20Manual%20V1.0_ES.pdf

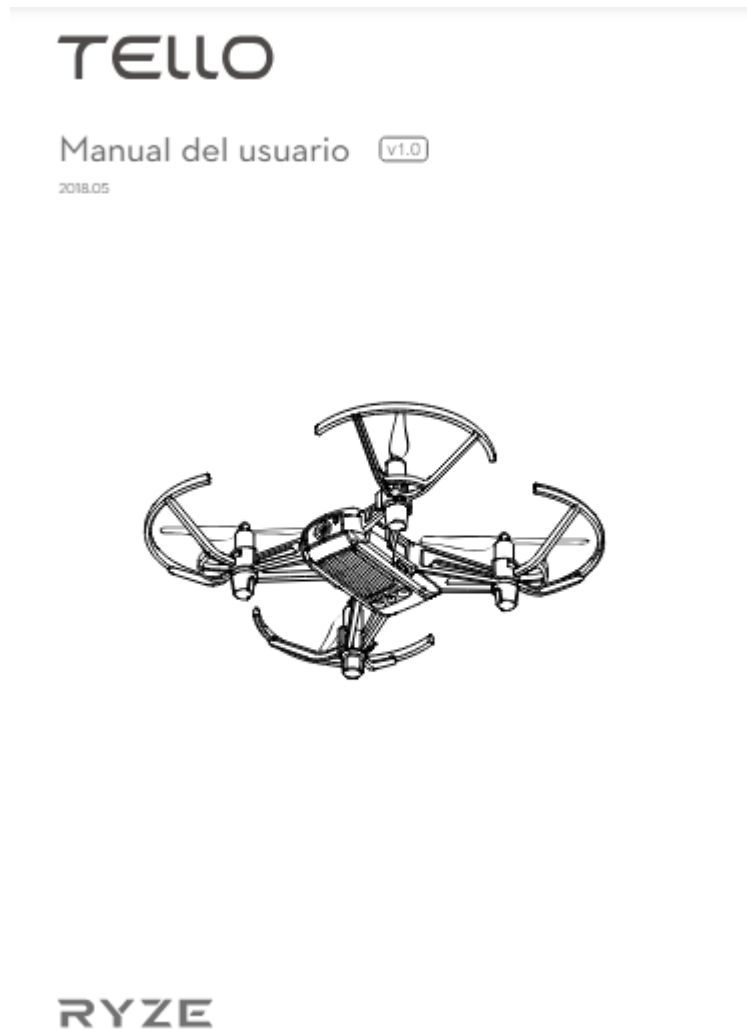


Figura D1. Portada Manual Usuario TELLO

Anexo E. Hardware Adquirido

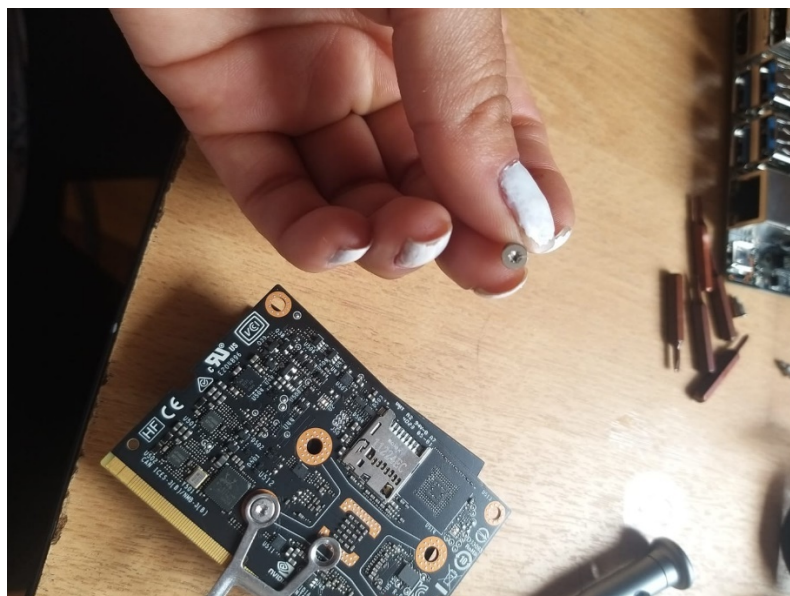


Figura E1. Ensamblaje Placa de desarrollo NVIDIA Jetson Nano



Figura E2. DJI Tello Edu

Anexo F. Visitas al cultivo de manzana



Figura F1. Investigadores en el cultivo de manzana



Figura F2. Pruebas de vuelo en el cultivo de manzana

Anexo G. Pruebas del sistema

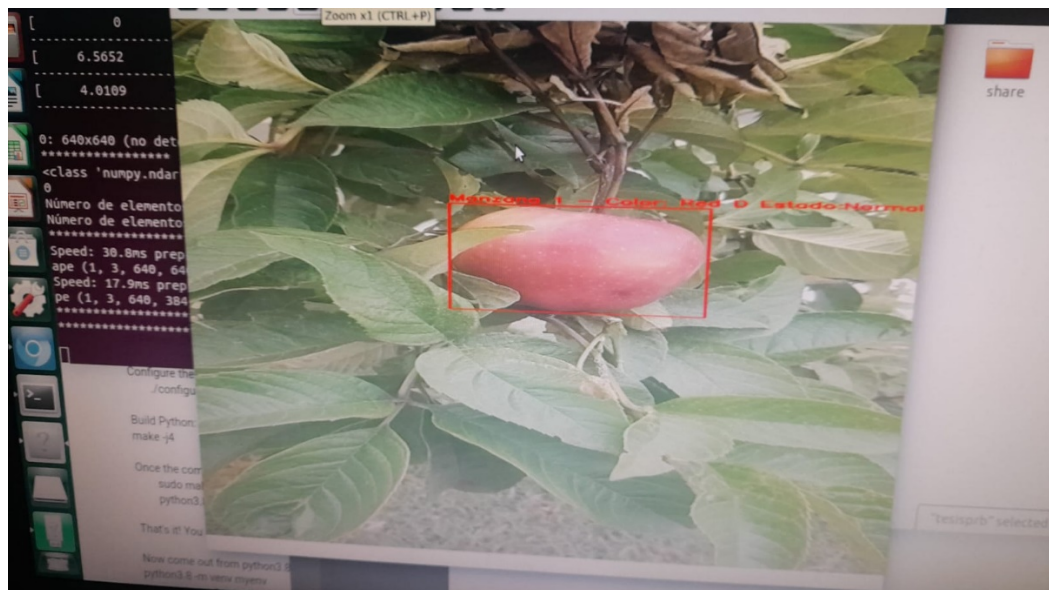


Figura G1. Primera prueba en el microcomputador de los modelos entrenados



Figura G2. Colección de imágenes de pruebas realizadas

Anexo H. Código del sistema

```
import threading
import time
from djitellopy import Tello
import cv2
from ultralytics import YOLO
import numpy as np
from sort import Sort
import os
import sqlite3
from flask import Flask, render_template, Response

# Inicializar variables
xmin2, ymin2, xmax2, ymax2 = 0, 0, 0, 0
label = "Normal"
color = (255, 255, 255)

# Diccionarios para llevar el registro de las manzanas malas, medias y
# colores por ID
estado_manzanas = {}
color_manzanas = {}

# Variables de conteo globales
contmalas = 0
contmedias = 0
cont_red_d = 0
cont_fuji = 0
cont_golden = 0
cont_granny = 0

# Flask app setup
app = Flask(__name__)

def contar_elementos(array):
    num_unos = 0
    num_ceros = 0
    for elem in array:
        if elem:
            if elem == 1:
                num_unos += 1
            elif elem == 0:
                num_ceros += 1
    return num_unos, num_ceros

def determinar_color(region_manzana, track_id):
    if region_manzana is None or region_manzana.size == 0:
```

```

        return 'otro'

    hsv_region = cv2.cvtColor(region_manzana, cv2.COLOR_BGR2HSV)
    lower_red = np.array([0, 100, 200])
    upper_red = np.array([1, 255, 255])
    lower_green = np.array([30, 100, 100])
    upper_green = np.array([40, 255, 255])
    lower_yellow = np.array([20, 100, 100])
    upper_yellow = np.array([30, 255, 255])
    lower_fuji = np.array([3, 100, 100])
    upper_fuji = np.array([15, 255, 255])

    mask_red = cv2.inRange(hsv_region, lower_red, upper_red)
    mask_green = cv2.inRange(hsv_region, lower_green, upper_green)
    mask_yellow = cv2.inRange(hsv_region, lower_yellow, upper_yellow)
    mask_fuji = cv2.inRange(hsv_region, lower_fuji, upper_fuji)

    avg_red = np.mean(cv2.bitwise_and(region_manzana, region_manzana,
mask=mask_red), axis=(0, 1))
    avg_green = np.mean(cv2.bitwise_and(region_manzana, region_manzana,
mask=mask_green), axis=(0, 1))
    avg_yellow = np.mean(cv2.bitwise_and(region_manzana,
region_manzana, mask=mask_yellow), axis=(0, 1))
    avg_fuji = np.mean(cv2.bitwise_and(region_manzana, region_manzana,
mask=mask_fuji), axis=(0, 1))

    if avg_red[2] > avg_red[0] and avg_red[2] > avg_red[1]:
        return 'Red D'
    elif avg_fuji[2] > avg_fuji[0] and avg_fuji[2] > avg_fuji[1]:
        return 'Fuji'
    elif avg_yellow[2] > avg_yellow[0] and avg_yellow[1] >
avg_yellow[0] and avg_yellow[2] > avg_green[2]:
        return 'Golden'
    elif avg_green[1] > avg_green[0] and avg_green[1] > avg_green[2]:
        return 'Granny'
    else:
        return 'fuji'

model = YOLO("yolov8x.pt")
model2 = YOLO("bestpodridoamarillodeteccion.pt")
tracker = Sort()

# Función para controlar el Tello
def controlar_tello():

    try:
        print("Conexión exitosa con el dron Tello.")

```

```

        # Esperar unos segundos después de la conexión antes de enviar
comandos
        time.sleep(5)

        # Realizar el despegue
        tello.takeoff()
        print("Despegue exitoso.")

        # Abre el archivo en modo lectura
        with open('archivo.txt', 'r') as archivo:
            # Lee las líneas del archivo
            lineas = archivo.readlines()

            # Itera sobre las líneas y determina la acción para cada
palabra clave
            for linea in lineas:
                palabras = linea.strip().split() # Elimina espacios y
divide las palabras
                for palabra in palabras:
                    if palabra == 'ADELANTE':
                        tello.send_rc_control(0, 50, 0, 0)
                        print('ADELANTE')
                    elif palabra == 'ATRAS':
                        tello.send_rc_control(0, -50, 0, 0)
                        print("ATRAS")
                    elif palabra == 'DERECHA':
                        tello.send_rc_control(50, 0, 0, 0)
                        print("DERECHA")
                    elif palabra == 'IZQUIERDA':
                        tello.send_rc_control(-50, 0, 0, 0)
                        print("IZQUIERDA")
                    elif palabra == 'ARRIBA':
                        tello.send_rc_control(0, 0, 50, 0)
                        print("ARRIBA")
                    elif palabra == 'ABAJO':
                        tello.send_rc_control(0, 0, -50, 0)
                        print("ABAJO")
                        elif palabra == 'ROTIZ':
                            tello.send_rc_control(0, 0, 0, -45)
                            print("ROTIZ")
                    elif palabra == 'ROTDER':
                        tello.send_rc_control(0, 0, 0, 45)
                        print("ROTDER")
                    elif palabra == 'ATERRIZAR':
                        tello.land()
                        print("ATERRIZAR")
                    else:
                        print("Palabra clave no válida:", palabra)

```

```

        time.sleep(2) # Espera después de cada comando
        tello.send_rc_control(0, 0, 0, 0)
        time.sleep(3)

    except Exception as e:
        print("Error:", e)

# Función para recibir y mostrar video del Tello
def recibir_video_tello():
    global contmalas, contmedias, cont_red_d, cont_fuji, cont_golden,
    cont_granny
    tello.streamon()

    while True:
        frame = tello.get_frame_read().frame
        if frame is not None:
            frame = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)

            results = model(frame, conf=0.3, show=False, show_boxes=False,
show_labels=False, stream=True, classes=[47])
            for res in results:
                boxes = res.bboxes.xyxy.cpu().numpy().astype(int)
                tracks = tracker.update(boxes)
                tracks = tracks.astype(int)

                carpeta_regiones = "regiones_manzanas"
                if not os.path.exists(carpeta_regiones):
                    os.makedirs(carpeta_regiones)

                for xmin, ymin, xmax, ymax, track_id in tracks:
                    region_manzana = frame[ymin:ymax, xmin:xmax]
                    ruta_archivo = os.path.join(carpeta_regiones,
f"region_manzana_{track_id}.jpg")
                    cv2.imwrite(ruta_archivo, region_manzana)

                    color_manzana = determinar_color(region_manzana,
track_id)

                    color_manzanas[track_id] = color_manzana

                if region_manzana is not None and region_manzana.size
!= 0:
                    results2 = model2(region_manzana, show=False,
show_boxes=False, show_labels=False, stream=True, classes=[0, 1],
conf=0.2)
                    for result2 in results2:
                        reg = result2.bboxes.cls.cpu().numpy()
                        num_unos, num_ceros = contar_elementos(reg)

```

```

        if num_unos >= 1:
            estado_manzanas[track_id] = "mala"
        elif num_ceros >= 1:
            estado_manzanas[track_id] = "medio"

    if color_manzana == "Red D":
        color = (0, 0, 255)
    elif color_manzana == "Fuji":
        color = (0, 165, 255)
    elif color_manzana == "Golden":
        color = (0, 255, 255)
    elif color_manzana == "Granny":
        color = (0, 255, 0)

    cv2.rectangle(frame, (xmin, ymin), (xmax, ymax), color,
3)
        cv2.putText(frame, f"Manzana {track_id} - Color:
{color_manzana} Estado:{estado_manzanas.get(track_id, 'Normal')}",
(xmin, ymin - 10), cv2.FONT_HERSHEY_SIMPLEX, 1, color, 4)

    ret, buffer = cv2.imencode('.jpg', frame)
    frame = buffer.tobytes()
    yield (b'--frame\r\n'
          b'Content-Type: image/jpeg\r\n\r\n' + frame + b'\r\n')

@app.route('/')
def index():
    return render_template('index.html')

@app.route('/video_feed')
def video_feed():
    return Response(recibir_video_tello(), mimetype='multipart/x-mixed-
replace; boundary=frame')

@app.route('/start_program', methods=['POST'])
def start_program():
    # Crear y lanzar el hilo para controlar el Tello y recibir video
    hilo_control = threading.Thread(target=controlar_tello)
    hilo_control.start()

    hilo_video = threading.Thread(target=recibir_video_tello)
    hilo_video.start()

    return '', 200

@app.route('/stop_program', methods=['POST'])
def stop_program():
    global tello # Hacer referencia a la variable global del dron

```

```

# Detener la transmisión de video desde el dron
tello.streamoff()
contmalas = list(estado_manzanas.values()).count("mala")
contmedias = list(estado_manzanas.values()).count("medio")
cont_red_d = list(color_manzanas.values()).count("Red D")
cont_fuji = list(color_manzanas.values()).count("Fuji")
cont_golden = list(color_manzanas.values()).count("Golden")
cont_granny = list(color_manzanas.values()).count("Granny")

print(f"Manzanas malas: {contmalas}")
print(f"Manzanas medias: {contmedias}")
print(f"Manzanas Red D: {cont_red_d}")
print(f"Manzanas Fuji: {cont_fuji}")
print(f"Manzanas Golden: {cont_golden}")
print(f"Manzanas Granny: {cont_granny}")
#-----

# Directorio con las imágenes
folder_path = 'regiones_manzanas'

# Conectar a la base de datos SQLite3 (o crearla)
conn = sqlite3.connect('imagenes.db')
cursor = conn.cursor()

# Crear tabla para almacenar imágenes y contadores
cursor.execute('''
CREATE TABLE IF NOT EXISTS imagenes (
    id INTEGER PRIMARY KEY AUTOINCREMENT,
    imagen1 BLOB,
    imagen2 BLOB,
    imagen3 BLOB,
    imagen4 BLOB,
    imagen5 BLOB,
    cont_malas INTEGER,
    cont_medias INTEGER,
    cont_red_d INTEGER,
    cont_fuji INTEGER,
    cont_golden INTEGER,
    cont_granny INTEGER
)
''')

# Función para convertir una imagen en un blob
def convert_to_binary_data(filename):
    with open(filename, 'rb') as file:
        blob_data = file.read()

```

```

        return blob_data

    # Contar y procesar archivos JPG en la carpeta
    count = 0
    imagenes_blob = []
    for filename in os.listdir(folder_path):
        if filename.lower().endswith('.jpg'):
            count += 1
            if count <= 5:
                filepath = os.path.join(folder_path, filename)
                blob_data = convert_to_binary_data(filepath)
                imagenes_blob.append(blob_data)

    # Si se han añadido 5 imágenes, insertarlas en la base de
datos
    if len(imagenes_blob) == 5:
        cursor.execute('''
            INSERT INTO imagenes (imagen1, imagen2, imagen3,
imagen4, imagen5, cont_malas, cont_medias, cont_red_d, cont_fuji,
cont_golden, cont_granny)
            VALUES (?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?)
            ''', (*imagenes_blob, contmalas, contmedias,
cont_red_d, cont_fuji, cont_golden, cont_granny))
        imagenes_blob = []

    # Insertar las imágenes restantes si hay menos de 5
    if len(imagenes_blob) > 0:
        # Rellenar con None para que siempre haya 5 columnas
        while len(imagenes_blob) < 5:
            imagenes_blob.append(None)
        cursor.execute('''
            INSERT INTO imagenes (imagen1, imagen2, imagen3, imagen4,
imagen5, cont_malas, cont_medias, cont_red_d, cont_fuji, cont_golden,
cont_granny)
            VALUES (?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?)
            ''', (*imagenes_blob, contmalas, contmedias, cont_red_d,
cont_fuji, cont_golden, cont_granny))

    # Confirmar cambios en la base de datos
    conn.commit()

    # Eliminar archivos JPG de la carpeta
    for filename in os.listdir(folder_path):
        if filename.lower().endswith('.jpg'):
            os.remove(os.path.join(folder_path, filename))

    # Cerrar la conexión a la base de datos
    conn.close()

```

```

    print(f'Se han procesado y subido a la base de datos las primeras 5
imágenes JPG junto con los contadores. Todas las imágenes han sido
eliminadas de la carpeta.')

    #-----
    # Detener cualquier hilo en ejecución
    for thread in threading.enumerate():
        if thread != threading.current_thread(): # Evitar detener el
hilo actual de Flask
            thread.join()

    return '', 200

if __name__ == "__main__":
    # Inicializar el Tello pero sin conectar automáticamente
    tello = Tello()
    tello.connect()

    # Iniciar la aplicación Flask sin conectar automáticamente al dron
    app.run(host='0.0.0.0', port=5000)

```