

UNIVERSIDAD TÉCNICA DEL NORTE



FACULTAD DE INGENIERÍA EN CIENCIAS APLICADAS CARRERA DE INGENIERÍA EN SISTEMAS COMPUTACIONALES

TEMA:

**COMPARATIVA DE PLATAFORMAS RASPBERRY PI 4 Y JETSON NANO PARA
APLICACIONES DE AGRICULTURA DE PRECISIÓN.**

TRABAJO DE GRADO PREVIO A LA OBTENCIÓN
DEL TÍTULO DE INGENIERA EN SISTEMAS COMPUTACIONALES

AUTORA:

MÓNICA CRISTINA ESCOBAR MOLINA

DIRECTOR:

MSC. MARCO REMIGIO PUSDÁ CHULDE

IBARRA – ECUADOR

2023



UNIVERSIDAD TÉCNICA DEL NORTE

BIBLIOTECA UNIVERSITARIA

AUTORIZACIÓN DE USO Y PUBLICACIÓN A

FAVOR DE LA UNIVERSIDAD TÉCNICA DEL NORTE

1. IDENTIFICACIÓN DE LA OBRA

En cumplimiento del Art. 144 de la Ley de Educación Superior, hago la entrega del presente trabajo a la Universidad Técnica del Norte para que sea publicado en el Repositorio Digital Institucional, para lo cual pongo a disposición la siguiente información:

DATOS DE CONTACTO			
CÉDULA DE IDENTIDAD:	1004370050		
APELLIDOS Y NOMBRES:	ESCOBAR MOLINA MÓNICA CRISTINA		
DIRECCIÓN:	IBARRA- BELLAVISTA DE SAN ANTONIO DE IBARRA CALLE JOSÉ CEVALLOS Y CUMANDÁ		
EMAIL:	mcescobarm@utn.edu.ec		
TELÉFONO FIJO:	-	TELÉFONO MÓVIL:	0996611203

DATOS DE LA OBRA	
TÍTULO:	“COMPARATIVA DE PLATAFORMAS RASPBERRY PI 4 Y JETSON NANO PARA APLICACIONES DE AGRICULTURA DE PRECISIÓN. “
AUTOR (ES):	ESCOBAR MOLINA MÓNICA CRISTINA
FECHA: DD/MM/AAAA	11/09/2023
PROGRAMA:	<input checked="" type="checkbox"/> PREGRADO <input type="checkbox"/> POSGRADO
TITULO POR EL QUE OPTA:	INGENIERÍA EN SISTEMAS COMPUTACIONALES
ASESOR /DIRECTOR:	MSc. MARCO REMIGIO PUSDÁ CHULDE

2. CONSTANCIAS

La autora manifiesta que la obra objeto de la presente autorización es original y se la desarrolló, sin violar derechos de autor de terceros, por lo tanto, la obra es original y que es el titular de los derechos patrimoniales, por lo que asume la responsabilidad sobre el contenido de esta y saldrá en defensa de la Universidad en caso de reclamación por parte de terceros.

Ibarra, a los 11 días del mes de septiembre de 2023

LA AUTORA:



ESCOBAR MOLINA MÓNICA CRISTINA

1004370050

CERTIFICADO DEL DIRECTOR



UNIVERSIDAD TÉCNICA DEL NORTE FACULTAD DE INGENIERÍA EN CIENCIAS APLICADAS

CERTIFICADO DIRECTOR DE TESIS

Por medio del presente yo MSc. Marco Pusdá, certifico que la Srta. **Mónica Cristina Escobar Molina** portadora de la cedula de ciudadanía numero **100437005-0**, ha trabajado en el desarrollo del proyecto de grado titulado "**COMPARATIVA DE PLATAFORMAS RASPBERRY PI 4 Y JETSON NANO PARA APLICACIONES DE AGRICULTURA DE PRECISIÓN**", previo a la obtención del título de Ingeniería en Sistemas Computacionales, lo cual ha realizado en su totalidad con responsabilidad y esmero.

Es todo cuanto puedo certificar en honor a la verdad.

En la ciudad de Ibarra, a los 7 días del mes de septiembre del 2023

Atentamente

0401200951 MARCO
REMIGIO PUSDA CHULDE

Firmado digitalmente por
0401200951 MARCO
REMIGIO PUSDA CHULDE
Fecha: 2023.09.07
11:35:41 -05'00'

Ing Marco Remigio Pusdá Chulde, MSc.
DIRECTOR DEL TRABAJO DE GRADO

DEDICATORIA

Dedico este trabajo de titulación a Dios quien confió mis esfuerzos y anhelos de superación en cada paso de este viaje académico, que ha sido mi faro en la oscuridad y ha fortalecido mi mente y corazón.

A mi familia, por su apoyo constante a lo largo de este viaje. A mis amigos, por su amistad y ánimo inagotable, que me han dado fuerzas en los momentos más desafiantes.

A mis docentes, cuya sabiduría y orientación me han guiado hacia la excelencia académica. A todos aquellos que, de alguna manera, han contribuido a mi crecimiento intelectual y personal.

AGRADECIMIENTO

Agradezco profundamente a **Dios**, fuente de inspiración y guía constante en mi vida, por brindarme la fortaleza y la determinación para alcanzar esta meta académica. Su amor y misericordia han sido mi refugio en momentos de dificultad y mi motivo de gratitud en momentos de éxito.

A mis padres **Rosita y Marco** que de alguna manera me han apoyado para cumplir mis metas.

A mi tutor **Ing. Marco PUSDÁ, MSc** y docentes de la carrera de Ingeniería en sistemas computacionales que supieron compartir sus conocimientos de una manera profesional y dedicada.

Al departamento de bienestar Universitario en especial a la **Lcda. Lila Cazar y la MSc. Mercedes Navarrete** que han sido un apoyo incondicional en esta trayectoria académica.

A mis amigos **Erika y Jonathan** quienes hicieron que la vida universitaria fuera más llevadera e inolvidable.

Y finalmente a la **MSc Ludmila Starodub** quien con su cariño su amabilidad me ha hecho sentir como en casa.

ÍNDICE DE CONTENIDOS

Contenido	
INTRODUCCIÓN.....	1
Problema	1
Antecedentes	1
Situación Actual.....	1
Prospectiva.....	2
Planteamiento del Problema	3
Objetivos.....	4
Alcance	4
Justificación.....	7
CAPITULO 1	8
1. Marco Teórico.....	8
1.1 Arquitecturas embebidas	8
1.1.1 El Raspberry Pi.....	9
1.1.2 Nvidia Jetson Nano.....	14
1.2 Drones en la agricultura de precisión.....	16
1.2.1 Funcionamiento de un Dron	17
1.2.2 Componentes de un Dron.....	19
1.2.3 Principales movimientos que realiza un dron	19
1.2.4 Clasificación de drones	20
1.2.5 Uso de drones en agricultura de precisión.....	21
1.2.6 Tipos de imágenes adquiridas con drones.....	23
1.3 Aplicaciones en agricultura de precisión	26
1.3.1 Monitoreo cultivos	27
1.3.2 Detección de malezas	28
1.3.3 Fertilización de suelos	29
1.3.4 Detección de enfermedades.....	30
1.4 Lenguajes de programación	31
1.4.1 Python	32
1.4.2 Matlab.....	34
1.4.3 C++	35
CAPÍTULO 2	38
2 Desarrollo.....	38
2.1 Análisis de las arquitecturas embebidas	38

2.2	Análisis del lenguaje de programación	39
2.3	Características del kit de desarrollo del algoritmo	40
2.3.1	SDK de Jetson nano	40
2.3.2	Sistema operativo versión.....	40
2.3.3	Kit de desarrollo de NVIDIA Jetson Nano	40
2.4	Desarrollo del algoritmo	44
2.4.1	Importación de Librerías.....	44
2.4.2	Leer Imagen y visualizar	45
2.4.3	Extracción de valores RGB y Cálculo E*G.....	45
2.4.4	Primer Filtro.....	46
2.4.5	Contorno de líneas.....	47
2.4.6	Total, de líneas detectadas.....	49
CAPÍTULO 3	51
3	RESULTADOS	51
3.1	Pruebas con imágenes offline.....	51
3.2	Definición de métricas a analizar	52
3.2.1	Tiempo de ejecución.....	52
3.2.2	Consumo de memoria	55
3.2.3	Consumo CPU.....	57
3.3	Obtener resultado del prototipo.....	61
3.3.1	Conteo de líneas reales y conteo del algoritmo	61
3.4	Análisis de resultados.....	64
3.4.1	Porcentaje de reconocimiento.....	64
3.4.2	Definir mejor reconocimiento de líneas según la semana y altura.....	68
3.5	Análisis de impactos	72
Conclusiones	73
Recomendaciones	74
Bibliografía	75

INDICE DE FIGURAS

Fig. 1 Arquitectura Raspberry Pi 4	13
Fig. 2 Arquitectura Nvidia JetsonNano	14
Fig. 3 Terminología "DRONE"	16
Fig. 4 Sistema multi hélice de un dron.....	18
Fig. 5 Drones de ala móvil	20
Fig. 6 Drones de ala Fija.....	20
Fig. 7 Imagen RGB tomada con dron	24
Fig. 8 Imagen 3D tomada con dron.....	25
Fig. 9 Imagen espectral tomada con dron.....	25
Fig. 10 Imagen infrarroja-térmica tomada con dron.....	26
Fig. 11 Código Importación de librerías.....	45
Fig. 12 Código leer imagen y visualizar.....	45
Fig. 13 Código extracción de valores RGB y cálculo E*G	46
Fig. 14 Código primer filtro	47
Fig. 15 Código contorno de líneas.....	48
Fig. 16 Resultado de la aplicación de contorno de líneas	49
Fig. 17 Código Total de líneas detectadas.....	50
Fig. 18 Resultado Cantidad de líneas detectadas en el cultivo	50
Fig. 19 Tiempo de Ejecución (T E) Semana 1.....	52
Fig. 20 Tiempo de Ejecución (T E) Semana 2.....	53
Fig. 21 Tiempo de Ejecución (T E) Semana 3.....	54
Fig. 22 Tiempo de Ejecución (T E) Semana 4.....	55
Fig. 23 gráfico de Tiempo promedio de ejecución semana 1-4.....	55
Fig. 24 Consumo Memoria RAM (C.M.) semana 1-4.....	56
Fig. 25 gráfico consumo promedio de memoria RAM (C.M.) semana 1-4.....	56
Fig. 26 Consumo de CPU promedio semana 1.....	57
Fig. 27 gráfico de consumo de CPU promedio semana 1	57
Fig. 28 Consumo de CPU promedio semana 2.....	58
Fig. 29 gráfico de consumo de CPU promedio semana 2	58
Fig. 30 Consumo de CPU promedio semana 3.....	59
Fig. 31 gráfico de consumo de CPU promedio semana 3	59
Fig. 32 Consumo de CPU promedio semana 4.....	60
Fig. 33 gráfico de consumo de CPU promedio semana 4	60
Fig. 34 gráfico de consumo de CPU promedio semana 1-4	61
Fig. 35 conteo de número de las líneas identificadas reales y por el algoritmo semana 1 ...	62
Fig. 36 conteo de número de las líneas identificadas reales y por el algoritmo semana 2 ...	62
Fig. 37 conteo de número de las líneas identificadas reales y por el algoritmo semana 3 ...	63
Fig. 38 conteo de número de las líneas identificadas reales y por el algoritmo semana 4 ...	64
Fig. 39 Porcentaje % de reconocimiento de líneas Semana 1	65
Fig. 40 Porcentaje % de reconocimiento de líneas Semana 2	66
Fig. 41 Porcentaje % de reconocimiento de líneas Semana 3	67
Fig. 42 Porcentaje % de reconocimiento de líneas Semana 4	67

Fig. 43 Mejor reconocimiento de líneas según altura semana 1	68
Fig. 44 Mejor reconocimiento de líneas según altura semana 2	69
Fig. 45 Mejor reconocimiento de líneas según altura semana 3	70
Fig. 46 Mejor reconocimiento de líneas según altura semana 4	70
Fig. 47 Promedio del mejor reconocimiento de líneas según la semana 1-4 y altura.....	71

INDICE DE TABLAS

Tabla 1 Modelos de Raspberry-Pi.....	11
Tabla 2 Características Raspberry Pi 4.....	13
Tabla 3 Características Kit desarrollador Nvidia Jetson Nano.....	15
Tabla 4 Comparativa de lenguajes de programación.....	39
Tabla 5 Especificaciones técnicas de Sistema Operativo Ubuntu Linux y procesador	40
Tabla 6 Pruebas de rendimiento del algoritmo.....	51
Tabla 7 Tiempo de Ejecución (T E) Semana 1.....	52
Tabla 8 Tiempo de Ejecución (T E) Semana 2.....	53
Tabla 9 Tiempo de Ejecución (T E) Semana 3.....	53
Tabla 10 Tiempo de Ejecución (T E) Semana 4	54
Tabla 11 Consumo Memoria RAM (C.M.) semana 1-4.....	56
Tabla 12 Consumo de CPU promedio semana 1	57
Tabla 13 Consumo de CPU promedio semana 2	58
Tabla 14 Consumo de CPU promedio semana3	59
Tabla 15 Consumo de CPU promedio semana 4	60
Tabla 16 Conteo de número de las líneas identificadas reales y por el algoritmo semana 1	61
Tabla 17 Conteo de número de las líneas identificadas reales y por el algoritmo semana 2	62
Tabla 18 Conteo de número de las líneas identificadas reales y por el algoritmo semana 3	63
Tabla 19 Conteo de número de las líneas identificadas reales y por el algoritmo semana 4	64
Tabla 20 Porcentaje % de reconocimiento de líneas Semana 1	64
Tabla 21 Porcentaje % de reconocimiento de líneas Semana 2.....	65
Tabla 22 Porcentaje % de reconocimiento de líneas Semana 3.....	66
Tabla 23 Porcentaje % de reconocimiento de líneas Semana 4.....	67
Tabla 24 Mejor reconocimiento de líneas según altura semana 1	68
Tabla 25 Mejor reconocimiento de líneas según altura semana 2.....	69
Tabla 26 Mejor reconocimiento de líneas según altura semana 3.....	69
Tabla 27 Mejor reconocimiento de líneas según altura semana 4.....	70
Tabla 28 Promedio del mejor reconocimiento de líneas según la semana 1-4 y altura	71

Resumen

El objetivo de esta comparativa es investigar las capacidades de las plataformas Raspberry Pi 4 y Jetson Nano para su uso en aplicaciones de agricultura de precisión. El propósito es evaluar si estas arquitecturas de hardware son adecuadas para el procesamiento de imágenes agrícolas en la automatización de tareas relacionadas con la agricultura.

Después de realizar una revisión exhaustiva de la literatura y llevar a cabo un análisis, se determinó que la computadora Jetson Nano era la opción más idónea para desarrollar e implementar un algoritmo destinado a la detección de líneas en cultivos de maíz. Esto se basó en su ventaja en términos de detección de objetos y procesamiento de imágenes, lo que permitiría lograr una mayor eficiencia, reducir el tiempo de ejecución y el consumo de recursos como memoria y CPU. Además, esta elección proporcionaría un mejor reconocimiento y clasificación de las líneas y malezas en los cultivos, con mayor precisión y velocidad, anticipando la toma de medidas para controlar su propagación.

Una vez que se implementó la aplicación, se llevaron a cabo evaluaciones de rendimiento del algoritmo utilizando un conjunto de imágenes adquiridas previamente con un dron. Los resultados mostraron una detección precisa de las líneas en los cultivos a alturas de 10 y 15 metros durante las semanas 3 y 4, superando el 90%. Además, se logró una reducción significativa en el tiempo de ejecución para la localización y un aumento en el porcentaje de reconocimiento de líneas en los cultivos de maíz, alcanzando una precisión superior al 85%.

Palabras clave: Jetson nano, Raspberry pi, agricultura de precisión

Abstract

The objective of this comparison is to investigate the capabilities of the Raspberry Pi 4 and Jetson Nano platforms for use in precision agriculture applications. The purpose is to evaluate whether these hardware architectures are suitable for agricultural image processing in the automation of agriculture-related tasks.

After conducting an exhaustive literature review and analysis, it was determined that the Jetson Nano computer was the most suitable option to develop and implement an algorithm for line detection in corn crops. This was based on its advantage in terms of object detection and image processing, which would achieve greater efficiency, reduce execution time and consumption of resources such as memory and CPU. Furthermore, this choice would provide better recognition and classification of lines and weeds in crops, with greater precision and speed, anticipating the taking of measures to control their spread.

Once the application was implemented, performance evaluations of the algorithm were carried out using a set of images previously acquired with a drone. The results showed an accurate detection of the lines in the crops at heights of 10 and 15 meters during weeks 3 and 4, exceeding 90%. In addition, a significant reduction in the execution time for localization and an increase in the percentage of line recognition in corn crops was achieved, reaching an accuracy greater than 85%.

Keywords: Jetson nano, Raspberry pi, precision agriculture

INTRODUCCIÓN

Problema

Antecedentes

La agricultura mundial viene enfrentando, a lo largo del tiempo, el desafío constante de aumentar la producción agrícola en respuesta a la creciente demanda de la población (Gutiérrez, 2022). Este aumento productivo se ha realizado con la expansión de nuevas áreas agrícolas, y aumento de nuevas tecnologías que permiten la sistematización de diversas actividades agrícolas para mejorar la producción de alimentos (Bongiovanni et al., 2016).

Existen diferentes aspectos inadecuados como baja productividad debido a la limitada utilización de tecnología para combatir plagas, malas hierbas, enfermedades entre otros, que son una problemática para los agricultores ocasionando sobrecostos en todas las actividades agrícolas (Meneses et al., 2015). Adicionalmente la no utilización de nuevas tecnologías ocasiona contaminación ambiental, agua, suelos y riesgos de intoxicación para los agricultores.

Todo lo expuesto anteriormente demanda un gran consumo de recursos, tanto económico, recurso humano y tiempo, para ello la agricultura de precisión es una buena alternativa la cual brinda muchos beneficios sobre datos reales en distintas áreas del campo agrícola, teniendo el objetivo en aumento de producción y manejo del cultivo, siendo indispensable trabajar con herramientas de percepción remota y sistemas de información geográfica con los cuales se realiza el análisis de áreas con diversos factores agronómicos (Solórzano & Jiménez, 2018).

La agricultura tradicional tiene desventajas con la agricultura de precisión debido a que el tratamiento de actividades agrícolas requiere del uso de productos fitosanitarios y químicos que deterioran los suelos con el pasar del tiempo, ocasionando la destrucción progresiva de los elementos químicos existentes en los suelos convirtiendo en terrenos estériles no aptos para la agricultura.

Situación Actual

En la actualidad el avance tecnológico ha alcanzado un nivel que le permite al productor medir, analizar, y manejar la variabilidad dentro de los lotes logrando adecuar el manejo de suelos y cultivo (Chora et al., 2018). Sin embargo, estas tecnologías pueden representar altos costos para los agricultores de nuestra zona debido a que los ingresos provenientes de la agricultura son mínimos y en ocasiones a pérdidas económicas (Bwambale

et al., 2022). Los altos costos tecnológicos en nuestro país se hacen inalcanzables para la mayoría de los agricultores que quieran sistematizar actividades agrícolas para mejorar la productividad de los cultivos, adicionalmente las instituciones gubernamentales no fomentan la innovación tecnológica por falta de recursos y políticas que beneficien a las personas dedicadas a la agricultura (Chora et al., 2018).

Los sistemas embebidos tales como Raspberry Pi y Jetson Nano / TX2 se presentan como semilleros para innovar en la agricultura ofreciendo soluciones económicas, sustentables, robustas y de código abierto para contribuir a la construcción colectiva de una seguridad alimentaria global (Manlove et al., 2021).

La agricultura de precisión en nuestro país tiene poca acogida en los agricultores, debido al desconocimiento de la tecnología en el campo agrícola, aun así, se ha venido dando énfasis en obtener por medio de procesamiento de imágenes, información sobre el estado del cultivo o número de plantas por hectárea y otras necesidades agrícolas (Cisternas et al., 2020). Un caso de estudio en Ecuador se ha implementado aplicaciones relacionadas con la agricultura de precisión en la Universidad Técnica de Manabí (UTM utilizando vehículos aéreos no tripulados (drones) para el tratamiento de fertilización y producción en 64 parcelas de algodón (FAO, 2020).

Prospectiva

La presente comparativa tiene como propósito la investigación de las Plataformas Raspberry Pi 4 y Jetson Nano para aplicaciones de agricultura de precisión con la finalidad de aprovechar los recursos de Hardware y comprobar si estas arquitecturas se adaptan al procesamiento de imágenes de cultivos en la sistematización de tareas agrícolas. Para el presente proyecto se utilizará imágenes previamente obtenidas desde un dron (offline) de cultivos agrícolas.

Después de haber realizado la correspondiente revisión literaria y llevar acabo un análisis comparativo que nos ayudará a definir la arquitectura más apropiada para la implementación de un algoritmo para detección de malezas o líneas de cultivo de maíz. Esto permitirá gestionar los recursos de manera eficiente dentro de la agricultura, para obtener una mayor productividad con bajo costo de la infraestructura tecnológica.

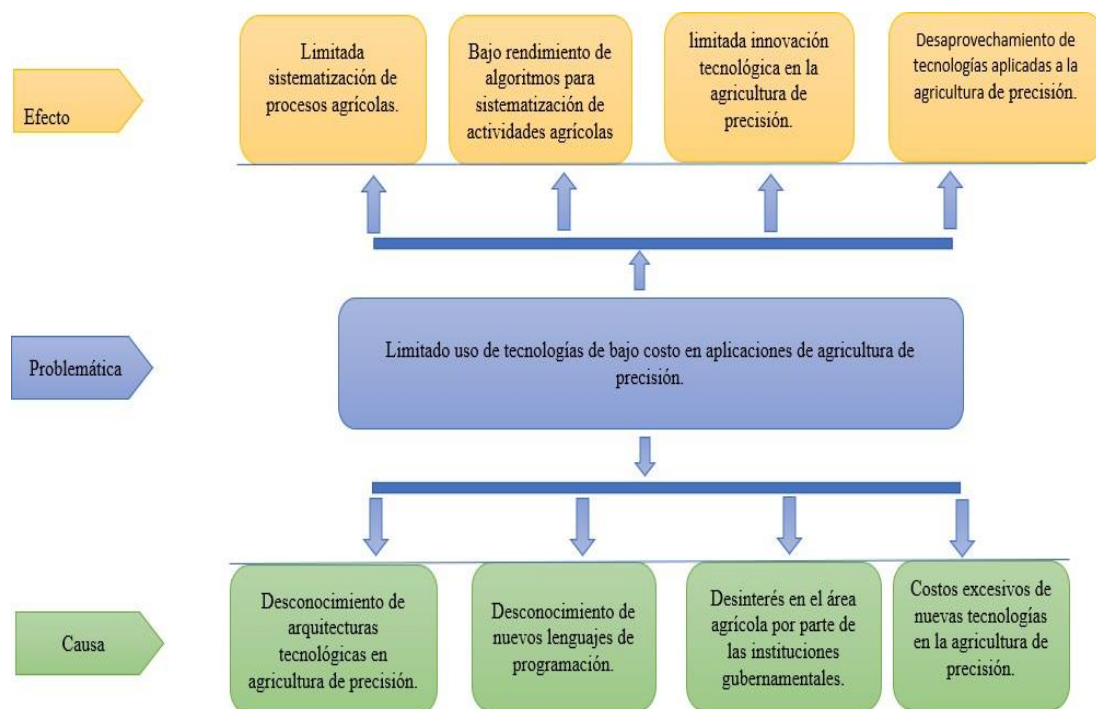
La utilización de arquitecturas heterogéneas como Jetson Nano / TX2, Raspberry Pi que son de bajo costo, permitirán optimizar procesos agrícolas por medio del control de datos ambientales y la gestión de actividades que se involucren en las labores diarias del campo, registro de enfermedades, plagas y malezas; aprovechando los beneficios del hardware y

software libre (Torky & Hassanein, 2020).

La tecnología UAV (Vehículos Aéreos no Tripulados), es un complemento para la captura de imágenes de alta calidad a un bajo costo, actualmente, este tipo de vehículos se están usando en varios campos de la ciencia, en la AP se utiliza de manera eficiente en diferentes actividades agrícolas como monitores de cultivos, detección de malezas, enfermedades entre otros.(Rabia et al. 2021) Las imágenes obtenidas con el dron permitirán procesar de manera eficiente algoritmos utilizando visión por computador (Medici et al., 2021).

Planteamiento del Problema

En nuestra zona1 netamente agrícola en cultivos de maíz no han sido sistematizado tareas para este tipo de cultivos debido al limitado uso de tecnologías de bajo costo en aplicaciones para agricultura de precisión, ya sea por desconocimiento de arquitecturas, nuevos lenguajes de programación, costos excesivos de nuevas tecnologías referente a la misma, así como también desinterés en el área agrícola por parte de las instituciones gubernamentales lo que trae consigo la limitada sistematización de procesos y bajos rendimientos de algoritmos en las actividades agrícolas como también el desaprovechamiento y escasa innovación tecnológica aplicadas en la AP.



Objetivos

Objetivo General

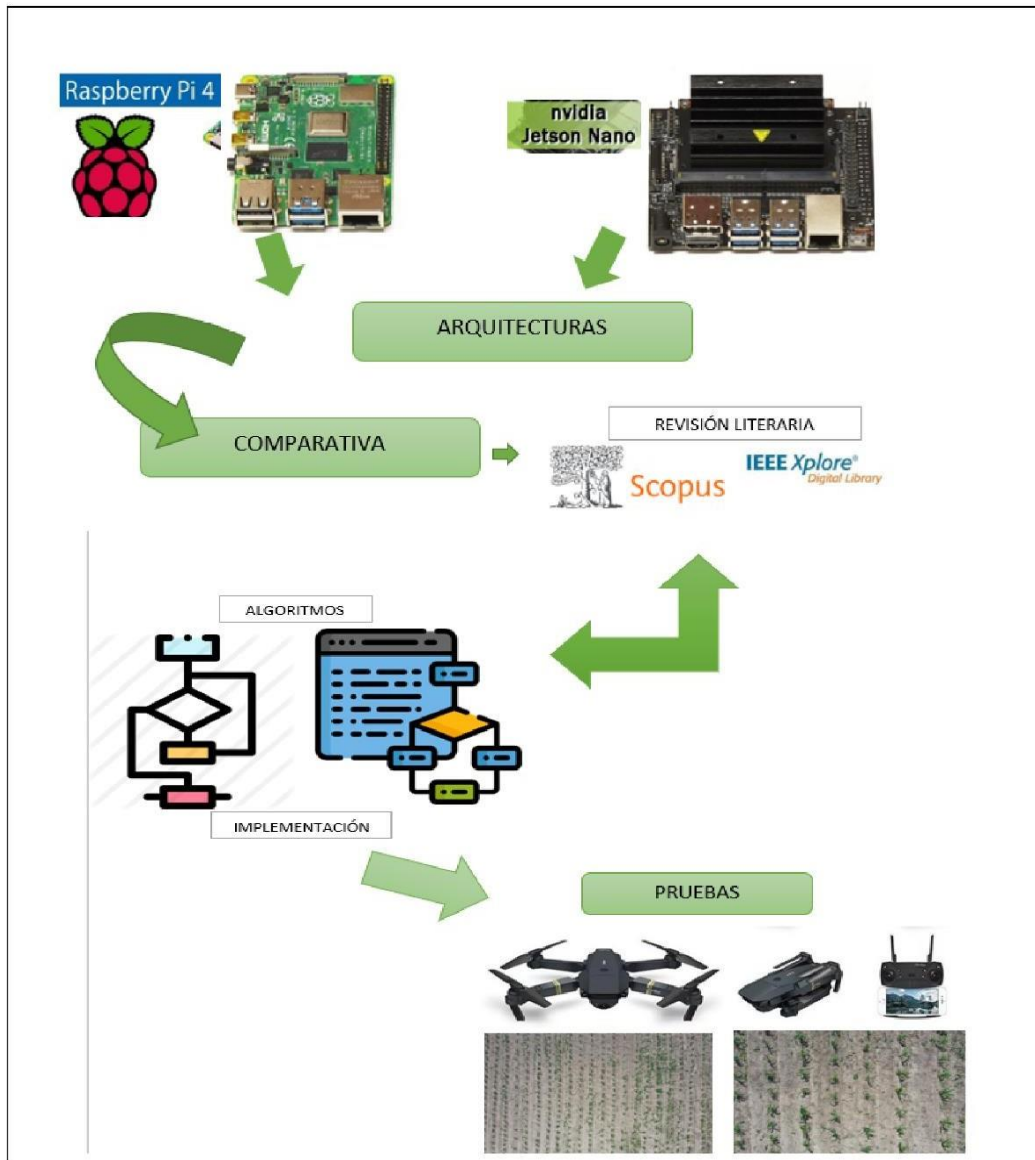
Comparar las Plataformas Raspberry Pi 4 y Jetson Nano en aplicaciones de agricultura de precisión.

Objetivos Específicos

- Elaborar un marco teórico sobre arquitecturas de hardware y herramientas de programación aplicables en agricultura de precisión.
- Desarrollar un algoritmo para actividades de agricultura de precisión en una Plataforma de bajo costo.
- Evaluar los resultados de la investigación propuesta.

Alcance

El alcance del presente proyecto consiste en la búsqueda y análisis de artículos científicos en las 2 revistas científicas IEEEExplore y Scopus que tiene acceso al UTN, relacionados con aplicaciones que se han desarrollado sobre Arquitecturas Raspberry Pi 4 y Jetson Nano en agricultura de precisión para definir cuál de ellas presenta mayores ventajas, para posteriormente realizar un algoritmo para detección de malezas o líneas de cultivo de maíz en la Arquitectura que tenga mejores características comparativas proporcionando nuevas opciones para la agricultura de precisión.



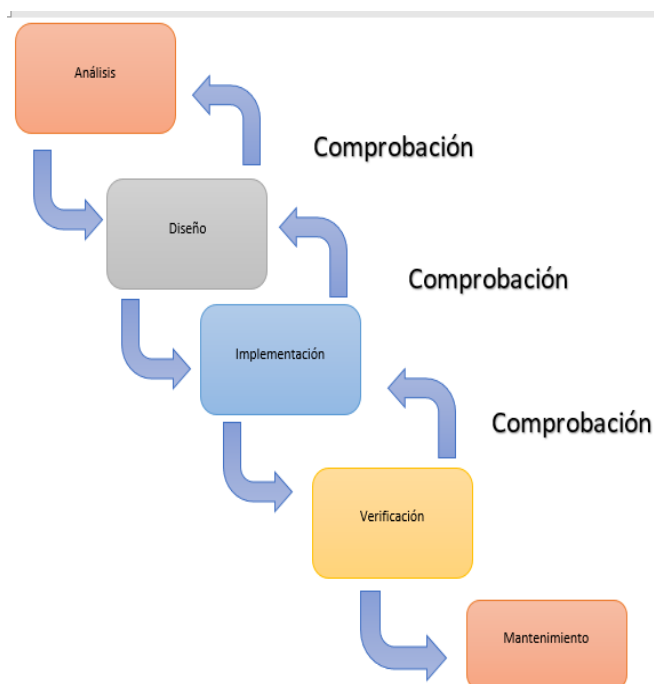
Metodología

Para el primer objetivo se realizó una revisión sistemática de literatura de investigaciones realizadas con Arquitecturas Raspberry Pi 4 y Jetson Nano en aplicaciones relacionadas con agricultura de precisión. La comparativa consiste en revisar sistematizaciones de actividades agrícolas utilizando imágenes agrícolas para identificar cuál de las arquitecturas embebidas de bajo costo es la más utilizada en este tipo de tareas que benefician a los agricultores.

Mientras que para el segundo objetivo que es el desarrollo de algoritmo, se consideró buenas prácticas de programación (R. C. Martin, 2012), y la metodología en cascada según los autores (Kong et al., 2021) y (Ma et al., 2022); que consiste en:

- **Análisis:** Donde se consideró los requisitos necesarios para sistematizar la detección de líneas de cultivo mediante imágenes offline adquiridas con dron.
- **Diseño:** Se llevó a cabo la adquisición, preprocesamiento, procesamiento y clasificación de las imágenes para el diseño del algoritmo.
- **Implementación:** Se realizó un prototipo utilizando Python con librerías o framework OpenCV para las diferentes fases del procesamiento de imágenes agrícolas en la arquitectura Jetson Nano.
- **Verificación:** Se realizó pruebas de funcionamiento utilizando imágenes RGB en formato jpg, con diferentes métricas como dimensiones de imágenes y características obtenidas en cultivos de maíz.
- **Mantenimiento:** Se corrigió posibles errores del algoritmo para detectar las líneas de cultivo presentes en las imágenes agrícolas.

Una vez que se implementó el aplicativo se realizó valoraciones de rendimiento del algoritmo con un conjunto de Imágenes off-line (no a tiempo real) previamente adquiridas con dron para evaluar los tiempos de ejecución.



Justificación

La presente comparativa tiene como finalidad la búsqueda y recolección de información de arquitecturas embebidas tales como (Raspberry Pi 4 y Jetson Nano) que nos ayudó a definir cuál de ellas presenta mayores ventajas y cuál es la más apropiada para la implementación de un algoritmo para detección de malezas o líneas de cultivo de maíz para realizar aplicaciones de procesamiento de imágenes en agricultura de precisión.

Esta investigación apoya al (Objetivo de Desarrollo Sostenible) ODS 2: Hambre cero, ya que tratará de mejorar la productividad de los cultivos de maíz. A la vez esto permitirá proveer de más alimento a la humanidad dándole la oportunidad de que todas las personas accedan a una buena alimentación tanto para el productor como para el consumidor apoyando a una de las metas de este objetivo (Moreira, 2022).

Justificación Teórica. - La comparativa de las Arquitecturas Raspberry Pi 4 y Jetson Nano en aplicaciones para agricultura de precisión forma parte de las herramientas tecnológicas que hacen tener mayor valor institucional y tiendan a ser pioneras en la región en donde se encuentran desempeñando su función.

Justificación Tecnológica. - Es importante la sistematización de actividades agrícolas utilizando tecnología aplicadas a la agricultura de precisión, ya que aporta diferentes beneficios en el sector agrícola mejorando la rentabilidad de los cultivos, además de reducir el uso de fertilizantes, costos y desperdicios minimizando impactos negativos en el planeta.

CAPITULO 1

1. Marco Teórico

1.1 Arquitecturas embebidas

Un sistema integrado se define como una colección de componentes electrónicos relacionados para realizar funciones específicas. En otras palabras, se diferencian de otro tipo de sistemas principalmente por tener una unidad central de procesamiento que se encarga de recibir, analizar y procesar los datos recibidos por los sensores, para luego enviarles señales a los actuadores para que éstos realicen la función o funciones específicas solicitadas (Barreto et al., 2022). Fue ya en los años 80 cuando surgieron los primeros sistemas embebidos con sistema operativo, sobre todo para cubrir las necesidades de comunicación, independizando las aplicaciones del hardware y permitiendo la reutilización de librerías (Alva & Alcorta, 2020).

El procesamiento central del sistema se realiza gracias a un microcontrolador, es decir, se diferencia de otro tipo de sistemas principalmente por poseer un procesador central, el cual se encarga de recibir, luego analizar y procesar los datos aceptados por los sensores, seguidamente envía señales a los actuadores y hace que realicen funciones específicas o requeridas, también incluye interfaces de entrada/salida, así como memoria de tamaño reducido en el mismo chip (Alarcón et al., 2020).

Los sistemas embebidos son diseñados regularmente para tareas que impliquen una computación en tiempo real, pero también destacan otros casos como son Arduino y Raspberry Pi, cuyo fin está más orientado al diseño y desarrollo de aplicaciones y prototipos con sistemas embebidos desde entornos gráficos (Luchetti, 2021).

Generalmente en un sistema embebido, los componentes están ubicados en la placa base (tarjeta gráfica, sistema de sonido, módem) y en varios casos estos dispositivos no son los mismos que normalmente vienen con un ordenador al que están acostumbrados. Normalmente, estos sistemas tienen una interfaz externa para monitorear y diagnosticar el estado del sistema. Dos de sus principales características de los sistemas integrados son el bajo coste y el consumo de energía. Dado que muchos sistemas integrados están diseñados para producir miles o millones de unidades, el costo unitario es un aspecto importante para considerar en la etapa de diseño (Sepúlveda-Cisneros et al., 2020).

Las principales características específicas de los sistemas embebidos son mencionadas por los autores (Barreto et al., 2022; Torres et al., 2019) a continuación:

- Disponen de unidad central. Los módulos de CPU normalmente pueden estar formados por microcontroladores, FPGAS, microprocesadores o una mezcla de ellos.
- Son sistemas confiables.
- Prácticamente sin mantenimiento. La mayoría de ellos no requieren ningún mantenimiento específico.
- Son sistemas seguros.
- Tienen software de protección incorporado, utilizan protocolos de cifrado y están diseñados para proteger contra intrusiones.
- Ahorran energía.
- Tienen una interfaz de usuario simple. Algunos de ellos cuentan con interfaces gráficas de usuario que facilitan la programación, configuración y/o gestión. Tenga en cuenta que la interfaz de usuario no está diseñada para uso general sino para un uso específico.

1.1.1 El Raspberry Pi

Se trata de un ordenador económico y de formato compacto, diseñado para proporcionar acceso al procesamiento de datos a todos los usuarios. Raspberry Pi es ampliamente utilizado para desarrollar prototipos y aprender a trabajar con computadoras y dispositivos electrónicos (Velasco, 2021).

El nombre de la computadora es un juego de palabras en inglés que se refiere a pastel de frambuesa. La primera parte de la nomenclatura hace referencia a la tradición de las empresas informáticas de tener nombres caracterizados por los nombres de frutas como Apple, Blackberry o Acorn (Digital Guide IONOS, 2022). La segunda parte, Pi, se abrevia como "Python Interpreter" ya que Python es el lenguaje de programación principal de Raspberry Pi (Digital Guide, 2021).

Generalmente todos los proyectos de Raspberry Pi se basan en hardware libre, está sugerido la instalación de Raspberry Pi OS anteriormente, Raspbian, que es una versión especializada de Debian, el sistema operativo gratuito basado en GNU/Linux y una versión específica de Windows 10 IoT (Calvo, 2022). El primer Raspberry Pi fue lanzado en febrero de 2012 por la Fundación Raspberry Pi. Esta organización nació con la idea de promover y enseñar informática en los centros educativos y universidades del Reino Unido (Solé, 2021). Para que sea lo más asequible posible, diseñaron esta PC pequeña y económica con

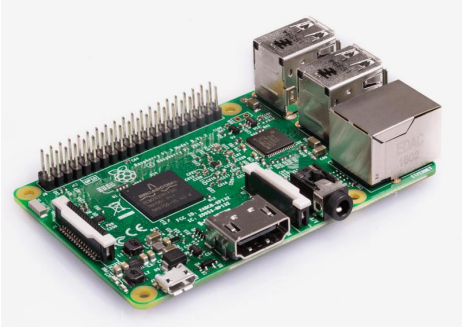
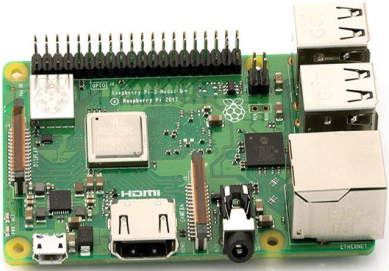
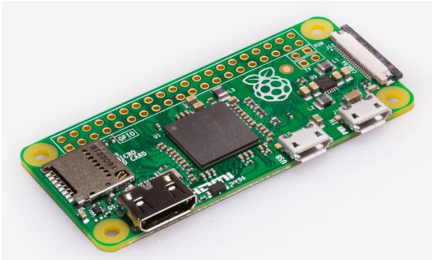

excelentes funciones, cuenta además con una gran conectividad y de conexiones GPIO (Entrada/Salida de Propósito General) que permiten desarrollar una gran variedad de proyectos educativos (Solé, 2021).

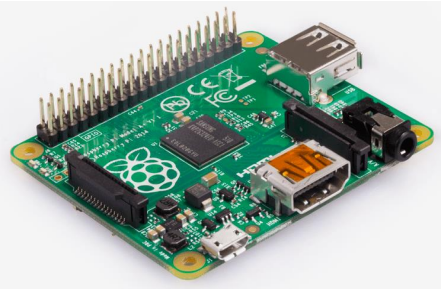

La Raspberry Pi funciona como una computadora de escritorio ya que tiene los mismos componentes en el nivel básico, sus placas se basan en alto rendimiento y eficiencia energética, vienen con modelos de RAM de modelo variable, tienen múltiples salidas de video y las versiones más nuevas tienen 4 pines. Conectores Jack para entrada de micrófono y salida de audio, también cuentan con un lector de tarjetas en el que se instala el sistema operativo y varios puertos USB, así como una gran cantidad de conectores GPIO para tener la posibilidad de crear un gran número de aplicaciones de gran tamaño (Huertos, 2019).

Las aplicaciones que permite la Raspberry Pi a nivel de usuario son amplias. Entre los usos más populares en el campo agrícola, destacan el autor (Digital Guide, 2021) las siguientes:

- **Medidor de madurez de frutas y verduras:** Las redes neuronales se entrenaron en conjuntos de datos que mostraban la madurez de diferentes tipos de frutas y verduras durante un período de 10 días en todas las etapas posibles. Los proyectos muy complejos pueden no ser adecuados para uso doméstico en el futuro, pero ofrecen grandes oportunidades para la agricultura. Por ejemplo, los agricultores pueden reducir significativamente el tiempo que lleva clasificar manualmente los cultivos podridos.
- **Monitor de calidad del aire:** Puede mejorar la calidad del aire, generalmente en las ciudades extensas., se pueden realizar mediciones que tengan en cuenta la humedad y la concentración de partículas con un medidor de calidad del aire.
- **Sistema de riego para tus plantas de interior:** Existe un sistema de riego para bonsáis mediante Raspberry Pi para asegurar que la planta recibe la cantidad de agua necesaria de forma automática.

Hasta el día de hoy existen varios modelos de computadores embebidos Raspberry-Pi entre los que citan los autores (Escalante & Vargas, 2019; MCI electronics, 2022) en la siguiente tabla comparativa:

MODELOS DE RASPBERRY-PI	
MODELO/IMAGEN	CARACTERÍSTICAS
<p>Raspberry Pi 3 B</p> 	<p>Procesador Broadcom BCM2837 a 1,2 GHz Wi-Fi 802.11 b/g/n Bluetooth 4.1 Memoria LPDDR2 de 1 GB 1x Puerto Ethernet 10/100 1x Conector de vídeo/audio HDMI 1x Conector 3.5mm audio/video compuesto. 4x Puertos USB 2.0 - 40x Pines GPIO Conector de pantalla DSI- Ranura de tarjeta microSD Dimensiones: 85 x 56 x 17 m</p>
<p>Raspberry Pi 3 B +</p> 	<p>Procesador Broadcom BCM2837B0 a 1.4GHz Wi-Fi 802.11 b/g/n 5Ghz Bluetooth 4.2 Memoria LPDDR2 de 1 GB Potencia a través de Ethernet (PoE) 1x Puerto Ethernet Gigabit 1x Conector de vídeo/audio HDMI 1x Conector 3.5mm audio/video compuesto. 4x Puertos USB 2.0 - 40x Pines GPIO Conector pant.DSI -Ranura de tarjeta microSD Dimensiones: 85 x 56 x 17 m</p>
<p>Raspberry Pi Zero</p> 	<p>1GHz, CPU de un solo núcleo 512 MB de RAM Puerto mini-HDMI Puerto micro-USB OTG Potencia Micro-USB Cabezal de 40 pines compatible con HAT Conector de cámara CSI (v1.3 solamente)</p>
<p>Raspberry Pi Zero W</p> 	<p>1GHz, CPU de un solo núcleo 512 MB de RAM Puerto mini-HDMI Puerto micro-USB OTG Cabezal de 40 pines compatible con HAT Conector de cámara CSI (v1.3 solamente) Wi-Fi 802.11 b/g/n Bluetooth 4.1 Bluetooth Low Energy (BLE)</p>

MODELO/IMAGEN	CARACTERÍSTICAS
<p data-bbox="341 275 539 304">Raspberry Pi A+</p> 	<p data-bbox="703 342 1327 584"> Procesador: Broadcom BCM2835 SoC Full HD 700 MHz. Memoria RAM: 512 MB SDRAM Almacenamiento a través de tarjetas microSD. Un puerto USB. Salida HDMI. Pines GPIO: 40. Consumo de energía: 600mA. </p>
<p data-bbox="331 656 549 685">Raspberry Pi 3 A+</p> 	<p data-bbox="703 663 1327 1088"> Broadcom BCM2837B0, Cortex-A53 (ARMv8) 64-bit SoC @ 1.4GHz 512MB LPDDR2 SDRAM 2.4GHz and 5GHz IEEE 802.11.b/g/n/ac wireless LAN, Bluetooth 4.2/BLE Extended 40-pin GPIO header- Full-size HDMI Single USB 2.0 ports CSI camera port for connecting a Raspberry Pi Camera Module DSI display port for connecting a Raspberry Pi Touch Display 4-pole stereo output and composite video port Micro SD port for loading your operating system and storing data 5V/2.5A DC power input </p>

La Raspberry Pi 4, es la versión más moderna cuenta con un conector Gigabit Ethernet y una tarjeta Wifi + Bluetooth integrada. El subsistema de gráficos fue reforzado comparativamente con las variantes anteriores, puesto que ahora tiene casi el doble de potencia. Este dispositivo en cuanto a la versión de 8 GB de RAM funciona de forma más holgada con toda clase de aplicaciones y programas que se necesite usar en el día a día, también tiene un conector de alimentación del tipo USB-C y dos conectores MIPI, uno para un display y el otro para una cámara (Solé, 2021).

Las dos salidas micro HDMI que posee la Raspberry Pi 4 hacen que sea viable usarla con una configuración de 2 monitores, lo que ya es una gran ventaja de cara a la productividad. Para iniciar el funcionamiento, es fundamental suministrar energía a la placa a través del conector designado. Además, se requiere cargar el sistema operativo deseado en una tarjeta de memoria SD con el fin de aprovechar todas sus capacidades. Asimismo, será necesario conectar un teclado, un ratón y un monitor de visualización de datos para interactuar con la placa. (Solé, 2021).

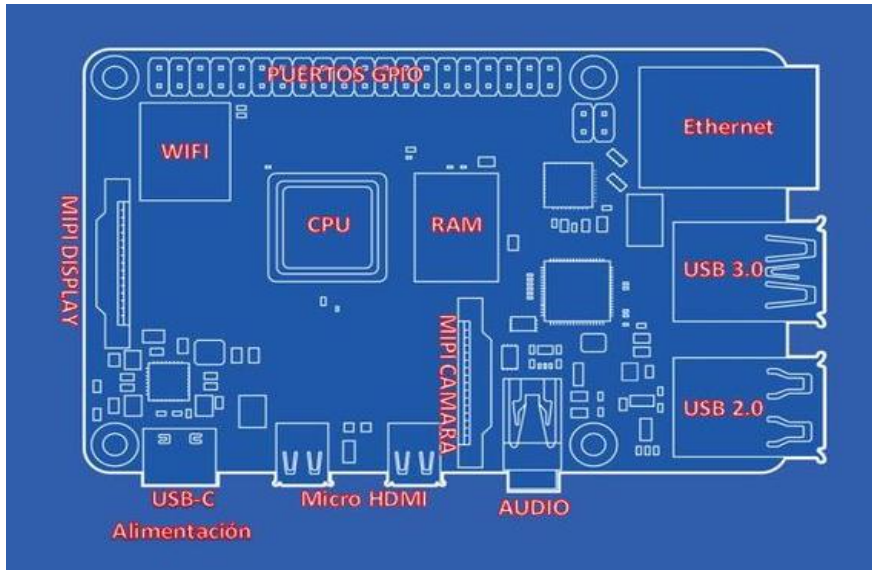


Fig. 1 Arquitectura Raspberry Pi 4

Fuente: (Solé Roberto, 2021).

Una de las características distintivas de Raspberry Pi 4 es que es más potente que Raspberry Pi3. Tanto la CPU como la GPU funcionan más rápido y de manera más eficiente, además de que obtiene configuraciones de RAM cada vez más rápidas, lo que permite que el GPU use RAM y ancho de banda de mayor calidad para hacer su trabajo (se trata de RAM compartida) (Barkstrom, 2019). A continuación, se detalla en la tabla 2 las características de la Raspberry.Pi4:

Tabla 2 Características Raspberry Pi 4

Fuente: (Barkstrom, 2019)

Raspberry Pi 4	
Procesador	Quad Core Cortex A-72 1,5 GHz
Memoria RAM	1, 2, 4, 8 GB LPDDR4
USB	2 x USB 2.0 2 x USB 3.0
Alimentación	USB Tipo-C
HDMI	2 x Micro HDMI
Ethernet	Gigabit sin limitaciones
Wi-Fi	2,4 / 5 GHz
Bluetooth	5.0

1.1.2 Nvidia Jetson Nano

NVIDIA® Jetson Nano™ es una computadora pequeña y potente que brinda alto rendimiento informático para ejecutar cargas de trabajo de inteligencia artificial modernas a un tamaño, potencia y costo sin precedentes, para creadores, estudiantes y desarrolladores estos pueden ejecutar marcos y modelos de IA para aplicaciones como clasificación de imágenes, detección de objetos, segmentación y procesamiento de voz. El kit para desarrolladores puede ser alimentado por micro-USB y viene con extensas E / S, que van desde GPIO a CSI, lo que facilita a los desarrolladores conectar un conjunto diverso de sensores para habilitar una variedad de aplicaciones de IA (Moreira, 2022). Es una plataforma fácil de usar, increíblemente eficiente en energía, consume tan poco como 5 vatios (NVIDIA®, 2021).

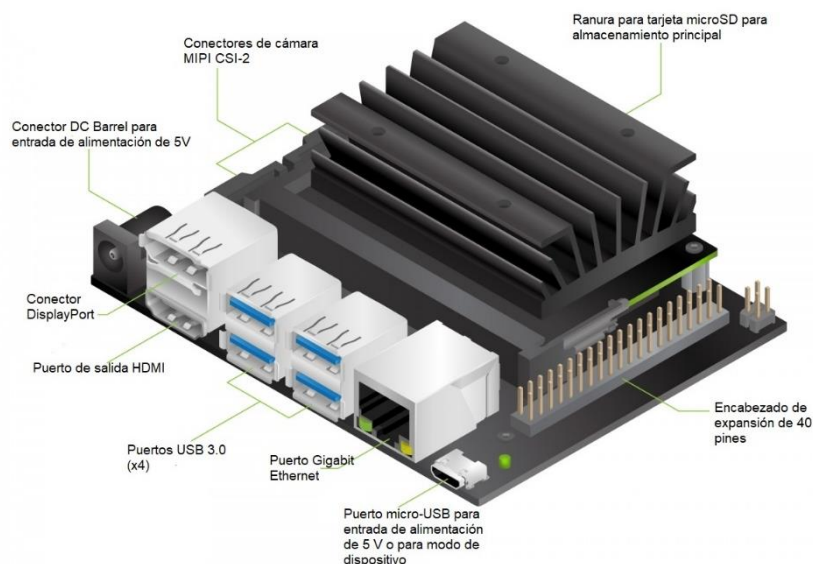


Fig. 2 Arquitectura Nvidia JetsonNano

Fuente: (NVIDIA®, 2021).

Jetson Nano es compatible con el completo NVIDIA® JetPack™ SDK que incluye un soporte de placa (BSP), SO Linux de escritorio completo con controladores NVIDIA, bibliotecas de software NVIDIA CUDA®, cuDNN y TensorRT™ para aprendizaje profundo, API de inteligencia y visión artificiales, computación GPU, procesamiento multimedia, herramientas de desarrollo, documentación, código de muestra y mucho más (A. Ruipérez, 2019).

La plataforma de Nvidia reduce y esfuerzo para desarrolladores de aplicaciones con artificial en arquitecturas Jeston Nano. Incluye todos los periféricos necesarios para desarrollar un sistema integrado utilizando visión por computadora, redes neuronales y más

(Pusdá Chulde, 2022). Cuenta con un sistema operativo base Linux con NVIDIA SDK para construir rápidamente sistemas de IA (Barahona, 2019; Plawrence, 2019). A continuación, en la tabla 3, se detalla las características específicas que contiene un Kit de desarrollador de Nvidia Jetson Nano mencionadas por los autores (LAB Linux, 2019; NVIDIA®, 2021):

Tabla 3 Características Kit desarrollador Nvidia Jetson Nano

Fuente: (LAB Linux, 2019; NVIDIA®, 2021)

ESPECIFICACIONES DEL KIT DE DESARROLLADOR JETSON NANO	
Módulo CPU Jetson	<ul style="list-style-type: none"> ✓ Nano GPU Maxwell de 128 núcleos ✓ Procesador Quad-core Arm A57 a 1,43 GHz ✓ Memoria del sistema – 4GB 64-bit LPDDR4 @ 25.6 GB / s ✓ Almacenamiento: ranura para tarjeta microSD (devkit) o flash de 16 GB eMMC (producción) ✓ Codificación de video – 4K @ 30 4x 1080p @ 30 9x 720p @ 30 (H.264 / H.265) ✓ Decodificación de video – 4K @ 60 2x 4K @ 30 8x 1080p @ 30 18x 720p @ 30 (H.264 / H.265) ✓ Dimensiones – 70 x 45 mm.
Zócalo	<ul style="list-style-type: none"> ✓ Conector SO-DIMM de 260 pines para el módulo Jetson Nano. ✓ Salida de video – HDMI 2.0 y eDP 1.4 (solo video) ✓ Conectividad – Gigabit Ethernet (RJ45) + encabezado PoE de 4 pines ✓ USB: 4x puertos USB 3.0, 1x puerto USB 2.0 Micro-B para modo de alimentación o dispositivo ✓ Cámara I / F – 1x carriles MIPI CSI-2 DPHY compatibles con Leopard Imaging LI-IMX219-MIPI-FF-NANO y el módulo de cámara Raspberry Pi V2 ✓ Expansión ✓ M.2 Socket Key E (PCIe x1, USB 2.0, UART, I2S e I2C) para tarjetas de red inalámbrica ✓ Cabezal de expansión de 40 pines con señales GPIO, I2C, I2S, SPI, UART ✓ Cabezal de botón de 8 pines con señales relacionadas con la alimentación del sistema, el restablecimiento y la recuperación forzada
Misc – Power LED	<ul style="list-style-type: none"> ✓ cabezal de ventilador de 4 pines
Fuente de alimentación	<ul style="list-style-type: none"> ✓ 5V / 4A a través del conector tipo barril de potencia o 5V / 2A a través del puerto micro USB; soporte opcional de PoE
Dimensiones	<ul style="list-style-type: none"> ✓ 100 x 80 x 29 mm

1.2 Drones en la agricultura de precisión

Un "drone" es un vehículo aéreo no tripulado (UAV). En otras palabras, es una máquina que vuela sin que nadie la conduzca. Su trayectoria está programada por software o controlada por un control remoto que se eleva utilizando la fuerza de giro de sus motores adherido en la hélice (Pinto, 2019). La terminología "DRONE" deriva del inglés, y su significado propio es "ZÁNGANO" o "ABEJA MACHO". Además, la expresión drone es denominada el zumbido de estos insectos (Gonzáles, 2022).



Fig. 3 Terminología "DRONE"

Fuente: (Ruipérez Pablo, 2017)

En sus inicios, eran destinados esencialmente al entorno militar, especialmente en la identificación del terreno y realizando ataques. Hoy en día, un dron es un vehículo aéreo no tripulado o UAV controlado por humanos desde una ubicación remota o que vuela de forma autónoma según el modo establecido, destacando su empleo en actividades agrícolas tanto vigilancia de los cultivos como irrigación del terreno (Parra, 2017).

En la actualidad, la agricultura internacionalmente se inclina por la estabilidad alimentaria a futuro, razón por la cual se está en la investigación de tecnologías modernas que apoyen a mitigar la insuficiente atención a los cultivos que producen costos innecesarios en el monitoreo de estos (Sánchez, 2019). A partir del siglo XIX hasta esta época se ha venido desarrollando y elevando la producción de máquinas o unidades no tripuladas como una tecnología potente empleada en el campo de la agricultura de precisión (Ríos, 2021).

Los diseños de UAV incluyen muchas formas, tamaños, configuraciones y características diferentes. Históricamente, los UAV eran simplemente aviones controlados remotamente (en inglés: drones), sin embargo, constantemente se utiliza más el control independiente de los UAV. Cabe resaltar que las aeronaves controladas remotamente en verdad no califican para ser denominadas como VANT, debido a que los vehículos aéreos pilotados remotamente (o por control remoto) son identificados como Aeronaves Radiocontroladas o Aeronaves R/C; esto ya que, claramente, los VANT son además sistemas autónomos que tienen la posibilidad de operar sin participación humana alguna a lo largo de su manejo en la tarea a la que se haya encomendado, o sea, tienen la posibilidad de despegar, volar y aterrizar automáticamente (Moreno, 2016).

La mayoría de los modelos generalmente cuentan con cuatro hélices, lo que les otorga el nombre de cuadricópteros. Sin embargo, es posible encontrar drones con tres, seis o incluso ocho hélices. Las hélices más habituales en los drones son las de dos palas, aunque algunos modelos utilizan hélices de tres o cuatro palas (AERLYPER, 2022). Para mantener estable el dron, dos de estas hélices giran en una dirección y las otras dos giran en la dirección opuesta. Cada hélice es impulsada por un pequeño motor accionado electrónicamente (Ruipérez, 2017).

Los vehículos aéreos no tripulados o UAV tienen la posibilidad de ser de uso civil y militar, los más pequeños muestran corto alcance y tienen la posibilidad de subir a alturas no mayores a 300m. Como dispositivos de uso civil tienen la posibilidad de capturar imágenes, audio y otro tipo de variables como temperatura, humedad, etc. Además, ciertos de estos dispositivos cuentan con GPS, lo que les posibilita desplazarse de un espacio a otro sin necesidad de un control remoto (Moreno, 2016).

1.2.1 Funcionamiento de un Dron

El funcionamiento del dron es básicamente el mismo que en un avión o helicóptero, el motor arranca, la hélice gira y puede empezar a volar. Luego se usa los controles para navegar el vuelo. Depende en gran medida del conocimiento y la habilidad del piloto. En algunos casos, el curso puede configurarse automáticamente. El modelo que hace esto tiene un GPS incorporado que puede cambiar y ajustar el vuelo hasta llegar a un punto determinado observando lo que captura la cámara. A veces se transmite o se graba en tiempo real (Ledesma et al., 2019) .

El dron se controla de forma remota y puede realizar una variedad de movimientos. Se puede girarlo hacia la derecha o hacia la izquierda alrededor de los ejes vertical y vertical, ascender verticalmente y girar hacia adelante y hacia atrás con respecto al eje horizontal.

Todos estos movimientos se controlan ajustando la fuerza motriz de cada hélice. Esto se hace con un dispositivo llamado controlador/a central de vuelo, cuya función principal es mantener el vuelo lo más estable posible para el dron (Chávez, 2018).

El núcleo del controlador incluye diversos elementos, siendo uno de los más destacados el magnetómetro. Este dispositivo se encarga de medir tanto la intensidad como la orientación del campo magnético, funcionando de manera similar a una brújula que señala el polo norte magnético. El controlador recibe información sobre la ubicación del dispositivo a través de la unidad GPS, que es un sistema de posicionamiento y navegación por satélite (AERLYPER, 2022). Algunos profesionales usan un magnetómetro separado del magnetómetro del controlador de vuelo para evitar interferencias (González, 2022; Moreno, 2016).

Un sistema de nivel múltiple, especialmente en aviones no tripulados, conduce a una dependencia especial y también ayuda a reducir los errores. En su interior un Dron tiene un motor enorme en el interior que puede lograr un mayor control sobre su altura y, por lo tanto, tiene una opción de carga más grande a lo largo del vuelo. Las hélices obtienen su energía de una fuente específica, y la mayoría de estos dispositivos cuentan con baterías removibles, lo que les permite mantenerse en el aire y funcionar durante períodos prolongados. La duración de vuelo puede extenderse utilizando baterías de larga duración." (González, 2022).

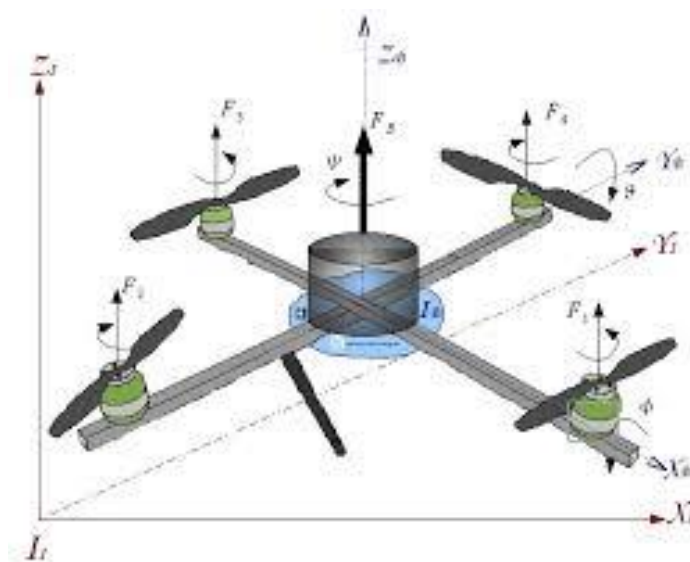


Fig. 4 Sistema multi hélice de un dron

Fuente (González Miguel, 2021)

1.2.2 Componentes de un Dron

El primer dron del que se tiene constancia data del año 1922 y lo construyó George de Bothezat, a pesar de que no consiguió levantarlo a más de cinco metros del suelo. Estas aeronaves sin tripulación humana están compuestas por varios componentes, que describen los autores (Raparelli & Bajocco, 2019) a continuación:

Componentes mecánicos

- Estructura. Parte donde se montan y se apoya el resto de los componentes. Su función principal es reducir al máximo las vibraciones producidas por los motores al hacer girar las hélices.
- Hélices. El número de hélices dependerá del número de rotores que tenga el dron. Su función es la de impulsar los motores y estabilizar el aparato en el aire.
- Motores. Van conectados a las hélices y se encuentran justo debajo de éstas, en la parte exterior de la estructura.

Componentes eléctricos

- Control electrónico de velocidad. Controla la velocidad y la dirección del dron.
- Batería. Alimenta a todos los componentes eléctricos.
- Control remoto. Es el dispositivo que controla el dron y que nos permitirá manejarlo desde donde estemos.
- Placa controladora. Su función es conseguir la estabilidad en el vuelo transmitiendo información al Control Electrónico de Velocidad.

1.2.3 Principales movimientos que realiza un dron

Los principales movimientos realizados por un dron se mencionan a continuación por el autor (Hispa Drones, 2019):

- Guiñada. Hacia la derecha o izquierda del eje vertical.
- Inclinación. Hacia la derecha o izquierda del eje longitudinal.
- Cabeceo. Rotación hacia delante o hacia atrás con respecto al eje transversal.
- Altitud. Elevación en vertical.

1.2.4 Clasificación de drones

Existen muchas posibles formas de clasificar los drones, en las siguientes figuras 5 y 6 se plantea una posible clasificación simplificada que muestra los principales tipos de drones según el autor (Ledesma et al., 2019):

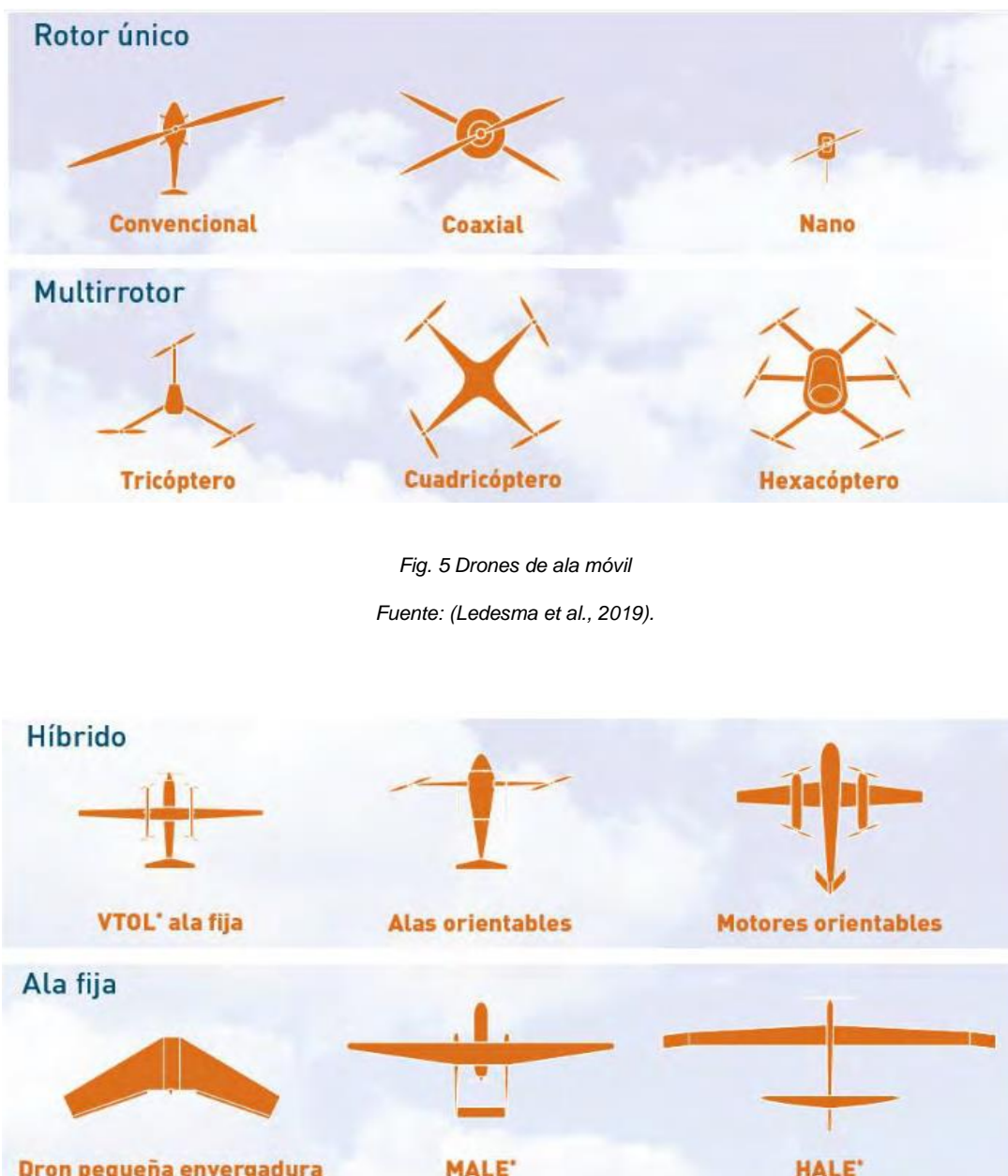


Fig. 5 Drones de ala móvil

Fuente: (Ledesma et al., 2019).

Fig. 6 Drones de ala Fija

Fuente: (Ledesma et al., 2019).

1.2.5 Uso de drones en agricultura de precisión

La Agricultura de precisión es un enfoque agronómico que implica la gestión personalizada de los cultivos, basada en la comprensión de las diferencias y variaciones presentes en una explotación agrícola. Esta tecnología tiene como objetivo adaptar la administración de las explotaciones agrícolas de acuerdo con las necesidades específicas de cada área de cultivo (Ledezma et al., 2019). Es decir, se persigue solucionar el problema allí donde se produce y con ello reducir costos y tratamientos innecesarios, para así optimizar el rendimiento, mejorando la rentabilidad de los cultivos y la disminución del impacto ambiental, ya que la aplicación de agroquímicos es dirigida y ajustada a los requerimientos reales de cultivo (Ledezma et al., 2019).

Para caracterizar esta variabilidad se utilizan herramientas tecnológicas como los Sistemas de Posicionamiento Global, sensores planta-clima-suelo e imágenes multiespectrales obtenidas a partir de satélites, aviones o drones (Ledezma et al., 2019).

La tecnología agrícola se desarrolla continuamente cada día, brindando nuevas posibilidades de aumentar la productividad y la seguridad de los agricultores. El dron es una herramienta que existe desde hace años, pero solo recientemente comenzó a usarse en la agricultura. La mayor accesibilidad de estas herramientas y el aumento de la digitalización en la agricultura han convertido a los drones en un aliado de la producción agrícola mundial (Ríos, 2021).

Los drones, o Vehículos Aéreos No Tripulados (UAV), cumplen múltiples funciones en la agricultura, como el mapeo de campos, la vigilancia y monitoreo de los cultivos, plagas y enfermedades, la eficiencia de irrigación, y la aplicación de plaguicidas, entre otros (A. López, 2021). Adicionalmente, traen múltiples beneficios, como la aplicación precisa, localizada y en áreas de difícil acceso, una menor exposición del aplicador, ahorro de agua y tiempo, y el aumento de la productividad del agricultor (CropLife, 2021).

Utilizados principalmente para capturar imágenes y proporcionar datos, los drones permiten el monitoreo permanente de un cultivo desde la siembra hasta la cosecha, también pueden ayudar a los agricultores a reaccionar más rápidamente ante las amenazas, como las malezas, insectos y hongos; ahorrar tiempo en la exploración de cultivos para tomar las acciones apropiadas; y mejorar la aplicación de las tasas variables de insumos en tiempo real; Los datos capturados por los UAV se procesan y se traducen en información útil para tener parámetros que aporten información como la salud de las plantas frente a infestaciones de plagas y enfermedades (Chávez, 2018).

La principal potencialidad del uso de los drones es la capacidad de obtener una vista panorámica de grandes extensiones de tierra cultivable. Si añadimos un sensor térmico y una cámara multiespectral, será más fácil y preciso controlar cuánto se riega el cultivo. Así, el agricultor puede controlar con exactitud qué zonas de su campo están demasiado secas o qué partes del cultivo están expuestas a un posible exceso de humedad. De esta manera, no solo se puede administrar mejor la cantidad de agua necesaria, sino también comprender cómo distribuir el agua para aumentar la eficiencia de las plantas (Hispa Drones, 2019).

Si bien las aplicaciones de los drones en la agricultura son diversas, todo este trabajo busca mapear algunas de las variables, a continuación, los autores (Ledesma et al., 2019; Pino et al., 2019) relacionan las más generalizadas:

- **Gestión eficiente del agua:** Cuando los cultivos experimentan estrés hídrico, es decir, falta de agua, las hojas tienden a cerrar sus estomas, lo que reduce la transpiración y aumenta la temperatura de las hojas. Para controlar esta situación, se utilizan sensores de temperatura que permiten estimar las necesidades de agua de cada planta. Esto facilita la aplicación precisa de la cantidad adecuada de agua, lo que a su vez ahorra recursos hídricos.
- **Aplicación localizada de herbicidas:** Para minimizar el uso de herbicidas y aplicarlos solo en las áreas donde existen malezas, es necesario detectar y mapear las malezas de manera precisa. Esto permite ajustar la dosis y el tipo de herbicida de manera eficiente.
- **Optimización del uso de fertilizantes:** La detección de deficiencias nutricionales en los cultivos mediante sensores multiespectrales permite la aplicación de fertilizantes solo cuando es necesario, evitando el desperdicio y reduciendo los costos.
- **Detección temprana de enfermedades y plagas:** Mediante la adquisición de imágenes multiespectrales en momentos específicos y su análisis, es posible detectar cambios en las plantas que podrían indicar la presencia de enfermedades o plagas. Esto ayuda a programar medidas de control de manera anticipada.
- **Monitoreo de áreas desinfectadas:** Los drones proporcionan una vista panorámica que facilita el seguimiento de las actividades realizadas en fincas y campos agrícolas.
- **Evaluación de la calidad de los cultivos:** Utilizando imágenes multiespectrales de drones junto con mediciones de campo, se pueden obtener indicadores basados en sistemas de información geográfica (SIG) que evalúan la calidad de los cultivos o la producción.
- **Inventario de cultivos:** Aunque en áreas extensas los drones no pueden igualar la resolución de los aviones y satélites, son útiles en lugares de difícil acceso o con

infraestructura limitada. Permiten realizar inventarios de cultivos en áreas más pequeñas de manera efectiva.

- **Conteo de árboles:** Los drones ofrecen una alternativa eficiente al conteo manual de árboles en un campo. Proporcionan información sobre todo el campo de manera simultánea y permiten evaluar la distribución de los árboles para optimizar su exposición al sol.
- **Estudios de cultivos:** Los estudios de cultivos a menudo se apoyan en imágenes multispectrales de aviones y satélites para identificar áreas afectadas. Sin embargo, la evaluación puede ser menos precisa cuando se trata de daños parciales en los cultivos. Los drones permiten obtener imágenes multispectrales de áreas específicas, lo que proporciona una herramienta objetiva en el proceso de evaluación incluso en casos de daño parcial.

1.2.6 Tipos de imágenes adquiridas con drones

Los tipos de imágenes de agricultura de precisión que obtienen los drones pueden llevar este tipo de sensores (herramientas agrícolas y tractores, robots automatizados) dotando a la agricultura de un sinnúmero de ventajas basadas en la recogida, análisis e interpretación de datos, gracias a las imágenes, profesionales agrícolas conocer las necesidades exactas de los insumos, tanto a nivel global como específico, los autores (Rodríguez & Espejo, 2020) y (Bollatti & Juárez, 2021) mencionan que tipos de imágenes se obtienen:

Imágenes RGB

Una imagen RGB, también conocida como imagen de color verdadero, se almacena como una matriz de datos $m \times n \times 3$, que define los componentes rojo, verde y azul para cada píxel individual.

Las imágenes en formato RGB no emplean paletas de colores. En lugar de ello, el color de cada píxel se define mediante la combinación de intensidades de los colores rojo, verde y azul, las cuales se almacenan en planos de color específicos en la ubicación correspondiente del píxel. En los archivos de gráficos, las imágenes RGB se guardan típicamente como imágenes de 24 bits, donde cada componente de color (rojo, verde y azul) se representa con 8 bits, lo que permite una gama potencial de 16 millones de colores. La precisión con la que se pueden representar las imágenes reales es la razón por la que se denominan "imágenes en color verdadero" (MathWorks, 2023).



Fig. 7 Imagen RGB tomada con dron

Fuente: (Bollatti & Juárez, 2021)

Imágenes 3D

La imagen 3D es un proceso para renderizar una imagen tridimensional en una superficie bidimensional creando la ilusión óptica de profundidad. Normalmente, las imágenes 3D utilizan dos lentes de cámara fijas o móviles, colocadas a poca distancia entre sí, para capturar imágenes de objetos tridimensionales. Este proceso recrea eficazmente la visión estereoscópica del ojo humano. La imagen aparece como dos imágenes planas que el ojo del espectador ve por separado, creando una ilusión visual de profundidad a medida que su cerebro combina las imágenes en una sola.

Las imágenes en 3D se generan a menudo mediante dos imágenes separadas, que se muestran una al lado de la otra o se superponen en una sola imagen. En la técnica de estereoscopia, se colocan dos fotografías estáticas una junto a la otra, y el espectador las observa con cada ojo por separado, es decir, la imagen izquierda con un ojo y la imagen derecha con el otro. Esta forma de fotografía estereoscópica tiene sus raíces en los primeros días de la fotografía y se basa en un proceso más sencillo de imágenes 3D, ya que solo requiere dos cámaras para capturar las dos imágenes estáticas necesarias. Además, estas imágenes pueden ser visualizadas independientemente por cada ojo sin la necesidad de dispositivos ópticos adicionales.

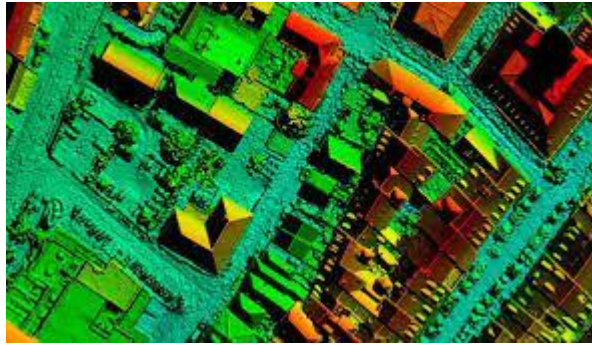


Fig. 8 Imagen 3D tomada con dron

Fuente: (Bollatti & Juárez, 2021)

Imágenes espectrales

Nos referimos a aquellas representaciones visuales de la información que son capturadas por un sensor introducido en una plataforma (normalmente embarcados en satélites artificiales o drones). Su principal distinción radica en la obtención de datos a partir de longitudes de onda específicas en el espectro electromagnético. Su principal propósito es adquirir información de la superficie terrestre y procesarla según sea necesario. Estas tecnologías son conocidas como imágenes hiperespectrales o multiespectrales y pueden ser de gran utilidad en el ámbito agrícola, ya que pueden mejorar la toma de decisiones y contribuir al avance de la agricultura de precisión (Bollatti & Juárez, 2021).

Las imágenes espectrales generalmente se generan mediante sensores que capturan la señal, la descomponen ópticamente en múltiples bandas y la envían al detector, lo que permite la creación de la imagen resultante.

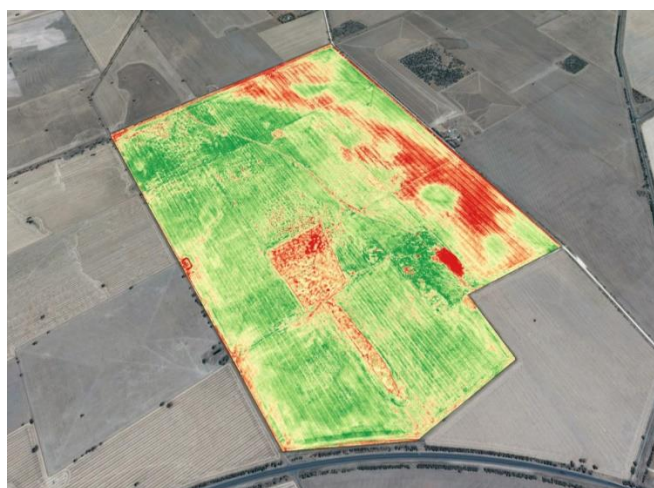


Fig. 9 Imagen espectral tomada con dron

Fuente: (Bollatti & Juárez, 2021)

Imágenes infrarrojas- térmicas

Los objetos irradian energía térmica desde la superficie. Las cámaras que capturan imágenes térmicas, también llamadas cámaras termográficas, tienen un sensor llamado micro bolómetro para identificar la energía térmica, los micro bolómetros están compuestos por una matriz de píxeles y diferentes componentes que pueden traducir la energía recibida en un valor de temperatura. A partir de esto, se produce un termograma para asignar los colores de tono de cada uno de los valores de temperatura elevados. Las imágenes térmicas capturan la luz infrarroja, identifican el calor presente en el objeto, hacen que la temperatura de transformación visualmente en la imagen, esta tecnología permite a los usuarios obtener lecturas y medir la temperatura de cada escena o punto de interés de una superficie o área.

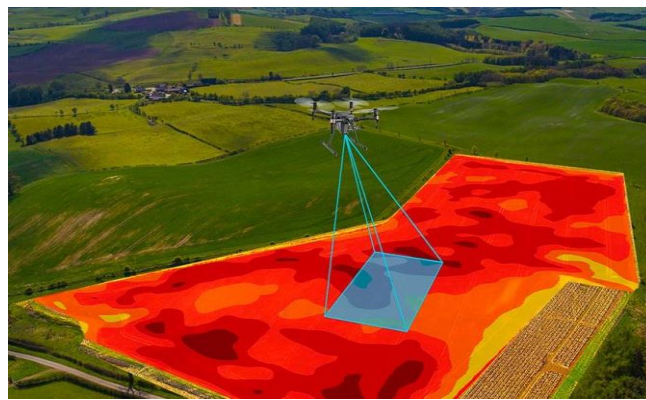


Fig. 10 Imagen infrarroja-térmica tomada con dron

Fuente: (Bollatti & Juárez, 2021)

1.3 Aplicaciones en agricultura de precisión

La agricultura de precisión, conocida también como agricultura específica por sitio, usa tecnologías de información espacial, tales como los sistemas de posicionamiento global (GPS) y sistemas de información geográfica (SIG), para mejorar las decisiones agronómicas de diferentes cultivos (Ríos, 2021).

La agricultura de precisión se fundamenta en la recopilación de datos, que se someten a análisis e interpretación con el objetivo de mejorar tanto la productividad como la sostenibilidad en la producción agrícola (Parada, 2021). Este enfoque emplea de manera intensiva tecnologías interconectadas. Las herramientas digitales relacionadas con la agricultura de precisión permiten la monitorización constante de una amplia gama de métricas, como niveles de precipitación, humedad, requerimientos nutricionales y tipos, así

como análisis de muestras de suelo, el entre otros. Esto facilita la supervisión detallada de los cultivos y la cría de ganado (Parada, 2021).

Además, las herramientas de comando y control centralizadas permiten reconocer cuándo hay que regar un campo determinado, lo que a su vez permite utilizar menor cantidad de agua para el riego (Parada, 2021).

1.3.1 Monitoreo cultivos

En la actualidad, tanto las explotaciones agrícolas como los inversores no tienen la posibilidad de permitirse escoger una parcela a ciegas. Dichas inversiones implican altos riesgos, por lo que es sumamente importante comprender cómo de prometedora es la tierra que se está comprando (EOS DATA ANALYTICS, 2020). Es entonces una vez que entra en juego el monitoreo de cultivos, que da diferentes resoluciones para examinar el estado de los campos y sus cultivos, tanto a grado nacional como local (Schram, 2020).

El monitoreo es la inspección regular y cuidadosa de las plantas cultivadas en todo el lapso de incremento: al hacer el monitoreo, el productor/la productora camina por medio de su cultivo intentando encontrar inconvenientes en las plantas como insectos y ácaros, patologías, malezas, mal provocado por tormentas y estreses del medio ambiente como sequía o deficiencias de nutrientes (Clare & Timothy, 2017). Este sienta las bases para compatibilizar el ambiente con la producción, debido a que, al tener una radiografía persistente de los lotes, la utilización de insumos fitosanitarios se optimiza. Se usan solo una vez que se necesita y se dictamina cual utilizar según la existencia de otros organismos (Igarzábal, 2016).

La importancia del monitoreo de cultivos es imprescindible ya que permite tener conocimiento sobre en qué momento surge y cómo evoluciona un problema fitosanitario es imprescindible para realizar tácticas de desempeño efectivas, que dañen menos al ambiente y que afecten en lo mínimo al rendimiento y calidad del cultivo. La elección de ocupar un tipo de medida de control dependerá del umbral y del grado de mal económico. Un conveniente monitoreo posibilita ofrecer un panorama real del grado de infección y/o infestación del patógeno en cuestión, partiendo de lo anterior, se considerarán hacer las ocupaciones de control que corresponden en tiempo y forma, sobre todo, cuando se tienen esquemas de Manejo Integrado de Plagas (MIP) y/o Enfermedades (MIE) (Abad & Farez, 2018).

Además, el seguimiento de los cultivos ayuda a identificar problemas en una etapa temprana, dando tiempo a los productores para abordarlos antes de que el cultivo enferme gravemente. Para cultivos de vida corta como tomates y maíz, la detección temprana de

problemas de plagas es importante para garantizar una cosecha oportuna antes de que sea demasiado tarde. Si los agricultores no aceptan la plaga, la detección temprana significará más tiempo para identificarla. El monitoreo deficiente de los cultivos puede conducir a pérdidas significativas en el cultivo (Delgadillo, 2019) .

Entre los primordiales beneficios que otorga el monitoreo de cultivos son mencionados por el autor (Pérez, 2018) a continuación:

- Ahorro en mano de obra, agua, energía y fertilizantes.
- Mejora la calidad de los productos cosechados.
- Aumenta la producción.
- Los precios de producción se disminuyen.
- Es viable hacer un estudio predictivo de efectos negativos por lluvias, plagas y patologías en las plantas.

1.3.2 Detección de malezas

La agricultura de precisión busca la aplicación de insumos en cultivos agrícolas en el sitio, el instante y la porción adecuados (A. F. Jiménez et al., 2020). La detección de malezas de un lugar específico es un plan de la AP que posibilita la reducción en la aplicación de herbicidas, minimizando precios de insumos, con efectos positivos para el medioambiente (Goñi et al., 2018).

La infestación de malezas fuepreciada como un elemento culpable de el decrecimiento en el rendimiento de los cultivos, que representa pérdidas económicas (Abouzahir et al., 2019). Estas afectaciones se generan al emerger las malezas en la línea de siembra, compitiendo con la planta por los nutrientes, el agua y la luz del sol, a lo largo de su fase de aumento (A. Jiménez et al., 2020). Las numerosas dosis de herbicida y la resistencia de las malas hierbas conforman un problema serio en la agricultura mundial, lo cual hace que la utilización de agroquímicos sin control tenga efectos negativos, entre los que resaltan el gasto innecesario de herbicida (pérdida económica), el mal medioambiental (contaminación del suelo y aguas subterráneas), y las trazas de agroquímicos en los alimentos (afectan la salud y estabilidad alimentaria) (Rehman et al., 2019; Wang et al., 2019).

A partir de finales del siglo XX, la aplicación colectiva de novedosas tecnologías ha mejorado las prácticas de administración de la agricultura y dio sitio al campo de la AP (di Cicco et al., 2017). Estas tecnologías se basan en el desempeño de los recursos de forma más eficiente, teniendo presente las condiciones del medio ambiente y las necesidades reales de las plantas (A. F. Jiménez et al., 2020). La AP vincula tácticas de compra y estudio de

información en campo y la aplicación de insumos según con límites establecidos por medio de sistemas capaces, en donde un aspecto primordial es el desempeño específico por lugar para una producción optimizada y eficiente de cultivos de campo (Farooq et al., 2018).

El desempeño específico de malezas con conceptos de agricultura de precisión busca utilizar el herbicida en el sitio, la porción y el instante adecuados, para mejorar la productividad, minimizar el desperdicio de insumos, sin influir al medioambiente. Para lograr este objetivo se implementan sistemas de control de malezas, en donde los sistemas robóticos juegan un papel importante. Dichos sistemas tienen que ser capaces de ubicar malezas en el campo y utilizar pulverizadores de herbicidas que dirigen su aplicación de manera directa sobre ellas (Lottes et al., 2018).

La detección de malezas en las imágenes utilizando la caracterización de manera y localización de los surcos del cultivo posibilita decidir el tipo y el número de malezas por imagen (A. F. Jiménez et al., 2020). El aumento de malezas pasa de manera no uniforme, sin embargo, estas se otorgan naturalmente agrupadas; o sea, tienen la posibilidad de crecer en (o entre) las filas del cultivo. La detección de las malas hierbas podría ser instantánea y estricta si se separa del suelo; este procedimiento puede solucionar los inconvenientes técnicos para la aplicación rigurosa de los pesticidas utilizando vehículos de tierra de navegación automatizada (Wang et al., 2019).

1.3.3 Fertilización de suelos

La fertilidad del suelo se refiere a su función de apoyar el crecimiento de las plantas mediante la producción de nutrientes esenciales para las plantas. Los principales factores que determinan la fertilidad son: físico, químico y biológico; todos importantes para obtener el grado esperado. Un desempeño correcto de las técnicas para la fertilidad del suelo pertenece a los gigantes fines de cualquier campesino, puesto que sus cosechas crecerán correctamente y no verá mermado el rendimiento de estas ni los ingresos (Meza et al., 2017).

Una de las funciones del suelo es proporcionar nutrientes a las plantas. El contenido de nutrientes en el suelo se llama fertilidad del suelo. Sin embargo, en algunos casos las tierras fértiles no son muy productivas. Esto significa que puede haber suelos con alto contenido de nutrientes, pero baja biomasa vegetal (Lottes et al., 2018). Esto se debe a que los nutrientes permanecen en el suelo, pero no pueden llegar a las plantas debido a varias limitaciones. Estos límites son acidez, sodio, salinidad, condición del agua, capacidad de almacenamiento de agua. En general, estas limitaciones son difíciles de resolver y requieren una inversión significativa, lo que limita la votación sobre alternativas productivas. Estos

problemas a menudo se extienden más allá del terreno y se extienden a grandes áreas. (AgroSpray, 2021).

Los nutrientes son fundamentales para la vida y para la permanencia de los habitantes del mundo (humana, animal y vegetal). Los fertilizantes son un tipo de sustancia, orgánica o inorgánica, que tiene nutrientes que son asimilables por las plantas (Muñoz & Muyulema, 2022). Se utiliza para conservar o aumentar el contenido de dichos recursos en el suelo, mejorar la calidad del sustrato a grado nutricional, excitar el incremento vegetativo de las plantas, etc (S. López et al., 2018).

La fertilidad del suelo puede potenciarse incorporando cultivos de defensa que añadan materia orgánica al suelo, lo cual optimiza su composición y promueve un suelo sano y fértil; usando abono verde o cultivando leguminosas para fijar el nitrógeno del viento por medio del proceso de fijación biológica de nitrógeno; implementando microdosis de fertilizante para reponer las pérdidas que se generan por medio de la absorción de las plantas y otros procesos; y disminuyendo al mínimo las pérdidas provocadas por la lixiviación por abajo del área de raíces de los cultivos, por medio de la gestión avanzada de agua y nutrientes (Herrera et al., 2017).

Aunque puede parecer solo el sitio que pisamos o donde se colocan las plantas, la verdad es que un suelo fértil tiene la función de proveer el agua y nutrientes necesarios para las plantas que en él habitan. Por consiguiente, la fertilidad del suelo en la agricultura es algo bastante a considerar para todos esos que se dedican al campo (AgroSpray, 2021). No solo las plantas se ven beneficiadas por esto, hay microorganismos y otros seres vivos, cuya aportación es una pieza más del engranaje, y sin los cuales todo el ecosistema se vendría debajo. Es por esto por lo que lograr conservar la fertilidad en niveles óptimos es lo que dará buenos resultados, tanto a corto como a extenso plazo (EOS, 2021).

1.3.4 Detección de enfermedades

Se trata de determinar la naturaleza y la causa de la enfermedad y, por tanto, requiere un análisis inteligente de las observaciones de campo, así como buenos sistemas de detección de patógenos ante problemas biológicos. En algunos casos, el diagnóstico puede ser tan simple como reconocer los síntomas en la planta, reconocer los signos del patógeno y confirmar su identidad si se ha informado previamente. Su complejidad y urgencia son comparables con una investigación de detectives, ya que las preocupaciones relacionadas a los problemas fitopatológicos llegan tarde y constantemente se olvida que es mejor prevenir que curar (Roldán Ortega et al., 2019).

Existen métodos para determinar las enfermedades de cualquier planta, como llevar muestras de tejido vegetativo a un laboratorio especializado o llevar a un ingeniero agrónomo experto al sitio del cultivo, en cualquiera de los dos métodos, la desventaja radica en el tiempo necesario para obtener los resultados (Roldán Ortega et al., 2019). Así también, pruebas de diagnóstico de calidad de suelos, análisis de aguas, pH, nutrientes y pesticidas, son útiles para identificar los factores abióticos que afecten al cultivo (Cabrera et al., 2019; Roldán Ortega et al., 2019).

En entornos agrícolas, resulta esencial reconocer la apariencia común de una planta como punto de referencia para llevar a cabo comparaciones morfológicas de manera precisa. Esto significa que, antes de evaluar cualquier desviación o anomalía en el crecimiento o aspecto de una planta, es necesario tener una comprensión clara de cómo debería lucir en condiciones normales. Conocer el nombre científico y la variedad o cultivar, ayudan a identificar el origen genético, grado de susceptibilidad, verificar la especificidad del patógeno y evitar confusiones con nombres genéricos del cultivo (Roldán Ortega et al., 2019). Examinar la variabilidad de signos y síntomas, ya sea por la falta o sobre desarrollo de tejidos, necrosis de órganos, apariencia anormal del cultivo, distribución de plantas afectadas (patrones de infección) o revisar si existe progresión de los síntomas, pueden, por ejemplo, ayudarnos a descartar el daño producido por químicos u otros agentes abióticos (Cabrera et al., 2019).

1.4 Lenguajes de programación

En la actualidad, los idiomas de codificación representan una herramienta esencial para abordar desafíos en diversas áreas de la ciencia y la tecnología. En particular, la metodología empleada para la resolución de una variedad de problemas en el ámbito de la Matemática consiste en plantear un algoritmo, programarlo en algún lenguaje de programación y ejecutar el programa en un ordenador (Martin et al., 2021).

La adopción de herramientas tecnológicas para programar las tareas de las maquinarias agrícolas contribuye a aumentar en forma exponencial la eficiencia y productividad en el manejo de información en el campo, se puede observar que la programación de operaciones o recursos en la agricultura, ha sido estudiada hace más de 50 años; se identifica como un tema de gran relevancia, no solo por los aportes al crecimiento y desarrollo de este sector de la economía, sino por los retos que genera la aplicación de las herramientas de programación, a las condiciones particulares del sector agrícola (Ramírez et al., 2018).

Así mismo, se detecta un desafío importante más allá de las condiciones de la agricultura como lo es, las variaciones de acuerdo con el cultivo (especificaciones del producto que se cultiva), la programación adecuada de recursos o actividades genera un impacto en los costos, de igual manera, se puede detectar factores importantes en la asignación o programación de actividades o recursos: condiciones del cultivo, forma de realizar las operaciones, condiciones meteorológicas, forma de programar y calidad del producto que pueden ser de utilidad como insumo para llevar a cabo investigaciones enfocadas a la optimización de actividades o recursos en la agricultura (Díaz et al., 2018). A continuación, se da a conocer tres tipos de lenguajes de programación Python, Matlab y C++, los cuales permiten la creación de algoritmos que pueden ser implementados en la agricultura de precisión para ayudar a sistematizar, monitorear, detectar malezas, fertilizar los cultivos, etc.

1.4.1 Python

Python es un lenguaje de programación, desarrollado por el matemático Guido Van Rossum entre finales de los años 80 y principios de los 90, cuyo nombre está inspirado en el grupo de humoristas británicos Monty Python (Alvarez, 2021). Se trata de un lenguaje muy sencillo y versátil que permite aprender a implementar código de forma muy fácil, obviando las complicadas reglas sintácticas de otros lenguajes (Marzal Varo et al., 2016).

Python es un lenguaje interpretado¹, las instrucciones o código fuente escrito por el programador en este lenguaje de alto nivel son traducidos por el intérprete² al lenguaje entendido por la máquina (lenguaje de máquina). Este proceso se repite cada vez que se ejecutan las distintas sentencias que componen el programa. Este lenguaje admite un modelo de programación orientado a objetos y, a menudo, proporciona una forma conveniente de crear programas con componentes reutilizables (Algar Diaz & Fernández de Sevilla Vellon, 2019).

Python dispone de un intérprete por la línea de comandos en el que se pueden introducir sentencias. Cada sentencia o instrucción se ejecuta y produce un resultado visible, que ayuda a clarificar la comprensión del código escrito y verifica la validez de los resultados de la ejecución de pequeñas porciones de código de forma inminente (Alvarez, 2021; Barceló, 2019).

Los autores (Algar Diaz & Fernandez de Sevilla Vellon, 2019; Alvarez, 2021) mencionan las siguientes propiedades, que hacen de Python un lenguaje ideal para iniciarse en el interesante mundo de la programación:

- Los programas escritos en Python son más compactos³ que los programas escritos en lenguajes como C, Java, etc.
- Considerado un idioma de muy alto nivel⁴.
- Como se mencionó anteriormente, es interactivo.
- Sintaxis simple, clara, legible e intuitiva. Todos estos componentes hacen que el código sea más fácil de leer y escribir.
- Fácilmente ampliable. El lenguaje en sí tiene muchas bibliotecas, funciones y tipos de datos diseñados para funcionar con Python.
- Este es un lenguaje que distingue entre mayúsculas y minúsculas, lo que significa que distingue entre mayúsculas y minúsculas.
- Soporta múltiples paradigmas: programación estructurada o imperativa y funcional. Por ejemplo, la esencia de este libro es utilizar Python para aprender modelado estructurado.
- Es multiplataforma: puedes usarlo en Windows, Unix/Linux, Mac/OS o Microsoft. • Gratuito en todos los ámbitos, incluido el empresarial. Esto contribuye a su creciente popularidad. Empresas como Yahoo, Disney, NASA, Red Hat, etc. todos usan esto.

Python es un lenguaje de programación de propósito general muy potente y flexible que también es simple y fácil de aprender. Es un lenguaje de alto nivel que permite manejar fácilmente todo tipo de estructuras de datos, tanto numéricas como textuales. A continuación, los autores (Covantec, 2019; García, 2022) presenta algunas ventajas y desventajas sobre el lenguaje Python:

Ventajas:

- Sencillo y rápido. - Este lenguaje facilita mucho la programación: "te obliga a adaptarte al modo del lenguaje de programación, Python proporciona una plantilla". Este es un gran idioma para los conjuntos, si necesita algo rápidamente (en el sentido del rendimiento del lenguaje), con algunas líneas que se han permitido.
- Elegante y flexible. - El lenguaje te brinda muchas herramientas: si necesitas una lista de diferentes tipos de datos, no necesitas declarar cada tipo de datos. Es un lenguaje flexible por lo que no necesitas preocuparte demasiado por los detalles.
- Programación saludable y eficiente. La programación en Python se está convirtiendo en un estilo de programación muy saludable: fácil de aprender, perfectamente orientado a reglas, depende de la perfección, sigue reglas, usa cadenas, variables. Además, es un lenguaje diseñado pensando en la productividad, lo que significa que Python aumenta la productividad al permitirle realizar su trabajo en el momento adecuado.

- Limpio. - El orden que mantiene Python es lo que más gusta a los usuarios, es muy legible, cualquier otro programador puede leer y trabajar en un programa escrito en Python. Los módulos están bien organizados, a diferencia de otros idiomas.
- Mano. - Es un lenguaje muy portátil (ya sea en Mac, Linux o Windows) en comparación con otros lenguajes. La filosofía de la batería incluida es que estas son las librerías que más necesitas en tu programación diaria, ya están incluidas en el traductor, no necesitas instalar adicionales para otros lenguajes.
- Comunidad. - Una cosa que es muy importante para el desarrollo de un lenguaje es la comunidad, la propia comunidad Python se encarga del lenguaje y casi todas las actualizaciones se realizan de forma democrática.

Desventajas

- Curva de aprendizaje. "El proceso de aprendizaje cuando ingresas al elemento web no es fácil".
- Almacenamiento. - La mayoría de los servidores no soportan Python y, si lo hacen, es un poco complicado de configurar.
- Contiene bibliotecas. - Algunas bibliotecas predeterminadas no son del agrado de toda la comunidad y prefieren utilizar bibliotecas de terceros.

A diferencia de otros lenguajes de programación como C, C++ o Java, Python es interpretado y dinámicamente tipado. Lo que quiere decir que no es necesario compilar el código fuente para poder ejecutarlo (interpretado) y que sus variables pueden tomar distintos tipos de objetos (dinámicamente tipado) (Villalvazo, 2020). Esto hace que el lenguaje sea sumamente flexible y de rápida implementación, aunque pierde en rendimiento y es más propenso a errores de programación que los anteriores lenguajes (Mochales, 2020).

Python es ideal para trabajar con grandes volúmenes de datos ya que, el ser multiplataforma, favorece su extracción y procesamiento, por eso lo eligen las empresas de Big Data. A nivel científico, tiene una gran biblioteca de recursos con especial énfasis en las matemáticas para aspirantes a programadores en áreas especializadas (Medrano, 2021). Y si esto fuera poco, es posible crear videojuegos, aunque no es tan eficiente como Java o C# (Robledano, 2019c).

1.4.2 Matlab

Es un programa que se centra en cálculos utilizando matrices, incluyendo muchos algoritmos para resolver problemas de aplicación y tecnología. Proporciona un entorno interactivo sencillo a través de una ventana en la que podemos introducir órdenes en modo

texto y en la que aparecen los resultados. Los gráficos se muestran en ventanas separadas y cada ventana dispone de una barra de menús que controla su funcionalidad (Benítez et al., 2019).

Lo que diferencia a MATLAB de otros sistemas informáticos es la facilidad para trabajar con vectores y matrices. Las operaciones comunes (suma, producto, exponenciación) operan en matrices de forma predeterminada sin más restricciones que la coherencia del tamaño caso por caso. Una de las características más notables de MATLAB son sus capacidades gráficas (Agud & Pla, 2020).

MATLAB tiene comandos de lenguaje de programación comunes para bucles y bifurcaciones condicionales, y los comandos contenidos en archivos ASCII se pueden ejecutar utilizando un editor como el Bloc de notas o el editor de archivos de comandos integrado. A continuación se describen algunas ventajas de usar Matlab según el autor (Pazmiño et al., 2018):

- La optimización adecuada se realiza de forma rápida y con gran precisión. • Soporte matemático ampliado, así como la capacidad de utilizar mayor precisión en los cálculos.
- Se pueden conectar en paralelo (MPI, PVM, OpenMP).
- Soporte ampliado para funciones ya desarrolladas.
- Creación rápida de prototipos. Se integra con dispositivos de hardware.
- Una comunidad muy grande. Existen foros en Internet donde se intercambian experiencias, se brinda apoyo y demostraciones a los usuarios.
- Comercial, fácil de conseguir y fácil de interactuar con otros desarrolladores.

1.4.3 C++

Es un lenguaje de programación que es una extensión del lenguaje C y permite la manipulación de objetos. Aunque existe desde hace muchos años, su inmenso poder lo convirtió en uno de los lenguajes de programación más buscados en 2019. Fue diseñado a mediados de los años 1980 por el danés Björn Stroustrup. El objetivo era ampliar el (entonces muy exitoso) lenguaje de programación C para incluir los mecanismos necesarios para manipular objetos. De ahí que C contenga modelos de programación estructurados y orientados a objetos, por eso se le llama lenguaje de programación multiparadigma. Por primera vez, el lenguaje C se conoció como "C con clase". Más tarde se cambió a C, abreviatura de "C aumentada", lo que indica que era una extensión del lenguaje de programación C (Hero C, 2021).

Las principales ventajas y desventajas de programar en C++ son mencionadas por los autores (Rivas, 2020; Robledano, 2019a) a continuación:

Ventajas:

- Alto rendimiento. Una de sus principales características es la alta eficiencia.
- Actualizaciones de idioma. A lo largo de los años, el lenguaje se ha actualizado para permitirle crear, conectarse y trabajar con datos complejos, así como implementar una serie de patrones de diseño.
- Multiplataforma
- Extensible: C y C son altamente extensibles. Casi todos los programas o sistemas están escritos o parte de ellos están escritos en estos lenguajes (desde los navegadores web hasta el propio sistema operativo).

Desventajas:

- C++ es que se trata de un lenguaje muy amplio (con muchos años y muchas líneas de código), debe tener una compilación por plataforma y su depuración se complica debido a los errores que surgen.
- EL manejo de librerías es más complicado que otros lenguajes como Java o .Net
- Su curva de aprendizaje muy alta.

Algunas de las características más importantes que posee el lenguaje C++ son mencionadas por los autores (García, 2022; Robledano, 2019a) a continuación:

- Compatibilidad con bibliotecas: A través de bibliotecas hay muchas funciones que están disponible y que ayudan a escribir código rápidamente.
- Orientado a Objetos: El foco de la programación está en los objetos y la manipulación y configuración de sus distintos parámetros o propiedades.
- Rapidez: La compilación y ejecución de un programa en C++ es mucho más rápida que en la mayoría de los lenguajes de programación.
- Compilación: En C++ es necesario compilar el código de bajo nivel antes de ejecutarse, algo que no ocurre en otros lenguajes.
- Punteros: Los punteros del lenguaje C, también están disponibles en C++.
- Didáctico: Aprendiendo programación en C++ luego es mucho más fácil aprender lenguajes como Java, C#, PHP, Javascript, etc.

Las aplicaciones del lenguaje C++ son muy extensas. Se menciona que los navegadores web, sistemas operativos, bases de datos, bibliotecas, aplicaciones gráficas, nubes, videojuegos, compiladores, etc. están escritos o una parte importante de su estructura

está programada en C++. A continuación, se detallan algunas aplicaciones desarrolladas en este lenguaje de programación por el autor (Robledano, 2019) :

- Base de datos: MySQL, una de las bases de datos más utilizadas, está escrita en C.
- Navegadores web: Utilizan C porque necesitan velocidad a la hora de mostrar los resultados en pantalla.
- Sistema operativo: La columna principal de Windows, Linux y Mac OS está escrita en C. Su potencia y velocidad lo convierten en el lenguaje de programación ideal para programar sistemas operativos.
- Compilador. Los compiladores de muchos lenguajes de programación están escritos en C.
- Videojuegos: C ++ todavía se utiliza en el mundo de los videojuegos, tanto para la programación de motores gráficos como para una parte concreta de los videojuegos.
- También tiene otras aplicaciones como dispositivos médicos, relojes inteligentes, etc. debido a su capacidad para estar más cerca del lenguaje de máquina que otros lenguajes de alto nivel.

CAPÍTULO 2

2 Desarrollo

2.1 Análisis de las arquitecturas embebidas

Tanto Raspberry Pi 4 como Jetson Nano son excelentes opciones para desarrollar aplicaciones de agricultura de precisión, pero cada uno tiene sus propias fortalezas y puntos a tener en cuenta.

Raspberry Pi 4 es un sistema versátil que cuenta con un procesador de cuatro núcleos ARM Cortex-A72, hasta 8 GB de opciones de RAM, puertos USB, HDMI, Ethernet y GPIO, lo que le permite conectarse a una amplia gama de sensores y dispositivos. Es una plataforma más accesible en términos de costo y ampliamente adoptada por la comunidad de desarrolladores, lo que significa que hay más recursos y documentación disponibles. Esto puede facilitar el desarrollo y la resolución de problemas (Süzen et al., 2020). Por otro lado, Jetson Nano generalmente se ejecuta en un sistema operativo Linux, como Ubuntu, que se puede instalar en una tarjeta microSD y está especialmente diseñada para aplicaciones de inteligencia artificial y visión por computadora. Utiliza un potente procesador de cuatro núcleos ARM Cortex-A57 y una GPU NVIDIA Maxwell con 128 núcleos CUDA. Esto le otorga una potencia de procesamiento significativamente mayor que la Raspberry Pi 4, lo que la hace particularmente adecuada para tareas computacionales intensivas como el análisis de imágenes y la detección de objetos, además NVIDIA proporciona una amplia documentación y una comunidad activa de usuarios en línea para obtener ayuda y soporte en caso de problemas o dudas, incluso se puede acceder a la misma a través de SSH (Secure Shell) para gestionarla de forma remota desde otro dispositivo (NVIDIA®, 2021; Plawrence, 2019).

Debido a los aspectos antes mencionados se consideró la computadora Jetson Nano como la opción más adecuada para el desarrollo e implementación de un algoritmo para la detección de líneas en cultivos de maíz teniendo presente la ventaja en el ámbito de detección de objetos y procesamiento de imágenes y así poder obtener mayor eficiencia, menor tiempo de ejecución, consumo de memoria, CPU y mayor reconocimiento y clasificación de las líneas y malezas del cultivo con mejor precisión y rapidez con antelación y tomar medidas para controlar su propagación.

2.2 Análisis del lenguaje de programación

Tabla 4 Comparativa de lenguajes de programación

Comparativa de lenguajes de programación			
Características	Python	Matlab	C++
Facilidad de Aprendizaje	Fácil aprendizaje debido a su sintaxis simple y legible.	Sintaxis amigables para matemáticas y simulaciones.	Curva de aprendizaje más empinada debido a la gestión de memoria y características avanzadas.
Sintaxis	Sintaxis clara y legible, similar al lenguaje humano.	Sintaxis orientada a la matriz para operaciones matriciales.	Sintaxis más compleja con múltiples características y reglas.
Rendimiento	Más lento que C++ debido a la interpretación. Puede ser acelerado con bibliotecas como NumPy.	Más rápido que Python en cálculos matemáticos, pero más lento que C++.	Compilado, lo que lo hace más rápido que Python y MATLAB en cálculos intensivos.
extensibilidad	Amplia variedad de bibliotecas y módulos de código abierto disponibles.	Extensible con archivos MEX para código C/C++ integrado.	Amplia variedad de bibliotecas y compatibilidad con código C/C++ nativo.
Aplicaciones Generales	Desarrollo web, análisis de datos, aprendizaje automático, automatización.	Ingeniería, física, matemáticas, análisis numérico, simulación.	Desarrollo de software de alto rendimiento, sistemas embebidos, juegos, aplicaciones de tiempo real.
Aplicación en Agricultura de Precisión	Análisis de datos de sensores agrícolas, control de drones, automatización de maquinaria agrícola.	Análisis de datos geoespaciales, gestión de cultivos, modelado de sistemas agrícolas.	Control de maquinaria agrícola, procesamiento de imágenes satelitales, sistemas de navegación GPS.

2.3 Características del kit de desarrollo del algoritmo


2.3.1 SDK de Jetson nano

NVIDIA JetPack SDK es la solución más completa para crear aplicaciones de IA aceleradas de extremo a extremo. JetPack SDK proporciona un entorno de desarrollo completo para el desarrollo de IA en el perímetro acelerado por hardware. Todos los módulos y kits para desarrolladores de Jetson son compatibles con JetPack SDK (NVIDIA®, 2021).

JetPack SDK incluye Jetson Linux Driver Package con cargador de arranque, kernel de Linux, entorno de escritorio Ubuntu y un conjunto completo de bibliotecas para la aceleración de computación GPU, multimedia, gráficos y visión por computadora (NVIDIA®, 2021). También incluye muestras, documentación y herramientas de desarrollo tanto para la computadora host como para el kit de desarrollo, y admite SDK de nivel superior como DeepStream para análisis de transmisión de video, Isaac para robótica y Riva para IA conversacional (Plawrence, 2019).

2.3.2 Sistema operativo versión

Tabla 5 Especificaciones técnicas de Sistema Operativo Ubuntu Linux y procesador

ESPECIFICACIONES TÉCNICAS DE SISTEMA OPERATIVO UBUNTU LINUX Y PROCESADOR	
	Ubuntu 18.04 LTS
Memoria	3,9 GIB
Procesador	ARMv8 Processor rev 1 (v8l) x4
Gráficos	NVIDIA Tegra X1 (nvgpu)/integrated
Tipo de sistema operativo	64-bit
Disco	30,4 GB

2.3.3 Kit de desarrollo de NVIDIA Jetson Nano

A continuación se detalla todo el Kit de desarrollo de la computadora NVIDIA Jetson Nano (NVIDIA®, 2021):

Supersónico Linux

NVIDIA Jetson Linux 34.1.1 (JetPack 5.0.1 DP) y Jetson Linux 34.1 (JetPack 5.0 DP) proporcionan el kernel de Linux 5.10, el cargador de arranque basado en UEFI, el sistema de archivos raíz basado en Ubuntu, los controladores NVIDIA, los firmwares necesarios, la cadena de herramientas y más. JetPack incluye Jetson Linux con estos aspectos destacados (Plawrence, 2019):

- Soporte para Jetson AGX Orin Developer Kit
- Soporte de emulación para emular el rendimiento de los módulos Jetson Orin NX en Jetson AGX Orin Developer Kit
- Núcleo LTS 5.10
- Sistema de archivos de referencia basado en Ubuntu 18.04
- Entorno de ejecución de confianza OP-TEE 1
- Cargador de arranque UEFI
- Compatibilidad con NVSCI: proporciona utilidades para la transmisión de paquetes de datos entre diferentes aplicaciones y para la comunicación entre procesos (IPC)
- Nueva Jetson Power GUI: herramienta para monitorear el estado térmico y de energía de la plataforma Jetson

TensorRT

TensorRT es un tiempo de ejecución de inferencia de aprendizaje profundo de alto rendimiento para clasificación de imágenes, segmentación y redes neuronales de detección de objetos. TensorRT se basa en CUDA, el modelo de programación paralela de NVIDIA, y optimiza la inferencia para todos los entornos de aprendizaje profundo. Incluye inferencia de aprendizaje profundo y optimizador de tiempo de ejecución para baja latencia y alto rendimiento para aplicaciones de aprendizaje profundo.

cuDNN

La biblioteca CUDA Deep Neural Network proporciona primitivas de alto rendimiento para entornos de aprendizaje profundo. Ofrece implementaciones altamente personalizables de rutinas estándar como convolución hacia adelante y hacia atrás, asociación, unificación y capas de activación. La versión de desarrollo JetPack 5.0/5.0.1 incluye cuDNN 8.3.2.

CUDA

El kit de herramientas CUDA proporciona un entorno de desarrollo completo para que los desarrolladores de C y C++ creen aplicaciones aceleradas por GPU. Este kit de herramientas incluye un compilador para GPU NVIDIA, bibliotecas matemáticas y herramientas para depurar y optimizar el rendimiento de las aplicaciones. La versión de desarrollo de JetPack 5.0/5.0.1 incluye CUDA 11.4.4.

API multimedia

El paquete proporciona API de bajo nivel para el desarrollo de aplicaciones flexibles. API de aplicación de cámara: libargus ofrece una API sincrónica de fotogramas de bajo nivel para aplicaciones de cámara, con control de parámetros de cámara por fotograma, compatibilidad con varias cámaras (incluidas las sincronizadas) y salidas de flujo EGL. Las cámaras CSI de salida RAW que necesitan ISP se pueden usar con el complemento libargus o GStreamer. En cualquier caso, se utiliza la API del controlador del sensor del controlador de medios V4L2.

API del controlador del sensor: la API V4L2 permite la decodificación de video, la codificación, la conversión de formato y la funcionalidad de escalado. V4L2 para codificar abre muchas características como control de tasa de bits, ajustes preestablecidos de calidad, codificación de baja latencia, compensación temporal, mapas de vectores de movimiento y más.

Visión por computador

VPI (interfaz de programación de visión) es una biblioteca de software que proporciona algoritmos de procesamiento de imágenes/visión artificial implementados en PVA 1 (acelerador de visión programable), GPU y CPU. OpenCV es una biblioteca de código abierto para visión artificial, procesamiento de imágenes y aprendizaje automático. JetPack 5.0/5.0.1 Vista previa para desarrolladores VPI 2.0

Nuevos algoritmos:

- Volteo de imagen horizontal y vertical en backends de CPU, GPU y VIC
- Acelerador de flujo óptico (OFA) backend para Stereo Disparity para Jetson AGX Orin
- Vistas de imagen / Recorte en backends de CPU y GPU

- Envoltura de búferes de cuda en enlaces de python para permitir el uso eficiente de VPI junto con otras bibliotecas
- Compatibilidad con enlaces de Python para el rastreador KLT
- JetPack 5.0/5.0.1 Developer Preview incluye OpenCV 4.5.4

Herramientas de desarrollo

CUDA Toolkit proporciona un entorno de desarrollo integral para desarrolladores de C y C++ que crean aplicaciones aceleradas por GPU de alto rendimiento con bibliotecas CUDA. El kit de herramientas incluye Nsight Eclipse Edition, herramientas de depuración y creación de perfiles, incluida Nsight Compute, y una cadena de herramientas para aplicaciones de compilación cruzada (Pusdá Chulde, 2022).

- NVIDIA Nsight Systems es una herramienta de creación de perfiles de todo el sistema de bajo costo que proporciona a los desarrolladores la información que necesitan para analizar y optimizar el rendimiento del software.
- NVIDIA Nsight Graphics es una aplicación independiente para depurar y crear perfiles de aplicaciones gráficas.
- NVIDIA Nsight Deep Learning Designer es un entorno de desarrollo integrado que ayuda a los desarrolladores a diseñar y desarrollar de manera eficiente redes neuronales profundas para la inferencia de datos en aplicaciones.
- SDK y herramientas compatibles.
- NVIDIA DeepStream SDK es un completo conjunto de herramientas de análisis para procesamiento multisensorial y análisis de audio y vídeo basados en IA.

Nativo de la nube

Jetson lleva Cloud-Native al borde y habilita tecnologías como contenedores y orquestación de contenedores. NVIDIA JetPack incluye NVIDIA Container Runtime con integración de Docker, lo que permite aplicaciones en contenedores aceleradas por GPU en la plataforma Jetson.

NVIDIA aloja varias imágenes de contenedor para Jetson en NVIDIA NGC . Algunos son adecuados para el desarrollo de software con muestras y documentación y otros son adecuados para la implementación de software de producción, que contienen solo componentes de tiempo de ejecución. Encuentre más información y una lista de todas las imágenes de contenedores en la página Cloud-Native on Jetson . Los aspectos destacados de JetPack 5.0/5.0.1 Developer Preview Cloud Native incluyen:

- El tiempo de ejecución de Nvidia Container deja de montar bibliotecas de nivel de usuario como CUDA, cuDNN y TensorRT desde el host hasta el interior del contenedor 1

Seguridad

Los módulos NVIDIA Jetson incluyen varias funciones de seguridad, como raíz de confianza de hardware, arranque seguro, aceleración criptográfica de hardware, entorno de ejecución de confianza, cifrado de disco y memoria, protección contra ataques físicos y más. Obtenga información sobre las funciones de seguridad saltando a la sección de seguridad de la guía Jetson Linux Developer. JetPack 5.0/5.0.1 Developer Preview no es compatible con Secure Boot, Disk and Memory Encryption. Estas características de seguridad estarán habilitadas con JetPack 5.0 Production Release.

Seguridad Funcional

El enfoque de NVIDIA Jetson para la seguridad funcional es dar acceso a la base de diagnóstico de errores de hardware que se puede utilizar en el contexto del diseño de sistemas relacionados con la seguridad. Jetson Safety extensión Package (JSEP) proporciona un marco de diagnóstico y notificación de errores para implementar funciones de seguridad y lograr el cumplimiento del estándar de seguridad funcional. JetPack 5.0/5.0.1 Developer Preview no es compatible con JSEP.

2.4 Desarrollo del algoritmo

2.4.1 Importación de Librerías

Para el desarrollo del algoritmo se trabajó con el editor Spyder (Python) de desarrollo integrado y multiplataforma de código abierto (IDE), para ello lo primero que se realizó fue la importación de librerías como OpenCV que permite la identificación de objetos, búsqueda de imágenes similares y reconocimiento de escenarios, Numpy se especializa en hacer cálculos numéricos y el análisis de datos, específicamente en volúmenes grandes de datos, Skimage.morphology import dilation, librería que posibilita dilatar, erosionar el tamaño y la forma de las imágenes, además de la librería Copy módulo crea copias de diferentes objetos de Python, principalmente colecciones mutables (listas y los diccionarios) y utilizar operaciones genéricas de copia profunda y superficial, como se muestra a continuación en la figura 11:

```

1  # Importación de Librerías
2  import cv2
3  import numpy as np
4  #from skimage.morphology import dilation, disk;
5  import copy
6  import math
7  from time import time
8

```

Fig. 11 Código Importación de librerías

Fuente: (Propia)

2.4.2 Leer Imagen y visualizar

Para la lectura y visualización de la imagen se realizó la identificación, lectura en formato JPG y cambio del tamaño de la imagen, como se muestra a continuación en la figura 12:

```

11
12  img = cv2.imread('C:\\Users\\MonyMonii\\Desktop\\imagenespruebas\\semana_2\\15m\\1.JPG')
13
14  img = cv2.resize(img, (960,540))
15

```

Fig. 12 Código leer imagen y visualizar

Fuente: (Propia)

2.4.3 Extracción de valores RGB y Cálculo E*G

Para la extracción de valores RGB y cálculo de escala de grises se dividió la imagen en componentes rojo, verde y azul, para luego definir el número de iteraciones y calcular a partir de la imagen dividida, se aplicó doble filtro para el cálculo de E*G de esta, luego se definió la anchura y altura de la imagen así como también el umbral de erosión y se inició las matrices de filtro de área de puntos en el cultivo aplicando la función kernel para suavizar la imagen como se muestra en la figura 13:

```

16 blue, green, red = cv2.split(img)
17
18 iterations=1
19
20 exg = 2*(red)-(green)-(blue)
21
22 w,h=exg.shape
23
24 humbral_erocion=1
25
26 filtro_area_puntos = np.zeros((w,h), np.uint8)
27 filtro_area_cultivos = np.zeros((w,h), np.uint8)
28
29 filtro_area_cultivos_p = np.zeros((w,h), np.uint8)
30 umbral,threshold = cv2.threshold(exg,0,255,cv2.THRESH_BINARY+cv2.THRESH_OTSU)
31
32 kernel = np.ones((2,2),np.uint8)

```

Fig. 13 Código extracción de valores RGB y cálculo E*G

Fuente: (Propia)

2.4.4 Primer Filtro

A continuación se aplicó erosión a la imagen con umbral binario usando un núcleo definido como kernel con un cierto número de iteraciones, la imagen se usó luego para encontrar contornos externos usando la función findContours() y se almacenó en una variable(contornos:filtro), seguidamente estos contornos se ordenó de forma descendente según su área utilizando la función contourArea() y se almacenó en una nueva variable(ordenar_contornos). Además de que se calculó la posición media de los contornos ordenados dividiendo la longitud de ordenar_contornos por 2 y redondeando a un número entero usando int(). Este valor se almacenó en la posicion_mediavariabile y se imprime en la consola.

Seguidamente se detectó los contornos en la imagen erodida y se ordenó de manera descendente según su área, luego se seleccionó los contornos de los cultivos requeridos en su posición relativa en la imagen y se dibujó en una nueva imagen (filtro_area_cultivos_p) para luego detectar las líneas verticales en la imagen (filtro_area_puntos) que atraviesan, como se muestra en la Figura 14 (a) y (b).

Primer Filtro

```
35 ##### primer filtro #####
36
37 erosion_filtro = cv2.erode(threshold,kernel,iterations=iterations)
38 contornos_filtro,hierarchy = cv2.findContours(erosion_filtro, cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_SIMPLE)
39 ordenar_contornos= sorted(contornos_filtro, key=cv2.contourArea, reverse= True)
40
41
42 posicion=int(len(ordenar_contornos))
43 posicion_media=int(posicion/2)
44 print('cantidad contornos',posicion_media)
45
46 ii=0
47 for c1 in ordenar_contornos:
48     ii=ii+1
49     if ii<posicion_media:
50         M = cv2.moments(c1)
51         if M['m00'] != 0:
52             cx = int(M['m10']/M['m00'])
53             cy = int(M['m01']/M['m00'])
54             cv2.line(filtro_area_puntos, (cx, cy), (cx, cy+20), (255),2)
55             cv2.drawContours(filtro_area_cultivos_p, [c1], -1, (255),1)
56
57
58 lineas = cv2.HoughLines(filtro_area_puntos, 7, np.pi, 200)
59
60 # Obtenga la longitud de la matriz de línea Hough
61 for i in range(len(lineas)):
62     for rho, theta in lineas[i]:
63         a = np.cos(theta)
64         b = np.sin(theta)
65         x0 = a * rho
66         y0 = b * rho
67         x1 = int(x0 + 1000 * (-b))
68         y1 = int(y0 + 1000 * (a))
69         x2 = int(x0 - 1000 * (-b))
70         y2 = int(y0 - 1000 * (a))
71
72         cv2.line(filtro_area_cultivos, (x1, y1), (x2, y2), (255), 2)
73
74
75 operacion and filtro=cv2.bitwise and(filtro area cultivos,filtro area puntos)
```



a) Primer filtro

b) Resultado de la imagen de líneas verticales

Fig. 14 Código primer filtro

Fuente: (Propia)

2.4.5 Contorno de líneas

Para el contorno de líneas se creó un borde rectangular `np.zeros` con el ancho y la altura y el tipo especificados `np.uint8`, dos matrices `recuadro_bordey` resultado `lineas` del mismo ancho y alto se inicializan con 0. Luego se dibujó un rectángulo `recuadro_borde` usando `cv2.rectangle` y se realizó una operación AND bit a bit entre `recuadro_bordey` `filtro_area_cultivos`, los contornos de la imagen resultante del paso anterior se encuentran usando `cv2.findContours` se almacenó en `contornos_lienas_dibujadas` y un bucle itera sobre los contornos que se encuentran en `contornos_lienas_dibujadas` como se muestra en la Figura 15:

```

86 ##### LINEAS #####
87
88 recuadro_borde = np.zeros((w,h), np.uint8)
89 resultado_lineas = np.zeros((w,h), np.uint8)
90 cv2.rectangle(recuadro_borde, (5,5),(h-5,w-5),255,-1)
91
92 operacion_and_lineas=cv2.bitwise_and(recuadro_borde,filtro_area_cultivos)
93
94 contornos_lienas_dibujadas,hierarchy = cv2.findContours(operacion_and_lineas, cv2.RETR_TREE, cv2.CHAIN_APPROX:
95
96 print('longitud',len(contornos_lienas_dibujadas))
97
98 ii=0
99 vector_distancia=np.zeros(len(contornos_lienas_dibujadas)-1, np.float16)
100 jj=0
101 var_tem_x=0
102 var_tem_y=0
103
104 #if ii>0 and ii<len(contornos_lienas_dibujadas):

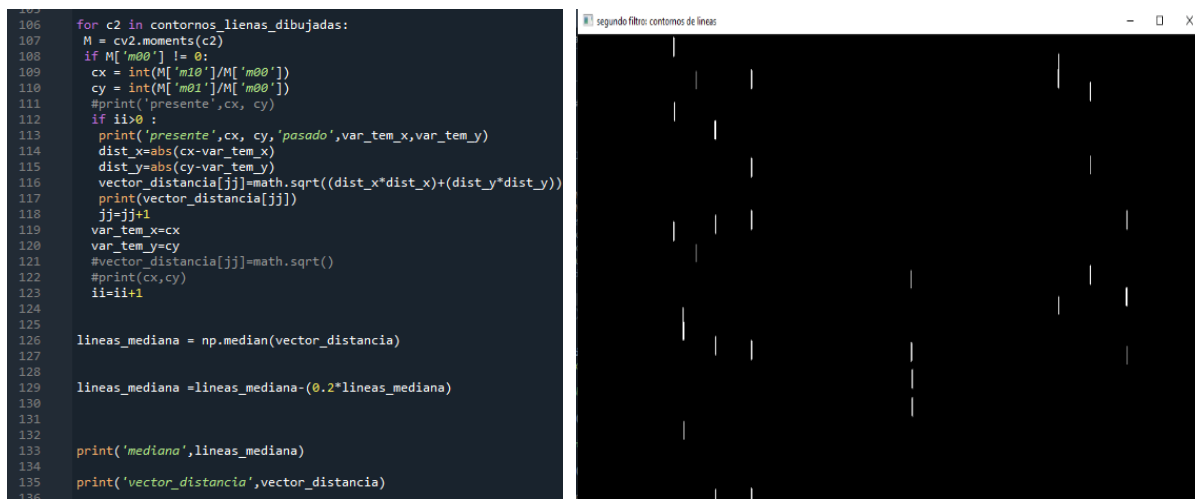
```

Fig. 15 Código contorno de líneas

Fuente: (Propia)

Posteriormente se calculó la distancia mediana entre contornos consecutivos en `contornos_lienas_dibujadas` para esto Primero, se recorre cada contorno `c2y` `contornos_lienas_dibujadas` calcula su centroide `cxy` `cyusa` `cv2.moments()`. Luego se verificó si este es el primer contorno en el bucle (es decir, `iies` mayor que 0). En caso contrario, calcula la distancia euclidiana entre la curva de nivel actual y la anterior mediante la fórmula $\text{math.sqrt}((\text{dist}_x*\text{dist}_x)+(\text{dist}_y*\text{dist}_y))$, donde `dist_xy` `dist_yson` las diferencias absolutas entre las coordenadas del centroide actual y las anteriores (`var_tem_xy` `var_tem_y`, respectivamente). Se almacenó esta distancia en la `vector_distancia` matriz en el índice `jj`, donde `jj` hay un contador que se incrementa en 1 después de cada cálculo de distancia y después de que se hayan calculado todas las distancias, el código calcula la distancia media entre contornos consecutivos usando `np.median(vector_distancia)`. Luego resta el 20 % de este valor medio y asigna el resultado a `lineas_mediana` mostrando los contornos de líneas en el cultivo como se muestra en la figura 16 (a) y (b):

Resultado de la aplicación de contorno de líneas



(a) Código contorno de líneas

(b) Resultado de la imagen contorno de líneas

Fig. 16 Resultado de la aplicación de contorno de líneas

Fuente: (Propia)

2.4.6 Total, de líneas detectadas

Este último bloque de código se agregó las líneas detectadas alrededor de los bordes de los cultivos en la imagen original. Se realizó un ciclo para recorrer todos los contornos encontrados en la imagen filtrada, calculando su centro de masa y la distancia entre cada uno de ellos. Luego se calculó la mediana de las distancias y se estableció un umbral para considerar que un contorno representa una línea si su distancia al contorno anterior es mayor que la mediana calculada.

Finalmente, se dibujó las líneas detectadas en la imagen original y se muestra el resultado. La cantidad de líneas detectadas se muestra en el título de la imagen. También se muestra el tiempo de ejecución del algoritmo como se muestra en la Figura 17.

```

137 ##### proceso de filtrado #####
138
139 ii=0
140 jj=0
141 for c2 in contornos_lienas_dibujadas:
142     M = cv2.moments(c2)
143     if M['m00'] != 0:
144         cx = int(M['m10']/M['m00'])
145         cy = int(M['m01']/M['m00'])
146         if ii<len(contornos_lienas_dibujadas)-1:
147             #cv2.line(img, (cx, 0), (cx, w), (255),2)
148             if vector_distancia[ii] > lineas_mediana:
149                 #cv2.circle(resultado_lineas, (cx, cy), 1, (255), -1)
150                 cv2.line(img, (int(cx), 0), (int(cx), w), (255),2)
151                 jj=jj+1
152                 # cv2.circle(img, (cx, cy), 1, (255), -1)
153                 #vector_distancia[jj]=math.sqrt()
154                 #print(cx,cy)
155                 ii=ii+1
156
157 cv2.line(img, (cx, 0), (cx, w), (255),2)
158 print(jj+1)
159 cv2.imshow('resultado linea-- Cantidad de lineas : '+str(jj+1), img)
160
161 #cv2.imshow('imagen',img)
162 cv2.imshow('imagen t',filtro_area_puntos)
163 cv2.imshow('imagen img',operacion_and_filtro)
164
165 cv2.waitKey(0)
166 cv2.destroyAllWindows()

```

Fig. 17 Código Total de líneas detectadas

Fuente: (Propia)

Es importante destacar que algunos parámetros, como el tamaño de los filtros y el umbral para la detección de líneas, se fundamentan empíricamente y pueden requerir definiciones según el tipo de imagen y el tamaño de los cultivos a detectar como se muestra a continuación en la figura 18:

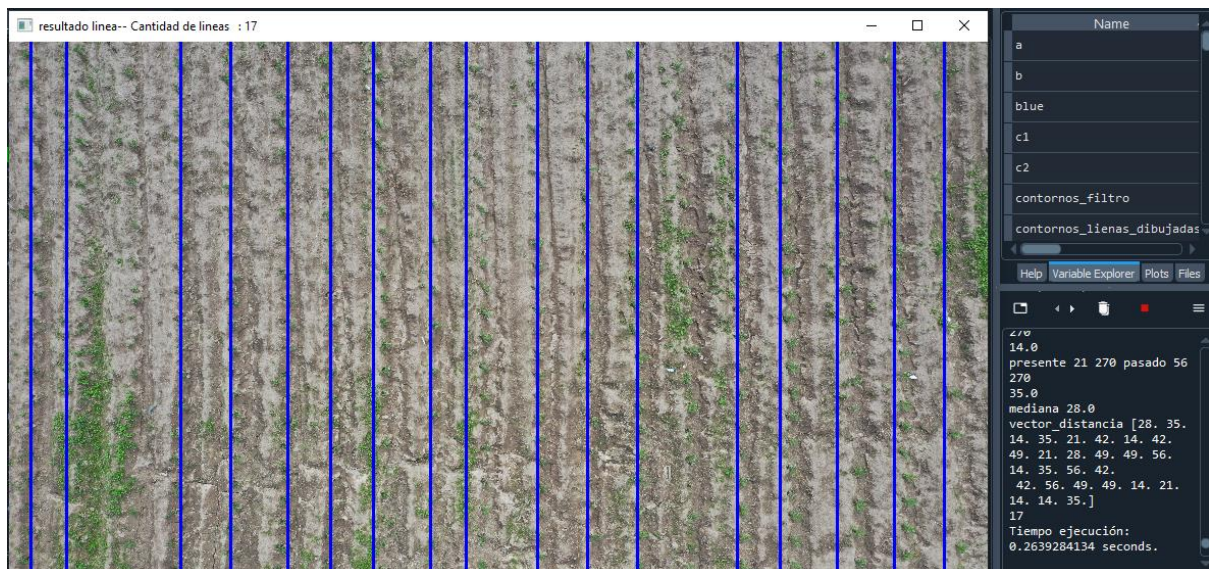


Fig. 18 Resultado Cantidad de líneas detectadas en el cultivo

Fuente: (Propia)

CAPÍTULO 3

3 RESULTADOS

3.1 Pruebas con imágenes offline

Para el análisis de resultados de la investigación una vez implementado el aplicativo se realizó pruebas de rendimiento del algoritmo con un conjunto de Imágenes off-line capturadas con dron a 10 y 15 metros de altitud de la semana 1 a la semana 4, de 5472 ancho por 3648 alto de pixeles en un cultivo de maíz. Para evaluar los tiempos de ejecución (TE) en segundos y el número de líneas identificadas con el algoritmo (N L I A) las imágenes tienen 24-bits de profundidad con representación de color en canal espectral R, G y B, en formato JPG, como se muestra en la tabla 6:

Tabla 6 Pruebas de rendimiento del algoritmo

Pruebas de rendimiento del algoritmo				
Semanas	Metros	Imagen	N L I A	T E (segundos)
1	10	1	12	0.158
		2	13	0.167
		3	12	0.314
		4	13	0.308
	15	1	15	0.307
		2	17	0.275
		3	15	0.311
		4	15	0.300
2	10	1	10	0.090
		2	10	0.078
		3	10	0.078
		4	10	0.119
	15	1	18	0.405
		2	18	0.339
		3	18	0.341
		4	18	0.347
3	10	1	12	0.175
		2	13	0.229
		3	11	0.220
		4	13	0.201
	15	1	17	0.173
		2	17	0.243
		3	17	0.279
		4	19	0.281
4	10	1	13	0.252
		2	14	0.258
		3	14	0.244
		4	13	0.220
	15	1	18	0.310
		2	18	0.319
		3	18	0.332
		4	16	0.333

3.2 Definición de métricas a analizar

3.2.1 Tiempo de ejecución

En la Tabla 7 y figura 19, se presenta los tiempos de ejecución (TE) promedio para el procesamiento de imágenes obtenidas con el dron del algoritmo ejecutado en la computadora Jetson Nano para la detección de líneas en los cultivos de maíz de la semana 1.

Tabla 7 Tiempo de Ejecución (T E) Semana 1

Tiempo de Ejecución (T E) Semana 1			
Semana	Metros	Imagen (segundos)	
1	10	1	0.158
		2	0.167
		3	0.314
		4	0.308
	15	1	0.307
		2	0.275
		3	0.311
		4	0.300

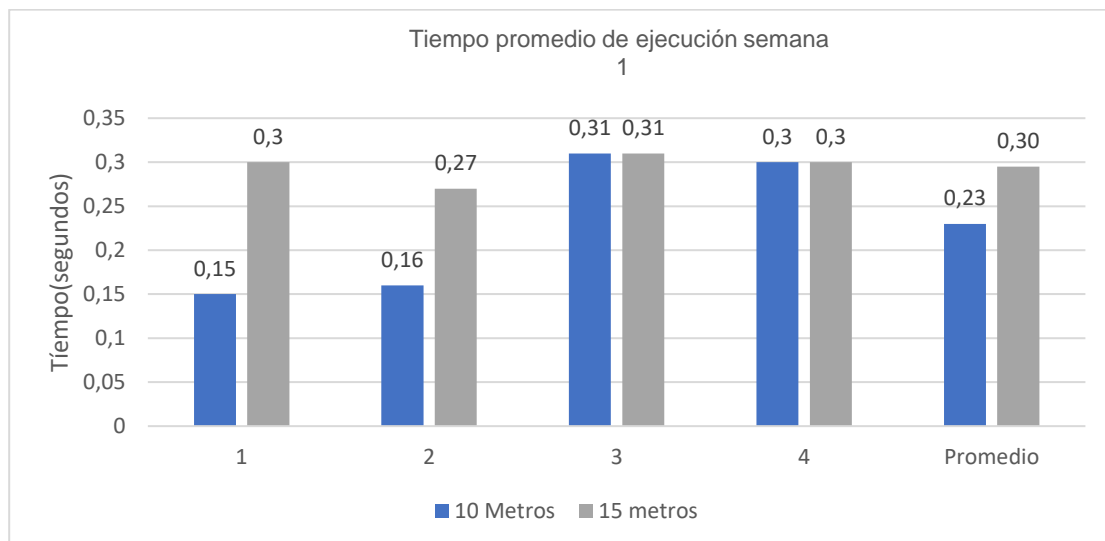


Fig. 19 Tiempo de Ejecución (T E) Semana 1

En la Tabla 8 y figura 20, se presenta los tiempos de ejecución (T E) promedio para el procesamiento de imágenes obtenidas con el dron del algoritmo ejecutado en la Jetson Nano para la detección de líneas en los cultivos de maíz de la semana 2.

Tabla 8 Tiempo de Ejecución (T E) Semana 2

Tiempo de Ejecución (T E) Semana 2			
Semana	Metros	Imagen	T E (segundos)
2	10	1	0.090
		2	0.078
		3	0.078
		4	0.119
	15	1	0.405
		2	0.339
		3	0.341
		4	0.347

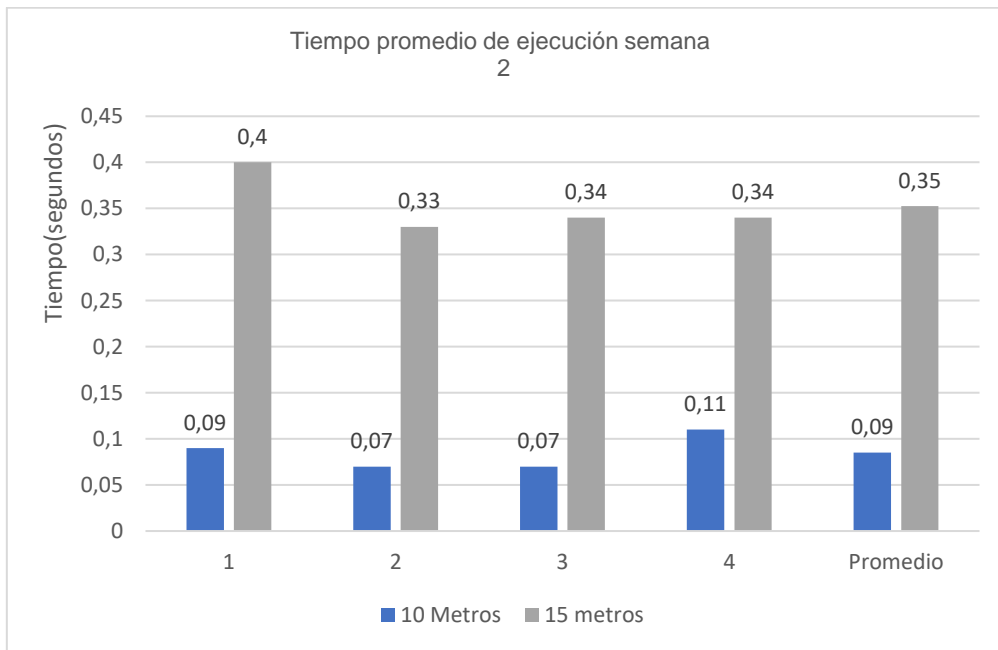


Fig. 20 Tiempo de Ejecución (T E) Semana 2

En la Tabla 9 y figura 21, se presenta los tiempos de ejecución (T E) promedio para el procesamiento de imágenes obtenidas con el dron del algoritmo ejecutado en la Jetson Nano para la detección de líneas en los cultivos de maíz de la semana 3.

Tabla 9 Tiempo de Ejecución (T E) Semana 3

Tiempo de Ejecución (T E) Semana 3			
Semana	Metros	Imagen	T E (segundos)
3	10	1	0.175
		2	0.229
		3	0.220
		4	0.201
	15	1	0.173
		2	0.243
		3	0.279
		4	0.281

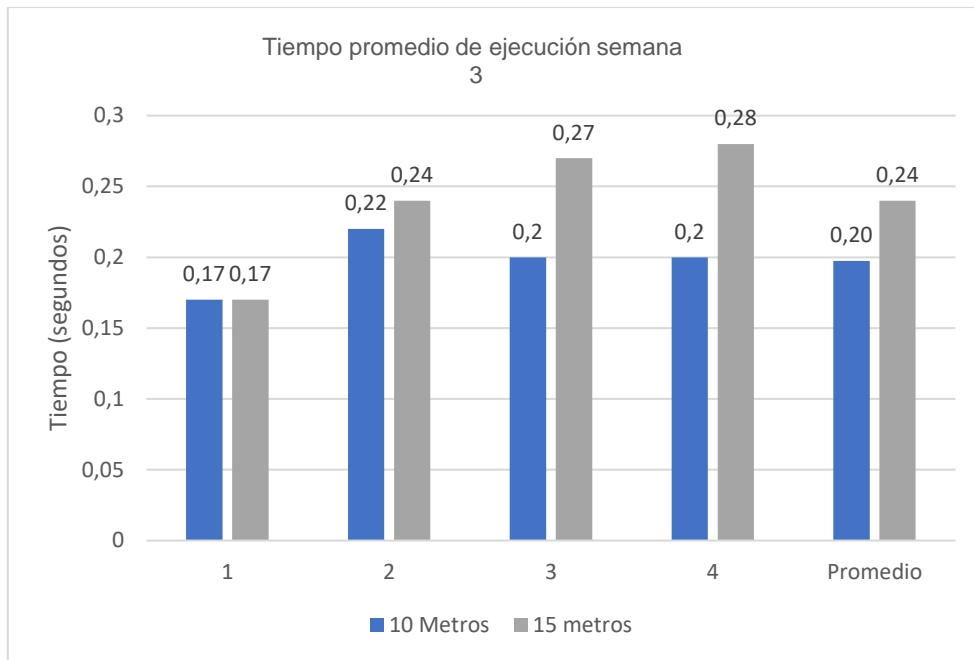


Fig. 21 Tiempo de Ejecución (T E) Semana 3

En la Tabla 10 y figura 22, se presenta los tiempos de ejecución (T E) promedio para el procesamiento de imágenes obtenidas con el dron del algoritmo ejecutado en la Jetson Nano para la detección de líneas en los cultivos de maíz de la semana 4.

Tabla 10 Tiempo de Ejecución (T E) Semana 4

Tiempo de Ejecución (T.E.) Semana 4			
Semana	Metros	Imagen	T E (segundos)
4	10	1	0.252
		2	0.258
		3	0.244
		4	0.220
	15	1	0.310
		2	0.319
		3	0.332
		4	0.333

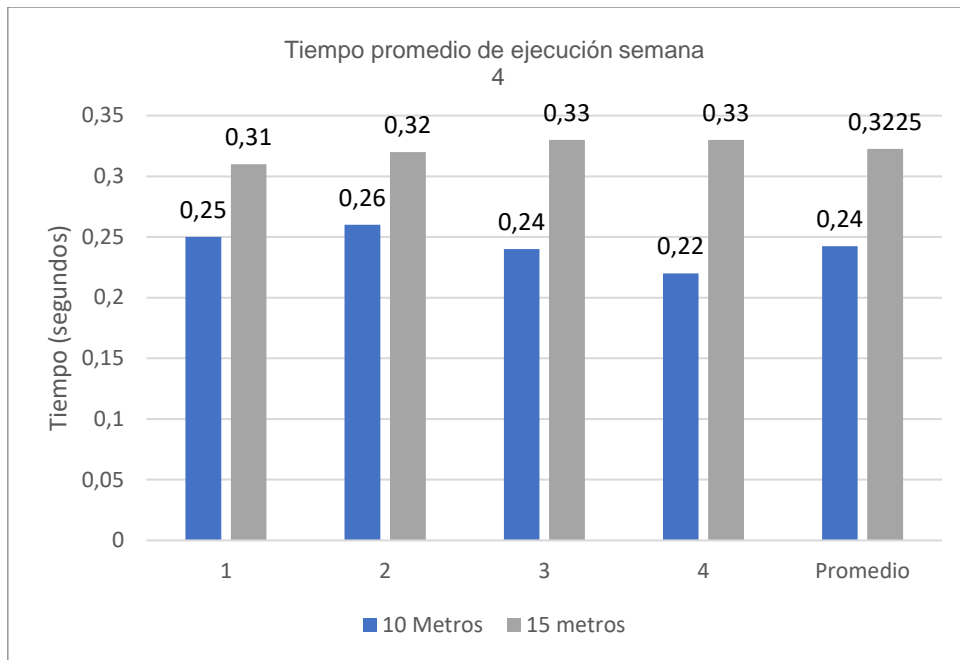


Fig. 22 Tiempo de Ejecución (T E) Semana 4

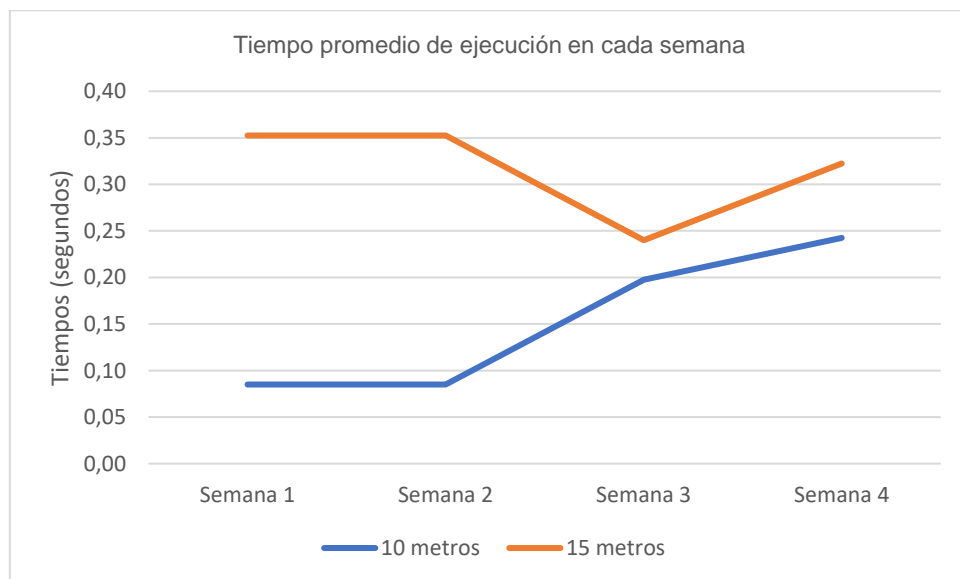


Fig. 23 gráfico de Tiempo promedio de ejecución semana 1-4

3.2.2 Consumo de memoria

En la Tabla 11, figuras 24 y 25, se presenta el consumo de Memoria RAM (C.M.) promedio para el procesamiento de imágenes obtenidas con el dron del algoritmo ejecutado en la Jetson Nano para la detección de líneas en los cultivos de maíz de la semana 1 a la 4 en imágenes de 10 y 15 metros de altura.

Tabla 11 Consumo Memoria RAM (C.M.) semana 1-4

Consumo Memoria RAM (C.M.) semana 1-4			
Semanas	Metros	Imagen	(C.M.) %
1-4	10	1	0.9
		2	
		3	
		4	
1-4	15	1	0.9
		2	
		3	
		4	

```

jetson-nano@jetsonnano-desktop: ~
1 [ | | 1.3%] Tasks: 164, 368 thr; 1 running
2 [ | | 0.6%] Load average: 0.25 0.41 0.44
3 [ | | 1.3%] Uptime: 00:30:58
4 [ | | 3.2%]
Mem[|||||||||||||1.75G/3.87G]
Swp[| 0K/1.93G]

PID USER PRI NI VIRT RES SHR S CPU% MEM% TIME+ Command
10606 jetson-na 20 0 10428 3744 2880 R 2.6 0.1 0:02.75 htop
10195 jetson-na 20 0 56176 36384 14112 S 1.3 0.9 0:06.40 /usr/bin/python3
    
```

Fig. 24 Consumo Memoria RAM (C.M.) semana 1-4

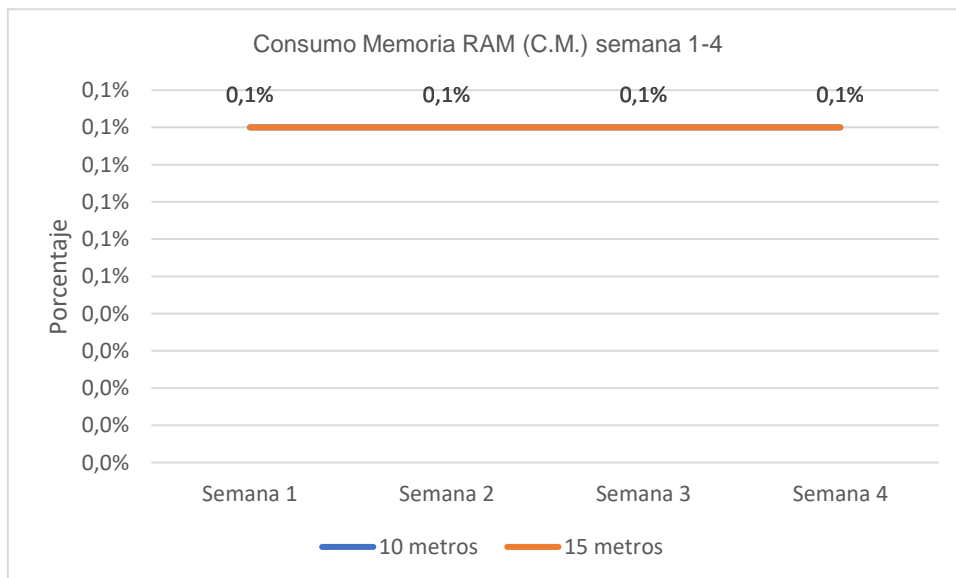


Fig. 25 gráfico consumo promedio de memoria RAM (C.M.) semana 1-4

3.2.3 Consumo CPU

En la Tabla 12, figuras 26 y 27, se presenta el consumo de CPU (C.CPU) % promedio para el procesamiento de imágenes obtenidas con el dron del algoritmo ejecutado en la Jetson Nano para la detección de líneas en los cultivos de maíz de la semana 1 en imágenes de 10 y 15 metros de altura.

Tabla 12 Consumo de CPU promedio semana 1

Consumo de CPU promedio semana 1		
Imagen	10 metros	15 metros
1	1,0%	1,3%
2	1,3%	1,2%
3	1,0%	1,0%
4	1,2%	1,3%
Promedio	1,1%	1,2%

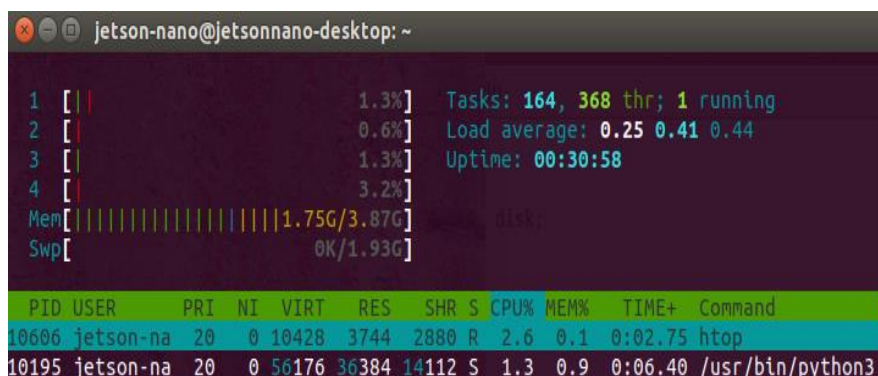


Fig. 26 Consumo de CPU promedio semana 1

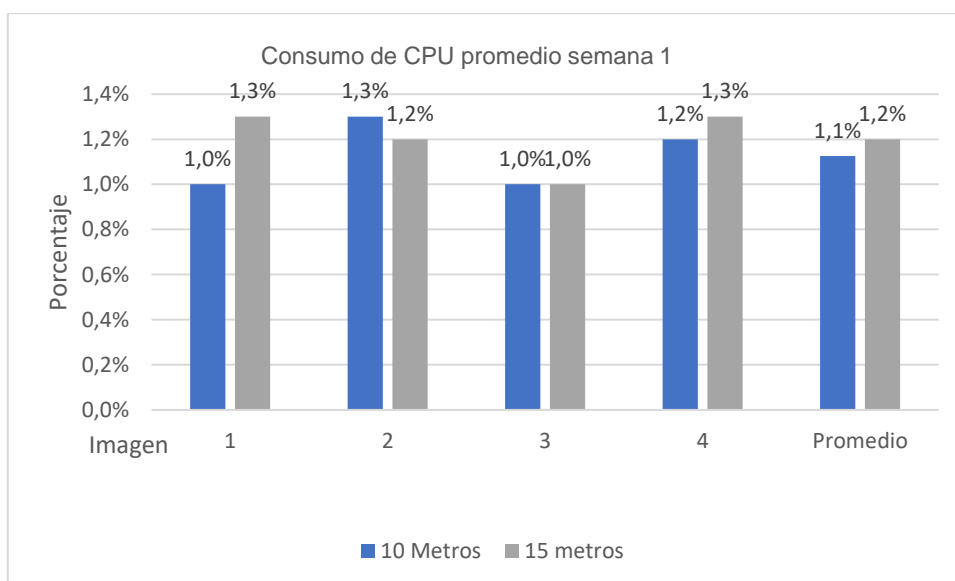


Fig. 27 gráfico de consumo de CPU promedio semana 1

En la Tabla13, figura 28 y 29, se presenta el consumo de CPU (C.CPU) % promedio para el procesamiento de imágenes obtenidas con el dron del algoritmo ejecutado en la Jetson Nano para la detección de líneas en los cultivos de maíz de la semana 2 en imágenes de 10 y 15 metros de altura.

Tabla 13 Consumo de CPU promedio semana 2

Consumo de CPU promedio semana 2		
Imagen	10 metros	15 metros
1	0,6%	0,4%
2	0,4%	0,6%
3	0,6%	0,4%
4	0,6%	0,5%
Promedio	0,6%	0,5%

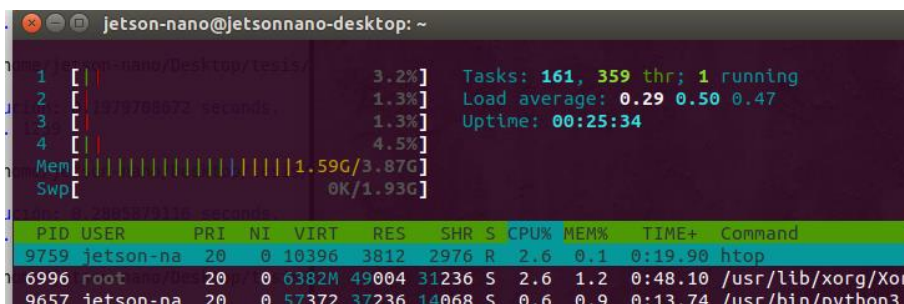


Fig. 28 Consumo de CPU promedio semana 2

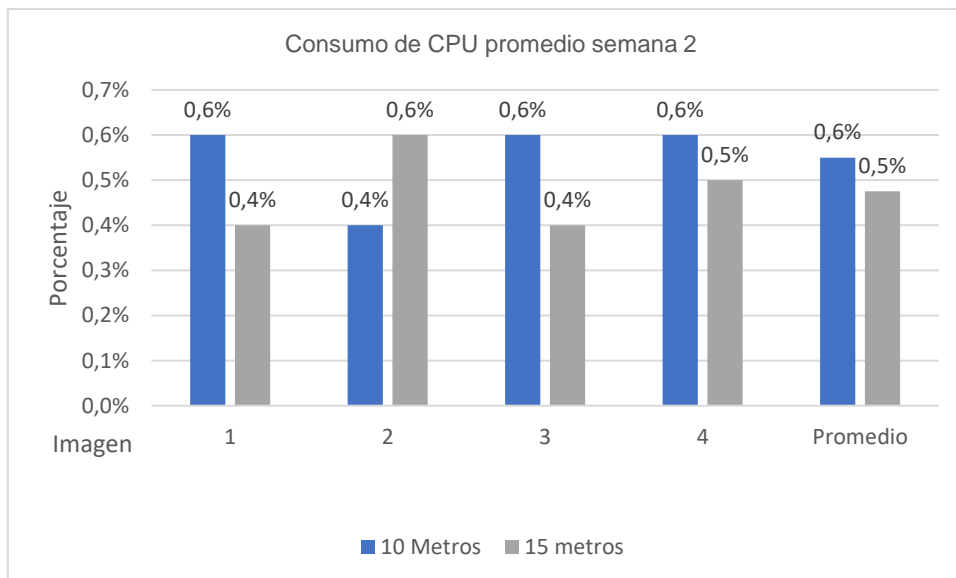


Fig. 29 gráfico de consumo de CPU promedio semana 2

En la Tabla 14, figura 30 y 31, se presenta el consumo de CPU (C.CPU) % promedio para el procesamiento de imágenes obtenidas con el dron del algoritmo ejecutado en la Jetson Nano para la detección de líneas en los cultivos de maíz de la semana 3 en imágenes de 10 y 15 metros de altura.

Tabla 14 Consumo de CPU promedio semana 3

Consumo de CPU promedio semana 3		
Imagen	10 metros	15 metros
1	0,15%	0,2%
2	0,2%	0,19%
3	0,13%	0,18%
4	0,13%	0,13%
Promedio	0,15%	0,2%

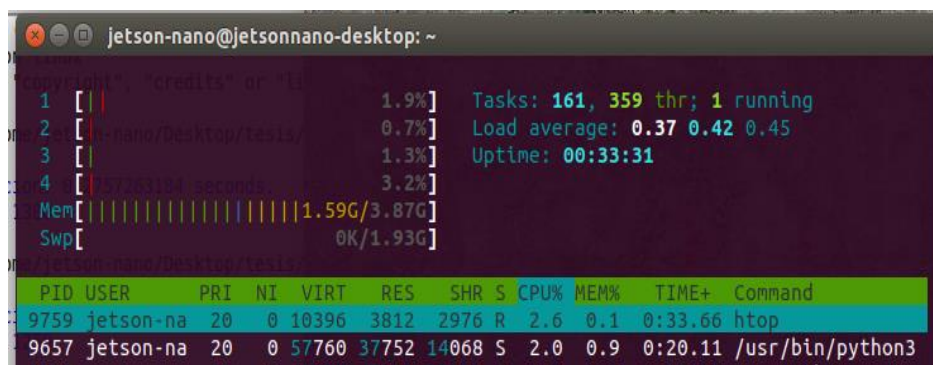


Fig. 30 Consumo de CPU promedio semana 3

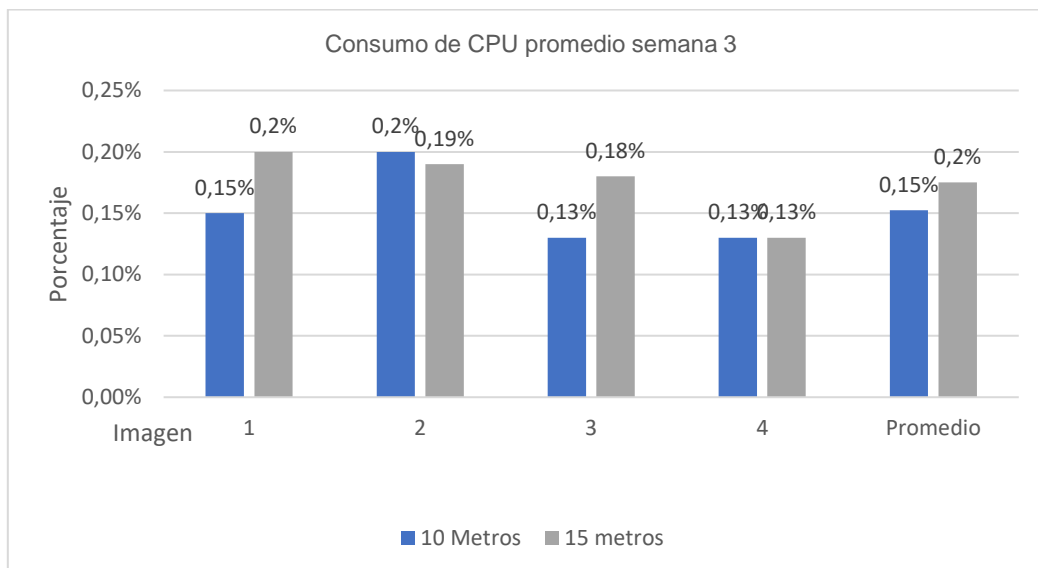


Fig. 31 gráfico de consumo de CPU promedio semana 3

En la Tabla 15, figura 32 y 33, se presenta el consumo de CPU (C.CPU) % promedio para el procesamiento de imágenes obtenidas con el dron del algoritmo ejecutado en la Jetson Nano para la detección de líneas en los cultivos de maíz de la semana 4 en imágenes de 10 y 15 metros de altura.

Tabla 15 Consumo de CPU promedio semana 4

Consumo de CPU promedio semana 4		
Imagen	10 metros	15 metros
1	0,13%	0,6%
2	0,9%	0,6%
3	0,13%	0,9%
4	0,80%	0,13%
Promedio	0,49%	0,6%

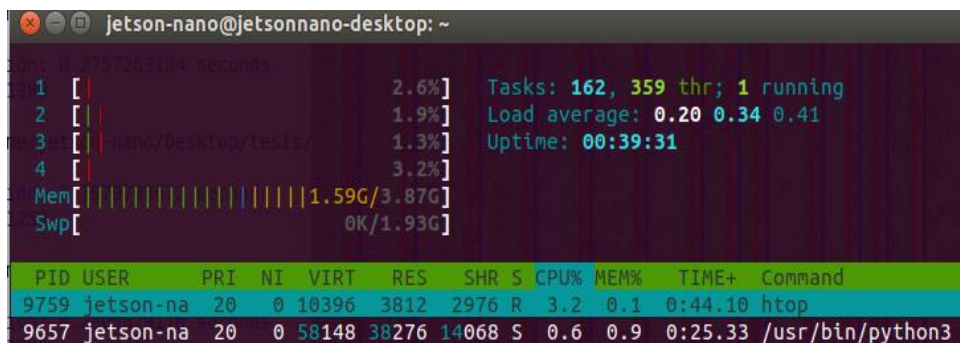


Fig. 32 Consumo de CPU promedio semana 4

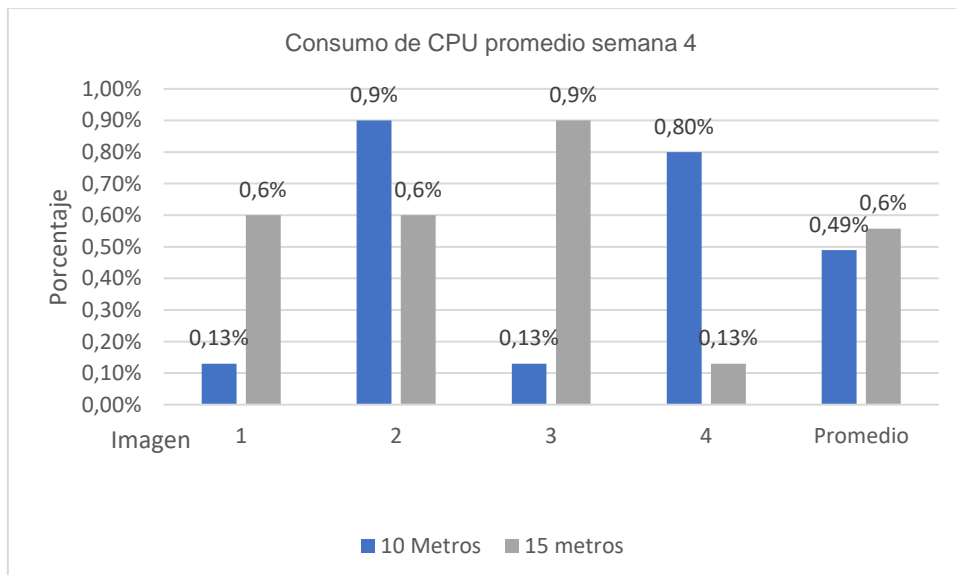


Fig. 33 gráfico de consumo de CPU promedio semana 4

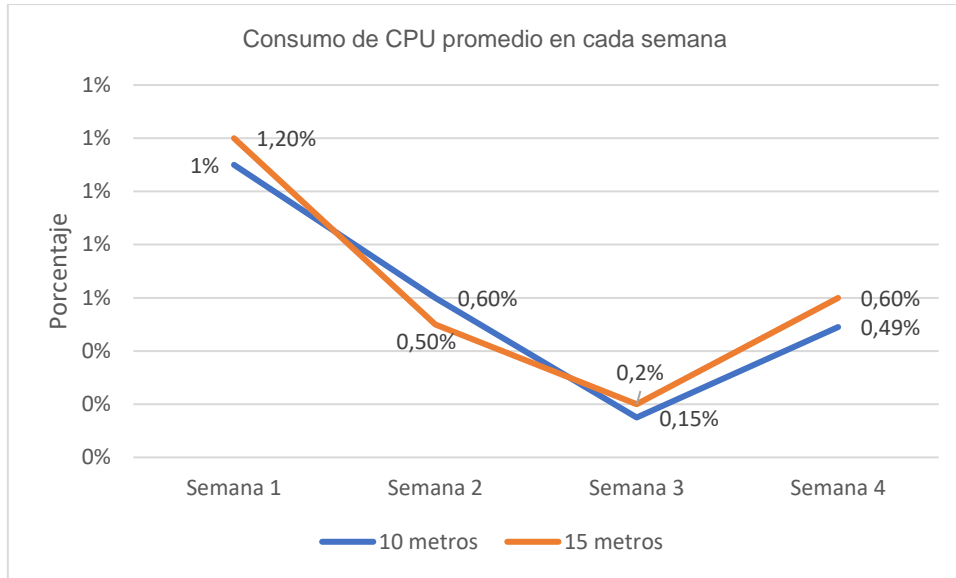


Fig. 34 gráfico de consumo de CPU promedio semana 1-4

3.3 Obtener resultado del prototipo

3.3.1 Conteo de líneas reales y conteo del algoritmo

En la Tabla 16 y figura 35, se presenta el conteo de número de las líneas identificadas reales (NLIR) y el conteo promedio de número de líneas identificadas por el algoritmo (NLIA) ejecutado en la Jetson Nano para el procesamiento de imágenes obtenidas con el dron para la detección de líneas en los cultivos de maíz en la semana 1.

Tabla 16 Conteo de número de las líneas identificadas reales y por el algoritmo semana 1

Conteo de líneas reales y conteo del algoritmo semana 1				
Semana	Metros	Imagen	NLIR	NLIA
1	10	1	13	12
		2	14	13
		3	13	12
		4	14	13
	15	1	18	15
		2	17	17
		3	18	15
		4	18	15

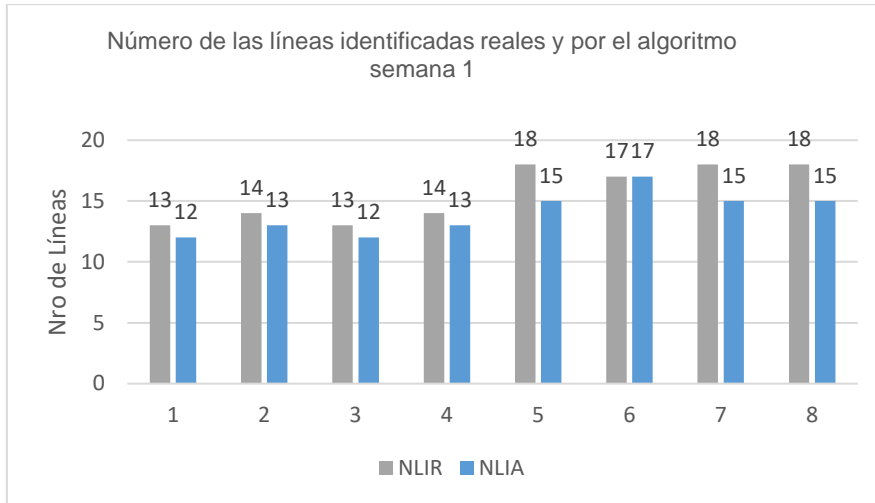


Fig. 35 Conteo de número de las líneas identificadas reales y por el algoritmo semana 1

En la Tabla 17 y figura 36, se presenta el conteo de número de las líneas identificadas reales (NLIR) y el conteo promedio de número de líneas identificadas por el algoritmo (NLIA) ejecutado en la Jetson Nano para el procesamiento de imágenes obtenidas con el dron para la detección de líneas en los cultivos de maíz en la semana 2.

Tabla 17 Conteo de número de las líneas identificadas reales y por el algoritmo semana 2

Conteo de líneas reales y conteo del algoritmo semana 2				
Semana	Metros	Imagen	NLIR	NLIA
2	10	1	13	10
		2	13	10
		3	13	10
		4	13	10
	15	1	18	18
		2	19	18
		3	18	18
		4	18	18

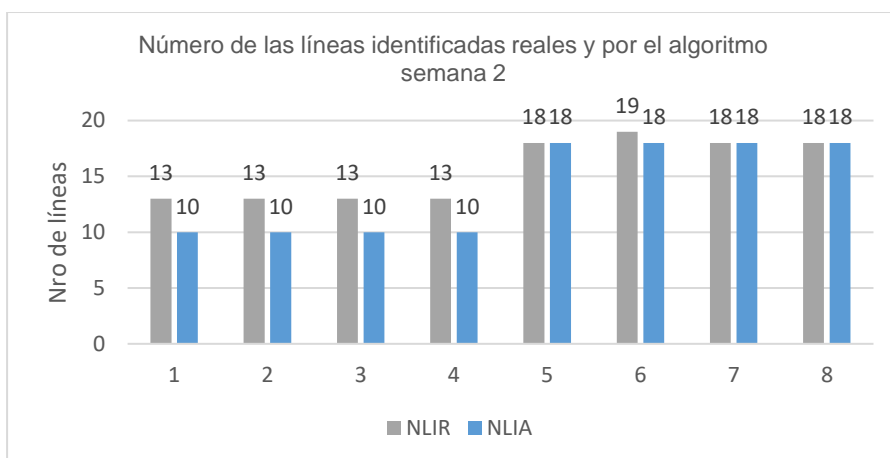


Fig. 36 Conteo de número de las líneas identificadas reales y por el algoritmo semana 2

En la Tabla 18 y figura 37, se presenta el conteo de número de las líneas identificadas reales (NLIR) y el conteo promedio de número de líneas identificadas por el algoritmo (NLIA) ejecutado en la Jetson Nano para el procesamiento de imágenes obtenidas con el dron para la detección de líneas en los cultivos de maíz en la semana 3.

Tabla 18 Conteo de número de las líneas identificadas reales y por el algoritmo semana 3

Conteo de líneas reales y conteo del algoritmo semana 3				
Semana	Metros	Imagen	NLIR	NLIA
3	10	1	14	12
		2	13	13
		3	13	11
		4	13	13
	15	1	20	17
		2	19	17
		3	19	17
		4	20	19

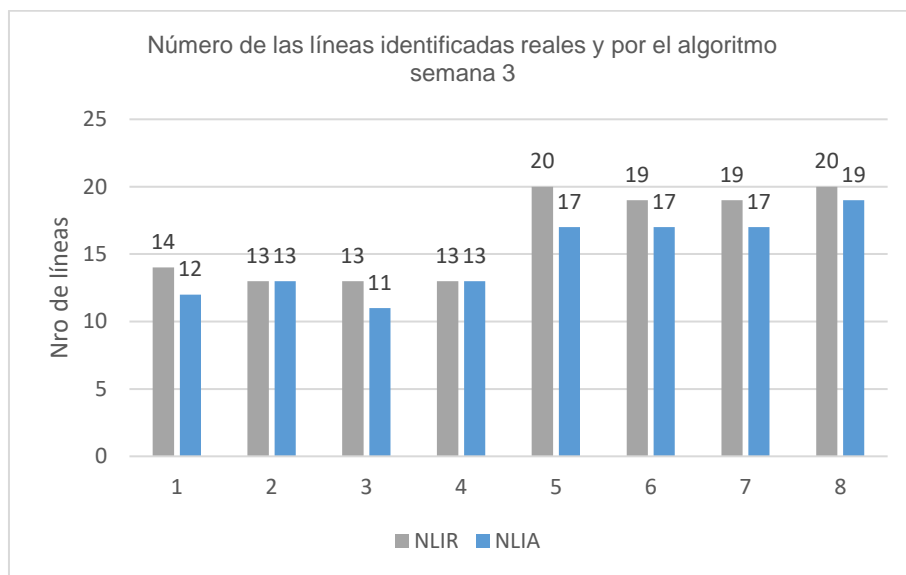


Fig. 37 Conteo de número de las líneas identificadas reales y por el algoritmo semana 3

En la Tabla 19 y figura 38, se presenta el conteo de número de las líneas identificadas reales (NLIR) y el conteo promedio de número de líneas identificadas por el algoritmo (NLIA) ejecutado en la Jetson Nano para el procesamiento de imágenes obtenidas con el dron para la detección de líneas en los cultivos de maíz en la semana 4.

Tabla 19 Conteo de número de las líneas identificadas reales y por el algoritmo semana 4

Conteo de líneas reales y conteo del algoritmo semana 4				
Semana	Metros	Imagen	NLIR	NLIA
4	10	1	15	13
		2	15	14
		3	15	14
		4	14	13
	15	1	18	18
		2	19	18
		3	19	18
		4	20	16

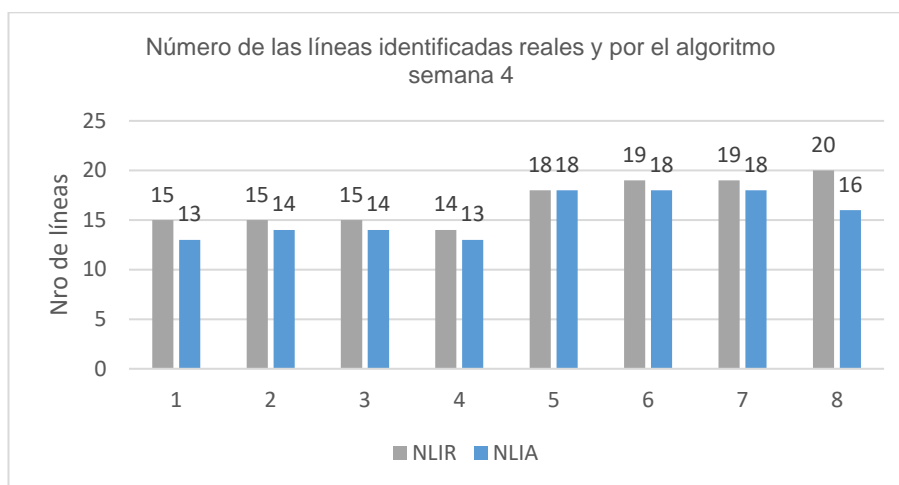


Fig. 38 Conteo de número de las líneas identificadas reales y por el algoritmo semana 4

3.4 Análisis de resultados

3.4.1 Porcentaje de reconocimiento

En la Tabla 20 y figura 39, se presenta el porcentaje % de reconocimiento de líneas en las alturas de 10 y 15 metros tomando como base el número de las líneas identificadas reales (NLIR) y promedio de número de líneas identificadas por el algoritmo (NLIA) ejecutado en la Jetson Nano para el procesamiento de imágenes obtenidas con el dron para la detección de líneas en los cultivos de maíz en la semana 1.

Tabla 20 Porcentaje % de reconocimiento de líneas Semana 1

Porcentaje % de reconocimiento de líneas Semana 1					
Semana	Metros	Imagen	NLIR	NLIA	%
1	10	1	13	12	92
		2	14	13	92
		3	13	12	92
		4	14	13	93
	15	1	18	15	83
		2	17	17	100
		3	18	15	83
		4	18	15	83

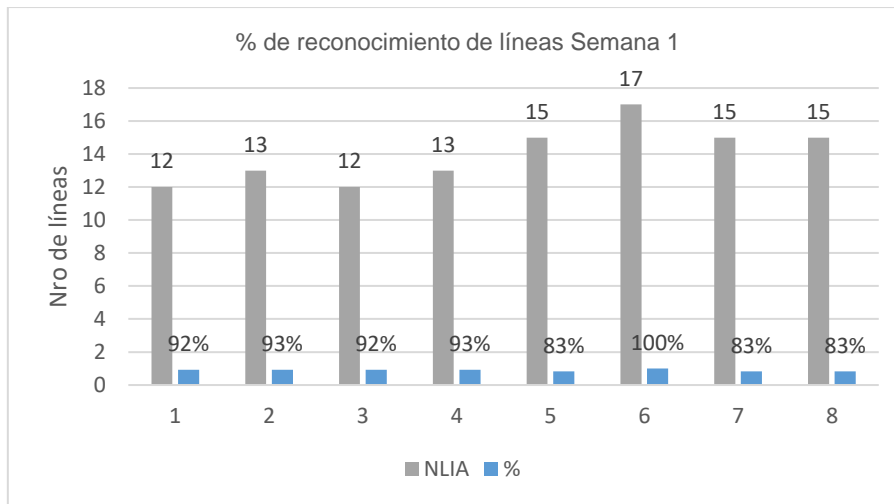


Fig. 39 Porcentaje % de reconocimiento de líneas Semana 1

En la Tabla 21 y figura 40, se presenta el porcentaje % de reconocimiento de líneas en las alturas de 10 y 15 metros tomando como base el número de las líneas identificadas reales (NLIR) y promedio de número de líneas identificadas por el algoritmo (NLIA) ejecutado en la Jetson Nano para el procesamiento de imágenes obtenidas con el dron para la detección de líneas en los cultivos de maíz en la semana 2.

Tabla 21 Porcentaje % de reconocimiento de líneas Semana 2

Porcentaje % de reconocimiento de líneas Semana 2					
Semana	Metros	Imagen	NLIR	NLIA	%
2	10	1	10	13	77
		2	10	13	77
		3	10	13	77
		4	10	13	77
	15	1	18	18	100
		2	18	19	95
		3	18	18	100
		4	18	18	100

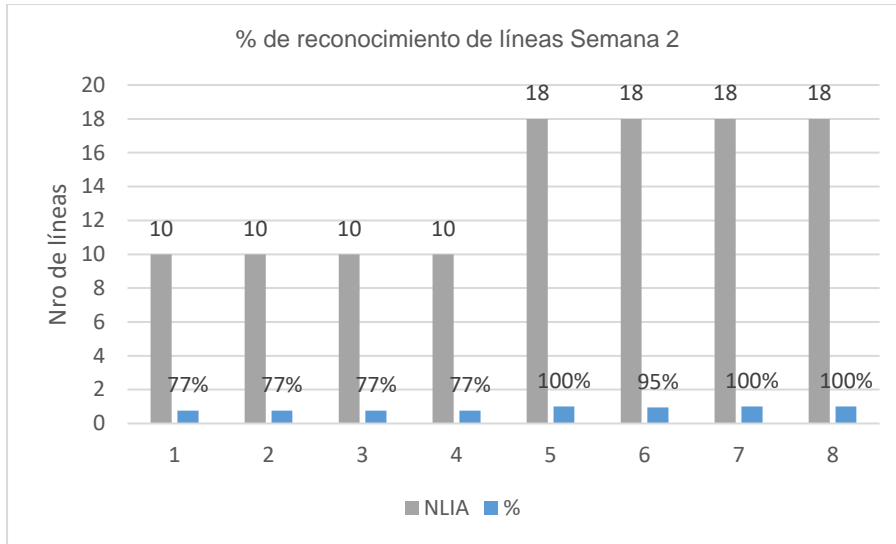


Fig. 40 Porcentaje % de reconocimiento de líneas Semana 2

En la Tabla 22 y figura 41, se presenta el porcentaje % de reconocimiento de líneas en las alturas de 10 y 15 metros tomando como base el número de las líneas identificadas reales (NLIR) y promedio de número de líneas identificadas por el algoritmo (NLIA) ejecutado en la Jetson Nano para el procesamiento de imágenes obtenidas con el dron para la detección de líneas en los cultivos de maíz en la semana 3.

Tabla 22 Porcentaje % de reconocimiento de líneas Semana 3

Porcentaje % de reconocimiento de líneas Semana 3					
Semana	Metros	Imagen	NLIR	NLIA	%
3	10	1	14	12	86
		2	13	13	100
		3	13	11	85
		4	13	13	100
	15	1	20	17	85
		2	19	17	89
		3	19	17	89
		4	20	19	95

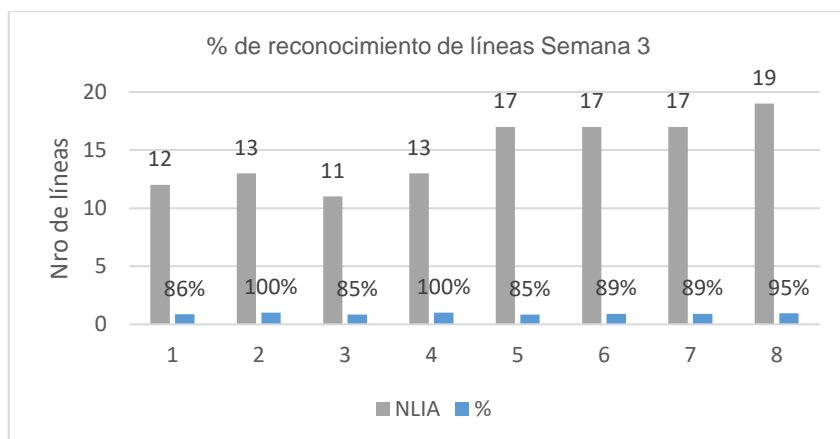


Fig. 41 Porcentaje % de reconocimiento de líneas Semana 3

En la Tabla 23 y figura 42, se presenta el porcentaje % de reconocimiento de líneas en las alturas de 10 y 15 metros tomando como base el número de las líneas identificadas reales (NLIR) y promedio de número de líneas identificadas por el algoritmo (NLIA) ejecutado en la Jetson Nano para el procesamiento de imágenes obtenidas con el dron para la detección de líneas en los cultivos de maíz en la semana 4.

Tabla 23 Porcentaje % de reconocimiento de líneas Semana 4

Porcentaje % de reconocimiento de líneas Semana 4					
Semana	Metros	Imagen	NLIR	NLIA	%
4	10	1	15	13	87
		2	15	14	93
		3	15	14	93
		4	14	13	93
	15	1	18	18	100
		2	19	18	95
		3	19	18	95
		4	20	16	80

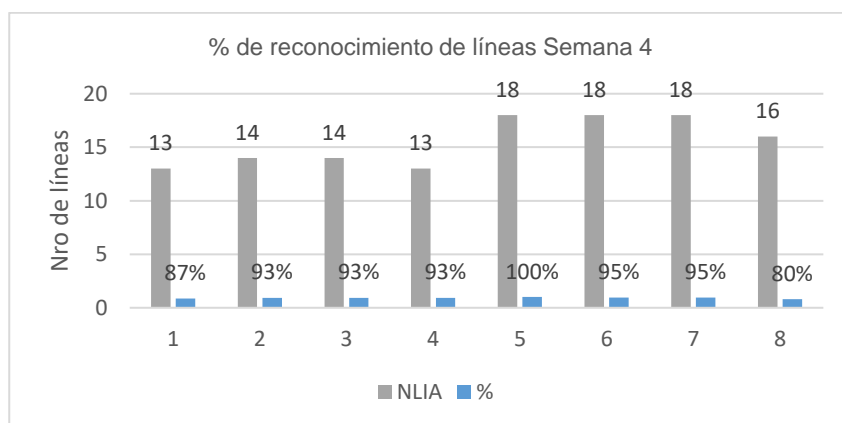


Fig. 42 Porcentaje % de reconocimiento de líneas Semana 4

3.4.2 Definir mejor reconocimiento de líneas según la semana y altura

En la Tabla 24 y figura 43, se presenta el mejor reconocimiento de líneas en las alturas de 10 y 15 metros basándose en el número de las líneas identificadas reales (NLIR) y promedio de número de líneas identificadas por el algoritmo (NLIA) ejecutado en la Jetson Nano para el procesamiento de imágenes obtenidas con el dron para la detección de líneas en los cultivos de maíz en la semana 1.

Tabla 24 Mejor reconocimiento de líneas según altura semana 1

Mejor reconocimiento de líneas según altura semana 1					
Semana	Metros	Imagen	NLIR	NLIA	%
1	10	1	13	12	93
		2	14	13	
		3	13	12	
		4	14	13	
	15	1	18	15	88
		2	17	17	
		3	18	15	
		4	18	15	

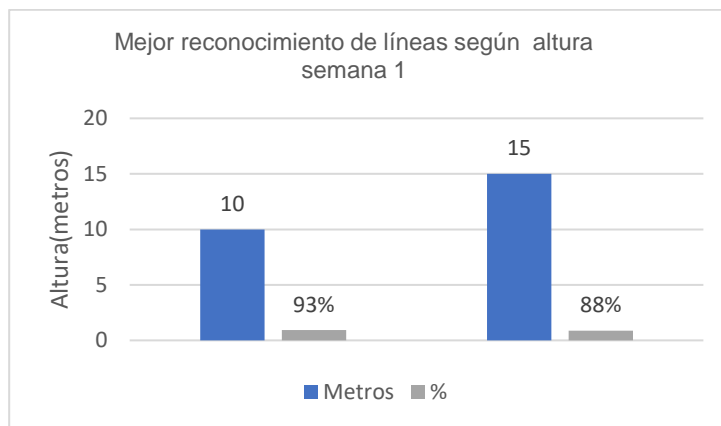


Fig. 43 Mejor reconocimiento de líneas según altura semana 1

En la Tabla 25 y figura 44, se presenta el mejor reconocimiento de líneas en la altura de 10 metros basándose en el número de las líneas identificadas reales (NLIR) y promedio de número de líneas identificadas por el algoritmo (NLIA) ejecutado en la Jetson Nano para el procesamiento de imágenes obtenidas con el dron para la detección de líneas en los cultivos de maíz en la semana 2.

Tabla 25 Mejor reconocimiento de líneas según altura semana 2

Mejor reconocimiento de líneas según altura semana 2					
Semana	Metros	Imagen	NLIR	NLIA	%
2	10	13	10	13	77
		13	10	13	
		13	10	13	
		13	10	13	
	15	18	18	18	99
		19	18	19	
		18	18	18	
		18	18	18	

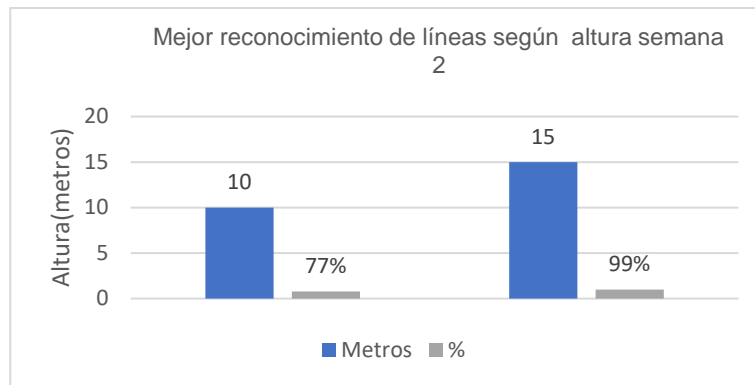


Fig. 44 Mejor reconocimiento de líneas según altura semana 2

En la Tabla 26 y figura 45, se presenta el mejor reconocimiento de líneas en las alturas de 10 y 15 metros basándose en el número de las líneas identificadas reales (NLIR) y promedio de número de líneas identificadas por el algoritmo (NLIA) ejecutado en la Jetson Nano para el procesamiento de imágenes obtenidas con el dron para la detección de líneas en los cultivos de maíz en la semana 3.

Tabla 26 Mejor reconocimiento de líneas según altura semana 3

Mejor reconocimiento de líneas según altura semana 3					
Semana	Metros	Imagen	NLIR	NLIA	%
3	10	1	14	12	93
		2	13	13	
		3	13	11	
		4	13	13	
	15	1	20	17	90
		2	19	17	
		3	19	17	
		4	20	19	

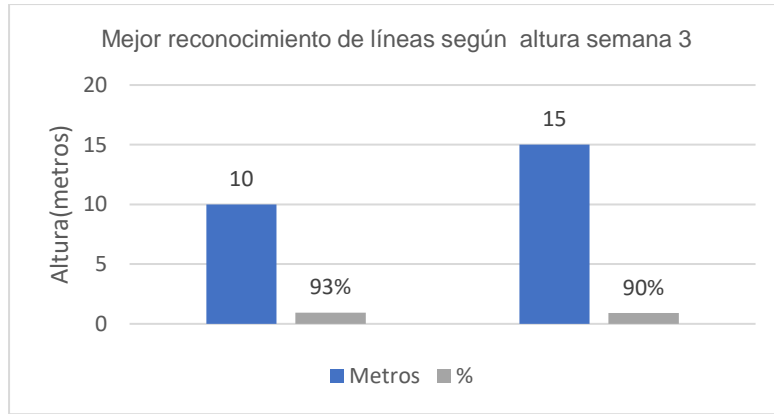


Fig. 45 Mejor reconocimiento de líneas según altura semana 3

En la Tabla 27 y figura 46, se presenta el mejor reconocimiento de líneas en las alturas de 10 y 15 metros basándose en el número de las líneas identificadas reales (NLIR) y promedio de número de líneas identificadas por el algoritmo (NLIA) ejecutado en la Jetson Nano para el procesamiento de imágenes obtenidas con el dron para la detección de líneas en los cultivos de maíz en la semana 4.

Tabla 27 Mejor reconocimiento de líneas según altura semana 4

Mejor reconocimiento de líneas según altura semana 4					
Semana	Metros	Imagen	NLIR	NLIA	%
4	10	1	15	13	92
		2	15	14	
		3	15	14	
		4	14	13	
	15	1	18	18	92
		2	19	18	
		3	19	18	
		4	20	16	

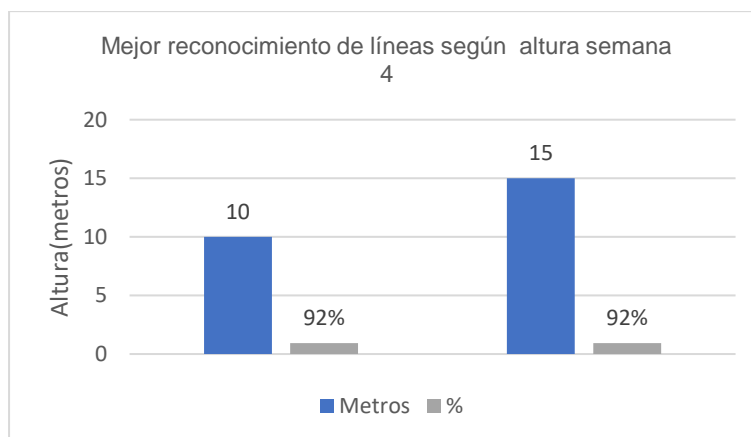


Fig. 46 Mejor reconocimiento de líneas según altura semana 4

En la Tabla 28 y figura 47, se presenta el promedio del mejor reconocimiento de líneas en las alturas de 10 y 15 metros ejecutado en la Jetson Nano para el procesamiento de imágenes obtenidas con el dron para la detección de líneas en los cultivos de maíz entre la semana 1 a la 4.

Tabla 28 Promedio del mejor reconocimiento de líneas según la semana 1-4 y altura

Promedio del mejor reconocimiento de líneas según la semana 1-4 y altura		
	10 metros	15 metros
Semana 1	93%	88%
Semana 2	77%	99%
Semana 3	93%	90%
Semana 4	92%	92%

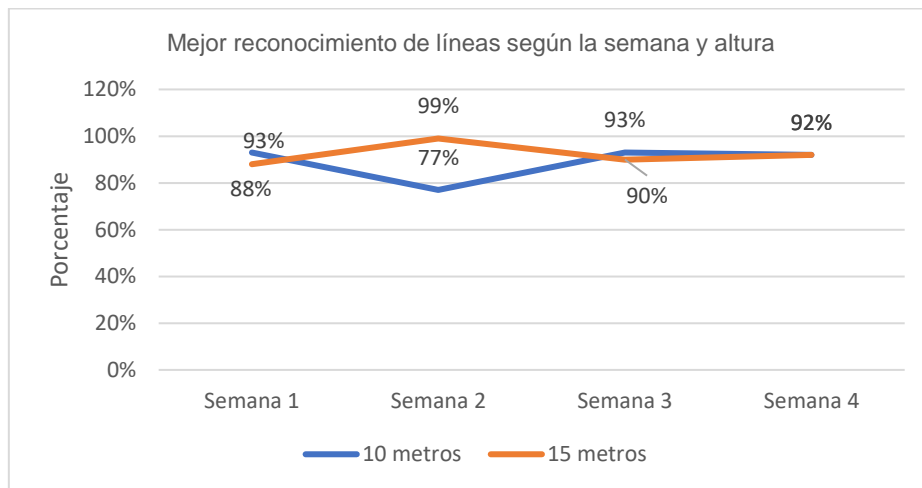


Fig. 47 Promedio del mejor reconocimiento de líneas según la semana 1-4 y altura

3.5 Análisis de impactos

En el presente trabajo de titulación se lograron ciertos resultados, los cuales se describen a continuación.:

Impacto Científico

Las plataformas Raspberry Pi 4 y Jetson Nano permiten la implementación de aplicaciones para agricultura de precisión en la sistematización de tareas con fines aplicativos e investigativos, aportando con el desarrollo científico local, regional e internacional enfatizando el potencial del hardware y software libre para soluciones efectivas y económicas.

Impacto Tecnológico

El algoritmo desarrollado brinda al sector agrícola una herramienta que ayuda con la detección de líneas en cultivos de maíz utilizando tecnología aplicadas a la agricultura de precisión. Todo el algoritmo se desarrolló en el hardware de la Jetson Nano con herramientas como el editor Spyder (Python) de desarrollo integrado y multiplataforma de código abierto (IDE), y librerías como OpenCV, Numpy, Skimage.morphology import dilation, Copy que permiten la identificación de objetos, búsqueda de imágenes similares y reconocimiento de escenarios, dilatar, erosionar el tamaño y la forma de las mismas, además de crear copias de distintos objetos de Python; La aplicación del algoritmo aporta beneficios para los agricultores mejorando la rentabilidad de los cultivos, además de reducir el uso de fertilizantes, y desperdicios siendo base esencial para transformar el trabajo agrícola generando un impacto positivo.

Impacto Económico

El prototipo se desarrolló basándose en un sistema integrado que utiliza hardware y software gratuitos, así como electrónica de bajo costo y que consume mucha energía conectada con diferentes pasos de procesamiento para monitorear continuamente las señales de contaminación del suelo. Esto reducirá el costo de implementación.

Conclusiones

De la revisión bibliográfica de las bases de datos bibliográficas como IEEEExplore y Scopus que tiene acceso la UTN sobre Jetson Nano / TX2 y Raspberry Pi 4, se concluye que las dos arquitecturas son de bajo costo, se adecuan fácilmente en la sistematización de actividades agrícolas, aportando beneficios en el sector agrícola para mejorar la rentabilidad de los cultivos, además reducir el uso de fertilizantes, costos de producción y desperdicios, minimizando impactos negativos en el planeta.

La aplicación de las buenas prácticas de programación y la metodología en cascada en el desarrollo del algoritmo para detectar líneas en los cultivos son metodologías que proporcionan una estructura clara y ordenada para el desarrollo de algoritmos computacionales aplicados a la agricultura de precisión, lo que contribuye a los equipos de desarrollo a trabajar de manera más eficiente, realizando un correcto análisis de requerimientos, un mejor diseño en cuanto al procesamiento y clasificación de las imágenes, una adecuada implementación y verificación del funcionamiento del proyecto así como también a producir código ordenado y bien estructurado con fines de escalamiento para soporte y mantenimiento.

Los vehículos aéreos no tripulados (drones) son un complemento para la toma de imágenes de alta calidad. Las imágenes adquiridas con drones son una herramienta valiosa para la toma de decisiones agrícolas y soluciona problemas en una amplia gama de actividades, ya que proporcionan información detallada que facilita el procesamiento con algoritmos computacionales.

Las métricas definidas en la toma de imágenes con el dron permitieron realizar una mejor evaluación del algoritmo ejecutado en la Jetson Nano. Dando como resultado la correcta verificación en la detección de líneas en los cultivos en las alturas de 10 y 15 metros en las semanas 3 y 4 que sobre pasa el 90 % como se muestra en la tabla 28.

Los resultados obtenidos con la implementación del algoritmo ejecutado en la arquitectura embebida Jetson Nano demuestran que se reduce el tiempo de ejecución en la localización y un mayor porcentaje de reconocimiento de líneas en los cultivos de maíz superando el 85% de precisión.

Recomendaciones

Realizar más estudios sobre este tipo arquitecturas de bajo costo aplicadas en la agricultura de precisión para que se pueda incentivar a la implementación de estas y así beneficiar a los agricultores de nuestro país, mejorando la rentabilidad de los cultivos.

Dar continuidad y mejora de los algoritmos en el procesamiento de imágenes para la detección de líneas en los cultivos de maíz, y así llegar a un mayor porcentaje de reconocimiento de malezas.

Para la toma de imágenes en los cultivos con drones tomar en cuenta la correcta altura a la hora de encender el equipo ya que algunos toman la altura desde su despeje.

Seguir utilizando la arquitectura Jetson Nano para futuras implementaciones. Esta arquitectura embebida está diseñada específicamente para aplicaciones de inteligencia artificial y visión por computadora, Dado que en esta investigación se ha obtenido resultados positivos en términos de reducción del tiempo de ejecución y un mayor porcentaje de reconocimiento de líneas en los cultivos de maíz.

Bibliografía

- Abad, J. A., & Farez, J. P. (2018). *Diseño e implementación de un sistema de monitoreo de variables climáticas que afectan al cultivo de café, en la plantación ASOPROCCSI ubicado en santa Isabel.*
- Abouzahir, S., Sadik, M., & Sabir, E. (2019). Enhanced approach for weeds species detection using machine vision. *2018 International Conference on Electronics, Control, Optimization and Computer Science, ICECOCS 2018.*
<https://doi.org/10.1109/ICECOCS.2018.8610505>
- AERLYPER. (2022). *Funcionamiento de un dron: lo que debes saber.*
<https://aerlyper.es/funcionamiento-de-un-dron-lo-que-debes-saber/>
- AgroSpray. (2021). *Importancia de la Fertilización Mineral del Suelo y su Impacto en la Siembra -.* <https://agrospray.com.ar/blog/fertilizacion-mineral/>
- Agud, L., & Pla, M. L. (2020). *Problemas de analisis de una variable y algebra lineal: soluciones analíticas y con Matlab.* Editorial de la Universidad Politecnica de Valencia.
<https://elibro.net/es/lc/utnorte/titulos/127797>
- Algar Diaz, M. J., & Fernandez de Sevilla Vellon, M. (2019). *Introduccion practica a la programacion con Python.* Editorial Universidad de Alcala.
<https://elibro.net/es/lc/utnorte/titulos/124259>
- Alva, J. L., & Alcorta, N. F. (2020). *Sistemas embebidos guía metodológica para su desarrollo.*
- Alvarez, L. (2021). *REPOSITORIO NACIONAL EN CIENCIAS Y TECNOLOGÍA FICHA DE REGISTRO DE TESIS.*
- Barahona, S. (2019). *Navegación autónoma basada en maniobras bajo estimación de posturas humanas para un robot omnidireccional kuka youbot.*
- Barceló, P. (2019). *Universidad de Sancti Spiritus “José Martí Pérez.”*
- Barkstrom, J. (2019). *Introducción a la Raspberry Pi Proyectos de computación de placa única.*
- Barreto, S., Chen He, J., & Niño, M. J. (2022). *Implementación de una Librería de Deep Learning en Lenguaje C para Plataformas Embebidas de Bajos Recursos Computacionales.*
- Benítez, L. ´ O. J., Jos´, J., Luis, J., & Pagoaga, H. (2019). *Introducción a MATLAB.*
- Bollatti, P., & Juárez, M. (2021). *Introducción al procesamiento de imágenes RGB tomadas por drones para monitoreo del estado de cultivos.*
- Bongiovanni, R., Mantovani, E. C., Best, S., & ROEL, A. (Ed.). (2016). *Agricultura de precisión integrando conocimientos para una agricultura moderna y sustentable.* Procisur/IICA.
- Bwambale, E., Abagale, F. K., & Anornu, G. K. (2022). Smart irrigation monitoring and control strategies for improving water use efficiency in precision agriculture: A review. *Agricultural Water Management, 260*, 107324.
<https://doi.org/https://doi.org/10.1016/j.agwat.2021.107324>

- Cabrera, J. A., Anrango, M. S., Yandú, M., & Lascano, S. (2019). *Dialnet-DeteccionDeEnfermedadesEnCultivosDePapaUsandoProce-8228807*.
- Calvo, L. (2022). *¿Qué es una Raspberry Pi y para qué sirve? - Blog*. <https://es.godaddy.com/blog/que-es-raspberry-pi/>
- Chávez, M. A. (2018). *Mejoramiento de la productividad mediante la implementación de drones en el cultivo de flores de verano y rosas en el grupo esmeralda ecuador*.
- Chora, D., Álvarez, G., & Espinoza, M. (2018). Raspberry Pi y Arduino: semilleros en innovación tecnológica para la agricultura de precisión. *Informática y Sistemas: Revista de Tecnologías de La Informática y Las Comunicaciones*, 2(1). <https://doi.org/10.33936/isrtic.v2i1.1134>
- Cisternas, I., Velásquez, I., Caro, A., & Rodríguez, A. (2020). Systematic literature review of implementations of precision agriculture. *Computers and Electronics in Agriculture*, 176, 105626. <https://doi.org/https://doi.org/10.1016/j.compag.2020.105626>
- Clare, L., & Timothy, M. (2017). *Monitoreo de cultivos para la detección temprana de plagas de insectos | ECHOcommunity.org*. <https://www.echocommunity.org/es/resources/78ba129d-56a3-43b6-abd9-dc963495f235>
- Covantec, R. L. (2019). *Introducción al lenguaje Python — Materiales del entrenamiento de programación en Python - Nivel básico*. <https://entrenamiento-python-basico.readthedocs.io/es/latest/leccion1/index.html>
- CropLife. (2021). *Uso de drones en la agricultura - CropLife Latin America*. <https://www.croplifela.org/es/actualidad/articulos/uso-de-drones-en-la-agricultura>
- Delgadillo, R. (2019). *Desarrollo de una Red de Sensores para el Monitoreo en Ambiente Web de Parámetros Físico-Químicos en Invernaderos de Plantas Ornamentales*.
- di Cicco, M., Potena, C., Grisetti, G., & Pretto, A. (2017). Automatic model based dataset generation for fast and accurate crop and weeds detection. *IEEE International Conference on Intelligent Robots and Systems, 2017-September*, 5188–5195. <https://doi.org/10.1109/IROS.2017.8206408>
- Díaz, I. K., Fierro, E., & Muñoz Pentón, M. (2018). *La enseñanza de la programación. Una experiencia en la formación de profesores de Informática*. <https://doi.org/10.18800/educacion.201802.005>
- Digital Guide. (2021). *Raspberry Pi: ideas y usos para 2022 - IONOS*. <https://www.ionos.es/digitalguide/servidores/know-how/un-vistazo-a-proyectos-basados-en-raspberry-pi/>
- Digital Guide IONOS. (2022, July 4). *30 proyectos con Raspberry Pi que explotan todo su potencial*. Digital Guide IONOS
- EOS. (2021). *Fertilidad Del Suelo: Cómo Mantenerla Y Recuperar Su Pérdida*. <https://eos.com/es/blog/fertilidad-del-suelo/>
- EOS DATA ANALYTICS. (2020, October 26). *Monitoreo Satelital De Cultivos Y Sus Condiciones*. <https://eos.com/es/blog/monitoreo-satelital-de-cultivos/>
- Escalante, D., & Vargas, D. (2019). *Raspberry pi: la tecnología reducida en placa*. www.raspberrypi.org,

- FAO. (2020, January 14). *Iniciativa pionera en Ecuador utiliza drones para identificar problemas y buscar soluciones para la producción eficiente de algodón*. Organización de Las Naciones Unidas Para La Alimentación y Agricultura.
- Farooq, A., Hu, J., & Jia, X. (2018). Weed classification in hyperspectral remote sensing images via deep convolutional neural network. *International Geoscience and Remote Sensing Symposium (IGARSS), 2018-July*, 3816–3819. <https://doi.org/10.1109/IGARSS.2018.8518541>
- García, R. (2022). *PROGRAMACIÓN CON GARCÍA*. <https://garciaprogramaciondirigidaaobjetos.blogspot.com/2022/08/la-programaciongarciaorientada-objetos.html>
- Goñi, O., Noguera, J., Leiva, L., & Tosini, M. (2018). *Sistema de detección malezas y cultivos*. <http://www.labset.exa.unicen.edu.ar>
- González, M. (2022). *¿Qué es un dron y cómo funciona?* <https://filmora.wondershare.es/drones/what-is-drone-how-does-it-work.html>
- Gutiérrez, I. A. (2022). *Universidad Nacional de Loja Facultad de la Energía, las Industrias y los Recursos Naturales No Renovables Carrera de Ingeniería Electromecánica*.
- Hero C. (2021). *entorno de desarrollo.docx - ENT Programacion en C/ C ++*.
- Herrera, P., Martínez, J., Rodríguez, A., & Cid, L. (2017). Efecto de dos sistemas de labranza sobre la infiltración en suelos Ferralíticos Rojos. *Revista Ingeniería Agrícola*, 7(4), 3–10.
- HispaDrones. (2019). *Beneficios del uso de drones en la agricultura » HispaDrones*. <https://www.hispadrones.com/principiantes/aprendizaje-consejos/beneficios-usar-drones-agricultura/>
- Huertos, A. (2019, June 28). *Lo que debes saber sobre la Raspberry Pi 4 antes de lanzarte a comprar una*. <https://computerhoy.com/listas/tecnologia/debes-saber-raspberry-pi-4-antes-lanzarte-comprar-446889>
- Igarzábal, D. (2016). *Los beneficios del monitoreo de cultivos por Daniel Igarzábal | Mundo Agro Cba | Noticias del Agro*. <https://mundoagrocba.com.ar/los-beneficios-del-monitoreo-de-cultivos-por-daniel-igarzabal/>
- Jiménez, A., Camargo, D., & García Ramírez, D. Y. (2020). Sistema Inteligente para el manejo de Malezas en el cultivo de Piña con Conceptos de Agricultura de Precisión. *Ciencia y Agricultura*, 17(3), 122–136. <https://doi.org/10.19053/01228420.v17.n3.2020.10830>
- Jiménez, A. F., Camargo, D. A., & García, D. Y. (2020). Sistema Inteligente para el manejo de Malezas en el cultivo de Piña con Conceptos de Agricultura de Precisión. *Ciencia y Agricultura*, 17(3), 122–136. <https://doi.org/10.19053/01228420.v17.n3.2020.10830>
- Kong, Q., Siau, T., & Bayen, A. M. (2021). Chapter 10 - Errors, Good Programming Practices, and Debugging. In Q. Kong, T. Siau, & A. M. Bayen (Eds.), *Python Programming and Numerical Methods* (pp. 157–173). Academic Press. <https://doi.org/https://doi.org/10.1016/B978-0-12-819549-9.00019-1>
- LAB Linux. (2019, April 21). *Nvidia Jetson Nano: una computadora para la implementación de aplicaciones AI*. <https://laboratoriolinux.es/index.php/-noticias-mundo-linux-ultimas->

noticias/23414-nvidia-jetson-nano-una-computadora-para-la-implementacion-de-aplicaciones-ai.html

- Ledesma, N. R., Reuter, F., & Pedenovi, A. (2019). *Serie didáctica n° 43 facultad de ciencias forestales los drones y sus aplicaciones a la ingeniería*.
- López, A. (2021). *Levantamiento Fotogramétrico con Dron del Predio Jorge Yáñez Castro*.
- López, S., Rodríguez, G., Martínez, C., Herrera, P., & Hernández, L. (2018). Régimen hídrico en un suelo Ferralítico cultivado con maíz bajo principios de agricultura de conservación. *Revista Ingeniería Agrícola*, 8(3), 3–11.
- Lottes, P., Behley, J., Chebrolu, N., Milioto, A., & Stachniss, C. (2018). Joint Stem Detection and Crop-Weed Classification for Plant-Specific Treatment in Precision Farming. *IEEE International Conference on Intelligent Robots and Systems*, 8233–8238. <https://doi.org/10.1109/IROS.2018.8593678>
- Luchetti, S. (2021, June 1). *Sistemas embebidos y sus características*.
- Ma, Y., Huang, Y., Wu, G., Liu, J., Liu, Y., Xiang, Y., Liu, Y., & Tang, Z. (2022). Decentralized monthly generation scheduling of cascade hydropower plants in multiple time scale markets. *International Journal of Electrical Power & Energy Systems*, 135, 107420. <https://doi.org/https://doi.org/10.1016/j.ijepes.2021.107420>
- Machuca, E. J. (2018). *Raspberry Pi y sus Aplicaciones*. <http://www.uca.edu.py>
- Manlove, J. L., Shew, A. M., & Obembe, O. S. (2021). Arkansas producers value upload speed more than download speed for precision agriculture applications. *Computers and Electronics in Agriculture*, 190, 106432. <https://doi.org/https://doi.org/10.1016/j.compag.2021.106432>
- Martin, C., Urquia, A., & Rubio, M. A. (2021). *Lenguajes de programacion*. UNED - Universidad Nacional de Educacion a Distancia. <https://elibro.net/es/lc/utnorte/titulos/184827>
- Martin, R. C. (2012). *Codigo-Limpio - Español*. 1–453.
- Marzal Varo, A., Garcia Sevilla, P., & Gracia Luengo, I. (2016). *Introduccion a la programacion con Python 3*. D - Universitat Jaume I. Servei de Comunicacio i Publicacions. <https://elibro.net/es/lc/utnorte/titulos/51760>
- MathWorks. (2023). *Tipos de Imagenes*. https://la.mathworks.com/help/matlab/creating_plots/image-types.html
- MCI electronics. (2022). *Raspberry Pi*. <https://raspberrypi.cl/raspberry/>
- Medici, M., Pedersen, S. M., Canavari, M., Anken, T., Stamatelopoulos, P., Tsiropoulos, Z., Zotos, A., & Tohidloo, G. (2021). A web-tool for calculating the economic performance of precision agriculture technology. *Computers and Electronics in Agriculture*, 181, 105930. <https://doi.org/https://doi.org/10.1016/j.compag.2020.105930>
- Medrano, E. (2021). *INTEGRACIÓN DEL PROTOCOLO DE TRANSPORTE MÓVIL EN UN SISTEMA INTELIGENTE DE VIGILANCIA DEL HOGAR*.
- Meneses, V. A. B., Téllez, J. M., & Velasquez, D. F. A. (2015). Uso de drones para el análisis de imágenes multispectrales en agricultura de precisión. *@limentech, Ciencia y Tecnología Alimentaria*, 13(1). <https://doi.org/10.24054/16927125.v1.n1.2015.1647>

- Meza, M., Paz, C., & Pereira, K. (2017). *Indicadores para el monitoreo de la calidad del suelo en áreas periurbanas. valle de Quillota, cuenca del Aconcagua, Chile*.
<https://www.redalyc.org/journal/339/33952871003/>
- Mochales, M. (2020). *Python, el lenguaje de programación más popular en 2022*.
<https://profile.es/blog/python/>
- Moreira, R. D. (2022). *DESARROLLO DE ALGORITMOS DE PROCESAMIENTO DE IMÁGENES AGRÍCOLAS OBTENIDAS POR DRONES PARA LA DETECCIÓN DE PLANTAS FALTANTES EN CULTIVOS DE MAÍZ*.
- Moreno, J. K. (2016). *Sistema de monitoreo, vigilancia y control mediante el uso de drones para los campos agrícolas–Tulcán*.
- Muñoz, V., & Muyulema, Y. (2022). *RESPUESTA AGRONÓMICA DE LAS VARIETADES DE CAFÉ GEISHA, SARCHIMOR Y MANABÍ CON FERTILIZACIÓN ORGÁNICA E INORGÁNICA EN EL CENTRO EXPERIMENTAL SACHA WIWA*.
- NVIDIA®, D. (2021). *Getting Started With Jetson Nano Developer Kit | NVIDIA Developer*.
<https://developer.nvidia.com/embedded/learn/get-started-jetson-nano-devkit>
- Parada, P. (2021). *Tecnologías y aplicaciones innovadoras en la agricultura de precisión*.
<https://blog.softtek.com/es/tecnolog%C3%ADas-y-aplicaciones-innovadoras-en-la-agricultura-de-precisi%C3%B3n>
- Parra, D. (2017). *Sistema multiagente para el manejo óptimo de un cultivo empleando la metodología prometheus*.
- Pazmiño, A., Jácome, J., Zabala, L., & Gavilanes, J. (2018). *MATLAB BÁSICO*.
- Pérez, R. (2018). *Proyecto de implementación de monitoreo, seguimiento y control automático para las variables que más impactan el cultivo de tomate de mesa en invernadero en el municipio de villa de Leyva -Boyacá*.
- Pino, E., Nacional, U., Grohmann, J. B., & Tacna, P. (2019). *Volumen 37, N° 1. Páginas 75-84 IDESIA (Chile) Marzo*. <https://doi.org/10.4067/S0718-34292019005000402>
- Pinto, R. (2019). *Drones: la tecnología, ventajas y sus posibles aplicaciones*.
- Plawrence. (2019). *Jetson Nano Developer Kit*.
- Pusdá Chulde, M. R. (2022). *Algoritmos para agricultura de precisión utilizando computación de alto rendimiento [Universidad Nacional de La Plata]*.
<https://doi.org/10.35537/10915/149069>
- Ramírez, V., Cárdenas, D. M., & Ruiz, S. (2018). Programación o planeación de actividades o recursos en la agricultura. Una revisión de literatura. *Revista EIA*, 15(30), 73–87.
<https://doi.org/10.24050/reia.v15i30.1151>
- Raparelli, E., & Bajocco, S. (2019). A bibliometric analysis on the use of unmanned aerial vehicles in agricultural and forestry studies. *International Journal of Remote Sensing*, 40(24), 9070–9083. <https://doi.org/10.1080/01431161.2019.1569793>
- Rehman, T. U., Zaman, Q. U., Chang, Y. K., Schumann, A. W., & Corscadden, K. W. (2019). Development and Field Evaluation of a Machine Vision Based In-Season Weed Detection System for Wild Blueberry. *Comput. Electron. Agric.*, 162, 1–13.
<https://doi.org/10.1016/j.compag.2019.03.023>

- Ríos, R. (2021). *Uso de los Drones o Vehículos Aéreos no Tripulados en la Agricultura de Precisión*. <https://www.redalyc.org/journal/5862/586268743010/html/>
- Rivas, A. (2020). *Ventajas y desventajas de C++ - Qué es, ejemplos y definición - Muy Tecnológicos*. <https://muytecnologicos.com/diccionario-tecnologico/ventajas-y-desventajas-de-c-mas-mas>
- Robledano, A. (2019a). *Qué es C++: Características y aplicaciones*.
- Robledano, A. (2019b). *Qué es C++: Características y aplicaciones | OpenWebinars*. <https://openwebinars.net/blog/que-es-cpp/>
- Robledano, A. (2019c). *Qué es Python: Características, evolución y futuro | OpenWebinars*. <https://openwebinars.net/blog/que-es-python/>
- Rodriguez, J. L., & Espejo, E. D. (2020). *Cartografía con drones (VANT s)*. Editorial UPTC. <https://elibro.net/es/lc/utnorte/titulos/193945>
- Roldán Ortega, B., Roshan Biswal, R., & Sánchez Delacruz, E. (2019). Detección de enfermedades en el sector agrícola utilizando Inteligencia Artificial Detection of Diseases in the Agricultural Sector using Artificial Intelligence. In *Research in Computing Science* (Vol. 148, Issue 7).
- Ruipérez, A. (2019). *Evaluación de la tarjeta NVIDIA Jetson TK1 para aplicaciones de video-vigilancia*.
- Ruipérez, P. (2017). *Diseño y fabricación de un dron*.
- Sánchez, A. F. A. (2019). LA AGRICULTURA DE PRECISIÓN EN LA EMPRESA AGRARIA DE CUBA. *Revista Derecho & Paz*, 1(40), 257–285. <https://doi.org/10.32713/rdp.v1i40.1133>
- Schram, J. (2020). *Monitoreo De Cultivos Por Satélite: Lo Que Hay Que Saber*. <https://eos.com/es/blog/monitoreo-satelital-de-cultivos/>
- Sepúlveda-Cisneros, O., Acevedo-Juárez, B., Saldaña-Durán, C. E., Castro, W., De-La-Torre, M., & Avila-George, H. (2020). *Desarrollo de un sistema embebido para un compostero doméstico inteligente*. <https://doi.org/10.17013/risti.41.112-129>
- Solé, R. (2021, June 18). *Raspberry Pi: Crea proyectos DIY por muy poco dinero*. <https://www.profesionalreview.com/2021/07/18/que-es-raspberry-pi/>
- Solórzano, B., & Jiménez, M. (2018). *Estudio de algoritmos en imágenes para conteo de población del cultivo de banano*. 1–12. http://www.uagraria.edu.ec/publicaciones/revistas_cientificas/18/067-2018.pdf
- Süzen, A. A., Duman, B., & Şen, B. (2020). Benchmark Analysis of Jetson TX2, Jetson Nano and Raspberry PI using Deep-CNN. *2020 International Congress on Human-Computer Interaction, Optimization and Robotic Applications (HORA)*, 1–5. <https://doi.org/10.1109/HORA49412.2020.9152915>
- Torky, M., & Hassanein, A. E. (2020). Integrating blockchain and the internet of things in precision agriculture: Analysis, opportunities, and challenges. *Computers and Electronics in Agriculture*, 178, 105476. <https://doi.org/https://doi.org/10.1016/j.compag.2020.105476>

- Torres, M., Correa, C., Fredy, J., & Gómez, V. (2019). *Configuración de un Control de Temperatura en un Sistema Embebido de Bajo Costo, usando Herramientas de Inteligencia Artificial y el Internet de las Cosas*. <https://doi.org/10.17013/risti.34.68-84>
- Velasco, R. (2021). *Sistema operativo para Raspberry Pi - Mejores para todos los usos*. <https://www.softzone.es/programas/sistema/mejores-sistemas-operativo-raspberry-pi/>
- Villalvazo, J. (2020). *Que ventajas tiene Python si lo comparamos con otros lenguajes de programacion?* <https://respuestasrapidas.com.mx/que-ventajas-tiene-python-si-lo-comparamos-con-otros-lenguajes-de-programacion/>
- Wang, A., Zhang, W., & Wei, X. (2019). A Review on Weed Detection Using Ground-Based Machine Vision and Image Processing Techniques. *Computers and Electronics in Agriculture*, 158, 226–240. <https://doi.org/10.1016/j.compag.2019.02.005>